UNIVERSIDAD DE OVIEDO

Programa de Doctorado en Informática
Departamento de Informática

Ensembles of Quantifiers

Ph.D. Thesis in Computer Science

Pablo José Pérez Gállego

Ph.D. Supervisors
Dr. Juan José del Coz Velasco
Dr. José Ramón Quevedo Pérez
Centro de Inteligencia Artificial

2017

RESUMEN DEL CONTENIDO DE TESIS DOCTORAL

1.- Título de la Tesis	
Español/Otro Idioma: **Algoritmos de cuantificación basados en combinación de modelos**	Inglés: **Ensembles of Quantifiers**

2.- Autor	
Nombre: **Pablo Pérez Gállego**	DNI/Pasaporte/NIE: **. . -M**
Programa de Doctorado: **Doctorado en Informática**	
Órgano responsable: **Comisión Académica del Programa de Doctorado**	

RESUMEN (en español)

Durante los últimos años hemos asistido a un crecimiento exponencial de los datos disponibles. Una consecuencia en la comunidad del aprendizaje automático ha sido identificar nuevas tareas, como son los problemas que requieren obtener estimaciones agregadas para conjuntos de ejemplos, en lugar de realizar predicciones individuales para cada uno de ellos. La motivación detrás de esta idea es que, a medida que los volúmenes de datos crecen, es menos importante centrarse en los individuos, que en cómo esos individuos se comportan colectivamente. El objetivo, por ejemplo, puede ser dar respuesta a preguntas como *¿Cuántos clientes están satisfechos con nuestro nuevo producto?*

La cuantificación es una de esas nuevas tareas de aprendizaje que necesitan producir predicciones agregadas. Su objetivo es estimar el número de casos que pertenecen a cada clase, es decir, predecir la distribución de las clases en el conjunto. El aspecto clave es que los datos usados para entrenar los cuantificadores pueden tener una distribución substancialmente diferente respecto a las muestras de test. El método más intuitivo para cuantificar es simplemente clasificar y contar los ejemplos que pertenecen a cada clase. Sin embargo, la literatura prueba que este enfoque puede ser mejorado.

El concepto de cambio en la distribución es crucial para entender los principios de la cuantificación y las propuestas de esta tesis. Un cambio en la distribución de los datos ocurre cuando la probabilidad conjunta de la descripción de los ejemplos y sus salidas varía entre entrenamiento y test, $P_T(\boldsymbol{x},y) \neq P_D(\boldsymbol{x},y)$. Dependiendo del problema, es posible especificar de antemano que componentes de $P(\boldsymbol{x},y)$ cambian y cuales pueden ser constantes. En cuantificación, la distribución de las clases, $P(y)$, cambia por definición, y se asume que $P(\boldsymbol{x}|y)$ es constante.

Nuestra hipótesis principal es que existe una conexión entre los problemas con cambios en la distribución y las propiedades de los *ensembles*, como es la diversidad, que ayuda a obtener mejores meta-modelos. La diversidad se puede introducir generando una muestra de entrenamiento distinta para cada modelo. Este punto conecta la cuantificación con los *ensembles*. Siguiendo esa idea, analizamos como hipótesis que los *ensembles* pueden ser apropiados en problemas que (i) tienen cambios en la distribución y (ii) esos cambios se pueden caracterizar.

La principal propuesta de esta tesis está basada en generar diferentes muestras de entrenamiento, cada una de ellas representando un cambio esperado en la distribución. Cuando una nueva muestra debe ser cuantificada, es probable que el meta-modelo contenga algún modelo que haya sido entrenado con una distribución de clases similar. En primer lugar nos centramos en la cuantificación binaria, proponiendo los llamados EoQ (*Ensembles of Quantifiers*) y nuestros experimentos demuestran que los EoQ tienen más precisión que cuantificadores individuales, incluso cuando las estrategias de combinación usadas son triviales.

Además, la tesis propone medidas de selección diseñadas para cuantificación. La idea es seleccionar los mejores modelos del *ensemble* en lugar de agregar todos ellos. Por ejemplo, para dar más importancia a los modelos cuya distribución de entrenamiento se parece más a la de la muestra a predecir. Se introducen tres nuevas medidas de selección y dos de ellas permiten realizar una selección dinámica. Los experimentos prueban que estas estrategias de selección superan en muchos casos a las estrategias tradicionales, como promediar las salidas de todos los modelos o seleccionarlos de acuerdo a su precisión.

Por último, se aplicaron los EoQ, junto con las medidas de selección propuestas, a un problema real, la cuantificación de sentimientos, en el que se estima la distribución de comentarios como positivos, neutrales y negativos. Se emplearon las cuatro últimas competiciones de SemEval y los EoQ resultaron tan efectivos como en otros problemas, combinando alta precisión y estabilidad.

RESUMEN (en inglés)

During the last years we are assisting to an exponential growth of data availability. One of the consequences within the machine learning community is the identification of a new useful learning task: to produce aggregated estimations for a full sample rather than giving a specific prediction for each instance. The motivation behind this is that, as the data volume increases, the less important is to focus on individuals, but on how these individuals behave together as a group. The goal is to answer questions like *"How many consumers are satisfied with our new product?"*

Quantification learning is one of the prototypical methods based on giving sample level predictions. Its objective is to accurately estimate the number of cases belonging to each class (or class distribution) in a test set, using a training set that may have a substantially different distribution. At a first sight, the most intuitive way to quantification is to count the predictions of a classifier over a test set. This method has already been proved to perform poorly, since it does not considerate distribution changes.

The concept of *distribution shift* is crucial to understand both quantification learning and our proposals in this thesis. Distribution shift occurs when the joint distribution of inputs and outputs changes between the training and testing phase, $P_T(\mathbf{x},y) \neq P_D(\mathbf{x},y)$. Depending on each concrete problem, it is possible to specify in advance which components in $P(\mathbf{x},y)$ are expected to change, and which ones may remain constant. As for quantification, class distribution, $P(y)$, changes by problem definition, and it is assumed that $P(\mathbf{x}|y)$ remains constant.

We have realized that these distribution shift characteristics are closely related to some properties of ensemble learning; diversity is one of such key factors contributing to a good performing ensemble. Usually, diversity is introduced by generating different training samples for each model. This connects distribution shift problems and ensemble learning: each model can be trained with a different data distribution. Following this idea, this work analyzes the hypothesis that ensembles may be especially appropriate in problems that: (i) suffer from distribution changes, (ii) it is possible to characterize those changes beforehand. Our main proposal is based on generating different training samples, with each one representing an expected distribution change. When a new test set is to be quantified, the ensemble is likely to contain some models trained with a similar class distribution. We have focused on binary quantification, and proposed EoQ (*Ensembles of Quantifiers*). Experimental results show that EoQ outperform the original counterpart algorithms, even when trivial aggregation rules are used.

Our next contribution is to improve EoQ by designing ensemble selection measures particularly devised for quantification. The idea is to select some potentially good ensemble models instead of aggregating all of them. For example, it would be possible to give more importance to those weak quantifiers whose training distribution is closer to the test sample distribution. We have proposed three quantification based selection measures and two of them are defined for dynamic ensemble selection. The experiments demonstrate that, in many cases, these selection functions outperform straightforward approaches, like averaging all models and using quantification accuracy to prune the ensemble.

The last contribution of this thesis is to evaluate the effectiveness of EoQ, together with the proposed selection measures, in a real-world application. We have studied the usage of a decomposition approach based on EoQ for a multi-class sentiment quantification task, where each comment is labeled as positive, negative or neutral. The last four *SemEval* competitions have been used as benchmarks, and we have found that ensembles behave as good as in other learning problems, combining robustness, stability and a competitive performance.

Agradecimientos

Me gustaría mostrar especialmente mi más sincero agradecimiento a Juanjo por toda su ayuda, esfuerzo y dedicación durante estos cuatro años, y por no dejarme que me durmiese en los laureles en más de una ocasión, sin ti esta tesis no hubiese llegado a buen puerto.

Agradecer también a mi otro director, Quevedo, por sus importantes contribuciones en cada uno de nuestros trabajos conjuntos. Así como a cada uno de los miembros del Centro de Inteligencia Artificial, grupo de Aprendizaje Automático, con los que he tenido el placer de colaborar durante estos años, y especialmente relevante fue la oportunidad que me concedieron de trabajar con ellos y adentrarme en el mundillo del "*Machine Learning*", allá por 2010.

A mis padres, por ayudarme y apoyarme siempre incondicionalmente, ya fuese en la cercanía, o en alguno de los momentos más lejanos que nos ha tocado vivir últimamente.

Dar las gracias también a mis compañeros de trabajo por permitir que me tomase ciertas libertades en los momentos de más apuro, especialmente a Nacho por animarme a darle el último empujón, y a mis compis de mesa, Carmen y Fer, por preocuparse de cada una de las evoluciones. A mis amigos, Abel, Gerardo, Greg, Olga (y más) ... por todo el apoyo durante este tiempo.

Por último, quiero agradecer también a las entidades que han ayudado a financiar esta investigación, en concreto, el Ministerio de Economía y Competitividad (MINECO) y el Fondo Europeo de Desarrollo Regional (FEDER), mediante la subvención TIN2015-65069-C2-2-R.

Resumen

Durante los últimos años hemos asistido a un crecimiento exponencial de los datos disponibles. Una de las consecuencias dentro de la comunidad dedicada al Aprendizaje Automático ha sido la identificación de nuevas tareas. Una de ellas la constituyen aquellos problemas que requieren obtener estimaciones agregadas para conjuntos de ejemplos, en lugar de realizar predicciones individuales para cada uno de ellos. La motivación detrás de esta idea es que, a medida que los volúmenes de datos crecen, es menos importante centrarse en los individuos, que en cómo esos individuos se comportan colectivamente. El objetivo, por ejemplo, puede ser dar respuesta a preguntas como *¿Cuántos consumidores están satisfechos con nuestro nuevo producto?*

La cuantificación es una de esas nuevas tareas de aprendizaje que están pensadas para producir predicciones agregadas. Su objetivo es estimar el número de casos que pertenecen a cada clase, o dicho de otra forma, predecir la distribución de las clases dado un conjunto de casos. El aspecto clave es que los datos usados para entrenar los cuantificadores pueden tener una distribución substancialmente diferente respecto a las muestras de ejemplos que luego deben predecirse. A primera vista, el método más intuitivo para cuantificar es simplemente clasificar los ejemplos individuales y contar los que pertenecen a cada clase. Sin embargo, los estudios realizados prueban que este enfoque puede ser mejorado ya que no tiene en cuenta los cambios que pueden ocurrir en la distribución de los datos.

El concepto de cambio en la distribución es crucial para entender los principios del aprendizaje de cuantificadores y las propuestas de esta tesis. Un cambio en la distribución de los datos ocurre cuando la probabilidad conjunta de la descripción de los ejemplos y las salidas correspondientes varía entre la fase el entrenamiento y la de test, $\mathbf{P}_T(\boldsymbol{x}, y) \neq \mathbf{P}_D(\boldsymbol{x}, y)$. Dependiendo de cada problema concreto, es posible especificar de antemano que componentes de $\mathbf{P}(\boldsymbol{x}, y)$ se espera que cambien y cuales deberían ser constantes. En cuantificación, la distribución de las clases, $\mathbf{P}(y)$, cambia por la propia definición del problema, y se asume habitualmente que $\mathbf{P}(\boldsymbol{x}|y)$ permanece constante.

Una de las hipótesis de este trabajo es que existe una conexión entre los problemas que sufren cambios en la distribución y las propiedades de los métodos basados en combinar modelos (*ensembles*). Una de esas propiedades es la diversidad que contribuye a obtener mejores meta-modelos. La diversidad puede ser introducida en un *ensemble* generando distintas muestras de entrenamiento para cada modelo

individual. Este el punto de conexión con los problemas con cambios en la distribución: cada modelo puede ser entrenado con una distribución diferente. Siguiendo esa idea, esta tesis analiza la hipótesis que los *ensembles* pueden ser especialmente apropiados en los problemas que (i) tienen cambios en la distribución y (ii) es posible caracterizar dichos cambios de antemano.

La principal propuesta de esta tesis está basada en generar diferentes muestras de entrenamiento, cada una de ellas representando un cambio esperado en la distribución. Cuando una nueva muestra debe ser cuantificada, es probable que el meta-modelo contenga algún modelo individual que haya sido entrenado con una distribución de clases similar. Este trabajo se ha centrado fundamentalmente en la cuantificación binaria, y se ha propuesto una adaptación de los *ensembles* que denominamos EoQ (*Ensembles of Quantifiers*). Los resultados experimentales realizados demuestran que los modelos EoQ tienen más precisión que los modelos individuales que los forman, incluso cuando las estrategias de combinación empleadas son triviales, como devolver la estimación media de todos los modelos individuales.

No obstante, la tesis propone también mejorar los EoQ mediante medidas de selección especialmente diseñadas para problemas de cuantificación. La idea es seleccionar los mejores modelos del *ensemble* en lugar de agregar todos ellos. Por ejemplo, sería posible dar más importancia a aquellos modelos cuya distribución de entrenamiento se parece más a la distribución de la muestra que se va a predecir. Se introducen para ello varias nuevas medidas de selección adaptadas a la cuantificación, permitiendo dos de ellas realizar una selección dinámica de modelos. Los experimentos llevados a cabo demuestran que estas estrategias de selección superan en muchos casos a las estrategias tradicionales, como promediar las salidas de todos los modelos o seleccionarlos de acuerdo a su precisión. Además, los resultados muestran que el rendimiento depende de la combinación entre el algoritmo de cuantificación base y la medida de selección empleada.

La cuantificación de las opiniones de los usuarios en redes sociales constituye una tarea interesante, no solo en campos relacionados con las ciencias sociales, sino también con la economía y los negocios. En ese sentido, se aplicaron los EoQ, junto con las medidas de selección propuestas, a un problema multi-clase de cuantificación de sentimientos en el que hay que estimar la distribución de tres clases de comentarios: positivos, neutrales y negativos. Se emplearon para ello las cuatro últimas competiciones de SemEval y los *ensembles* resultaron tan efectivos como los son en otros problemas de aprendizaje, combinando alta precisión y estabilidad.

Conclusiones

En este trabajo se ha estudiado cómo pueden adaptarse los métodos basados en combinación de modelos (*ensembles*), a problemas que se caracterizan porque la distribución de los datos cambia entre la fase de entrenamiento y una vez el modelo se despliega y se usa en la práctica. La idea principal de este trabajo consiste en, basándose en dicha asunción, diseñar *ensembles* que contengan la necesaria diversidad. Para lograrlo, se generan diferentes muestras de entrenamiento para representar los cambios esperados en la distribución. En primer lugar, se probó experimentalmente esta idea en problemas de cuantificación binaria, que se caracterizan porque la prevalencia de las clases, $\mathbf{P}(y)$, cambia, pero en donde podemos asumir que $\mathbf{P}(\boldsymbol{x}|y)$ permanece constante. Las muestras de entrenamiento se generan bajo dicha asunción de aprendizaje. Como resultado, se espera que el meta-modelo formado por los cuantificadores individuales aprendidos con las citadas muestras esté mejor preparado para hacer predicciones sobre nuevas muestras no vistas en las que la prevalencia de las clases varíe significativamente. Los resultados experimentales sobre conjuntos de referencia que se exponen en el Capítulo 3 demuestran que los *ensembles* de cuantificadores (EoQ , *Ensembles of Quantifiers*) propuestos mejoran el rendimiento de cuantificadores individuales generados con los algoritmos de cuantificación existentes.

Por tanto, una de las contribuciones más importantes de este trabajo es proponer un método teóricamente razonable para usar algoritmos basados en combinación de modelos en el contexto de la cuantificación binaria. Sin embargo, es digno de mencionar que la importancia de la solución propuesta no solamente se limita a los problemas de cuantificación, sino que la misma idea también podría aplicarse en otros problemas de aprendizaje en los que se sabe que la distribución cambia, y donde además es posible caracterizar de alguna forma dichos cambios, como por ejemplo los problemas en los que se da el denominado *covariate shift*. Además, creemos que este trabajo abre nuevas líneas de investigación dentro del aprendizaje de *ensembles*, en aspectos como la inclusión de diversidad en función de los cambios esperados en la distribución.

La segunda contribución de este trabajo es proponer nuevos criterios para la selección de modelos que pueden usarse para implementar tanto estrategias de selección estáticas, que aplican los mismos modelos en todos los casos, descartando el resto, como estrategias dinámicas, donde se pueden usar un grupo de modelos distinto en función de la muestra que deba predecirse. Además, algunos de los criterios propuestos están diseñados para ciertos algoritmos concretos de

cuantificación, explotando sus peculiaridades. Los experimentos del Capítulo 4 muestran que estos criterios usados en combinación con estrategias simples de selección, mejoran la capacidad predictiva de los EoQ. Otra importante conclusión de ese estudio experimental es que el rendimiento depende en buena medida de una correcta combinación entre el algoritmo de cuantificación usado para aprender los cuantificadores individuales y el criterio empleado para seleccionar los modelos resultantes. En concreto, los criterios diseñados *ad hoc* para un algoritmo concreto funcionan mejor con dicho método que con otros algoritmos de cuantificación, lo que en parte valida su diseño.

El hecho de que la cuantificación requiera hacer estimaciones sobre conjuntos de ejemplos y no predicciones sobre instancias individuales, requiere que los trabajos realizados hasta ahora relacionados con el aprendizaje de *ensembles* deban adaptarse a este nuevo tipo de problemas. De acuerdo con los resultados obtenidos en este trabajo, los criterios de selección que parecen proporcionar resultados más prometedores son aquellos que miden la similitud entre la distribución de la muestra de test que debe predecirse, y la distribución de las muestras con las que se entrenaron los cuantificadores individuales que forman el meta-modelo.

Para finalizar, en este trabajo se ha analizado la aplicabilidad de los EoQ propuestos en problemas multi-clase de cuantificación de sentimientos en donde las opiniones de los usuarios pueden ser etiquetadas como positivas, neutrales o negativas. Los EoQ retornan en este caso la distribución de probabilidad de estas tres clases dada una muestra que contiene un conjunto de opiniones. Los experimentos del Capítulo 5 realizados empleando los conjuntos de datos correspondientes a las competiciones organizadas por SemEval, permiten concluir que el rendimiento de los EoQ es al menos competitivo con respecto a los cuantificadores individuales, y en varios casos los supera. De hecho, los EoQ son los vencedores en la mayoría de los conjuntos de SemEval, y además se pueden observar algunas propiedades interesantes, como es su robustez frente al sobre-ajuste.

Abstract

During the last years we are assisting to an intense and exponential growth of data availability. One of the consequences within the Machine Learning community is the identification of a new useful learning task: to produce aggregated estimations for a full sample rather than giving a specific prediction for each instance. The motivation behind this is that, as the data volume increases, the less important is to focus on individuals, but on how these individuals behave together as a group. The goal is to answer questions like *how many consumers are satisfied with our new product?*

Quantification learning is one of the prototypical methods based on giving sample level predictions. Its objective is to accurately estimate the number of cases belonging to each class (or class distribution) in a test set, using a training set that may have a substantially different distribution. At a first sight, the most intuitive way to quantification is to count the predictions of a classifier over a test set. This method has already been proved to perform poorly, since it does not considerate distribution changes between training and testing.

The concept of *distribution shift* is crucial to understand both quantification learning and our proposals in this dissertation. Distribution shift occurs when the joint distribution of inputs and outputs changes between the training and testing phase, $\mathbf{P}_T(\boldsymbol{x}, y) \neq \mathbf{P}_D(\boldsymbol{x}, y)$. Depending on each concrete problem, it is possible to specify in advance which components in $\mathbf{P}(\boldsymbol{x}, y)$ are expected to change, and which ones are expected to remain constant. As for quantification, class distribution, $\mathbf{P}(y)$, changes by problem definition, and it is assumed that $\mathbf{P}(\boldsymbol{x}|y)$ remains constant.

We have realized that these distribution shift characteristics are closely related to some of the most desirable properties for ensemble learning. Diversity is one of the key factors contributing to a good performing ensemble. Usually, it is introduced by generating different training samples for each model. This is the connecting point between distribution shift problems and ensemble learning: each model can be trained with a data distribution that may be different from the original training set distribution. Following this idea, this dissertation analyzes the hypothesis that ensembles can be especially appropriate in problems that: (i) suffer from distribution changes, (ii) it is possible to characterize those changes beforehand.

Our main proposal in this dissertation is based on generating different training samples, with each one representing an specific and expected distribution change. When a new test set is to be quantified, the ensemble is likely to contain some models trained with a similar class distribution. We have focused on binary quantification problems, and proposed an ensemble version adaptation called EoQ (*Ensembles of quantifiers*). For each weak quantifier a different training sample with a random prevalence is generated. Experimental results show that these ensemble adaptations outperform the original counterpart algorithms, even when trivial aggregation rules are used.

Our next contribution in this dissertation is to improve EoQ by designing ensemble selection measures particularly devised for quantification problems. The idea is to select some potentially good ensemble models instead of aggregating all of them. For example, it would be possible to give more importance to those weak quantifiers whose training distribution is closer to the new test sample distribution. We have proposed three quantification based selection measures, where two of them are defined for dynamic ensemble selection. The experiments demonstrate that, in many cases, these selection functions outperform straightforward approaches, like averaging all models and using quantification accuracy to prune the ensemble. Moreover, the results show that performance heavily depends on the combination of the base quantification algorithm and the selection measure.

Quantification of user-generated opinions in social media constitutes an interesting task, not only in social science related fields, but also to economics and business. The last contribution of this dissertation is to evaluate the effectiveness of EoQ, together with the proposed selection measures, in a real-world quantification application. We have studied the usage of a decomposition approach based on EoQ for a multi-class sentiment quantification task, where each comment is labeled as positive, negative or neutral. These quantifiers are designed to predict the probability class distribution given a new unlabeled set of comments. The last four *SemEval* competitions have been used as benchmarks, and we have found that ensembles behave as good as in other learning problems, combining robustness, stability and a competitive performance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter is intended to provide the reader with a conceptual review of the main concepts that will be further developed along this dissertation. We start by introducing the problem that is behind our research work, and we try to motivate our original approach to solve it. Then, we establish the main goals and contributions of this dissertation, summing up: to introduce, design, improve and check for the effectiveness of a novel approach to tackle quantification via ensemble learning. Finally, we enumerate and analyze the list of publications, both in conferences and journals, this dissertation has led to, and we describe the outline of the rest of this document.

1.1 Motivation

During the last years we are assisting to an intense and exponential growth of data availability; two new players have joined the match, namely, *Big Data* and *Cloud Computing*. We are emphasizing next the causes for this rise: i) businesses going digital, focusing on data acquisition and retention in order to migrate to data-driven business models, ii) technology improvements making data storage and transmission costs to decrease, iii) ubiquitous existence of sensors monitoring, tracking and recording data (GPS enabled cellphones, credit/debit cards, email devices, etc.), and iv) the explosion of social network related user created data. Some examples of these new large volumes of available data are the following: *Twitter* user generated opinion feeds, *app* log data describing how users interact with the applications, *Google* search data feeds, or support-call logs detailing user support issues.

The machine learning community is well aware of this, time and time more noticeably, explosion in data availability. As a consequence, a new learning task has recently started to be identified as a real necessity: to produce aggregated estimations for a full sample rather than giving an specific prediction for each instance. The motivation behind this is that, as the data volume increases, the less important is to focus on individuals, but on how these individuals behave together as a group. Under this framework, the learning evaluation unit changes from

data instances, to data samples. For instance, there is an increasing demand for automatic methods to track overall consumer opinions [Gao and Sebastiani, 2015a]. The goal is to answer questions like *how many consumers are satisfied with our new product?*. This task requires effective algorithms focused on estimating the class distribution from a sample. Notice that the goal is not to label individual examples (solved using traditional classification algorithms), but to obtain estimations at aggregated level.

We do think that sample level based learning will acquire more attention from the research community once the utility notion of giving estimates for samples, vs the traditional example classification task, gets fully assimilated. There exist situations, particularly in real-world problems, in which training an accurate classifier is whether complicated or costly. Some real-world concepts are difficult to distinguish for state-of-the-art classifiers, and special purpose learners are expensive. Still, under these circumstances, it is possible to give accurate estimations for the whole sample. Some problems are inherently based on predicting characteristics for a sample, rather than for each instance comprising it. This is especially true in the case of tracking trends over time tasks [Rakthanmanon et al., 2012], such as early detection of epidemics and endangered species, risk prevalence and ecosystems evolution. In strategy planning applications usually is sufficient, and sometimes even more relevant, to have an aggregated estimation. For example, assigning priorities and resources in a product support department by quantifying specific product issues.

1.1.1 Problem definition

Quantification is one of the prototypical learning methods based on giving sample level predictions. It is a supervised learning task introduced by Forman in the recent years [Forman, 2008], aimed at estimating the class distribution in a test set, using a training set that may have been drawn from a different distribution. Given a test set, a quantifier outputs an estimation of the number of examples belonging to each class, without requiring individual predictions for each instance.

It is not a famous task within the machine learning community yet, principally because of two reasons: i) it has been mistakenly addressed as a trivial problem, and ii) the omnipresent training set random sampling and cross validation procedures, prevent the testing distribution from mismatching the training one. Many practitioners tackle quantification by inducing a classifier over the training set, and then classifying and counting the test examples. This straightforward approach has already been proved to perform poorly, because the classifier does not take into account the class distribution shift, which by definition, appears in quantification problems [Forman, 2008]. As a consequence, the associated quantifier inherits the classifier bias, which could lead to a major degradation when the data distribution effectively changes.

Interestingly, quantification methods have been lately applied to solve several real-world problems from different application areas. A typical example is related to sentiment analysis in social networks [Gao and Sebastiani, 2015b], e.g., to estimate the percentage of positive comments about a certain topic of interest during a concrete time period [González et al., 2016b]. Automatic methods based on quantification have been used to predict the proportion of membrane-intact sperm cells, which is one of the analyzed aspects in fertility studies and was traditionally carried out by human experts [Alaiz-Rodríguez et al., 2008]. Quantifiers have been also applied to quantify recurrent issues based on the analysis of technical-support call logs recorded by customer services [Forman et al., 2006]. Early detection of such issues is useful, among others, to identify the problems that may become epidemics. In Epidemiology, King and Lu [2008] proposed a quantification approach to estimate the cause of death distribution in a population using, what it is called, verbal autopsies (surveys on deceased's symptoms provided by relatives). This procedure is cheap because physician reviews are not required. Plankton abundance estimation González et al. [2013]; Solow et al. [2001] is another paradigmatic quantification task, whose goal is to estimate the abundance of some taxonomic groups given plankton samples. Other tentative application scopes include network-behavior analysis [Tang et al., 2010], quality control [Sánchez et al., 2008], or credit scoring [Hand, 2006].

The main difference between classification and quantification is that a quantifier produces estimates for *bags* or samples (groups of instances), instead of making predictions for individual examples like a classifier does. Thus, given that the sample is the evaluation unit, a more complex methodology is necessary to accurately estimate the performance of a quantifier. It is required a large group of test samples covering the whole prevalence (class distribution) domain. Notice that, as the class distributions change by problem definition, it is recommendable to represent these changes in the testing set. Having available this prevalence-diverse test samples group is not always a situation, and artificially generating it is not a trivial task neither.

The concept of data distribution drift, also named as dataset shift [Moreno-Torres et al., 2012], is crucial to understand quantification learning from a conceptual point of view. Supervised learning makes the assumption that the unknown distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$, from where the examples are drawn independently and identically distributed (i.i.d. assumption), does not change between the training and testing or production phases. That is to say that the probability distribution of the problem is faithfully represented in the training set. However, in practice, this assumption often does not hold in real-world applications [Quiñonero Candela et al., 2009; Storkey, 2009]. Dataset shift occurs when the joint distribution of inputs and outputs changes between the training phase and the deployment of the model, $P_T(\boldsymbol{x}, y) \neq P_D(\boldsymbol{x}, y)$. This may occur for several reasons, but usually due to the fact that samples are captured at different times and/or places in a changing environment.

Let's see how we can characterize quantification learning from a dataset shift point of view. Depending on the causal relation between the covariates, $\boldsymbol{x}$, and the class variable, y, Fawcett and Flach [2005] identifies two class of problems: $\mathcal{X} \rightarrow \mathcal{Y}$ and $\mathcal{Y} \rightarrow \mathcal{X}$. The joint distribution is written as $\mathbf{P}(y|\boldsymbol{x})\mathbf{P}(\boldsymbol{x})$ in the former, and as $\mathbf{P}(\boldsymbol{x}|y)\mathbf{P}(y)$ in the latter. Moreno-Torres et al. [2012] proposes a categorization for those learning tasks presenting a drift into the distribution. The type of dataset shift problem is determined by the components that are expected to change, and by the ones expected to remain constant. This constitutes a key statement within the scope of this dissertation, since it means that it is possible to characterize in advance the distribution drift we are expecting to take place in a concrete problem or application. In particular, quantification, also referred to as *prior probability shift* problem, is defined in terms of $\mathbf{P}(\boldsymbol{x}|y)\mathbf{P}(y)$, with $\mathbf{P}(y)$ changing by own problem definition, and $\mathbf{P}(\boldsymbol{x}|y)$ remaining constant between training and testing (some proper quantification methods, described in Section 2.4, assume this fact as a base hypothesis).

1.1.2 Proposing ensembles of quantifiers

At this point, we are already aware of the existence of problems in which:

(i) data distribution changes between training and testing phases, and

(ii) it is possible to characterize those changes beforehand, i.e. which components are expected to get modified, and which ones to stay unaltered.

Once upon a time, a few months after the start of this dissertation, we realized that those characteristics were surprisingly related to some of the most desirable properties for ensemble learning. Two decidedly important design decisions one has to face when constructing an ensemble are: to select an appropriate procedure in order to introduce diversity into the ensemble, and to choose an aggregation function to produce the final ensemble prediction. Usually, diversity is obtained by generating different training samples. As for our interests, diversity could be achieved by producing training samples representing the expected distribution changes (assuming we can a priori define them). Moreover, it was also possible to introduce specific quantification knowledge into the definition of the aggregation function. Back then, our intuition was that ensembles clearly manifested symptoms of being a promising method for solving quantification problems.

An ensemble is a meta-model that combines the predictions of its single models to obtain consensus decisions. Diversity is one of the key factors contributing to a good performing ensemble: the solution implicitly represents some kind of agreement between the participating models. However, this cooperation is

not beneficial without diversity. Diversity means that individual models of the ensemble must produce different predictions given a testing case [Banfield et al., 2005; Kuncheva and Whitaker, 2003; Brown et al., 2005]. If ensemble models always make the same prediction, their combination does not lead to any improvement. Only when models are different and each of them excels for a particular region of the input space, the ensemble obtains a better performance than their individual models. An ensemble tends to reduce the variance of its base classifier, thus reducing the risk of obtaining a particularly bad response from a single model. Intuitively, the same idea is highly present in the human decision making processes; a set of opinions is more rich than an isolated opinion, especially when there exist a high degree of diversity within the opinions. Thus, from a practical point of view, ensembles generally perform better than a single-model solution [Bauer and Kohavi, 1999; Breiman, 1996; Freund et al., 1996; Opitz and Maclin, 1999], although this cannot be guaranteed [Fumera and Roli, 2005]. The fusion strategy defines how model predictions are combined, and depends, in part, on the learning task. Majority voting is the most popular strategy in classification problems, while averaging is the predominant rule for regression and probability estimation tasks. In any case, ensemble models can be treated equally or not, assigning a different weight to each one; usually the weight is proportional to its accuracy or precision.

In this dissertation we propose that ensemble learning results appropriate and effective when applied to problems verifying the next properties: (i) are known to suffer from distribution changes between training and testing phases, (ii) we are able to characterize the distributional changes beforehand (i.e. we can define the conditions that make $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ to change). The reason is that each model of the ensemble can be trained with a training set whose data distribution may be different from the distribution of the original dataset. This fact can be further exploited in learning problems that suffer a characterizable drift in the distribution. From an ensemble point of view, we can make the most of this knowledge in order to introduce diversity into the weak models, a desirable property to boost its efficacy [Dietterich, 2000b; Kuncheva and Whitaker, 2003; Banfield et al., 2005; Brown et al., 2005; Wang and Yao, 2009]. The bulk of our idea consists in generating different training samples, with each one representing an specific and expected distribution change. When a new test set is to be quantified, the ensemble is likely to contain some models trained with a similar class distribution.

In order to prove the validity of our idea, we have applied it to *binary quantification* problems. In the case of binary quantification the set of class values is restricted to two and the objective is to correctly estimate the number of positive examples (prevalence). Quantification tasks fit perfectly our requirements, since, by definition, the class probabilities may change, and we are also able to characterize and to restrict ourselves to a certain types of changes. To the best of our knowledge this is the first attempt to solve quantification by using ensembles. This approach is different to other propositions that have been suggested to

tackle tasks that present some sort of drift in the distribution [Kuncheva, 2004], especially *concept drift* problems [Widmer and Kubat, 1996]. Those methods are based on removing, modifying or adding new models to the ensemble, mainly because the concept $\mathbf{P}(y|\boldsymbol{x})$ changes throughout time and it is not possible to exploit any prior knowledge. Our approach is different in the sense that, as we know the characteristics of the expected changes, we can use that knowledge to build an enriched ensemble from the beginning without the need of subsequent modifications. A detailed presentation of our proposition is shown in Section 3.2.

1.2 Summary of contributions

The contributions of this dissertation are fourfold:

(i) We have presented the first ensemble method for quantification. Interestingly, our approach is not just a straightforward application of well-known ensemble algorithms, like bagging or boosting, but a specially devised method for binary quantification. Our proposal takes into account, and exploits, the main characteristic of quantification tasks: how the data distribution is expected to change.

(ii) The second contribution is to propose several selection criteria appropriately adapted for some state-of-the-art quantification algorithms. Additionally, some of them are static, but two of them are defined for dynamic ensemble model selection. The idea in the latter case is to select the best ensemble models for each testing sample.

(iii) Besides, these ensembles of quantifiers have been exhaustively tested using benchmark datasets, analyzing their behavior from different perspectives. This includes the proposed selection measures, for instance, studying which selection criterion is more appropriate for each base quantifier. In this sense, the experiments reported in this dissertation establish a solid point of departure for future research in the field of ensembles for quantification learning.

(iv) Finally, ensembles of quantifiers have been applied to tackle a real-world application, namely sentiment quantification. The method, originally devised for binary quantification, was extended to the multi-class case using the one-vs-all approach. This algorithm has been tested over four datasets coming from *SemEval* competitions. Our experiments conclude that ensembles provide a competitive performance and some times outperform single quantifiers.

1.3 List of publications

The results of this research work have been published in four articles, three of them in journal papers (1 accepted and 2 currently under review) and one was presented at the Spanish Conference for Artificial Intelligence.

(i) Journal papers:

[1] Pablo Pérez-Gállego, José Ramón Quevedo, Juan José del Coz: Using ensembles for problems with characterizable changes in data distribution: A case study on quantification. Information Fusion, vol 34, pp. 87–100. 2017. ISSN: 1566-2535
DOI: 10.1016/j.inffus.2016.07.001
Category: Computer Science, Artificial Intelligence
Impact factor: 4.353 (2015), 9/130 Q1

[2] Pablo Pérez-Gállego, Alberto Castaño, José Ramón Quevedo, Juan José del Coz: New Static and Dynamic Quantifier Selection Measures, NeuroComputing, Under Review. ISSN: 0925-2312
Category: Computer Science, Artificial Intelligence
Impact factor: 2.392 (2015), 31/130 Q1

[3] Pablo Pérez-Gállego, Alberto Castaño, José Ramón Quevedo, Juan José del Coz: Ensembles of Quantifiers for Sentiment Quantification, Decision Support Systems, Under Review. ISSN: 0167-923
Category: Computer Science, Artificial Intelligence
Impact factor: 2.604 (2015), 25/130 Q1

(ii) Conference papers:

[4] Pablo Pérez-Gállego, José Ramón Quevedo, Juan José del Coz: Uso de ensembles en problemas con cambios de distribución caracterizables. In proceedings of the 16th Conference of the Spanish Association for Artificial Intelligence, CAEPIA 2015.

1.4 Document outline

The rest of this document is organized as follows:

- Chapter 2 provides an in-depth explanation to the quantification problem. This includes a formal definition of the learning task, the notation used along this document, the description of appropriate evaluation methods, a

list of potential applications and a comparison between classification and quantification from a theoretical point of view. Following, we analyze one of the main properties of quantification tasks, which is that data distribution changes. Finally, the state-of-the-art quantification algorithms used in the experiments are described.

- Chapter 3 first gives a general overview of the principal ensemble learning characteristics. Then, our main proposal to design an ensemble method for binary quantification is deeply discussed. The chapter ends with an experimental study in which the proposed ensemble method, using three different base quantification algorithms, is compared to single quantifiers. The conclusion is that ensembles of quantifiers significantly outperforms individual quantifiers in several benchmark datasets. (Based on Pérez-Gállego et al. [2015, 2017c])

- Chapter 4 proposes three new ensemble selection criteria specially devised for ensembles of quantifiers: one of them is static in essence and the other two allow dynamic selection. The resulting approaches are experimentally compared with the method proposed in Chapter 3, and we conclude that selection strategies help to improve the performance of ensembles of quantifiers.

- Chapter 5 is aimed at experimentally checking the effectiveness of our proposals in a real-world application. Concretely, ensembles of quantifiers combined with two selection measures are applied to four multi-class sentiment quantification tasks taken from *SemEval* competitions. The goal of such tasks is to quantify the proportion of positive, negative and neutral users' comments sampled from Twitter. Ensemble approaches again provides at least competitive performance and in several cases outperform single quantifiers. (Based on Pérez-Gállego et al. [2017b])

- Chapter 6 summarizes the main findings and contributions of this thesis and outlines several directions for future research.

Chapter 2

Quantification: notation, definitions and methods

Quantification learning is one of the core concepts in this dissertation. All of the novel methods we propose in Chapters 3 and 4, as well as the application in Chapter 5, are ensemble adaptations for the quantification problem. Given that quantification learning is a fairly new task within the Machine Learning community, we do think it is necessary to dedicate a full chapter of this dissertation to provide an in-depth explanation to the problem.

This chapter exhibits a formal view of quantification, defined in terms of a traditional machine learning task. We start presenting a widely accepted definition for the problem and its formal notation. A brief comparison between the classical classification problem and quantification is included. Then, we describe the necessities for a special evaluation framework in quantification problems. Following, we analyze one of the main properties of quantification tasks: distributional changes between training and test sets. Finally, we go into details about the theory behind some base quantifier algorithms that are employed in the experimental setting in Chapters 3, 4 and 5.

2.1 Quantification learning

Quantification learning is a supervised learning based task that has emerged in the past years introduced by Forman in [Forman, 2005]. As stated by Forman, the task of quantification is *to accurately estimate the number of positive cases (or class distribution) in a test set, using a training set that may have a substantially different distribution* [Forman, 2008]. More practically, it is aimed at, given a test set, producing an estimation of the number of examples belonging to each class, without requiring individual predictions for each instance.

Intuitively, it seems a less complex problem than classification, since it is not necessary to give accurate estimations for each example. We could think of a first trivial approach to quantification by inducing a classifier, predicting the class

of each example, and counting the number of predicted examples for each class, obtaining a class histogram. As we shall see later in this chapter, this baseline approach of classifying and counting has already been proved to perform poorly [Forman, 2008].

Next, we are formally introducing the notation for the quantification problem under the framework of a supervised learning task. In order to formally state the differences between the classification and quantification problems, we are also including the notation for classification tasks.

2.1.1 Notation

Let $\mathcal{X}$ be an input space and $\mathcal{Y}$ an output space. A supervised learning task requires as input a training dataset $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ randomly drawn from an unknown probability distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ from the product $\mathcal{X} \times \mathcal{Y}$. Each instance, $\boldsymbol{x}_i \in \mathcal{X}$, is represented by an attribute vector $(x_{i,1}, x_{i,2}..., x_{i,d})$ and a target class or label, $y_i \in \mathcal{Y} = \mathscr{L} = \{\ell_1, \ldots, \ell_l\}$.

The objective in a multi-class classification task is to approximate an unknown function $f : \mathcal{X} \to \mathcal{Y}$, by inducing a hypothesis or model, h, defined into some hypothesis space $\mathcal{H}$, from D:

$$h : \mathcal{X} \longrightarrow \{\ell_1, \ldots, \ell_l\}, \tag{2.1}$$

that is able to assign the correct class label, given a new individual test example. When the number of classes are just two, they are usually denoted as $\{+1, -1\}$, and the task is named binary classification.

As regards the quantification learning problem, the main difference with respect to the previous classification definition, is that predictions are not made for each single instance, but for a complete sample. Precisely, a multi-class quantifier returns the class probability distribution for a given sample by inducing a hypothesis $\bar{h}$:

$$\bar{h} : \mathbb{N}^{\mathcal{X}} \longrightarrow [0, 1]^l, \tag{2.2}$$

where $\mathbb{N}^{\mathcal{X}}$ is used to represent the number of times that each possible instance, $\boldsymbol{x}$, from $\mathcal{X}$ appears in the sample, implying that a sample can contain duplicate examples according to the representation defined by $\mathcal{X}$. $\mathbb{N}^{\mathcal{X}}$ denotes the set of all functions from $\mathcal{X}$ to $\mathbb{N}$. The quantifier, $\bar{h}$, returns a predicted vector, $[\hat{p}_1, \ldots, \hat{p}_l]$, in which each value $\hat{p}_j$ represents the probability of class ℓ_j in the given sample. Thus, it holds that $\sum_{j=1}^{l} \hat{p}_j = 1$. The goal is to predict a class probability distribution as close as possible to the true one, $[p_1, \ldots, p_l]$.

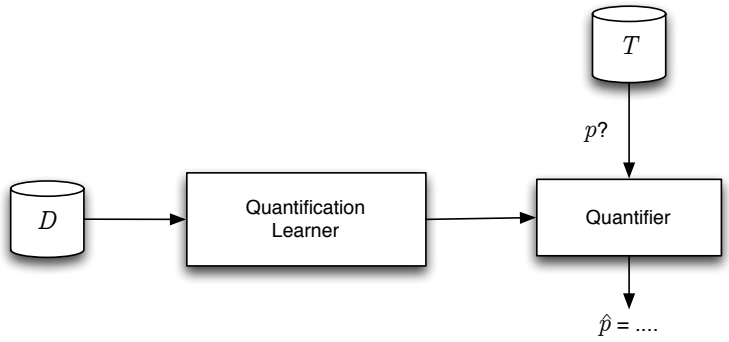

Figure 2.1: Binary quantification learning

Binary quantification

The binary quantification problem is a special case in which each instance is labeled as positive or negative, $y_i \in \{+1, -1\}$. In this situation, the class distribution can be summarized with the actual proportion of positives or *prevalence* p (the proportion of negatives is $n = 1 - p$). The binary quantification goal is to induce a model or quantifier:

$$\bar{h} : \mathbb{N}^{\mathcal{X}} \longrightarrow [0, 1], \tag{2.3}$$

able to estimate the prevalence, $\hat{p}$, of the positive class for an unseen test set, T, i.e. $\hat{p} = \bar{h}(T) = \mathbf{P}_T(y = +1)$, that may have a significantly different class distribution (see Figure 2.1). From a learning point of view, the most important assumption made by quantification methods is that the class probability distribution $\mathbf{P}(y)$ changes between training and test, but $\mathbf{P}(\boldsymbol{x}|y)$ remains constant. This issue is deeply discussed in Section 2.3.

2.1.2 Multi-class quantifiers based on decomposition

Taking into account that most of the existing quantification algorithms are designed for binary problems, the most straightforward approach to tackle multi-class quantification is to apply some kind of decomposition strategy, for instance, the well-known one-versus-all approach [Forman, 2008], using any binary quantification algorithm. Such approach comprises the following steps:

(i) Given a training set $D = \{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$ with l different class values, l binary quantifiers, $\{h_1, \ldots, h_l\}$ of the form $h_j : \mathbb{N}^{\mathcal{X}} \longrightarrow [0, 1]$ are learned,

one for each class ℓ_j. The dataset for training h_j is composed by the same examples in D, in which those from class ℓ_j are labeled as positives, and the rest are labeled as negatives. Recall that any binary quantification algorithm, like the ones described in Section 2.4, could be used as base learner.

(ii) In the testing phase, each binary quantifier, h_j, is applied to obtain an initial estimation of the prevalence for class ℓ_j, $\hat{p}_j^0$.

(iii) These initial prevalences are finally normalized in order to ensure that their sum is 1:

$$\hat{p}_j = \frac{\hat{p}_j^0}{\sum_{k=1}^{l} \hat{p}_k^0}. \tag{2.4}$$

This shall be the approach applied in the experiments in Chapter 5. A multi-class quantifier is required in order to give an estimation of the class distribution in a sentiment analysis problem comprised by three different class labels: positive, neutral and negative.

2.1.3 Differences between classification and quantification

Both classification and quantification tasks are defined under the same supervised learning framework, as we have seen in Section 2.1.1. However, the denotation of Equations 2.1 and 2.2 (classification and quantification, respectively), clearly demonstrate that they address different learning problems. It is worth noting that it is possible to solve quantification by just using classifiers, though. All that is necessary is to have a classifier giving a class estimate for each example, and then count.

One of the principal differences is the evaluation unit: while a classifier evaluates an individual example, a quantifier produces estimates for samples or bags (groups of examples). The direct consequences are twofold. First, traditional classification evaluation procedures are not longer valid because a representative group of samples is required for estimating the quantifier performance. Having just one test sample is not enough for giving an accurate estimate. Moreover, it would be recommendable to have enough prevalence diversity within the test sample group, knowing that class distributions, by definition, may change. Second, given that a quantifier produces class distribution estimates for the sample, it does not make sense to use the same performance measures as in classification.

A core disparity between both problems is that the learning assumptions are totally different. Supervised classification generally assumes that the unknown distribution from where the examples are drawn independently and identically distributed, does not change. On the contrary, class distribution in quantification

problems changes between training and test (see Section 2.3) by the own definition of the learning task. In symbols, $\mathbf{P}_D(y = +1) \neq \mathbf{P}_T(y = +1)$. Otherwise the quantification problem would be trivial: a quantifier could predict just the prevalence of the positives observed in the training set: $h(T) = \mathbf{P}_D(y = +1)$.

2.2 Evaluation of quantification methods

Evaluating the performance of a quantifier model is more complicated than for other supervised learning models, as we already highlighted in Section 2.1.3. In classification, an individual prediction for a test case can be determined right or wrong independently of others, and the overall goodness of a method can be estimated on a group of instances. In contrast, a quantifier outputs a single prediction for a whole batch of test cases. This makes quantifier evaluation a more complex task because we need a representative group of samples, otherwise the performance of a quantifier will not be accurately estimated. This batching requirement calls for a unique research methodology [Forman, 2008]. Two are the principal conditions this methodology shall accomplish:

(i) a large number of test sample quantifier performance measurements should be aggregated in order to avoid getting a biased estimation, and

(ii) all the prevalence domain should be represented within the collection of testing samples, in order to account for the class distribution changes that, by problem definition, may occur.

The methodology we have employed in this dissertation for assessing the performance of our ensemble based quantifiers in Chapters 3 and 4 is the following. As it was not available any test set from the dataset benchmark, we decided to create artificial drifted testing sets. Under a cross validation framework, for each test partition, we generated 100 test samples by random sampling with replacement, in which the positive class prevalence p was randomly/uniformly (respectively in Chapters 3 and 4) selected from 0% to 100%.

Given a collection of testing samples, $\{T_1, \ldots, T_s\}$, the performance of a quantifier, $\bar{h}$, is computed as:

$$Performance(\bar{h}, L, \{T_1, \ldots, T_s\}) = \frac{1}{s} \sum_{j=1}^{s} L(\bar{h}, T_j), \tag{2.5}$$

in which $L(\cdot, \cdot)$ represents a quantification loss function. In some cases, the experimental design may require adapted versions of well-known validation methods, like cross-validation [González et al., 2016a]. Usually L must compare the predicted class distribution and the actual one.

The next two sections are devoted to present a review of the quantification loss functions that we have used in Chapters 3, 4 and 5.

2.2.1 Binary loss functions

First, let us to formally introduce the concept of prevalence, p, under the framework of a binary problem. Table 2.1 shows a confusion matrix for a binary problem, which describes the performance of a classifier on a test data set T whose true values are known. P represents the actual number of positive examples on T, and N the count of negatives. The classifier is applied to the test set to estimate its classes and we obtain P', the number of positive predicted examples, and N', the number of negative predicted ones. TP, FN, TN and FP represent the count of *true positives*, *false negatives*, *true negatives* and *false positives*.

Table 2.1: Confusion matrix for a binary problem

	P	N
P'	TP	FP
N'	FN	TN
$(S = P + N = P' + N')$		

The true prevalence, p, of the test set can be obtained as:

$$p = \frac{P}{S} = \frac{TP + FN}{S}, \tag{2.6}$$

end the estimated prevalence, $\hat{p}$, as:

$$\hat{p} = \frac{P'}{S} = \frac{TP + FP}{S}. \tag{2.7}$$

In the case of binary quantification loss function, it is enough to compute the difference between the predicted prevalence of the positive class, $\hat{p}$, and the actual prevalence, p.

Estimation bias

The estimation bias of a quantifier is computed as the estimated prevalence minus the actual prevalence of the test set:

$$bias(\bar{h}, T) = \hat{p} - p = \frac{P' - P}{S} = \frac{FP - FN}{S}. \tag{2.8}$$

This metric measures whether the model tends to overestimate or underestimate the proportion of the positive class. According to Forman [Forman, 2008], this is not a useful function to measure the overall performance of a quantifier over a set of samples, since positive and negative bias may average out.

Mean Absolute Error (MAE)

The *Absolute Error* (AE) [Forman, 2005, 2006, 2008] between the actual and the predicted prevalence is defined as:

$$AE(\bar{h}, T) = |\hat{p} - p| = \frac{|P' - P|}{S} = \frac{|FP - FN|}{S}. \tag{2.9}$$

Averaging it over a group of test samples we have:

$$MAE(\bar{h}, \{T_1, \ldots, T_s\}) = \frac{1}{s} \sum_{j=1}^{s} |\hat{p}_j - p_j|. \tag{2.10}$$

MAE has been traditionally the preferred loss function in binary quantification because it is easy to interpret and simple to calculate.

Mean Squared Error (MSE)

As an alternative to AE, Bella et al. [2010] proposed to calculate the *Squared Error*, SE, which is defined as:

$$SE(\bar{h}, T) = (\hat{p} - p)^2 = \left(\frac{P' - P}{S}\right)^2 = \left(\frac{FP - FN}{S}\right)^2. \tag{2.11}$$

Averaging it over a group of test samples we have:

$$MSE(\bar{h}, \{T_1, \ldots, T_s\}) = \frac{1}{s} \sum_{j=1}^{s} (\hat{p}_j - p_j)^2. \tag{2.12}$$

While MSE is not as easy to interpret as MAE, it is more sensitive to large errors.

Kullback-Leibler Divergence (KLD)

Kullback-Leibler Divergence (KLD), also known as normalized cross-entropy [Forman, 2008; Esuli and Sebastiani, 2010] can be applied in the context of quantification. It can be defined for binary quantification as:

$$KLD(\bar{h}, T) = p \cdot \log\left(\frac{p}{\hat{p}}\right) + (1-p) \cdot \log\left(\frac{1-p}{1-\hat{p}}\right). \tag{2.13}$$

We shall extend the description of *KLD* in Section 2.2.2, as it is more frequently used in multi-class environments.

2.2.2 Multi-class loss functions

Most of the binary quantification loss functions can be easily extended to multi-class quantification.

The most widely used [Barranquero et al., 2013, 2015; Tang et al., 2010; Alaiz-Rodríguez et al., 2008] loss function for multi-class quantification problems is probably the Kullback-Leibler Divergence (*KLD*), previously described for the binary case. *KLD* is a special case of a two classes of divergences, f-divergences and Bregman divergences, and measures the non-symmetric difference between two probability distributions, in our case $[p_1, \ldots, p_l]$ and $[\hat{p}_1, \ldots, \hat{p}_l]$:

$$KLD(\bar{h}, T) = \sum_{j=1}^{l} p_j \cdot \log\left(\frac{p_j}{\hat{p}_j}\right). \tag{2.14}$$

KLD ranges from 0 (the optimum) to $+\infty$ and has some advantages, for instance that it is more suitable than other measures for averaging over different test prevalences, but also some drawbacks, mainly that it is not a true metric because it does not obey the triangle inequality and is not symmetric in general. It is worth noting that *KLD* is not defined when the predicted prevalence, $\hat{p}_j$, of any class is 0 or 1, which can be especially common in cases where the number of classes is large. In these cases, it is still possible to use the *KLD* loss function by normalizing it via the logistic function, as proposed by Gao and Sebastiani [2016].

The other multi-class performance measure that we have considered in this dissertation is the *Mean Absolute Error* (*MAE*). Its extension to multi-class problems is defined as:

$$MAE(\bar{h}, T) = \frac{1}{l} \sum_{j=1}^{l} |\hat{p}_j - p_j|. \tag{2.15}$$

A deeper discussion on quantification measures functions can be found in [González et al., 2017].

2.3 Characterizing data distribution changes

Our proposal in Chapter 3 assumes that there exist a series of problems or applications in which data distribution changes between the training and test phases. Moreover, for some of them, as it is the case of quantification, it is possible to determine in advance which elements change, and which ones remain constant. Thus, once the potential distribution changes have been defined, it is possible to generate diverse training samples simulating them. This dissertation section is aimed at giving a formal and detailed description about these data distribution changing environments.

Supervised learning makes the assumption that the unknown distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$, from where the examples are drawn independently and identically distributed (i.i.d. assumption), does not change between the training and testing or production phases. Or to state it differently, it is assumed that the training set truly reflects the probability distribution of the problem. However, in practice, this assumption gets often violated in real-world applications [Quiñonero Candela et al., 2009; Storkey, 2009]. This situation is referred to as *dataset shift* in the research community, and it takes place when $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ changes from training to testing data [Kull and Flach, 2014; Moreno-Torres et al., 2012]. Characterizing those changes in the distribution sometimes depend on the target application, and even though it can be challenging in some cases, it is surprisingly simple, even trivial, in other problems.

A categorization and a discussion about problems presenting distribution changes, or using the current terminology, problems suffering from *dataset shift*, can be found for instance in [Moreno-Torres et al., 2012; Kull and Flach, 2014]. Supervised learning problems are defined by a set of covariates, $\boldsymbol{x}$, a class variable, y, and the examples are drawn at random from the joint probability distribution of both. To better understand dataset shift it is important to realize how the data is generated according to the causal relationship between covariates and the class variable, since it determines the kind of changes in the distribution that a problem may suffer from. In this sense, a taxonomy proposed in [Fawcett and Flach, 2005] identifies two types of problems:

(i) $\mathcal{X} \rightarrow \mathcal{Y}$ in which the class value y is causally determined by the covariate values $\boldsymbol{x}$, and

(ii) $\mathcal{Y} \rightarrow \mathcal{X}$ where the covariates $\boldsymbol{x}$ causally depend on the class label y.

Spam detection constitutes an example of the first type of problems: the mail content determines whether it is spam or not. On the other hand, medical diagnosis problems are a typical example of the second; suffering from a determined disease y cause a series of symptoms $\boldsymbol{x}$ to appear.

Given an instance $\boldsymbol{x}$ and a class value y, their joint probability $\mathbf{P}(\boldsymbol{x}, y)$ can be written as $\mathbf{P}(y|\boldsymbol{x})\mathbf{P}(\boldsymbol{x})$ in $\mathcal{X} \rightarrow \mathcal{Y}$ problems and $\mathbf{P}(\boldsymbol{x}|y)\mathbf{P}(y)$ in the case of $\mathcal{Y} \rightarrow \mathcal{X}$ problems. Dataset shift arises when any of these elements change between training and test, that is to say $\mathbf{P}_{tr}(\boldsymbol{x}, y) \neq \mathbf{P}_{tst}(\boldsymbol{x}, y)$. Thus, several types of dataset shift problems can be identified depending on the elements that change:

(i) *covariate shift*: $\mathbf{P}(\boldsymbol{x})$ changes but $\mathbf{P}(y|\boldsymbol{x})$ remains constant,

(ii) *prior probability shift*: $\mathbf{P}(y)$ changes but $\mathbf{P}(\boldsymbol{x}|y)$ does not, and

(iii) *concept shift* (o *drift*): $\mathbf{P}(y|\boldsymbol{x})$ changes but $\mathbf{P}(\boldsymbol{x})$ does not ($\mathcal{X} \rightarrow \mathcal{Y}$ problems), or $\mathbf{P}(\boldsymbol{x}|y)$ changes but $\mathbf{P}(y)$ remains constant ($\mathcal{Y} \rightarrow \mathcal{X}$ problems).

Quantification problems constitute an obvious case of *prior probability shift*. In fact, both terms refer to the same concept. In quantification class probabilities, $\mathbf{P}(y)$, change by own definition of the problem. Otherwise, it would be a straightforward task in which the optimal estimate would always be to predict the original training test prevalence p. We do know that is not the case.

Let's illustrate the *prior probability shift* situation with an example. A typical application of quantification learning is to estimate the prevalence of positive and negative opinions. Imagine that we want to track the opinions about a product in Twitter during a period of time and give just an estimate on how many are positives (and negatives), without predicting individual opinions (this would be a classification task). In such a problem, when the class distribution $\mathbf{P}(y)$ changes (e.g. the number of positive opinions increases), the opinions maintain the same distribution when the class is fixed. The way in which users express their opinions does not change from one day to another, there would be very good opinions using strong words expressing that felling, moderately positive comments and so on. When can assume in that case that $\mathbf{P}(\boldsymbol{x}|y)$ is constant.

2.3.1 Ensembles to tackle dataset shift

In practice, there are many important applications suffering from distribution changes to a greater or lesser extent. These kind of problems are interesting from an ensemble learning point of view, because some of the aforementioned changes can be easily characterized. In fact, our idea described in Chapter 3 makes the most of these expected changes by replicating them into the weak ensemble models.

Although not directly comparable to our proposition, ensembles have been applied before for problems that present a shift in the distribution [Kuncheva, 2004], mainly in concept drift tasks [Widmer and Kubat, 1996; Žliobaitė, 2010]. The main idea is to build an ensemble with models created at different moments in time and the goal is to have at least one model representing each distinct concept. The ensemble maintains a *memory* of models representing past concepts because some

of them may become useful again in the future [Hosseini et al., 2013]. Different strategies for training such ensembles can be employed, for instance, to divide historical sequential data into non overlapping blocks [Wang et al., 2003; Tsymbal et al., 2008; Karnick et al., 2008] or using different sized training windows [Stanley, 2003; Kolter and Maloof, 2007; Scholz and Klinkenberg, 2007]. The former group of techniques are useful for sudden and incremental drifts, and the later implicitly assume that once off sudden drift has happened. After the set of models is trained, adaptivity is achieved by defining a combination or fusion rule. Basically, the combination rule consists in assigning weights to the individual models at each point in time. The weights express the expected competence of the model and may depend on different factors, typically the estimated performance [Tsymbal et al., 2008; Scholz and Klinkenberg, 2007] or historical performance in the past [Karnick et al., 2008; Stanley, 2003; Kolter and Maloof, 2007].

However, our approach differs in several aspects with respect to these concept drift methods. The first difference is due to the fact that the concept does not change in quantification applications. For instance, the concept of what a positive opinion is does not change for a given sentiment analysis problem. The ensembles in concept drift tasks are usually designed to maintain a memory of models, representing the evolution of the concept and allowing to reuse models that were obtained in the past and are valid again. This approach is not aplicable for quantification tasks because because the concept is always the same, there are no *old* and *new* models in quantification. This is particularly true for $\mathcal{Y} \rightarrow \mathcal{X}$ problems for obvious reasons. The second difference is that models for concept drift based on ensembles are trained with successive samples. In our case, the samples are not given but are generated according to the expected changes in data distribution.

2.4 Binary quantification methods

This section of the dissertation is devoted to provide an in-depth review of the quantification base methods that have been used in the experimental sections of Chapters 3, 4 and 5. We start by analyzing the most obvious way to quantification: *Classify and Count* (CC). Then we review *Adjusted Count* (AC), the approach proposed by Forman to reduce the systematic bias of CC. Following, we illustrate the probabilistic versions of CC and AC, PCC and PAC, respectively. Finally, we describe a conceptually different method, HDy, and briefly comment a series of alternative quantification approaches that have not been employed in this dissertation. Notice that multi-class quantifiers implemented in Section 5, are based on these binary quantifiers, which have been extended to the multi-class framework by applying the one-vs-all strategy (See Section 2.1.2).

2.4.1 The intuitive way: Clasify and Count (CC)

From a practical point of view, the most intuitive way to quantification consists in classifying each instance on a test set, and then counting the number of examples belonging to each class. This method is referred to as *Classify and Count* (CC) in the literature, and has already been proved to perform poorly [Forman, 2008]. Many practitioners still apply this method when facing quantification tasks, ignoring more sophisticated quantification algorithms. More formally, the procedure to classify and count is the following:

(i) a classifier is induced from the training set,

(ii) such classifier is used to classify a testing sample, T, and

(iii) the number of examples predicted as positives is counted in order to compute $\hat{p}$.

It is worth nothing that, by following this approach, a perfect classifier (the one not making generalization errors) would lead to a perfect quantifier. However, producing a perfect classifier is almost impossible or costly, especially in real-world applications. Real-world concepts are usually complicated to induce, and special purpose classifiers are costly to develop. Most of times an imperfect classifier is all that is available. In that scenario, the underlying quantifier inherits the imperfect classifier bias, which shall get magnified if the test class distribution changes from the training set one.

The principal problem of such approach is that the induced classifier does not take into account that the distribution may change, which, by problem definition, will happen. In fact, CC performance tends to notably drop when p significantly changes, by going down or up a high degree. The main reason is because the underlying classifier will have a systematic bias, tending to produce false positives or false negatives. As Forman demonstrated by his theorem (see [Forman, 2008], page 169), the derived quantifier will underestimate the prevalence, $\hat{p} \ll p$, when p increases and the classifier tends to produce false negatives. And the opposite, the quantifier will also overestimate p, $\hat{p} \gg p$, when p decreases and the classifier produces more false positives. Thus, it is impossible for CC to perform well in the whole prevalence domain.

Figure 2.2 depicts this behavior for two different datasets. The perfect quantifier is shown with the thick line and we can observe that CC (dashed line) only makes perfect predictions for a certain value of p. CC underestimates the prevalence when it increases from that point; and the other way around, CC tends to overestimate the prevalence for lower values of p with respect to the optimal point. The CC bias depends on the classifier accuracy. This behavior was studied by Forman [2008], both theoretically (the aforementioned theorem), and analytically (see Figure 1,

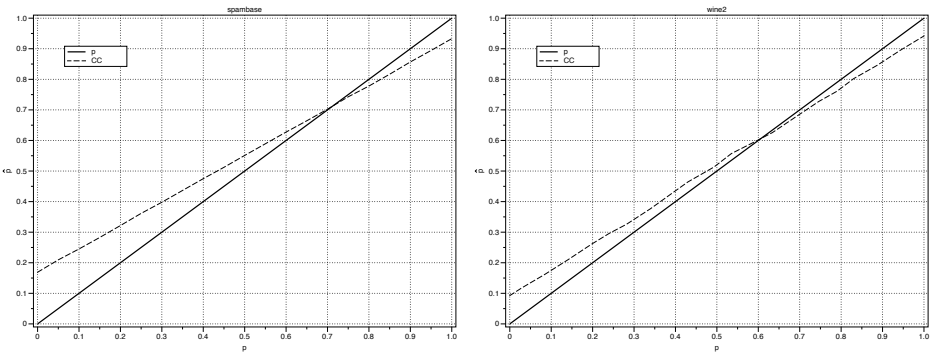

Figure 2.2: Bias of the CC approach when the prevalence varies. Testing sets with different prevalences were generated by random sampling with replacement from the original training set (*spambase* dataset on the left figure, and *wine2* on the right one; See Table 3.1 for more details on both datasets). Each result is the average prediction of 100 sample sets with the same prevalence

page 166). In Section 3.3.3 we shall extend this experiment showing that proper quantification methods, including the ensemble methods proposed in this work, alleviate this issue.

2.4.2 Adjusted Count (AC)

Probably, the most popular quantification algorithm is the AC (*Adjusted Count*) method. It was proposed by Forman [Forman, 2008] in order to reduce the systematic bias that CC inherits from the underlying classifier. Its popularity is due to the fact that AC is designed from an elegant and well-founded mathematical derivation. Actually, the AC method is built on top of CC, in the sense that it also uses a classifier and counts the examples belonging to each class. The AC method starts estimating the class probability distribution using the CC approach. Conceptually, the key aspect is that AC makes the learning assumption that within-class probability densities, $\mathbf{P}(\boldsymbol{x}|y)$, do not change, i.e. $\mathbf{P}_T(\boldsymbol{x}|y) = \mathbf{P}_D(\boldsymbol{x}|y)$. When such assumption holds, the prevalence of the positive class predicted in the first step by the CC method depends on three elements: the actual prevalence p and two characteristics of the underlying classifier, the *true positive rate* (tpr) and the *false positive rate* (fpr):

$$tpr = \frac{TP}{P} \ , \ fpr = \frac{FP}{N} \tag{2.16}$$

in which P and N represent the count of actual positives and negatives, while TP and FP represent the count of true positives and false positives predicted by the

model.

These two rates can be used to correct the prevalence estimation given by a classifier. In fact, the probability of a classifier making a positive prediction, $\hat{p}_{CC} = \mathbf{P}(h(\boldsymbol{x}) = +1)$, in a binary problem can be expressed as a function of the ground truth prevalence p:

$$\begin{aligned}\hat{p}_{CC} &= \mathbf{P}(h(\boldsymbol{x})=+1|y=+1)\cdot\mathbf{P}(y=+1) + \mathbf{P}(h(\boldsymbol{x})=+1|y=-1)\cdot\mathbf{P}(y=-1)\\ &= tpr\cdot\ p + fpr\cdot\ n\\ &= tpr\cdot\ p + fpr\cdot\ (1-p).\end{aligned}$$

The first term ($tpr \cdot p$) represents that CC only predicts as positives the tpr fraction of the actual positives (p), but also classifies as positives the negative examples in the second term ($fpr \cdot (1-p)$), that is, the fpr fraction of the negatives ($1-p$). Noting that both rates (tpr and fpr) do not change when $\mathbf{P}(\boldsymbol{x}|y)$ remains constant: the classifier will generate the same proportion of true positives and false positives, independently of the fact that the probabilities of both classes vary.

The previous expression may be very useful if we are able to accurately estimate tpr and fpr. Solving for p, the ground truth prevalence can be written depending on $\hat{p}_{CC}$, tpr and fpr:

$$p = \frac{\hat{p}_{CC} - fpr}{tpr - fpr}. \tag{2.17}$$

Therefore, in order to obtain a better estimation of the true prevalence p it will suffice to:

(i) train a classifier using D,

(ii) estimate the tpr and the fpr of this classifier by using, for instance, a validation set or cross validation over the training set,

(iii) classify and count the test examples towards obtaining $\hat{p}_{CC}$, and

(iv) obtain the true prevalence p by substituting the calculated values in (2.17).

The crucial step, which is the estimation of tpr and fpr, is usually performed using cross-validation over the training set [Forman, 2008]. It is important to emphasize that such estimations are independent of the distribution changes, since it was assumed that $\mathbf{P}(\boldsymbol{x}|y)$ remained constant. Notice that theoretically AC should output perfect quantification estimates as long as this assumption is fullfiled. However, in practice, inaccurate tpr and fpr estimations and/or the assumption not getting satisfied often lead to imperfect quantifications.

The AC correction is, in general, applicable to any quantifier built on top of a classifier, like in the case of quantifiers based on decision trees [Milli et al.,

2013], on nearest neighbors [Barranquero et al., 2013], on structured classifiers that optimize quantification measures [Barranquero et al., 2015] or even on probabilistic classifiers, with the necessity of adapting the correction to the probabilistic domain [Bella et al., 2010].

2.4.3 Probabilistic CC and AC (PCC and PAC)

Probabilistic Classify & Count (PCC) and Probabilistic Adjusted Count (PAC), introduced in [Bella et al., 2010], are the probabilistic versions of CC and AC, respectively. The key difference between both families of methods is that CC and AC are usually implemented using a crisp classifier, while PCC and PAC requires a probabilistic classifier. In binary quantification, this classifier generally returns the probability of a given example being positive. Under such setting, PCC obtains the prevalence of the positive class averaging the probabilities returned for all the examples in the testing sample:

$$\hat{p}_{PCC} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{P}(y_i = +1|\boldsymbol{x}_i). \tag{2.18}$$

Acknowledging that PCC may suffer from the same issues than CC, especially when there are significant changes in the class probability distribution, the authors propose a probabilistic version of AC, denoted as PAC here, that corrects the estimate computed by PCC using a probabilistic variant of (2.17):

$$\hat{p}_{PAC} = \frac{\hat{p}_{PCC} - FP^{pa}}{TP^{pa} - FP^{pa}} \,. \tag{2.19}$$

The ratios TP^{pa} and FP^{pa} are estimated applying the following equations:

$$TP^{pa} = \frac{\sum_{i \in D^+} \mathbf{P}(y_i = +1|\boldsymbol{x}_i)}{|D^+|}, \tag{2.20}$$

$$FP^{pa} = \frac{\sum_{i \in D^-} \mathbf{P}(y_i = +1|\boldsymbol{x}_i)}{|D^-|}, \tag{2.21}$$

in which D^+ and D^- denote the set of positive and negative examples in D, respectively. TP^{pa} and FP^{pa} are therefore the averaged probability of the positive examples and the averaged probability of the negatives. Both rates may be estimated using the same procedures described above for the tpr and the fpr in the AC method.

2.4.4 HDy

The following method, termed HDy [González-Castro et al., 2013], is totally different to the previously commented approaches, since it does not rely on

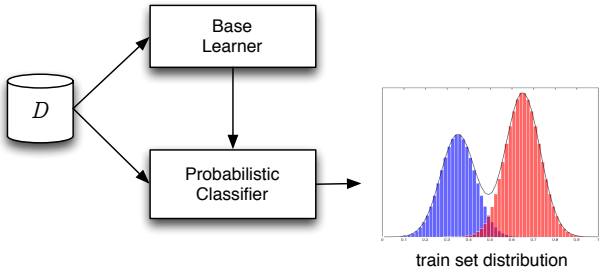

Figure 2.3: HDy Training phase: a probabilistic classifier is learned and applied to obtain the distribution of D

classifying, counting and correcting. Its core idea consists in measuring distribution similarities. More precisely, HDy compares training and test distributions, and seeks for the prevalence $\hat{p}$ that would make the training distribution, modified by $\hat{p}$ and obeying that $\mathbf{P}(\boldsymbol{x}|y)$ does not change, the most similar to the test distribution.

In [González-Castro et al., 2013], the authors propose two different methods to represent both distributions: HDx uses directly the feature vectors $\boldsymbol{x}$, while HDy employs the predictions of a probabilistic model induced from the training set. The later not only outperforms the former, but it also has the advantage of working in a space with a smaller dimensionality. In fact, the main advantage of HDx is that it does not require a classification model, but its main limitation is that its computational complexity increases with the number of features. Thus, the use of HDx is unfeasible in high dimensional spaces.

In the case of HDy, its representational space has just $L-1$ dimensions, being L the number of classes. For a binary problem it has only one dimension: the model predicts the probability of an example to belong to the positive class. The process is quite simple: the range [0..1] is partitioned into b bins, and a histogram is built with each classified example being assigned to one of the bins depending on its probabilistic score. In the training stage, the probabilistic classifier and the actual distribution of the training set are obtained (see Figure 2.3).

In the testing phase, the distribution of the test set is obtained following the same procedure:

(i) computing the probabilistic scores of the testing examples using the probabilistic classifier, and

(ii) calculating the corresponding histogram using the same number of bins b.

Then, the idea is to modify the training histogram to match the testing histogram varying $\hat{p}$. For instance, in the example depicted in Figure 2.4 in which red color

represents the distribution of the positives, $\hat{p}$ should decrease (and of course $\hat{n} = 1 - \hat{p}$ should increase in the same quantity) with respect to the prevalence of the training set in order to match the testing distribution.

The strategy followed for modifying the original training set histogram is a simple linear search. $\hat{p}$ is moved over the range [0..1] in small steps and the histogram associated is calculated applying the following equation:

$$\frac{|D'_i|}{|D'|} = \frac{\left|D_i^+\right|}{|D^+|} \cdot \hat{p} + \frac{\left|D_i^-\right|}{|D^-|} \cdot (1 - \hat{p}), \tag{2.22}$$

in which $|D^+|$ is the number of examples in D belonging to the positive class and $\left|D_i^+\right|$ the number of positive examples in D belonging to the i-th bin. $|D^-|$ and $\left|D_i^-\right|$ are analogously referred to the negative class. These components are pre-calculated in the training phase (Figure 2.3).

Similarity between both histograms is measured with the *Hellinger Distance* metric:

$$HD(D', T) = \sqrt{\sum_{i=1}^{b} \left(\sqrt{\frac{|D'_i|}{|D'|}} - \sqrt{\frac{|T_i|}{|T|}} \right)^2}, \tag{2.23}$$

where D' and T represent the modified distribution of the training set computed using (2.22) and the testing distribution, respectively. $|D'_i|$ is the number of examples of the modified distribution D' belonging to the i-th bin and $|D'|$ is the training set cardinality. $|T_i|$ and $|T|$ refer in the same way to the test set distribution.

Figure 2.4 schematically shows the testing phase of HDy. For the different values of $\hat{p} \in [0..1]$ considered, the Hellinger distance to the test set is calculated using (2.23). The $\hat{p}$ value minimizing the distance to the test set will be the estimated prevalence. Note that, as all the bins of the train set distribution are altered uniformly using $\hat{p}$, the assumption of $\mathbf{P}(\boldsymbol{x}|y)$ being constant is not violated.

2.4.5 Alternative approaches

As stated by Forman in Forman [2006], one of the main drawbacks of the AC method is that it usually fails to deliver a reasonable performance when the training class distribution presents a high imbalance. Under this circumstance, classifiers tend to assign the majority class to the most of the instances, i.e. when the positive class is scarce, the estimated *tpr* would be low, resulting in a small denominator ($tpr - fpr$) in the correcting Equation 2.17. The smaller the denominator, the bigger the correction, which could be risky. Forman proposes to train a linear SVM classifier and to calibrate its threshold in order to increase the number of true positives, even at the expense of getting many more false positives. The objective is to reduce the variance on the *tpr* and *fpr* estimation, and to avoid

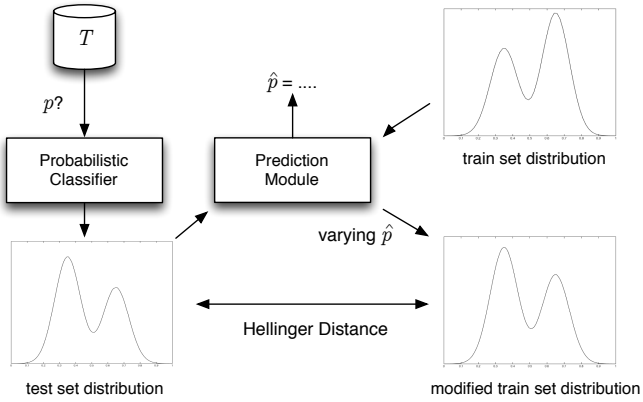

Figure 2.4: HDy Testing phase: the test distribution is computed using again the probabilistic classifier. The most important step is to compute a modified distribution of D, varying $\hat{p}$ in (2.22). Similarity between both distributions is calculated using the Hellinger Distance (2.23)

small $(tpr - fpr)$ denominators. A number of proposals are presented depending on the concrete threshold selection policy: X, Max and $T50$. The *Median Sweep* method obtains an estimation of Equation 2.17 at every threshold, and returns the median of these estimates [Forman, 2006, 2008].

Contrasting with most of the methods we have reviewed so far, that are based on classifying, counting and correcting, a number of alternative methods rely on adapting the base algorithms to the quantification problem. The training phase of these proposals is specifically devised for quantification estimation. A multi-class quantification method called *Quantification trees*, based on decision trees, is introduced in Milli et al. [2013]. A quantification measure is taking into account in order to select the best possible split. Barranquero et al. [2013] study the effectiveness of applying weighted nearest neighbor algorithms to the binary quantification problem. They support the idea of NN algorithms allowing to implement more efficient methods for estimating tpr and fpr in the training step. A different idea, introduced by Esuli and Sebastiani [2010], consists in optimizing a pure quantification measure in the training phase without taking into account classification performance. A new measure, termed *Q-measure*, which balances quantification and classification performance simultaneously is proposed by Barranquero et al. [2015]. Including the aforementioned HDy algorithm, another family of quantification proposals are based on matching the training an testing distribution. They are usually comprised by a mechanism to represent distributions, and a criterion to compare two given distributions. Representative approaches include the $Mixture\ Model$ [Forman, 2005], a method based on the

EM algorithm [Saerens et al., 2002], and methods based on the Hellinger distance (HDx and HDy) [González-Castro et al., 2013].

There exist a series of variations to the original formulated quantification problem. Forman proposed several solutions for the cost quantification problem [Forman, 2006, 2008], quantification for regression methods have been mainly established by Bella et al. [2013], and ordinal quantification has been tackled in Da San Martino et al. [2016].

Chapter 3

Designing Ensembles of Quantifiers

From our point of view, the most important contribution of this dissertation is to introduce a new problem context in which ensemble algorithms can take the most out of its singular characteristics. These kind of problems, in which the data distribution changes over time, are often referred to as *dataset shift* in the research community. Taking into consideration the taxonomy describing those data distribution changes presented in Section 2.3, we do claim that ensemble learning can be an effective technique to deal with those emerging changes, as long as it is possible to identify them beforehand. The reason is that each model of the ensemble can be trained with a training set whose data distribution may be different from the distribution of the original dataset; each ensemble weak model will represent an specific and expected distribution change. From a conceptual point of view, this approach obeys one of the main ensemble learning principles, which is to build a collection of models with some kind of diversity [Banfield et al., 2005; Dietterich, 2000b; Kuncheva and Whitaker, 2003; Brown et al., 2005].

This chapter describes our proposal to design and build ensembles representing these expected changes in the distribution Pérez-Gállego et al. [2015, 2017c]. However, before we can keep moving on this subject, we need to specify which kind of *dataset shift* problems we are aiming to solve at. It is necessary to delimit which elements in $\mathbf{P}(\boldsymbol{x}, y)$ are expected to change, and which ones remain constant. We are focusing on *Prior probability shift* problems (also known as *quantification* problems), in which by definition, class probabilities $P(y)$ change. Formally, it holds that $\mathbf{P}_{tr}(\boldsymbol{x}, y) = \mathbf{P}_{tst}(\boldsymbol{x}, y)$, but $\mathbf{P}_{tr}(y) \neq \mathbf{P}_{tst}(y)$. Our requirements are decidedly satisfied by the *quantification* task since, by definition, data distribution may change, and we are able to characterize and restrict ourselves to a certain types of changes.

We also include in this chapter all the experimental evidence we have found to support our ideas. More concretely, we have compared three well known quantifiers (CC, AC and HDy), to its corresponding quantifier ensemble versions. For the sake of simplicity in the experiment design phase, we have decided to focus only on *binary quantification* problems (Section 2.1.1), which are a special case having

the number of classes limited to two. However, in Chapter 5 we describe how we have applied our idea to a multi-class real-world sentiment analysis quantification problem.

Before going into details about designing and building ensembles of quantifiers, we think that an introduction of the formal definition, notation, and a general overview of the principal ensemble characteristics, might be beneficial for the reader. We also expect it to be useful for better understanding why ensemble learning is a well-suited technique for those problems in which the data distribution changes, as it occurs in quantification.

3.1 A general view on ensembles

Ensemble learning consists in constructing a meta-model that results from the combination of a set of individual models, using a particular aggregation rule. Ensembles get benefited from the existent diversity in the model set, producing a solution that implicitly represents some sort of agreement between the individual models. From a practical point of view, they generally perform better than a single-model solution [Bauer and Kohavi, 1999; Breiman, 1996; Freund et al., 1996; Opitz and Maclin, 1999], although this cannot be guaranteed [Fumera and Roli, 2005]. An ensemble limits the risk of obtaining a particular bad response from a single model; formally this is due to the fact that the ensemble tends to reduce the variance of its base learning algorithm. Intuitively, the same idea is highly present in the human decision making processes; a set of opinions is more rich than an isolated opinion, especially when there exist a high degree of diversity within the opinions.

3.1.1 Notation

Let us introduce some notation for ensembles under the framework of supervised learning. Let $\mathcal{X}$ be an input space and $\mathcal{Y}$ an output space. There exist a training set $D = \{(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_n, y_n)\}$ drawn from an unknown distribution $\mathbf{P}(\mathbf{X}, \mathbf{Y})$ from the product $\mathcal{X} \times \mathcal{Y}$. Usually each example, $\boldsymbol{x}_i$, is represented by an attribute vector $(x_{i,1}, x_{i,2}..., x_{i,d})$ and a target class y_i that may belong to a discrete set in classification problems or to $\mathbb{R}$ in the case of a regression problem. The objective is to approximate an unknown function $f : \mathcal{X} \to \mathcal{Y}$ by generating a function h, called hypothesis, defined into some hypothesis space $\mathcal{H}$. To do so, many algorithms search for the best single hypothesis h that approximates f taking into account the training set D, the selected hypothesis space and a target loss function. In contrast, an ensemble produces a hypothesis h resulting from the combination of a set of m (weak) hypothesis $\{h_1, h_2, ..., h_m\}$, in which each model h_j is usually

learned using a subsample D_j generated from D. The combination of the set of hypothesis or models is performed by means of a particular aggregation strategy.

3.1.2 Aggregation and selection measures

Normally, an ensemble method has at least two main steps: 1) in the training phase, the ensemble models are obtained using some base learning algorithm; and 2) in the prediction phase, the outputs of these models are combined to produce a collective decision for each testing example. This fusion strategy mainly depends on the learning task, being majority voting the predominant rule for classification, and averaging for regression and probability estimation. In any case, ensemble models can be treated equally or not, assigning a different weight to each one; usually the weight is proportional to its accuracy or precision.

Ensemble model selection (also referred to as ensemble pruning by some authors [Tsoumakas et al., 2009]) is a broadly studied subject in the literature [Caruana et al., 2004; Tsoumakas et al., 2008; Britto et al., 2014]. The underlying idea consists in, instead of aggregating all the ensemble models, choose an adequate model subset, according to a defined selection or search strategy, yielding to a better ensemble accuracy. Some studies reveal that ensembling *many* models may be better that ensembling all of them [Zhou et al., 2002]. Ensemble selection strategies can be categorized into static or dynamic. Static based approaches select a fixed subset of models and discard the rest, which are not longer considered. On the other hand, dynamic methods select a different model subset for each testing example.

3.1.3 Diversity and accuracy

It is widely accepted in the literature that a necessary and sufficient condition for an ensemble to be more accurate than any of its lone members is that the models are enough accurate and diverse [Hansen and Salamon, 1990; Dietterich, 2000a]. By enough accurate we are referring to models that can do better than just random guessing on unseen examples. The error rate of these weak models should be $< 1/2$ (assuming a binary classification problem). Diversity implies models making different generalization errors on new data points (uncorrelated errors). Thus, the main characteristic of a well designed ensemble is that it is able to reduce the bias and variance on its predictions because the various errors of the models "average out". For example, the probability of a majority vote to be wrong on an ensemble comprised of m hypothesis, is equal to the area under the binomial distribution where more than $m/2$ hypothesis are wrong. Figure 3.1 depicts this case for two ensembles with $m = 30$ (left) and $m = 50$ (right), and each hypothesis having an error rate of 0.3. The probability of $m/2$ or more classifiers being simultaneously wrong at the same time (the ensemble being wrong) is 0.0169

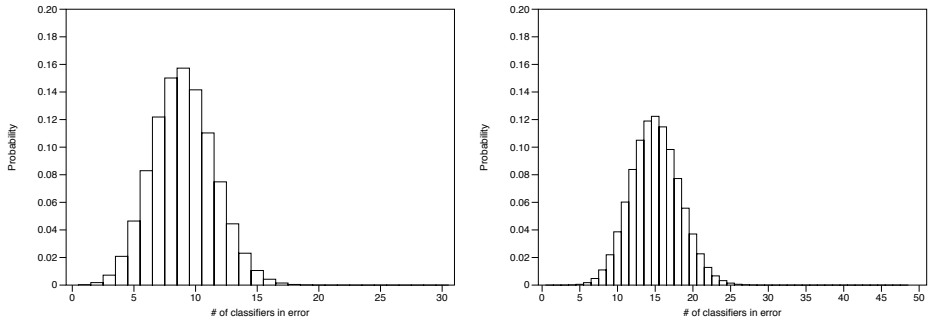

Figure 3.1: The probability that exactly m of 30 or 50 hypotheses (left/right) will make an error, assuming each hypothesis has an error rate of 0.3 and makes its errors independently of the other hypothesis

and 0.0024 respectively, which is much less that the individual error rate of the models [Dietterich, 2000a].

Different strategies have been proposed with the objective of introducing diversity into an ensemble. The most popular one is based on modifying the training set. The learning algorithm is then executed multiple times over different training sets. The most representative algorithms following this approach are *Bagging* [Breiman, 1996] and *Boosting* [Freund et al., 1996; Freund and Schapire, 1997]. *Bagging* uses a random sampling with replacement strategy from the original set in order to generate each training sample. This strategy is effective when the learners are unstable and tend to be reactive to little variations within the input space, as in the case of neural networks or decision trees. On the other hand, *Boosting* assigns a weight to each original example. Iteratively, these weights are updated giving more importance to the misclassified examples. The bigger the weight for an input data point, the bigger the chance for it to appear in a new sample.

3.2 Designing ensembles for quantification

As we have already been discussing, there exist problems, like covariate shift or quantification, in which we:

(i) are aware of the distribution changing between training and test,

(ii) are able to characterize the kind of changes to emerge, or at least,

(iii) can make a previous assumption about the nature of such changes.

Therefore, the core idea in this dissertation contribution is that we can take advantage of that knowledge to build ensembles with an appropriate diversity that are prepared to effectively represent the expected changes in the distribution. The ensemble learning task is separated into three phases:

(i) sample generation with each one representing an expected distribution change,

(ii) model training on each generated sample, and

(iii) combination of the individual estimates to produce the final ensemble prediction.

The most important step is certainly the first one, which must be adapted depending on the dataset shift problem being tackled. In this dissertation chapter we are showing a simple adaptation for the binary quantification problem. In other cases, the adaptation may require from additional knowledge about the particular application nature, as in the case of covariate shift, with $\mathbf{P}(\boldsymbol{x})$ being the changing component. In the binary quantification case it is more straightforward. It is important to keep in mind the learning assumption of the problem: quantification mainly stands for $\mathcal{Y} \rightarrow \mathcal{X}$ problems with $\mathbf{P}(y)$ changing and $\mathbf{P}(\boldsymbol{x}|y)$ remaining constant. Once the expected distribution changes are defined, the goal is to generate training samples a priori representing them, so it is possible to effectively react to its presence. Thus, sampling is not aimed at correctly predicting concrete examples, even though it is a characteristic to also be considered in the future, but just at representing the expected distribution changes.

3.2.1 Sample generation, training and evaluation procedures

We propose the following procedure in order to generate each training sample. First, the sample prevalence p_j is randomly selected in [0..1]. Then, simple random sampling with replacement is performed within the positive class examples until the number of positive examples, given by the chosen prevalence p_j, is obtained. This process guarantees $\mathbf{P}(\boldsymbol{x}|y)$ to be constant. The same operation is repeated for the negative class, with its prevalence being equal to $n_j = 1 - p_j$. By changing the prevalence of each generated sample we obtain the desired diversity. This procedure is repeated until the number of defined training samples m is reached. Figure 3.2 describes this process.

The next step is to train the base quantifier algorithm over each generated sample. In the case of the quantifier using the AC correction (2.17) it is necessary to estimate the *tpr* and the *fpr* using each sample D_j. An important detail to highlight is that the generated samples D_j should have an adequate number of examples in order to be able to produce accurate estimates for *tpr* and *fpr*. In our

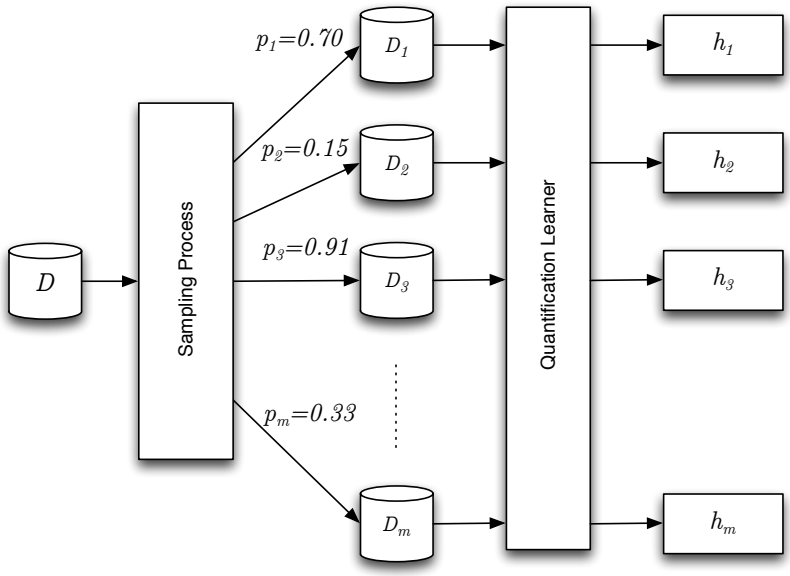

Figure 3.2: Training stage of the ensemble. Each sample D_j is generated with a different prevalence p_j

case, we generate samples with the same cardinality as the original training set but with a different prevalence in general.

Finally, given an unlabeled test set T, each ensemble learned model h_j is applied and an estimate of the test set prevalence, p'_j, is obtained. The definitive ensemble prediction is computed by applying a combination function. In the experiments reported in this chapter we have just used a simple aggregation function, the arithmetic mean. Nevertheless, we do think that using quantification devised aggregation and selection measures may lead to significant improvements in the ensemble accuracy. Chapter 4 is specially dedicated to analyze the benefits of using quantification conceived aggregation and selection measures.

3.2.2 Benefits of using ensembles for quantification

We believe that applying ensemble techniques to the quantification problem results in more benefits than its application in other kind of problems, like classification. The advantage of using a combination of models instead of a single model, with the risk of not performing well in some cases, is generic to every kind of problem in which ensemble technique is applied to. However, in the quantification case there are additional benefits.

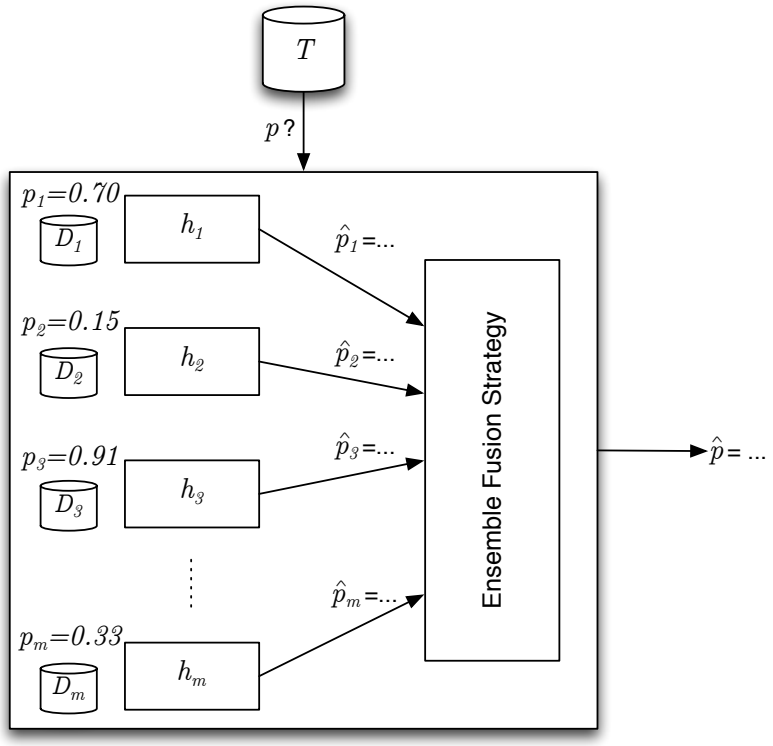

Figure 3.3: Prediction stage of the proposed ensemble for binary quantification. Each model h_j is applied over test set T, obtained different values for the estimated prevalence (p'_j) of T. These values are combined using a combination strategy to obtain the final estimate p'

First of all, we actually know how to explicitly introduce diversity into the ensemble. In the second place, using multiple models when the base quantifier corrects its estimates leads to major improvements. The AC correction (2.17), while theoretically produces perfect quantifications, is, in practice, risky. When having inaccurate *tpr* and *fpr* estimates, the correction will cause inappropriate changes. As we shall show in Section 3.3.2, AC manages to produce very accurate results for some problems, while it significantly drops its performance for others. It not only depends on the underlying classifier accuracy, but principally on the correction quality. Using ensembles reduces both risks. Not only the diversity itself results in an ensemble specific benefit, but also the combining strategy lessens more risks than in a typical classification scenario.

3.3 Experimental setting

The objective of the performed experiments was to empirically verify the effectiveness of applying ensembles to a problem noticeably suffering from data distribution changes, as it is the case of quantification. These experiments compare the performance of a baseline quantifier, CC, and two state-of-the-art quantification algorithms, AC and HDy, to the performance of its ensemble adapted versions, ECC, EAC and EHDy. The aim is to improve the performance of AC and HDy using ensembles.

3.3.1 Experimental design

We basically chose AC and HDy as quantifier representatives because they are based on very contrasting approaches, thus increasing the study significance. On the one hand, AC, as an emblematic quantification algorithm based on classifying and counting with posterior estimate corrections using *tpr* and *fpr*. On the other hand, HDy, even though it also makes use of a classifier to represent sample distributions, it is based on measuring distribution similarities rather than in classifying, counting and correcting. Additionally, other three ensemble-based algorithms, named as CC(bag), AC(bag) and HDy(bag), were included in the experiments to better study the influence of the design decisions made to propose EAC and EHDy. These three methods use bagging [Breiman, 1996] as the underlying classifier, applied with different base learners.

The compared algorithms can be grouped according to two different criteria. The main criterion depends on the quantification approach: CC methods (CC, CC(bag) and ECC), AC methods (AC, AC(bag) and EAC) and HDy methods (HDy, HDy(bag) and EHDy). Notice that the differences between the algorithms in the same group are due to the different classifier or approach being used.

But from another point of view, we can group the algorithms considering the models they use. This is important for the fairness of the experiments. First, we have those quantifiers that only need a single non-ensemble classifier. To this group belong CC, AC and HDy. In the experiments all these methods share the same underlying model in order to perform a fair comparison between them. The second group is formed by CC(bag), AC(bag) and HDy(bag). The bagging classifier is also the same. Finally, ECC, EAC and EHDy implement the ensemble strategy proposed in this chapter, using samples with a different prevalence. Again, the ensemble is composed by exactly the same models. Hence, the performance differences between the algorithms in the same group, according to the second criterion, are only due to the different way in which the predictions made by the classifier are used; the differences between them are not due to the classifier.

Table 3.1: Summary of datasets: n is the number of examples, d is the dimension of the input space, $P(N)$ the number of positive (negative) examples and p the prevalence of the positive class

Dataset	*Identifier*	n	d	P	N	p
Balance Scale (left)	balance.1	625	4	288	337	46%
Balance Scale (balanced)	balance.2	625	4	49	576	8%
Balance Scale (right)	balance.3	625	4	288	337	46%
Breast Cancer Wisconsin	breast-cancer	683	9	444	239	65%
Contraceptive Method Choice (no use)	cmc.1	1473	9	629	844	43%
Contraceptive Method Choice (long term)	cmc.2	1473	9	333	1140	23%
Contraceptive Method Choice (short term)	cmc.3	1473	9	511	962	35%
Cardiotocography Data Set (normal)	ctg.1	2126	22	1655	471	78%
Cardiotocography Data Set (suspect)	ctg.2	2126	22	295	1831	14%
Cardiotocography Data Set (pathologic)	ctg.3	2126	22	176	1950	8%
Pima Indians Diabetes Data Set	diabetes	768	8	268	500	35%
Statlog German Credit Data	german	1000	24	700	300	70%
Haberman's Survival Data	haberman	306	3	81	225	26%
Johns Hopkins University Ionosphere DB	ionosphere	351	34	126	225	36%
Iris Plants Database (versicolour)	iris.2	150	4	50	100	33%
Iris Plants Database (virginica)	iris.3	150	4	50	100	33%
Mammographic Mass	mammographic	830	5	403	427	49%
Page Blocks Classification (5)	pageblocks.5	5473	10	115	5358	2%
Phoneme	phoneme	5404	5	1586	3818	29%
Semeion Handwritten Digit (8)	semeion	1593	256	155	1438	10%
Sonar, Mines vs. Rocks	sonar	208	60	97	111	47%
Spambase Data Set	spambase	4601	57	1813	2788	39%
SPECTF Heart Data	spectf	267	44	55	212	21%
Tic-Tac-Toe Endgame Database	tictactoe	958	9	332	626	35%
Blood Transfusion Service Center Data Set	transfusion	748	4	178	570	24%
Wisconsin Diagnostic Breast Cancer	wdbc	569	30	212	357	37%
Wine Recognition Data (1)	wine.1	178	13	59	119	33%
Wine Recognition Data (2)	wine.2	178	13	71	107	40%
Wine Recognition Data (3)	wine.3	178	13	48	130	27%
Wine Quality Red (6-10)	wine-q-red	1599	11	855	744	53%
Wine Quality White (6-10)	wine-q-white	4898	11	3258	1640	67%
Yeast	yeast	1484	8	429	1055	29%

Datasets

The experiments were carried out over 32 datasets, specifically all of those used in [Barranquero et al., 2013; González-Castro et al., 2013], except *iris.1*, *acute.a* and *acute.b* (perfect quantifiers are obtained, so they are trivial problems), and *coil*, *lettersG* and *lettersH* because the experiments did not finish in a reasonable time. As we shall see in this description of the experimental setting used, the experiments reported are quite complex due to several reasons, mainly the use of ensembles, the grid-search applied for tuning the classifier parameters and the testing phase, which is more time consuming than in classification problems, because we need to estimate the prevalence for different test sets. The testing phase is even slower

when using ensembles. Table 3.1 contains the characteristics of the finally used datasets. Some of them are originally binary, while others are binary adaptations of multi-class problems. For instance, *iris.2* is a binary problem in which the original class 2 has been mapped to the positive class, with the remaining classes being mapped to the negative class. In all cases, the instances with missing values were removed.

Base classifiers

We decided to use probabilistic base classifiers because HDy uses the scores provided by the underlying classifier to represent the distributions. This decision avoids unbounded predictions and homogenizes the obtained histograms. Such histograms were generated with 8 bins. The results of HDy with other values for this parameter were similar (with 4 bins), or worse (with 12 and 20 bins).

Three different base classifiers were employed to verify that the obtained results do not depend on a particular classifier. Each base learner is applied in combination with the nine compared methods. The selected learners were:

(i) Naïve Bayes (*NB*), the most classical probabilistic classifier,

(ii) Logistic Regression (*LR*), to induce linear models, and

(iii) *SVM* with the Gaussian kernel (*RBF*) defined as

$$K(\boldsymbol{x}, \boldsymbol{x}') = \exp(-\gamma \cdot ||\boldsymbol{x} - \boldsymbol{x}'||^2), \tag{3.1}$$

and probabilistic output Platt [2000] to produce non linear models.

Parameter tuning

In the case of *LR* and *SVM-RBF*, the regularization parameter (C) was set through a search in $C \in [10^{-3}, \ldots, 10^{3}]$, by optimizing the geometric mean in binary classification using a 5-fold cross validation with 2 repetitions (CV5x2). As regards *SVM-RBF*, the parameter γ was selected within the values $[0.001, 0.005, 0.01, 0.05, 0.1, 1]$ following the same procedure. *NB* uses gaussian distributions to deal with numerical attributes.

The geometric mean of *tpr* and *tnr* (*true negative rate*), defined as

$$GM = \sqrt{\frac{TP}{P} \cdot \frac{TN}{N}}, \tag{3.2}$$

measures the ability of a classifier to balance sensitivity (accuracy on positive examples) and specificity (accuracy on the negative examples). It was selected to

deal with imbalanced datasets [Barandela et al., 2003], see details in Table 3.1. Moreover, in the case of *LR* and *SVM-RBF* we equally weighted both classes by using the $-w$ parameter in LibLinear and LibSVM [Fan et al., 2008]. The weight of each class multiplies the regularization parameter C, so both classes have the same influence in the loss term of the optimization problem. Both decisions are tailored at obtaining well suited classifiers even under important class imbalance scenarios, a usual situation in quantification problems. Without class weighting, and by only optimizing the classifier accuracy, both CC and AC performed significantly worse and the difference with respect to the ensemble versions got accentuated, due to the classifier tending to predict the majority class in most of the cases.

Evaluation procedure and loss functions

The estimation of *tpr* and fpr, a critical step for AC-based approaches, was obtained through a 10-fold cross-validation over the training examples. Notice that for EAC, each model of the ensemble requires to estimate its *tpr* and fpr. The size of the ensemble approaches was set to $m = 30$. We decided to generate samples D_j with the same size as the original training set D in order to be able to get accurate estimates of the *tpr* and the *fpr* when needed. The prevalence p_i was randomly chosen in the interval $[5\% - 95\%]$ for each sample. We deliberately avoided values near 0% and 100% because in these points it may arise difficulties when estimating both *tpr* and *fpr*, due to the lack of examples in one of the classes. Given the selected prevalence p_i, random sampling with replacement (to ensure $\mathbf{P}(\boldsymbol{x}|y)$ remains constant) was used to generate the number of examples required for each class.

The results reported in the next section were obtained using 5-fold cross validation with two repetitions (CV5x2). For each test partition 100 samples were generated with replacement, in which the positive class prevalence p was randomly selected ranging from 0% to 100%. Thus, each result in the tables represents the mean of 1000 quantifications. The error metric represented is the mean squared error (MSE). Similar results are obtained by analyzing the mean absolute error (MAE).

3.3.2 Experimental results

Table 3.2 shows the scores for each method using Logistic Regression as base classifier. It is worth noting that EAC and EHDy obtains the best results overall, not only better than those of their counterparts, AC and HDy, but also better than the scores of the bagging methods (AC(bag) and HDy(bag)). Analyzing the results from that point of view, it seems that the use of ensembles helps to produce a slight improvement in performance, comparing AC vs AC(bag) and HDy vs HDy(bag). This improvement is boosted when the proposals of this dissertation are applied,

Table 3.2: Mean squared error using Logistic Regression as base classifier. The score of the best performer in a group for each dataset is in bold

dataset	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
balance.1	0.0031	0.0031	**0.0025**	0.0025	0.0025	**0.0021**	0.0014	0.0013	**0.0012**
balance.2	0.1309	0.1436	**0.1123**	0.1833	0.2185	**0.1328**	0.3708	0.3298	**0.1530**
balance.3	0.0021	0.0021	**0.0019**	**0.0010**	**0.0010**	**0.0010**	**0.0006**	**0.0006**	**0.0006**
breast-cancer	**0.0008**	0.0010	0.0009	0.0007	0.0008	**0.0006**	**0.0004**	**0.0004**	**0.0004**
cmc.1	**0.0436**	0.0439	0.0451	0.0123	0.0121	**0.0114**	0.0118	0.0117	**0.0105**
cmc.2	**0.0429**	**0.0429**	0.0432	0.0164	0.0167	**0.0148**	0.0109	0.0107	**0.0100**
cmc.3	**0.0521**	0.0524	0.0538	0.0318	0.0281	**0.0238**	0.0184	0.0178	**0.0156**
ctg.1	**0.0056**	**0.0056**	0.0059	0.0016	0.0017	**0.0013**	0.0006	**0.0005**	0.0007
ctg.2	0.0092	**0.0080**	0.0110	0.0013	**0.0010**	**0.0010**	**0.0010**	**0.0010**	0.0011
ctg.3	**0.0020**	0.0025	0.0024	**0.0011**	0.0014	0.0013	0.0015	0.0017	**0.0014**
diabetes	0.0211	**0.0208**	0.0229	0.0102	0.0071	**0.0066**	0.0057	**0.0051**	**0.0051**
german	**0.0274**	**0.0274**	0.0300	0.0091	0.0094	**0.0085**	**0.0070**	0.0071	0.0075
haberman	0.0777	0.0757	**0.0715**	0.0780	0.0773	**0.0662**	0.0966	0.0856	**0.0735**
ionosphere	**0.0152**	0.0236	0.0213	**0.0068**	0.0088	0.0108	**0.0131**	0.0148	0.0185
iris.2	0.0448	0.0492	**0.0438**	0.0677	0.0765	**0.0457**	0.0329	0.0326	**0.0189**
iris.3	0.0018	**0.0014**	0.0015	0.0017	**0.0009**	**0.0009**	0.0017	0.0014	**0.0011**
mammographic	0.0168	0.0163	**0.0148**	0.0101	0.0077	**0.0068**	0.0048	**0.0042**	0.0044
pageblocks.5	0.0078	**0.0077**	0.0264	0.0046	**0.0043**	0.0052	0.0016	0.0016	**0.0010**
phoneme	**0.0253**	0.0255	0.0254	**0.0017**	0.0019	0.0018	0.0010	**0.0009**	0.0010
semeion.8	**0.0070**	0.0089	0.0082	**0.0017**	0.0024	0.0046	0.0054	0.0042	**0.0037**
sonar	**0.0230**	0.0250	0.0254	0.0357	0.0387	**0.0187**	0.0362	0.0333	**0.0220**
spambase	0.0079	0.0050	**0.0047**	0.0021	0.0004	**0.0003**	**0.0002**	**0.0002**	**0.0002**
spectf	**0.0393**	0.0407	0.0408	0.0424	0.0515	**0.0278**	0.0549	0.0475	**0.0332**
tictactoe	0.0480	**0.0476**	0.0492	0.0365	0.0369	**0.0289**	0.0203	0.0198	**0.0172**
transfusion	0.0471	**0.0441**	0.0446	0.0293	0.0239	**0.0194**	**0.0214**	0.0234	0.0230
wdbc	0.0066	0.0066	**0.0063**	0.0037	0.0033	**0.0031**	**0.0023**	0.0025	0.0024
wine.1	0.0042	0.0044	**0.0037**	0.0035	0.0038	**0.0026**	0.0024	0.0027	**0.0021**
wine.2	0.0094	0.0067	**0.0052**	0.0089	0.0057	**0.0041**	0.0058	0.0041	**0.0038**
wine.3	0.0016	0.0016	**0.0014**	0.0021	0.0020	**0.0015**	0.0009	0.0010	**0.0008**
wine-quality-red	0.0234	**0.0232**	0.0238	0.0082	**0.0062**	0.0069	0.0051	**0.0044**	0.0047
wine-quality-white	0.0333	**0.0326**	0.0328	0.0030	0.0027	**0.0023**	0.0017	0.0016	**0.0015**
yeast	**0.0308**	0.0316	0.0321	0.0088	0.0085	**0.0072**	0.0043	0.0045	**0.0039**
Average	0.0254	0.0260	0.0255	0.0196	0.0207	0.0147	0.0232	0.0212	0.0139
Average ranking	7.1875	7.2500	7.0625	5.5469	5.3125	3.5156	3.7500	3.1719	2.2031

as we can see analyzing the differences between AC(bag) and EAC and between HDy(bag) vs EHDy.

If we focus on the comparison between the original quantifiers, AC/HDy, and the ensemble adapted versions, EAC/EHDy, we can observe the ensemble versions winning most of the times. With respect to the EAC vs AC comparison, the former wins on 26 occasions, losses just 5 times and there is only one draw. As concerns EHDy, it wins on 22 occasions to HDy, losses on 6 and there are 4 draws. A more detailed examination reveals that ensemble versions defeats are usually produced by small margins in low quadratic error problems, while, in contrast, in more difficult problems generally they get the victory by relatively large margins. These results are relevant since AC is supposed to produce, at least theoretically, perfect quantifications independently of the classifier accuracy. However, in practice, the correction (2.17) often works well in a range close to the training set prevalence,

but its effectiveness drops as the test set prevalence moves away. Using ensembles reduces the impact of this situation, since the training sample generation process ensures all the prevalence domain to be represented within the ensemble models.

Finally, the trivial approximation of classifying and counting, provided by CC-based methods, attains worse scores than the other approaches. This result is in line with the experiments reported in the quantification literature, in which CC is often outperformed by proper quantifiers. Notice that the performance of CC does not improve even when an ensemble is used as classifier, in fact, the scores of the three methods (CC, CC(bag) and ECC) are quite similar, but ECC obtains the best ranking. The reason for such results is motivated by the CC approach itself. When the classifier is not perfect, CC is only able to provide good estimates for a small range of the prevalence, for the rest of the values the estimates are worse. In the best scenario, bagging may improve the accuracy of a single logistic regression model, but it cannot correct the weakness of the CC approach. This behavior shall be analyzed graphically in the next section.

Statistical analysis

These results can be analyzed in different ways from an statistical point of view. First of all, and following [García and Herrera, 2008], we have performed an statistical comparison in two steps:

(i) a Friedman test rejects the hypothesis of all the methods performing at the same level, and

(ii) pairwise comparisons using the Bergmann-Hommel test with $\alpha = 0.05$ indicate that EHDy and EAC are significantly better than CC and AC.

The difference between EHDy and HDy is not significant though. Neither it is between EAC and AC(bag), and between EHDy and HDy(bag). However, it is important to note that all these comparisons are deeply influenced by the inclusion of CC approaches, a group of algorithms that systematically perform worse than the other methods, and especially because there exist dependencies and correlations between the studied methods. Table 3.5 contains the complete results of the Bergmann-Hommel test comparing EHDy and EAC with the rest of the methods, considering both MAE and MSE.

Having in mind that our objective was to compare the ensemble adapted versions to its original counterparts, Wilcoxon signed-rank test seems more appropriate for the performed experiments. This test reveals that EAC performs significantly better than AC ($p = 0.0001$), with the same behavior being found for EHDy with respect to HDy ($p = 0.0013$). Table 3.6 shows the p-values for the comparison with all the methods.

Table 3.3: Mean squared error using Naïve Bayes as base classifier. The score of the best performer in a group for each dataset is in bold

dataset	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
balance.1	0.0288	**0.0090**	0.0210	0.0135	0.0044	**0.0025**	0.0031	**0.0016**	0.0017
balance.2	0.2758	0.2758	**0.0979**	0.2758	0.2758	**0.0935**	0.2758	0.2758	**0.2753**
balance.3	0.0297	**0.0085**	0.0208	0.0082	0.0037	**0.0020**	0.0023	0.0019	**0.0014**
breast-cancer	0.0007	0.0007	**0.0006**	0.0007	0.0006	**0.0005**	0.0005	0.0005	**0.0004**
cmc.1	0.0672	0.0786	**0.0498**	0.0451	0.0266	**0.0137**	0.0147	0.0101	**0.0097**
cmc.2	0.1100	0.1223	**0.0431**	0.0307	0.0299	**0.0126**	0.0126	0.0120	**0.0109**
cmc.3	0.3391	0.3391	**0.0641**	0.3391	0.3391	**0.0485**	0.0211	0.0186	**0.0147**
ctg.1	0.0118	0.0119	**0.0073**	0.0026	0.0024	**0.0018**	0.0023	**0.0016**	**0.0016**
ctg.2	0.0125	0.0133	**0.0089**	0.0018	**0.0014**	0.0015	0.0022	**0.0017**	0.0018
ctg.3	0.0115	0.0148	**0.0052**	0.0035	0.0045	**0.0030**	0.0048	0.0045	**0.0036**
diabetes	0.0318	0.0360	**0.0261**	0.0133	0.0144	**0.0076**	0.0109	**0.0090**	**0.0090**
german	0.0852	0.0898	**0.0396**	0.0704	0.0196	**0.0096**	0.0166	0.0099	**0.0094**
haberman	0.2257	0.2895	**0.0752**	0.2689	0.2746	**0.0573**	0.0725	0.0856	**0.0562**
ionosphere	0.0119	0.0126	**0.0105**	0.0052	0.0055	**0.0047**	0.0054	**0.0053**	0.0054
iris.2	0.0054	0.0069	**0.0043**	0.0066	0.0078	**0.0034**	0.0035	0.0029	**0.0026**
iris.3	0.0099	**0.0044**	**0.0044**	0.0134	0.0047	**0.0040**	0.0052	0.0044	**0.0040**
mammographic	0.0148	0.0145	**0.0137**	0.0054	0.0056	**0.0047**	0.0046	**0.0034**	0.0037
pageblocks.5	**0.0437**	0.0530	0.0897	0.0162	**0.0125**	0.0905	**0.0120**	0.0146	0.1049
phoneme	**0.0173**	0.0178	0.0278	0.0012	0.0016	**0.0011**	**0.0010**	**0.0010**	0.0012
semeion.8	0.0098	0.0099	**0.0072**	0.0033	**0.0026**	0.0028	0.0046	0.0044	**0.0030**
sonar	**0.0223**	0.0234	0.0236	0.0235	0.0253	**0.0147**	0.0196	**0.0141**	0.0173
spambase	0.0072	0.0067	**0.0062**	**0.0004**	0.0005	**0.0004**	**0.0004**	**0.0004**	**0.0004**
spectf	0.0363	0.0341	**0.0316**	0.0391	0.0356	**0.0295**	0.0416	0.0394	**0.0271**
tictactoe	0.1083	0.0684	**0.0619**	0.1266	**0.0322**	0.0336	0.0249	0.0181	**0.0171**
transfusion	0.0703	0.1410	**0.0519**	0.0398	0.1165	**0.0265**	0.0316	0.0262	**0.0203**
wdbc	0.0027	0.0025	**0.0022**	0.0018	0.0017	**0.0012**	0.0014	**0.0013**	0.0014
wine.1	0.0011	0.0013	**0.0010**	**0.0010**	0.0014	**0.0010**	**0.0009**	0.0013	**0.0009**
wine.2	**0.0016**	0.0020	0.0018	**0.0014**	0.0021	0.0015	**0.0017**	0.0020	**0.0017**
wine.3	0.0010	0.0009	**0.0008**	0.0012	0.0009	**0.0007**	0.0009	**0.0005**	0.0006
wine-quality-red	0.0251	**0.0229**	0.0262	0.0078	0.0061	**0.0053**	**0.0050**	0.0062	0.0051
wine-quality-white	0.0367	**0.0355**	0.0367	0.0031	0.0032	**0.0022**	0.0024	**0.0021**	0.0024
yeast	0.0935	0.1018	**0.0385**	0.0269	0.0169	**0.0068**	0.0122	0.0077	**0.0075**
Average	0.0546	0.0578	0.0281	0.0437	0.0400	0.0153	0.0193	0.0184	0.0194
Average ranking	7.4531	7.6562	6.3125	5.6875	5.6562	2.6562	4.0312	3.1406	2.4062

Naïve Bayes and SVM with Gaussian kernel result analysis

Table 3.3 reports the experimental results when Naïve Bayes is used as probabilistic classifier. On average, it seems that these results are a little bit worse than those obtained using logistic regression, but the conclusions are similar. However, there are two interesting changes. In this case, ECC performs much better than CC and CC(bag). It also occurs in the comparison between EAC and AC(bag); now the difference is significant.

Studying the ensemble versions results, they again outperform single quantifiers. On the one hand, EAC wins 28 times, AC wins just 2 and there are 2 ties. The difference between EHDy vs HDy is less pronounced, with 23 victories for EHDy, 6 ties and 3 victories for HDy. Statistically analyzing the results by using a Bergmann-Hommel test, we can see that EAC is significantly better than CC

Table 3.4: Mean squared error using SVM with RBF kernel as base classifier. The score of the best performer in a group for each dataset is in bold

dataset	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
balance.1	**0.0000**	0.0006	0.0003	**0.0001**	0.0005	0.0002	**0.0002**	0.0003	**0.0002**
balance.2	**0.2701**	0.2758	0.0337	0.3064	0.2758	**0.0456**	0.0713	0.1328	**0.0613**
balance.3	**0.0001**	0.0003	0.0003	**0.0001**	0.0002	0.0002	**0.0001**	0.0003	**0.0001**
breast-cancer	0.0006	0.0006	**0.0005**	0.0005	**0.0004**	**0.0004**	0.0004	**0.0003**	**0.0003**
cmc.1	0.0444	0.0447	**0.0392**	**0.0063**	0.0083	0.0099	0.0104	**0.0100**	0.0158
cmc.2	0.2283	0.2047	**0.0495**	0.0504	0.0461	**0.0147**	0.0216	0.0206	**0.0164**
cmc.3	0.1195	0.1328	**0.0581**	0.0404	0.0415	**0.0236**	0.0197	**0.0180**	0.0209
ctg.1	0.0103	0.0170	**0.0082**	**0.0020**	0.0034	0.0063	0.0073	0.0048	**0.0041**
ctg.2	0.0265	0.0416	**0.0111**	**0.0025**	0.0060	0.0097	0.0315	0.0186	**0.0161**
ctg.3	0.0142	0.0288	**0.0064**	**0.0045**	0.0051	0.0049	0.0200	0.0151	**0.0055**
diabetes	0.0488	0.0524	**0.0309**	0.0144	0.0161	**0.0134**	0.0406	**0.0174**	0.0265
german	0.0804	0.0797	**0.0344**	0.0174	0.0206	**0.0095**	0.0136	**0.0115**	0.0131
haberman	0.1971	0.1992	**0.0684**	0.1289	0.1304	**0.0402**	0.0714	0.0742	**0.0547**
ionosphere	**0.0031**	0.0043	0.0032	0.0024	0.0026	**0.0023**	0.0027	0.0033	**0.0022**
iris.2	0.0030	0.0032	**0.0026**	0.0032	0.0039	**0.0022**	0.0039	0.0039	**0.0018**
iris.3	0.0018	0.0051	**0.0016**	0.0018	0.0049	**0.0011**	0.0019	0.0033	**0.0012**
mammographic	**0.0118**	0.0131	0.0129	**0.0038**	0.0042	0.0040	0.0036	**0.0034**	0.0035
pageblocks.5	0.2900	0.2645	**0.1735**	0.2476	**0.1576**	0.1819	**0.1743**	0.2410	0.2019
phoneme	0.0094	0.0117	**0.0076**	**0.0005**	**0.0005**	0.0016	0.0024	**0.0016**	0.0026
semeion.8	**0.0055**	0.0113	0.0641	**0.0007**	0.0019	0.0621	0.0081	**0.0058**	0.1110
sonar	**0.0106**	0.0113	0.0107	0.0123	0.0144	**0.0079**	0.0083	**0.0060**	0.0105
spambase	**0.0034**	**0.0034**	0.0049	0.0007	**0.0005**	0.0012	0.0015	**0.0010**	0.0020
spectf	0.1213	0.1309	**0.0529**	0.1473	0.1796	**0.0394**	0.1025	0.0820	**0.0508**
tictactoe	**0.0004**	0.0009	0.0007	0.0004	0.0004	**0.0003**	**0.0005**	0.0006	0.0006
transfusion	0.3301	0.2884	**0.0646**	0.2981	0.1254	**0.0520**	0.0812	0.0762	**0.0573**
wdbc	**0.0033**	**0.0033**	0.0043	**0.0021**	0.0022	0.0023	0.0133	0.0131	**0.0076**
wine.1	**0.0107**	0.0297	0.0298	**0.0150**	0.0302	0.0206	0.0078	0.0108	**0.0074**
wine.2	0.0283	0.0291	**0.0240**	0.0195	0.0220	**0.0179**	0.0234	0.0271	**0.0231**
wine.3	0.0530	0.0793	**0.0339**	**0.0296**	0.0369	0.0305	0.0623	0.0540	**0.0507**
wine-quality-red	0.0215	**0.0207**	0.0291	0.0079	0.0073	**0.0051**	0.0053	**0.0047**	0.0066
wine-quality-white	0.0561	0.0567	**0.0357**	**0.0018**	0.0023	0.0027	0.0040	**0.0027**	0.0068
yeast	0.1087	0.1129	**0.0358**	0.0118	0.0131	**0.0050**	0.0080	**0.0063**	0.0068
Average	0.0660	0.0674	0.0292	0.0431	0.0364	0.0193	0.0257	0.0272	0.0247
Average ranking	6.4531	8.1094	5.8438	3.9219	4.7031	2.7656	4.8906	4.5469	3.7656

and its counterpart AC, while EHDy is also significantly better than these two methods, but it is not with respect to HDy. After applying a Wilcoxon signed-rank test, we found that EAC performs significantly better than AC ($p = 0.00002$), and also EHDy with respect to HDy ($p = 0.0002$).

The situation changes when using SVM with RBF kernel as base classifier, see the scores in Table 3.4. The ensemble versions still perform better than their counterparts; EAC vs AC: 17 wins, 0 ties and 15 losses; EHDy vs HDy: 20 wins 2 ties and 10 losses. However, statistical tests reveal that the differences are no longer significant. In the EHDy-HDy comparison, test results are near to be significant though ($p = 0.078$). In our opinion, this behavior is explained by two facts:

(i) SVM with RBF kernel are complex models and the risk of overfitting increases, and

Table 3.5: Average ranking for all methods using different performance measures. Symbols § y † indicate a significant difference ($p < 0.05$) between EAC and EHDy and the corresponding method using a Bergmann-Hommel test, respectively

LR	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
MSE	7.1875§†	7.2500§†	7.0625§†	5.5469§†	5.3125†	3.5156	3.7500	3.1719	2.2031
MAE	6.9219§†	6.7500§†	7.2812§†	5.7344§†	5.6875§†	3.6094	3.5938	3.2344	2.1875

NB	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
MSE	7.4531§†	7.6562§†	6.3125§†	5.6875§†	5.6562§†	2.6562	4.0312	3.1406	2.4062
MAE	7.1562§†	7.0000§†	6.5312§†	5.9531§†	5.8125§†	2.9531	3.9688	2.9688	2.6562

SVM	CC	CC(bag)	ECC	AC	AC(bag)	EAC	HDy	HDy(bag)	EHDy
MSE	6.4531§†	8.1094§†	5.8438§†	3.9219	4.7031§	2.7656	4.8906§	4.5469	3.7656
MAE	6.0938§†	7.8594§†	6.2500§†	4.0938	4.7344	2.9375	4.7188	4.4375	3.8750

(ii) SVM is not originally a probabilistic classifier, with the probabilities resulting from an output post-process.

These circumstances have a negative impact on the stability of the results, leading to some models with a poor performance on one hand, and accurate models on the other, i.e. a greater model performance variance. Notice that CC and CC(bag) with SVM-RBF are the worst performing algorithms by far considering the three base learners.

Table 3.5 contains a summary of the obtained results for all the base classifiers containing both the mean squared error (MSE), and the mean absolute error (MAE). Interestingly, EAC and EHDy rank most of the times in the two first positions. The only exception is when logistic regression is the base learner. The reason is because HDy algorithm clearly outperforms AC in that case. There are significant differences for Naïve Bayes and logistic regression, both for MSE and MAE scores.

3.3.3 Graphical analysis

An additional experiment was performed in order to graphically analyze the behavior of the quantifiers. The goal was to study the performance at specific prevalence values. The differences with respect to the first experiment are twofold: the number of subsamples generated for each test set, 210 instead of 100, and the values of the prevalence that were used. In this case, only 21 different values were considered, those nearest to [0:0.05:1]. Notice that sometimes it is impossible to obtain some of these values depending of the number of examples in the test set.

Table 3.6: The two proposed algorithms, EAC and EHDy, are statistically compared with the rest of the methods using Wilcoxon sign-rank tests. The table shows the p-value of each comparison

LR		CC	CC(bag)	ECC	AC	AC(bag)	HDy	HDy(bag)
MSE	EAC	4.097e-08	8.338e-07	1.495e-06	0.0001433	0.0011340	0.4564072	0.2536064
	EHDy	1.287e-06	1.429e-07	3.174e-05	4.392e-05	5.642e-05	0.0013742	0.0018011
MAE	EAC	6.519e-09	8.847e-09	4.417e-06	0.0002057	0.0006889	0.2099543	0.1497597
	EHDy	1.727e-07	5.122e-08	6.379e-08	1.119e-05	7.716e-06	0.0012973	0.0016667

NB		CC	CC(bag)	ECC	AC	AC(bag)	HDy	HDy(bag)
MSE	EAC	7.517e-06	7.517e-06	2.011e-06	2.716e-05	7.653e-05	0.0083313	0.7420193
	EHDy	9.761e-06	4.209e-07	6.045e-05	7.935e-05	4.050e-05	0.0002547	0.0110552
MAE	EAC	1.287e-06	1.733e-06	5.122e-08	1.263e-05	4.570e-05	0.0803250	0.7719339
	EHDy	1.733e-06	3.512e-06	7.716e-06	2.924e-05	2.479e-05	0.0002778	0.1206512

SVM		CC	CC(bag)	ECC	AC	AC(bag)	HDy	HDy(bag)
MSE	EAC	3.736e-05	6.302e-06	2.692e-05	0.1322391	0.0152215	0.0038533	0.0152188
	EHDy	6.136e-05	2.312e-06	0.0148032	0.4388932	0.4890153	0.0786387	0.1681790
MAE	EAC	5.433e-05	5.871e-07	6.905e-07	0.1396090	0.0341199	0.0038655	0.0146764
	EHDy	0.0001472	3.512e-06	0.0058128	0.6378833	0.6511483	0.2241815	0.4773475

In that case, the nearest possible value is used. Hence, there are 100 results for each prevalence (10 folds and 10 subsamples per fold, 210 divided by 21 prevalence values). The next figures represent in different ways such results.

Figure 3.4 shows the bias error (defined as $p-p'$) of CC, AC, HDy, EAC and EHDy over four datasets. Apparently, it seems that the overall performance of AC, HDy, EAC and EHDy is quite good in terms of bias: the average value predicted is near the true prevalence. But, notice that negative and positive biases cancel each other in the computation of bias, thus the graphics do not show the magnitude of the errors in both directions.

However, bias graphics allow us to observe two important facts. First, CC follows the exact behavior analyzed theoretically by Forman in [Forman, 2008]: CC attains a perfect quantification at one point of p, and from there, CC underestimates the prevalence when p increases, and overestimates the prevalence when p decreases from CC's optimal point. The deviation with respect to the true prevalence depends on the classifier accuracy. If the classifier is perfect, the bias is 0. But the bias increases when the accuracy decreases. This occurs for *spambase* and *yeast* datasets. The bias is smaller in the case of *wine2*, and tiny for *iris3*. The big issue of CC is that it is almost impossible to obtain perfect classifiers for real applications. The other interesting aspect of bias analysis is that ensemble versions, EAC and EHDy, have more bias than AC and HDy, mainly in the tails. This is in part due to the use of the mean as the aggregation strategy.

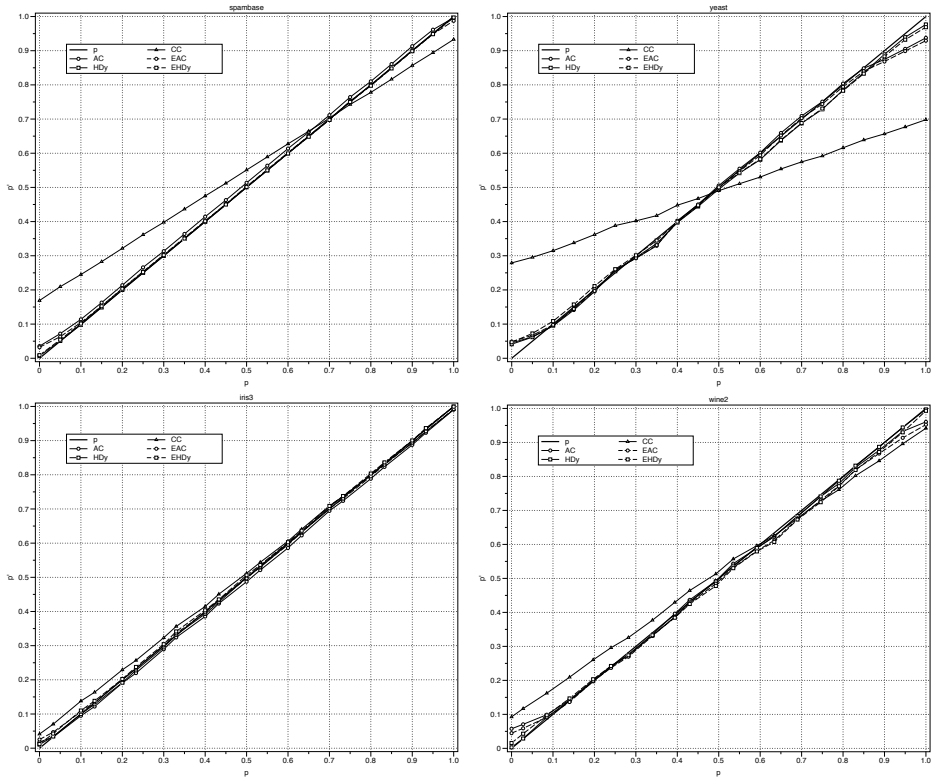

Figure 3.4: Analysis of bias when the prevalence varies in [0:0.05:1]. The figure compares five algorithms: CC, AC, EAC, HDy and EHDy. Each result comes from the prediction average of 100 sample sets with the same prevalence

Figure 3.5 and Figure 3.6 represent the distribution of the predictions using box plots. Each box plot comprises a group of 100 predicted test sets for each prevalence. These graphics demonstrate that the quantification problem is more difficult that it seems at first glance. The errors are lower in the case of *spambase* and *iris3* datasets (Figure 3.5), but quite big for *yeast* and *wine2* datasets (Figure 3.6). The most interesting fact in these graphics is that the variability of EAC and EHDy is lower than that of AC and HDy. Notice that the body of the boxplot is usually smaller, and also they often have shorter whiskers and less outliers. For instance, these aspects can be easily observed in the case of EAC vs AC over *spambase* dataset. This result is theoretically expected due to the use of ensembles. It is well known that ensembles tends to reduce the variance of the underlying classifier.

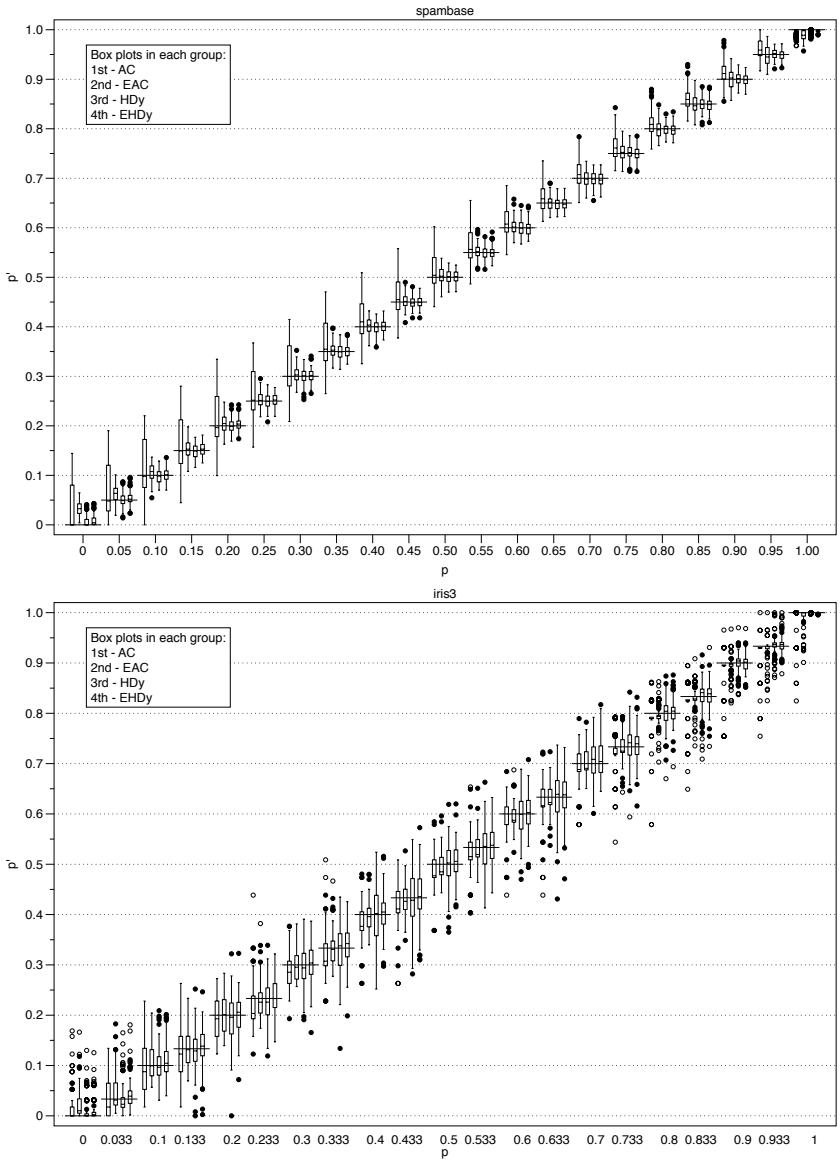

Figure 3.5: The figure shows the distribution of the predictions for *spambase* (39% of positives in the training set) and *iris3* (33%) datasets using a box plot. The horizontal lines represent the true prevalence for each group. Only four methods are displayed: AC, EAC, HDy and EHDy. Such methods appear always in the same order for each prevalence: first AC, then EAC, HDy is the third method and the last one is EHDy

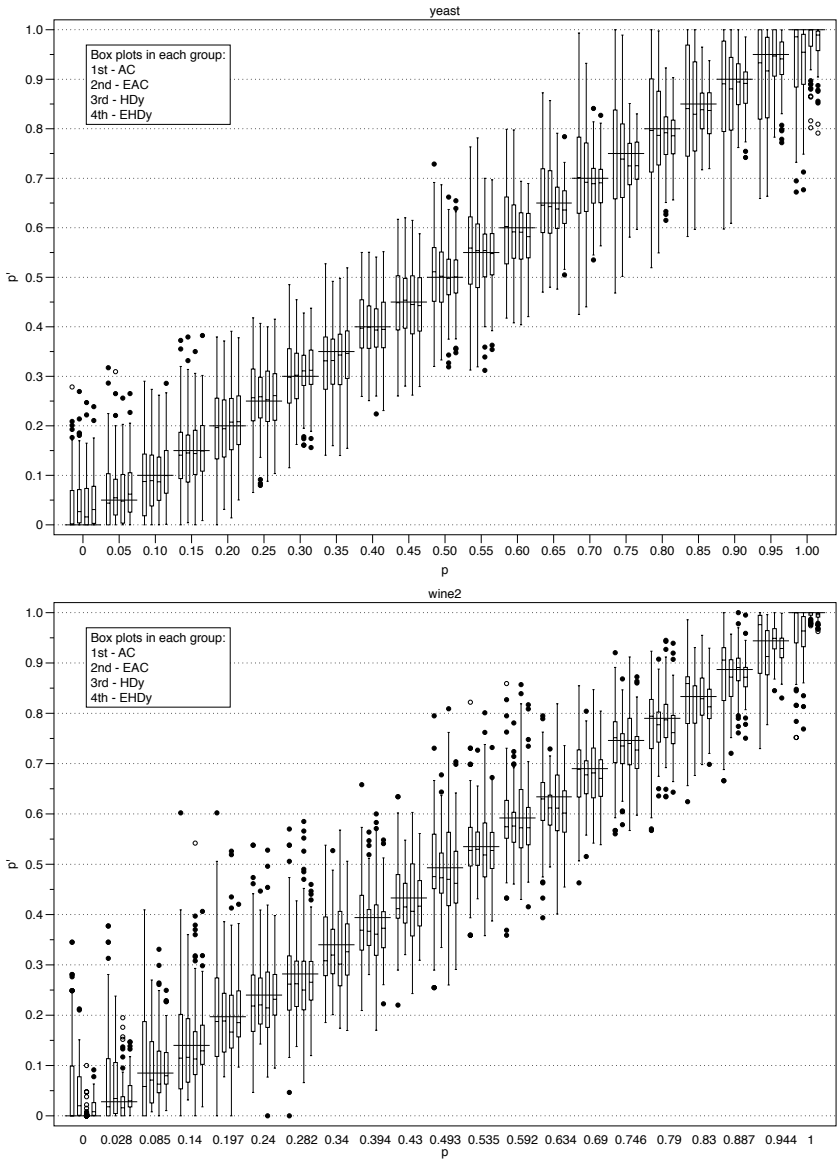

Figure 3.6: The figure shows the distribution of the predictions for *yeast* (29% of positives in the training set) and *wine2* (40%) datasets using a box plot. The horizontal lines represent the true prevalence for each group. Only four methods are displayed: AC, EAC, HDy and EHDy. Such methods appear always in the same order for each prevalence: first AC, then EAC, HDy is the third method and the last one is EHDy

Chapter 4

Selection measures for ensembles of quantifiers

In the previous chapter of this dissertation we introduced the novel concept of utilizing ensemble learning techniques to problems suffering from data distribution changes between training and test phases. More specifically, we applied it to the binary quantification problem in which, by definition, class probabilities change. The core idea was to generate different training samples, one for each weak quantifier, simulating the distributional changes we are expecting to take place on the new test samples. The final ensemble prevalence estimation was simply calculated by averaging all the base quantifier predictions. From here, we are referring to this method as *Ensembles of Quantifiers* (EoQ).

We do think that the performance of EoQ can be significantly improved by using more sophisticated aggregation strategies. These also includes consider using some sort of quantification specific knowledge in order to construct the definitive ensemble estimation. For example, it would be possible to give more importance to those weak quantifiers whose training distribution is closer to the new test sample distribution. This whole chapter of the dissertation is devoted to study quantifier devised selection measures for EoQ. The idea is to select some potentially good ensemble models instead of aggregating all of them. Our principal objective is to analyze the relationship between the base quantifiers and the selection measures; maybe the best select criterion depends on the base quantifier being used, or maybe it is independent and we are able to find a single measure outperforming the rest.

In this chapter we are proposing three new ensemble selection criteria specially devised for quantification [Pérez-Gállego et al., 2017a]. One of them is static in essence and the other two are dynamic. We present an exhaustive study in which we compare, for each considered base quantifier, the performance of applying four different selection measures (the three new proposed ones, plus a baseline criterion based on the quantifier accuracy). The results obtained by aggregating all of the ensemble models (original EoQ) are also included. In fact, we extend the experiments from the previous chapter by adding PCC and PAC as base quantifiers.

4.1 Designing selection measures for EoQ

There exist several ways of combining EoQ. In contrast to using all models with the same importance, the main alternatives are ensemble weighting and ensemble selection. The former consists in assigning different weights to the models comprising the ensemble. The idea is that the best models contribute more to the final prediction than the worst ones. The second method is based on selecting the best subset of models, discarding the rest. It is worth noting that ensemble selection is a particular case of the weighting strategy, in which the weights are just 1 (selected) or 0 (discarded). Notice that they can be used together; selecting the best models but combining them with different weights. From a conceptual point of view, both approaches present a comparable complexity in the sense that they may have the same elements:

(i) a selection/ranking criterion to assess and compare the model contribution to the ensemble, and

(ii) a method able to assign the concrete weight to each model or to select the best subset of models.

These two problems should be solved separately to reduce the complexity of obtaining optimal EoQ [Ko et al., 2006]. In fact, assigning optimal weights is a problem hard to solve and the search problem to find the best group of models is NP-complete [Tamon and Xiang, 2000]: the searching space has $2^m - 1$ different non-empty subsets, being m the ensemble size.

Following the above reasoning, in this dissertation we have just focused on selection measures/criteria: the main aim is not to obtain the best possible combination strategy with optimal weights or with an optimal selection, but to study what kind of criterion works better for selecting or ranking the ensemble models. The selection/ranking criteria can be better analyzed if they work in isolation, without an additional method to select models or to assign weights. Our main goal is to study the quality of the rankings provided by different criteria. We shall analyze, for instance, the performance of different subsets of models, increasing the size by including first those models with highest ranking. Our primal interest is to find out which is the best selection measure for some state-of-the-art base quantifiers. The use of selection algorithms proposed in the literature, like hill-climbing search [Caruana et al., 2004], clustering [Zhang and Cao, 2014], greedy search [Partalas et al., 2008], weight thresholding [Krawczyk and Woźniak, 2016], reinforcement learning [Partalas et al., 2009], genetic algorithms [Ruta and Gabrys, 2005], quadratic integer programming [Zhang et al., 2006] or hybrid methods [Lin et al., 2014] is out of the scope of this dissertation.

We have considered two different groups of selection measures. The first group leads to static ensemble selection, meaning that the same subset of models is used

for all testing samples, discarding the rest. The second group allows dynamic selection, that is, applying a different subset for each sample. Static selection principal advantages are mainly that predictions can be delivered faster and less memory is required to store the models. On the other hand, dynamic ensemble selection allows a flexible structure of the ensemble and an adaptive behavior [Lin et al., 2014].

4.1.1 Static measures: ACC and MAX

The most commonly used static selection measures are based on the accuracy/precision of the models. The idea is straightforward: select the best (strongest) models, discarding the worst (weakest) ones. This kind of selection criterion is defined in terms of performance measures. For instance, accuracy is a natural choice for classification problems: select the models with the highest accuracy. We also apply here this approach, named as ACC, but adapted to binary quantification. This means to use a binary quantification measure instead of classification accuracy. In the experiments of this chapter, the models are ranked and selected according to their MSE scores (2.12). It is worth noting that, in addition to the selected performance measure, it is important to define the validation process to estimate such errors. As it was discussed in Section 2.2, in quantification, it is essential to employ a representative collection of samples with a suitable range of prevalences. In our experiments, error estimation for each model is made by using the training samples generated for the rest of models.

In this dissertation we propose a new selection criterion that it is related to ACC, but adapted to the characteristics of some paradigmatic quantification algorithms. This selection measure is called MAX, and is especially devised for AC and PAC quantifiers. MAX was inspired by the method of the same name proposed in [Forman, 2008]. The idea is to select those models that maximize the difference $tpr - fpr$ for the AC method and $TP^{pa} - FP^{pa}$ for the PAC method. This means that MAX measure prefers those models with a large value for the denominator in the correction equations, (2.17) and (2.19). A smaller value in the denominator implies a bigger correction, which it is risky. It is preferable a smooth correction, in the sense that the prediction of CC/PCC is rather accurate and only needs a small correction.

Notice that this criterion is somehow similar to ACC, but at the same time, presents subtle differences that become relevant for imbalanced domains, in which the class distribution is skewed, since examples of one class (usually the negative class) appear much more frequently. For balanced domains, a model with a higher accuracy will also have a high tpr and a low fpr, resulting in a large value for the difference between both. But in an imbalanced situation, one model may have a high accuracy, but a small value for $tpr - fpr$, for instance, if the model predicts always the negative (majority) class ($tpr = 0$, $fpr = 0$).

4.1.2 Dynamic measures: P_{tr} and DS

The main characteristic of a quantification problem, the fact that data distribution changes, makes that dynamic quantifier ensemble selection seems a much more appealing approach than static selection. Our proposal is to select those models that were trained using a training distribution more similar to the distribution of the testing sample. The key issue is how to compare such distributions in a computationally efficient manner, avoiding other complex approaches like, for instance, density estimation [Silverman, 1986; Scott, 2015]. We propose two different and novel criteria in this dissertation.

The first selection criterion assumes that both distributions are similar when the prevalence of the positive class is also similar in both distributions. The problem is that we know the prevalence of the positive class in the training distribution, but we totally ignore the prevalence of the testing sample. This is precisely what we want to estimate. Our idea is to use all the models in the ensemble to obtain a first estimate of the prevalence for the testing sample. Then, in a second round, we will rank the models according to the difference between the prevalence of their corresponding training samples, and the first estimation obtained using all the models. The aim is to select the models that are close to such first estimate, discarding those that are far away.

This criterion, called P_{tr} in the experiments, seems very simplistic at first sight, but is closely related to several formal algorithms, namely AC, PAC and HDy. Actually, P_{tr} makes exactly the same learning assumption that those quantification algorithms, that is, P_{tr} also assumes that $\mathbf{P}(\boldsymbol{x}|y)$ does not change. When such assumption holds, a change in the data distribution is only due to a variation in the prevalence of the classes. Notice that HDy follows a similar idea when applies linear search to obtain the value of the prevalence that makes the combination of the positive distribution and the negative distribution most similar to the testing distribution (2.22).

The second criterion, called DS (Distribution Similarity), is inspired in the HDy algorithm. The idea is to compare the distribution of the y values for the testing and the training samples. The procedure is analogous to the one described for the HDy algorithm in Section 2.4.4. First, during the training phase, the training distribution of each ensemble model is summarized computing the histogram of its y values using the own model. The same y-histogram is computed for the testing sample, and both histograms are then compared using the Hellinger distance (2.23). The models are then ranked according to these distances. This procedure is similar to the one depicted in Figure 2.4.

Notice that, from a computational point of view, P_{tr} and DS are very similar. Both methods require to apply all models over the testing sample: in the former case, to calculate a first estimation for the testing sample prevalence, and in the latter, to obtain the distribution of the testing examples output values. Comparing

these two approaches with ACC and MAX criteria, presented above, we find that P_{tr} and DS are less efficient in the testing phase, because static approaches do not require to make any calculation during the testing phase (the best models are selected in the training phase). And for the same reason, ACC and MAX are less efficient in the training phase, because they require to estimate quantification accuracy or the difference between some rates (e.g. $tpr - fpr$).

4.2 Experimental setting

The experiments described here analyze the performance of the different selection measures we presented and discussed in the previous section. The main goal is to prove that selecting a group of models by applying these proposed criteria, outperforms the straightforward approach of taking the average prediction of all the ensemble models. Additionally, ensembles were built considering a group of representative base quantifiers to extend the significance of the experiments. The idea was to analyze whether the best criterion depends on the base quantifier or, in contrast, one of the criterion prevails over the rest, independently of the used quantification method.

4.2.1 Experimental design

We considered five different base quantification algorithms to learn the ensemble models: CC, PCC, AC, PAC and HDy (see Section 2.4). The corresponding ensemble versions shall be denoted as ECC, EPCC, EAC, EPAC and EHDy, respectively. In this sense, this section extends the results reported in Section 3.3 by the addition of EPCC and EPAC. Notice that all these quantification methods need an underlying classifier. We decided to use probabilistic classifiers for all of them because some of the algorithms, namely PCC, PAC, require a probabilistic classifier, and it is also preferable for others, like HDy. Additionally, this decision allows us to ensure that all the ensembles are comprised by the same models for a given dataset. Thus, despite we are just interested in studying the selection criteria for each EoQ separately, the reader could also compare all ensembles methods with the new selection measures.

Two distinct aggregation strategies were studied. First, we computed the average prediction of all the models, that is, the strategy employed by original EoQ in the previous chapter. This approach is denoted as ALL in the results. Our goal was to analyze if these results can be significantly improved. The second strategy is to select 50% of the models, just the best models given a new test sample, according to the four selection measures we have presented in this chapter:

(i) ACC: selects those models with the highest quantification accuracy, or lowest

error measure in terms of mean square error (2.12),

(ii) MAX: picks those models that maximize the denominator of equations (2.17) and (2.19) depending on the base quantifier used: $tpr - fpr$ for ECC and EAC, and $TP^{pa} - FP^{pa}$ for the probabilistic-oriented methods (EPCC, EPAC and EHDy),

(iii) P_{tr}: chooses those models that were trained with a sample that had a similar prevalence than that of the testing sample, and

(iv) DS: selects those models whose training distribution is most similar to the distribution of the testing sample, measured in terms of the Hellinger distance. Probabilities were discretized in 8 bins to compare both distributions.

Datasets

Thirty two datasets were used in the experiments, exactly the same employed in Section 3.3.1. Table 3.1 describes their main properties. Approximately half of them are multi-class problems (*balance*, *cmc*, *ctg*, *iris*, *pageblocks* and *wine*) in which one of the classes was mapped as the positive class and the others comprised the negative class. The rest are originally binary problems. The most important characteristic for binary quantification problems is the prevalence of the positive class (last column in Table 3.1). Notice that this value ranges from 2% to 70%, thus providing enough variability.

Base classifiers and parameter tunning

The probabilistic classifier employed was Logistic Regression [Fan et al., 2008]. The regularization parameter (C) was selected through a search in the interval $[10^{-3}, \ldots, 10^{3}]$, optimizing the geometric mean using CV5x2 (cross validation, 5 folds and 2 repetitions) over the training examples. As in the previous chapter, geometric mean (3.2) was chosen to deal with imbalanced datasets which frequently appear in quantification tasks. Besides, the positive class and the negative class were balanced ($-w$ parameter in LibLinear [Fan et al., 2008]) to obtain good classifiers even for severe imbalanced cases. The estimates for (tpr, fpr) in (2.17) and (TP^{pa}, FP^{pa}) in (2.19) were obtained using CV10x1 over the training data.

Evaluation procedure and loss functions

The number of models to build the ensembles was set to $m = 50$. The procedure for generating the training sample for each model was the following: the prevalence of each training sample, p_i, was uniformly selected from $[5\% - 95\%]$, and then,

the examples belonging to the sample were chosen using random sampling with replacement (trying to maintain $\mathbf{P}(\boldsymbol{x}|y)$ constant). The size of the sample was always equal to the size of the original training set.

Results in Section 4.2.2 and Section 4.2.3 were computed with CV5x2 experiments. 101 samples were generated with each test fold in order to adequately measure quantification performance. The prevalence of these samples varies uniformly ranging from 0% to 100%. Thus, each reported result in the next sections corresponds to the average of 1010 quantification tests. In order to study the behavior of the considered approaches, we have chosen two of the measures discussed in Section 2.2.1, MAE (2.10) and MSE (2.12).

4.2.2 Experimental results

Tables 4.1-4.5 show the MAE scores for the ensemble method using one particular base quantifier and applying different selection measures. For instance, Table 4.1 reports the results using CC as the base quantifier. The difference between the scores are just due to the selection strategies, because the ensemble models are the same. It is worth noting that ALL is the worst approach in this case. All selection approaches clearly boost the results of ALL. The best two methods are DS and ACC. In fact, they obtain the best results in 28 out of 32 datasets; ALL is not the best in any.

Table 4.2 shows the scores when PCC is the base quantifier of the ensemble method. Here, it is clear that ACC is the best performer: not only it ranks first in terms of average ranking, but it also obtains the best score for 25 datasets. The second best approach is MAX: so it seems that, in this case, the static selection and the criteria based on the performance of the underlying quantifiers are better than dynamic selection and those measures based on distribution similarity. Again, ALL does not obtain the best result in any of the datasets and is outperformed by all selection approaches.

Table 4.3 contains the MAE scores for the ensemble method based on the AC quantifier. Three approaches obtain similar results with this method: DS, ALL and MAX. The difference between them is small, both in terms of average ranking and the number of wins (13, 8 and 10 wins respectively). Interestingly, MAX achieves better performance than ACC, something that did not happen with ECC and EPCC. This could be somehow expected because MAX criterion was inspired by the equation that defines the AC quantifier.

Table 4.4 shows the results for the EPAC ensemble. MAX is the selection function that achieves the best average ranking, winning in 19 cases. This result is in line with the one described before for EAC ensemble. It makes sense that MAX outperforms other selection measures because it is designed for EPAC ensembles. The results for the rest of the approaches are pretty similar, being DS the second

Table 4.1: Mean absolute errors using different selection functions for the ensemble version of the Classify & Count (CC) quantifier. The best score for each dataset is in bold

ECC - Ensembles of CC

dataset	ALL	ACC	MAX	P_{tr}	DS
balance.1	0.0404	0.0417	**0.0396**	0.0397	**0.0396**
balance.2	0.3024	0.3116	0.3032	0.3042	**0.2875**
balance.3	0.0365	0.0373	0.0367	**0.0359**	0.0360
breast-cancer	0.0783	0.0757	0.0753	0.0624	**0.0619**
cmc.1	0.1809	0.1801	0.1808	0.1794	**0.1778**
cmc.2	0.1995	0.1982	0.1987	0.1981	**0.1978**
cmc.3	0.0624	**0.0596**	0.0620	0.0609	0.0605
ctg.1	0.0796	**0.0749**	0.0778	0.0803	0.0792
ctg.2	0.0379	0.0378	0.0377	0.0362	**0.0361**
ctg.3	0.1514	**0.1492**	0.1506	0.1502	0.1508
diabetes	0.1488	0.1470	0.1487	0.1429	**0.1418**
german	0.0983	**0.0948**	0.0962	0.0962	0.0956
haberman	0.2064	**0.2030**	0.2042	0.2090	0.2085
ionosphere	0.1307	**0.1145**	0.1247	0.1283	0.1269
iris.2	0.1766	**0.1671**	0.1731	0.1779	0.1701
iris.3	0.0232	0.0233	0.0253	0.0218	**0.0215**
mammographic	0.0237	0.0228	0.0229	0.0218	**0.0217**
pageblocks.5	0.1391	**0.1218**	0.1258	0.1451	0.1441
phoneme	0.0596	**0.0478**	0.0518	0.0643	0.0615
semeion.8	0.1337	0.1318	0.1332	0.1321	**0.1315**
sonar	0.1406	0.1327	0.1398	**0.1308**	0.1356
spambase	0.1795	**0.1778**	0.1790	0.1781	0.1782
spectf	0.1598	0.1580	0.1587	0.1498	**0.1444**
tictactoe	0.1865	0.1857	0.1862	0.1844	**0.1827**
transfusion	0.1703	**0.1701**	0.1704	**0.1701**	0.1702
wdbc	0.0625	**0.0602**	0.0633	0.0629	0.0631
wine.1	0.0443	**0.0398**	0.0461	0.0449	0.0442
wine.2	0.0538	**0.0496**	0.0549	0.0540	0.0526
wine.3	0.0210	0.0210	0.0223	0.0201	**0.0197**
wine-quality-red	0.1333	0.1326	0.1333	0.1315	**0.1312**
wine-quality-white	0.1462	**0.1435**	0.1454	0.1491	0.1473
yeast	0.1381	**0.1343**	0.1365	0.1362	0.1358
Avg. rank	4.1250	2.2500	3.5781	2.9375	**2.1094**

Table 4.2: MAE results using different selection functions for the ensemble version of the Probabilistic Classify & Count (PCC). The best score for each dataset is in bold

EPCC - Ensembles of PCC

dataset	ALL	ACC	MAX	P_{tr}	DS
balance.1	0.0861	**0.0750**	0.0755	0.0836	0.0847
balance.2	0.2609	0.2639	0.2631	0.2612	**0.2587**
balance.3	0.0575	**0.0531**	0.0539	0.0553	0.0554
breast-cancer	0.1416	0.1278	0.1278	0.1292	**0.1272**
cmc.1	0.2184	**0.2155**	0.2157	0.2175	0.2158
cmc.2	0.2302	**0.2286**	0.2287	0.2299	0.2293
cmc.3	0.0809	**0.0767**	0.0782	0.0789	0.0785
ctg.1	0.1044	**0.0994**	0.1003	0.1028	0.1020
ctg.2	0.0485	0.0473	0.0477	**0.0462**	**0.0462**
ctg.3	0.2076	**0.2014**	0.2016	0.2054	0.2072
diabetes	0.1932	0.1897	0.1903	0.1907	**0.1885**
german	0.1334	**0.1280**	0.1291	0.1322	0.1319
haberman	0.2276	**0.2225**	0.2226	0.2265	0.2287
ionosphere	0.1283	**0.1169**	0.1285	0.1209	0.1221
iris.2	0.2085	**0.2008**	0.2017	0.2056	0.2093
iris.3	0.0427	**0.0378**	0.0399	0.0393	0.0391
mammographic	0.0296	**0.0271**	0.0274	0.0276	0.0276
pageblocks.5	0.1166	**0.1066**	0.1089	0.1182	0.1168
phoneme	0.1192	**0.0984**	0.1042	0.1248	0.1199
semeion.8	0.1866	**0.1813**	0.1814	0.1841	0.1859
sonar	0.1893	0.1805	0.1829	**0.1772**	0.1835
spambase	0.2148	**0.2128**	**0.2128**	0.2145	0.2134
spectf	0.1913	0.1865	0.1871	0.1825	**0.1817**
tictactoe	0.2159	0.2138	0.2139	0.2154	**0.2133**
transfusion	0.2136	**0.2097**	0.2105	0.2142	0.2131
wdbc	0.0776	**0.0723**	0.0734	0.0753	0.0751
wine.1	0.0610	**0.0548**	0.0587	0.0588	0.0587
wine.2	0.0788	**0.0693**	0.0730	0.0768	0.0757
wine.3	0.0331	**0.0277**	0.0314	0.0312	0.0299
wine-quality-red	0.1756	**0.1725**	0.1731	0.1753	0.1738
wine-quality-white	0.1901	**0.1862**	0.1870	0.1922	0.1902
yeast	0.1814	**0.1732**	0.1761	0.1836	0.1801
Avg. rank	4.5625	**1.4062**	2.5156	3.5625	2.9531

best method.

Table 4.5 contains the latest scores, those corresponding to EHDy ensembles. As one could expect in this case, DS is the best criterion. However, the difference with respect to ALL and ACC is small. The worst method is MAX, maybe because the correction idea has nothing to do with EHDy ensembles.

Results for MSE are very similar to those previously discussed for MAE. The complete MSE scores have been omitted but the average rankings and their corresponding statistical analysis are included in the next section.

4.2.3 Statistical analysis

The obtained results may be analyzed using several statistical tests. First, we compare all the methods together taking into account their average ranks [García and Herrera, 2008]. This procedure comprises two steps:

(i) a Friedman test to reject/accept the null hypothesis (all approaches perform equally well from a statistical point of view), and

(ii) a set of pairwise comparisons using the Bergmann-Hommel test to analyze if one particular method is significantly better than other.

We are mainly interested in comparing ALL with the proposed selection measures. The average rankings of all ensemble methods using MAE and MSE are presented in Table 4.6.

The Friedman test rejects the null hypothesis (so there are significant differences among the methods) except for EPAC ($p = 0.3084$) and EHDy ($p = 0.2248$), both using MSE as the performance measure. Analyzing the pairwise comparisons using the Bergmann-Hommel test ($\alpha = 0.05$), we can observe that most of the significant differences occur for ECC and EPCC: all the selection measures are significantly better than ALL, except MAX when MAE is the performance measure, and P_{tr} for MSE. For the rest of ensemble models (EAC, EPAC and EHDy), the only significant difference is between ALL and ACC (MSE scores for EAC). There are other cases in which the difference is close to be significant, for instance MAX versus ALL (EAC and MAE, $p = 0.057$).

Regarding the rest of pairwise comparisons (not involving ALL), we see that ACC is significantly better than other selection functions for ECC and EPCC ensembles, for instance ACC is significantly better than DS, P_{tr} and MAX for EPCC ensembles, but it is worse for EAC, EPAC and EHDy ensembles, e.g. ACC is significantly worse than DS(EAC), MAX(EPAC). This suggests that ACC is a better criterion for those quantifiers that are based on the classify and count approach; more sophisticated quantifiers require other specific selection measures like the ones proposed in the paper.

Table 4.3: Mean absolute errors using different selection functions for the ensemble version of the Adjusted Count (AC) quantifier. The best score for each dataset is in bold

EAC - Ensembles of AC

dataset	ALL	ACC	MAX	P_{tr}	DS
balance.1	0.0371	0.0374	**0.0349**	0.0367	0.0368
balance.2	0.3284	0.3926	0.4067	0.3163	**0.2970**
balance.3	**0.0252**	0.0258	0.0257	**0.0252**	**0.0252**
breast-cancer	0.0559	0.0570	0.0598	0.0524	**0.0521**
cmc.1	0.0952	0.0947	0.0970	0.0950	**0.0940**
cmc.2	0.1186	0.1180	0.1122	0.1176	**0.1167**
cmc.3	0.0272	0.0271	**0.0268**	0.0283	0.0282
ctg.1	0.0254	0.0257	0.0254	0.0254	**0.0253**
ctg.2	0.0285	0.0289	**0.0283**	0.0284	**0.0283**
ctg.3	0.0662	**0.0658**	0.0663	0.0678	0.0666
diabetes	**0.0692**	0.0721	**0.0692**	0.0701	0.0702
german	0.0659	0.0677	**0.0630**	0.0683	0.0680
haberman	0.1929	0.1941	0.1892	0.1910	**0.1880**
ionosphere	**0.0853**	0.0915	0.0947	0.0891	0.0884
iris.2	0.1810	0.1857	**0.1737**	0.1782	0.1740
iris.3	0.0208	0.0213	0.0229	0.0208	**0.0202**
mammographic	**0.0186**	0.0194	0.0188	0.0192	0.0189
pageblocks.5	0.0535	0.0561	**0.0527**	0.0544	0.0546
phoneme	**0.0128**	0.0131	0.0130	0.0143	0.0139
semeion.8	0.0335	0.0341	0.0336	**0.0331**	**0.0331**
sonar	**0.1072**	0.1089	0.1092	0.1134	0.1105
spambase	0.0793	0.0798	**0.0791**	0.0799	0.0803
spectf	0.1165	**0.1109**	0.1193	0.1213	0.1159
tictactoe	0.1427	0.1426	**0.1376**	0.1423	0.1423
transfusion	0.1070	0.1082	0.1106	0.1087	**0.1054**
wdbc	0.0450	0.0455	**0.0447**	0.0452	0.0450
wine.1	0.0395	**0.0389**	0.0398	0.0402	0.0394
wine.2	0.0483	**0.0479**	0.0491	0.0508	0.0492
wine.3	0.0262	0.0261	0.0238	0.0239	**0.0237**
wine-quality-red	0.0637	0.0639	0.0647	0.0633	**0.0632**
wine-quality-white	**0.0371**	**0.0371**	0.0378	0.0377	0.0379
yeast	**0.0672**	0.0697	0.0699	0.0710	0.0681
Avg. rank	2.6875	3.5469	2.8438	3.4531	**2.4688**

Table 4.4: MAE results using different selection functions for the ensemble version of the Probabilistic Adjusted Count (PAC) quantifier. The best score for each dataset is in bold

EPAC - Ensembles of PAC

dataset	ALL	ACC	MAX	P_{tr}	DS
balance.1	0.0340	0.0338	0.0335	**0.0332**	0.0333
balance.2	0.3486	0.4748	0.4497	0.3253	**0.3039**
balance.3	0.0257	0.0262	0.0259	0.0257	**0.0256**
breast-cancer	0.0506	0.0511	0.0531	**0.0475**	0.0476
cmc.1	0.0843	**0.0800**	0.0808	0.0832	0.0840
cmc.2	0.0979	0.0980	**0.0967**	0.0981	0.0980
cmc.3	0.0240	0.0237	**0.0235**	0.0242	0.0241
ctg.1	0.0289	0.0294	**0.0282**	0.0285	0.0284
ctg.2	0.0311	0.0310	**0.0305**	0.0308	0.0307
ctg.3	0.0564	0.0559	**0.0545**	0.0553	0.0554
diabetes	**0.0674**	0.0680	0.0708	0.0681	0.0684
german	0.0521	0.0536	**0.0516**	0.0534	0.0534
haberman	0.2040	0.2096	**0.1984**	0.2023	0.2017
ionosphere	**0.0784**	0.0809	0.0884	0.0829	0.0825
iris.2	0.1660	0.1689	0.1662	0.1662	**0.1652**
iris.3	0.0288	0.0277	0.0283	0.0276	**0.0269**
mammographic	0.0179	0.0181	**0.0178**	0.0182	0.0181
pageblocks.5	0.0456	0.0459	**0.0438**	0.0473	0.0469
phoneme	0.0151	**0.0130**	0.0141	0.0188	0.0163
semeion.8	0.0311	0.0316	**0.0309**	0.0311	0.0311
sonar	0.1088	0.1092	**0.1085**	0.1097	0.1123
spambase	0.0865	0.0888	**0.0856**	0.0861	0.0864
spectf	0.1137	0.1129	**0.1125**	0.1142	0.1145
tictactoe	0.1233	0.1248	**0.1189**	0.1220	0.1221
transfusion	0.1150	0.1181	**0.1103**	0.1153	0.1166
wdbc	0.0409	**0.0401**	0.0413	0.0411	0.0410
wine.1	0.0407	0.0419	**0.0402**	0.0406	0.0405
wine.2	0.0550	0.0549	**0.0524**	0.0575	0.0560
wine.3	0.0253	0.0244	0.0243	0.0236	**0.0234**
wine-quality-red	0.0602	0.0597	0.0605	**0.0590**	0.0591
wine-quality-white	0.0336	0.0335	**0.0327**	0.0342	0.0343
yeast	0.0609	0.0616	**0.0608**	0.0631	0.0623
Avg. rank	3.1406	3.5000	**2.1719**	3.2344	2.9531

Table 4.5: Mean absolute errors using different selection functions for the ensemble version of the HDy quantifier. The best score for each dataset is in bold

EHDy - Ensembles of HDy

ddataset	ALL	ACC	MAX	P_{tr}	DS
balance.1	0.0282	**0.0277**	0.0288	0.0281	0.0279
balance.2	0.3537	0.3573	0.3469	**0.3312**	0.3406
balance.3	**0.0195**	0.0197	0.0205	0.0201	0.0200
breast-cancer	0.0482	0.0478	0.0571	0.0422	**0.0412**
cmc.1	0.0783	0.0764	0.0813	**0.0776**	**0.0776**
cmc.2	0.0891	**0.0889**	0.0906	0.0894	0.0891
cmc.3	0.0210	**0.0197**	0.0223	0.0198	0.0198
ctg.1	0.0260	0.0260	0.0261	**0.0252**	**0.0252**
ctg.2	0.0266	0.0267	0.0268	0.0264	**0.0262**
ctg.3	0.0482	0.0483	0.0483	0.0476	**0.0474**
diabetes	0.0704	0.0673	0.0719	0.0657	**0.0664**
german	**0.0501**	0.0507	0.0506	0.0518	0.0518
haberman	0.1980	0.2107	0.1950	0.1926	**0.1904**
ionosphere	0.1221	**0.0992**	0.1151	0.1234	0.1207
iris.2	**0.1221**	0.1288	0.1267	0.1252	**0.1221**
iris.3	0.0229	0.0236	**0.0227**	0.0253	0.0246
mammographic	0.0162	0.0152	**0.0149**	0.0157	0.0156
pageblocks.5	0.0252	0.0255	0.0252	0.0256	**0.0251**
phoneme	0.0102	0.0101	0.0103	**0.0100**	0.0101
semeion.8	**0.0239**	0.0240	0.0241	0.0240	0.0240
sonar	0.1095	0.1088	0.1164	0.1105	**0.1082**
spambase	0.0785	0.0796	**0.0772**	0.0802	0.0799
spectf	0.1206	0.1207	0.1208	0.1193	**0.1085**
tictactoe	**0.1047**	0.1063	0.1012	0.1049	**0.1047**
transfusion	0.1160	0.1215	0.1101	0.1180	**0.1148**
wdbc	0.0363	**0.0351**	0.0387	0.0364	0.0362
wine.1	0.0328	0.0334	0.0334	**0.0322**	0.0325
wine.2	0.0440	0.0429	**0.0425**	0.0443	0.0432
wine.3	0.0177	**0.0168**	0.0181	0.0191	0.0180
wine-quality-red	0.0542	**0.0543**	0.0552	**0.0543**	**0.0543**
wine-quality-white	**0.0320**	0.0321	0.0325	0.0322	0.0322
yeast	0.0581	**0.0559**	0.0592	0.0582	0.0588
Avg. rank	2.8594	2.8438	3.7656	3.1719	**2.3594**

Table 4.6: Average ranking for all ensemble methods using MAE (top) and MSE (bottom) as performance measures. Symbol † indicates that the selection function in the corresponding column is significantly better than ALL using a Bergmann-Hommel test ($p < 0.05$). Symbol § indicates the opposite

MAE average rankings

Algorithm	ALL	ACC	MAX	P_{tr}	DS
ECC	4.1250	2.2500 †	3.5781	2.9375†	**2.1094**†
EPCC	4.5625	**1.4062**†	2.5156 †	3.5625†	2.9531 †
EAC	2.6875	3.5469	2.8438	3.4531	**2.4688**
EPAC	3.1406	3.5000	**2.1719**	3.2344	2.9531
EHDy	2.8594	2.8438	3.7656	3.1719	**2.3594**

MSE average rankings

Algorithm	ALL	ACC	MAX	P_{tr}	DS
ECC	4.0156	**1.1250**†	3.5938 †	2.9531	2.3125 †
EPCC	4.6094	**1.4219**†	2.5156 †	3.4844†	2.9688 †
EAC	**2.4531**	3.6250 §	3.1250	3.1875	2.6094
EPAC	2.9219	3.4219	**2.5938**	2.9219	3.1406
EHDy	2.9688	3.1250	3.4688	2.8750	**2.5625**

However, the Bergmann-Hommel test discussed before presents some issues [Benavoli et al., 2016], basically because it depends on the set of compared methods: the outcome of the comparison between a pair of approaches also depends on the performance of the rest of the methods included in the study. This problem can be even worse in our experiments because some of the approaches are correlated (ACC and MAX, and DS and P_{tr}). Following [Benavoli et al., 2016] we perform multiple comparisons using the Wilcoxon signed-rank test. The obtained p-values are shown in Table 4.7.

The Wilcoxon tests confirm that all the selection functions are significantly better than ALL for ECC and EPCC, both for MAE and MSE. This includes MAX for CC ensembles, whose difference was not significant using the Bergmann-Hommel test. However, the most interesting results from our point of view are that:

(i) MAX significantly outperforms ALL for ensembles based on PAC,

(ii) DS also significantly improves the scores of ALL for HDy ensembles, and

(iii) ALL is significantly better than ACC for EAC ensembles.

The first two results point out that ad-hoc criteria, specifically designed to work with a concrete quantifier, can help to boost the performance of the ensemble

Table 4.7: p-values for the comparison between ALL and the proposed selection functions using the Wilcoxon signed-rank test

ECC	ACC	MAX	P_{tr}	DS
MAE	0.000069	0.012080	0.034120	0.000381
MSE	0.000044	0.006810	0.079819	0.000633

EPCC	ACC	MAX	P_{tr}	DS
MAE	0.000002	0.000002	0.000641	0.000008
MSE	0.000005	0.000008	0.002938	0.000031

EAC	ACC	MAX	P_{tr}	DS
MAE	0.004080	0.537180	0.180012	0.658280
MSE	0.004387	0.316767	0.626407	0.834729

EPAC	ACC	MAX	P_{tr}	DS
MAE	0.099776	0.016635	0.975385	0.945307
MSE	0.266549	0.015474	0.903040	0.637021

EHDy	ACC	MAX	P_{tr}	DS
MAE	0.929713	0.263913	0.940368	0.026648
MSE	0.577523	0.745263	0.483749	0.084681

methods. MAX criterion is defined to prefer those models that maximize $TP^{pa} - FP^{pa}$ which is the denominator of the correction formula (2.19) of PAC models. Therefore, it is reasonable that the performance of MAX for EPAC is good. Even more interesting is the case of the combination of EHDy and DS. The idea behind HDy quantifiers is to match the test distribution with the training distribution. Thus, it makes sense that best models in the EHDy ensemble are those trained with a distribution similar to the test distribution. In fact, EHDy with DS obtains the best overall results. If we perform Wilcoxon signed rank tests between the best method for each ensemble we obtain that: EHDy-DS is significantly better for MAE than: ECC-DS ($p = 5.771e{-}06$), EPCC-ACC ($p = 1.603e{-}05$), EAC-DS ($p = 0.0011$) and EPAC-MAX ($p = 4.392e - 05$); and for MSE is also better than the rest with the following p-values: ECC-ACC ($p = 2.313e - 06$), EPCC-ACC ($p = 2.313e - 06$), EAC-ALL ($p = 0.00941$) and EPAC-MAX ($p = 0.0009569$). Be aware, however, that these differences are not just motivated by the selection

measure, but by the combination of both elements.

4.2.4 Analysis of the selection strategy

The results discussed in the previous section were obtained selecting the 50% of the models. The main question regarding this policy, is what is the behavior of the ensemble methods when this percentage varies. The goal of this section is to graphically analyze this issue. Figures 4.1-4.20 show the performance for the ensemble method using one particular base quantifier and varying the percentage of selected models between 2% (using just the best model) and 100% (ALL strategy). The selection measure applied for each ensemble was the best performer in the previous experiment. For instance, Figures 4.1-4.4 depict the MAE scores using the combination of ECC ensembles and DS selection measure for all the benchmark datasets. The difference between the scores are just due to the ability of the selection measure to rank the best models of the ensemble.

By analyzing the results for the ECC-DS combination (Figures 4.1-4.4), we can observe that taking just the best model (2%) or averaging all of them (100%), achieves bad results: the best model rule wins just in 4 datasets (*balance2*, *ctg2*, *ionosphere* and *wine3*) and ALL in 3 cases (*iris2*, *wdbc* and *wine1*). The most usual behavior is that the scores tend to improve when the percentage of selected model increases from 2% until some point, in which the minimum (best) score is obtained. From that point the scores degrade again. This pattern occurs in 19 out of 32 datasets. A paradigmatic case is, for instance, the figure corresponding to *balance1* dataset. The position of the minimum depends on the problem: sometimes is reached early, between 10% and 30% (e.g. *breast cancer*), most of the times in the middle of the range, but it hardly happens for values greater than 70% (such cases mostly correspond for the datasets in which ALL is the winner rule).

Figures 4.5-4.8 show the graphs for the EPCC-ACC method. These results are quite surprising and totally different from those observed for the rest of the ensemble methods. In the most frequent pattern, the minimum is reached with the best model, and the scores degrade from there, almost linearly, when the percentage of selected models increase. This occurs in 26 datasets. There are other 4 cases (*ionosphere*, *iris3*, *semeion* and *spectf*) in which the pattern is quite similar and the difference is that the minimum is reached when taking between 10% and 25% of the models. Only 2 datasets show a clearly different behavior (*balance2* and *ctg3*). These results suggest that EPCC ensembles contain many bad PCC models, probably because it is difficult to provide good probabilities for some domains.

Figures 4.9-4.12 contain the results of the ensemble method based on AC quantifiers selecting the models using the DS criterion. Here, we observe that taking all the models produces good results in more cases than in the two previous methods. ALL rule attains the best result in 6 datasets (*cmc1*, *german*, *ionosphere*,

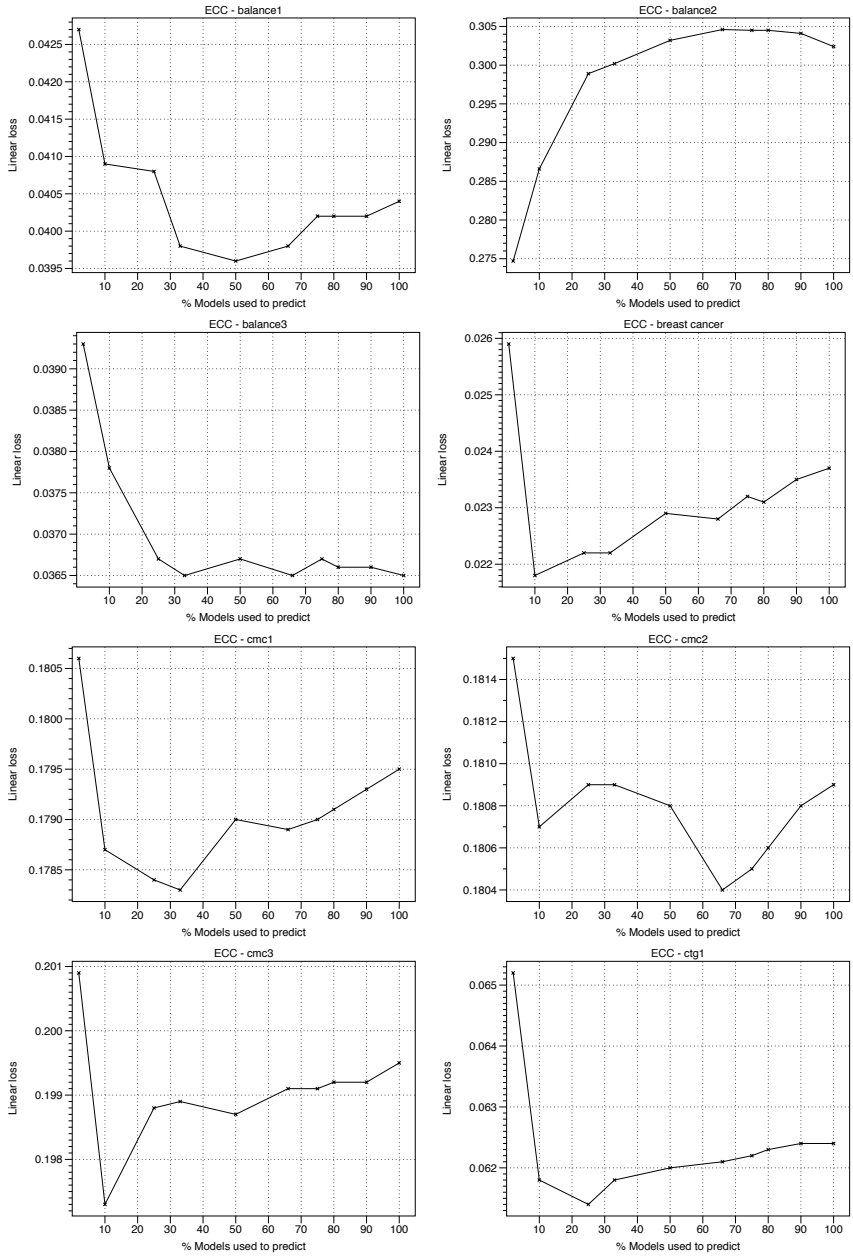

Figure 4.1: MAE scores selecting a different percentage of models using ECC-DS (part I)

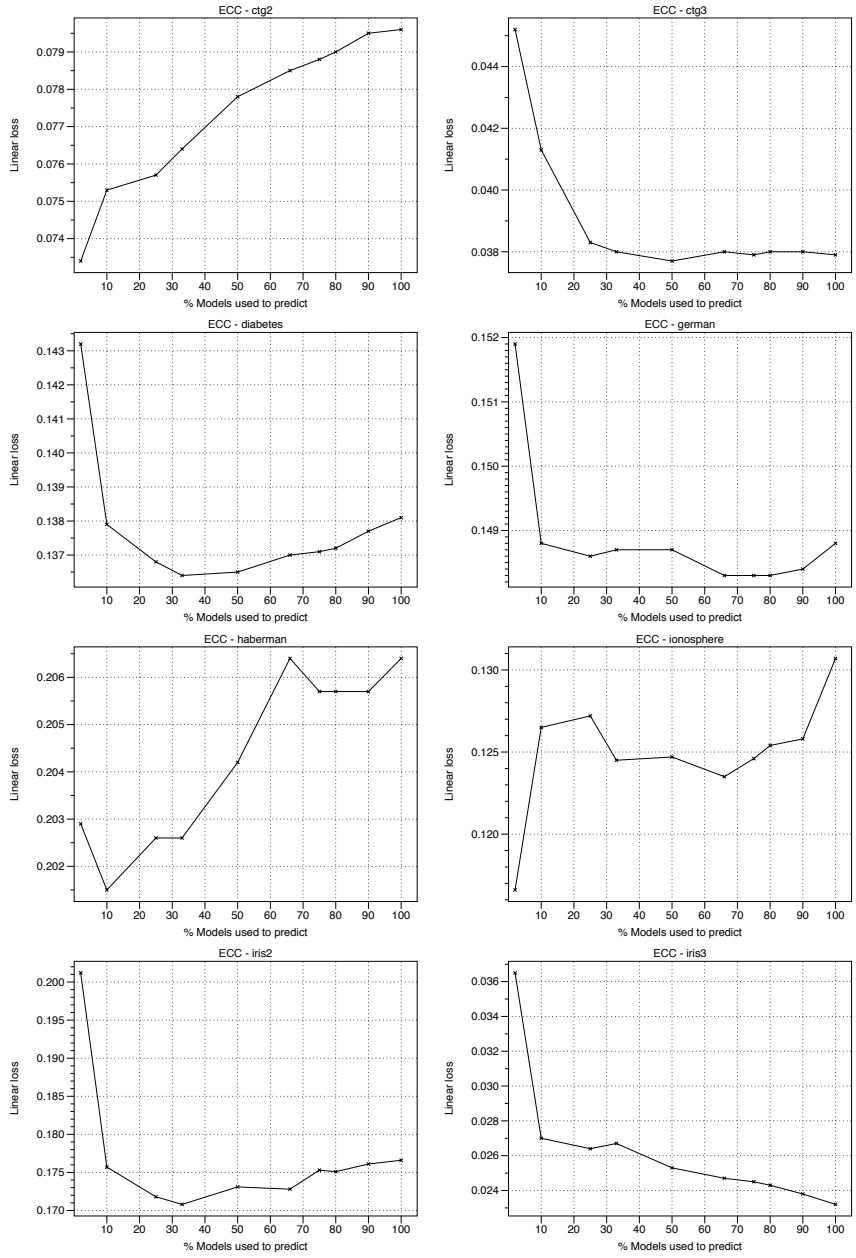

Figure 4.2: MAE scores selecting a different percentage of models using ECC-DS (part II)

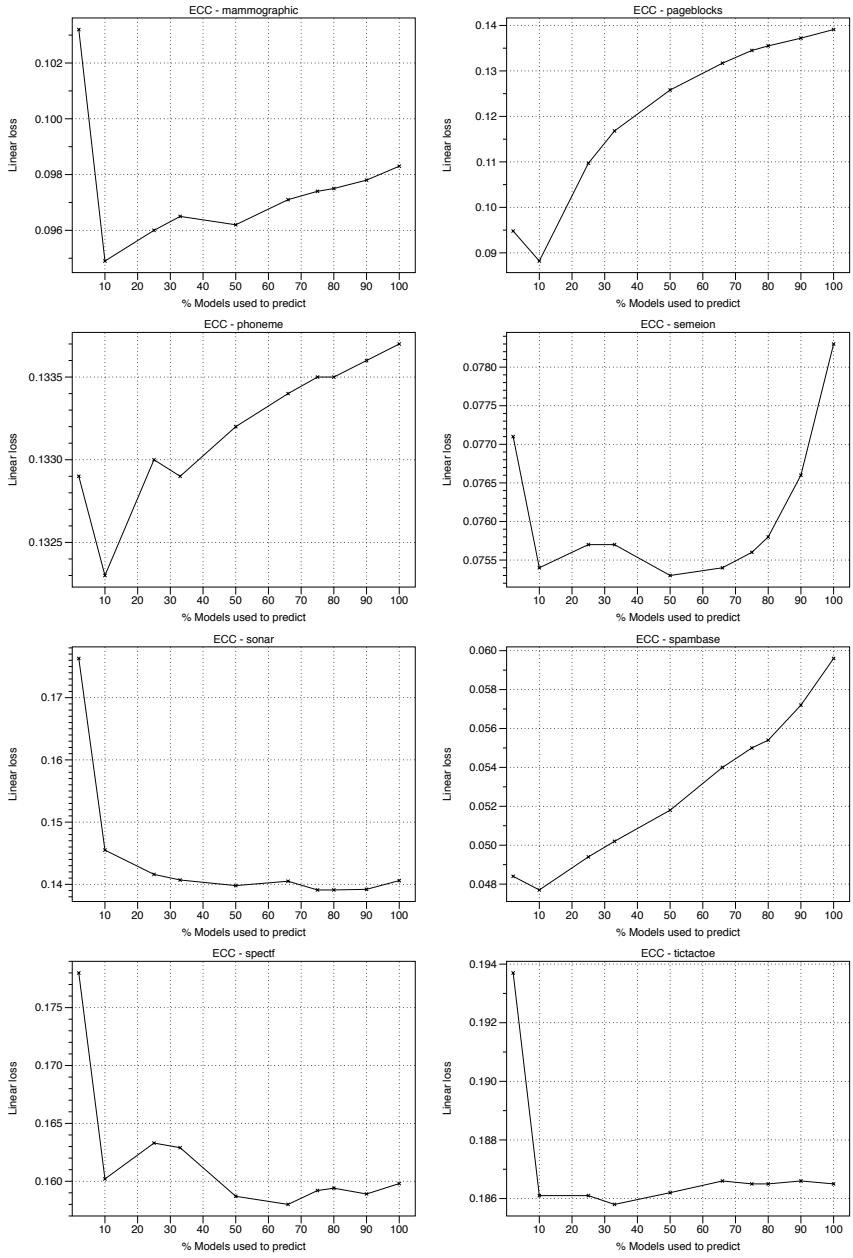

Figure 4.3: MAE scores selecting a different percentage of models using ECC-DS (part III)

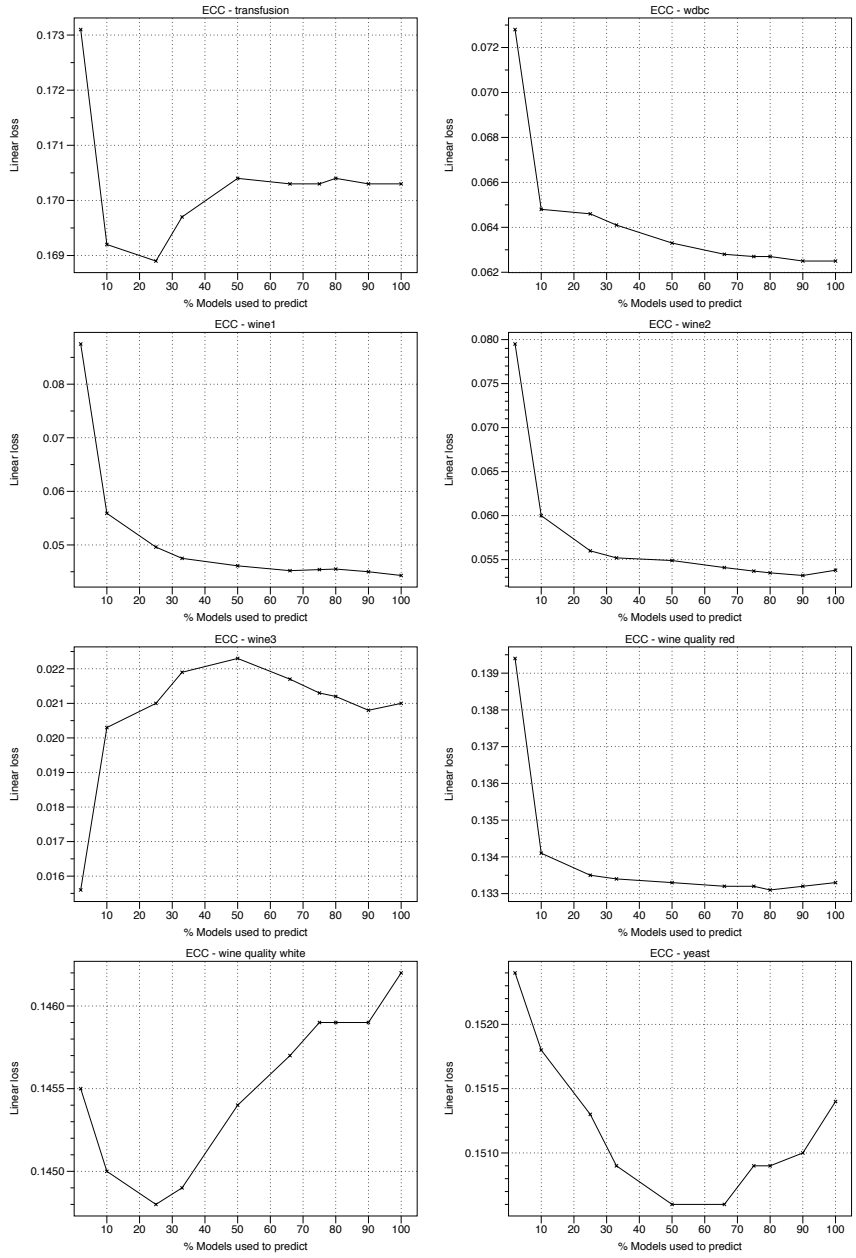

Figure 4.4: MAE scores selecting a different percentage of models using ECC-DS (part IV)

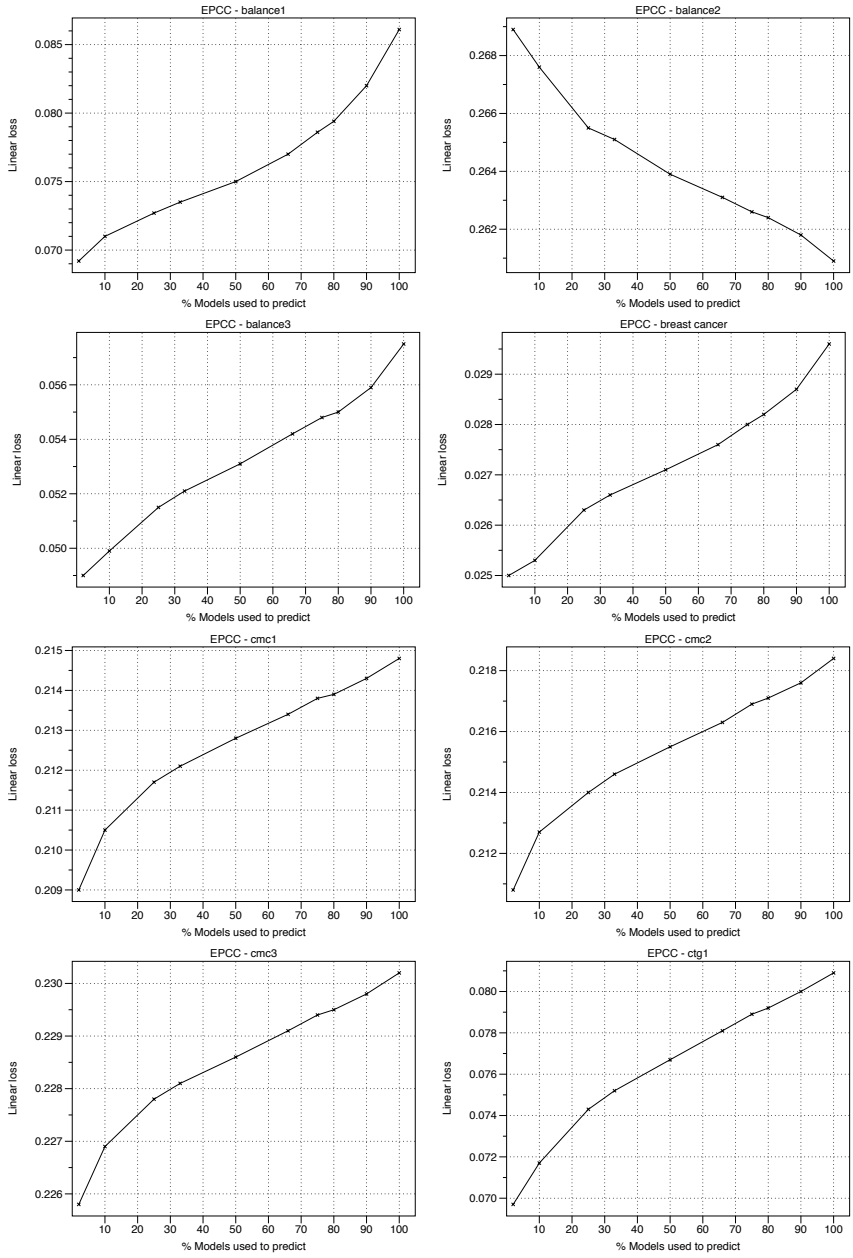

Figure 4.5: MAE scores selecting a different percentage of models using EPCC-ACC (part I)

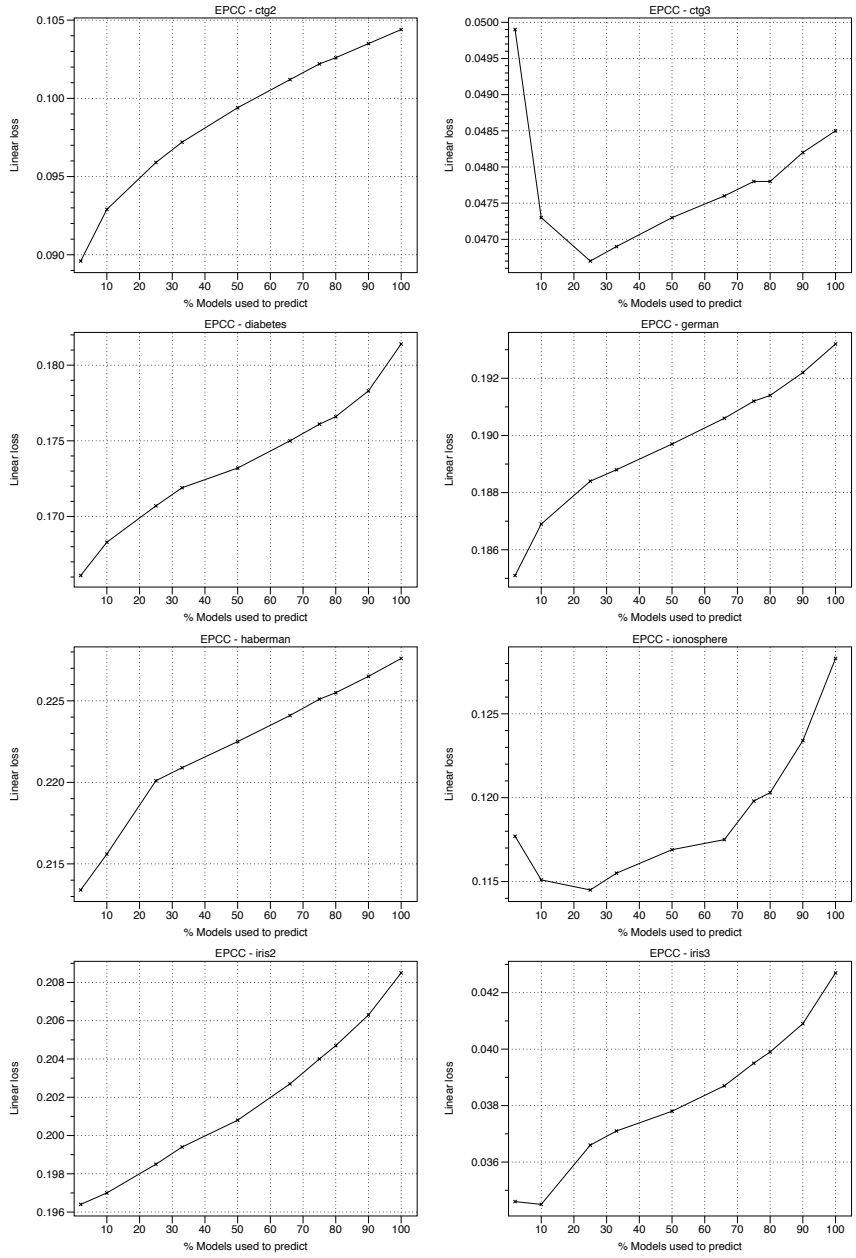

Figure 4.6: MAE scores selecting a different percentage of models using EPCC-ACC (part II)

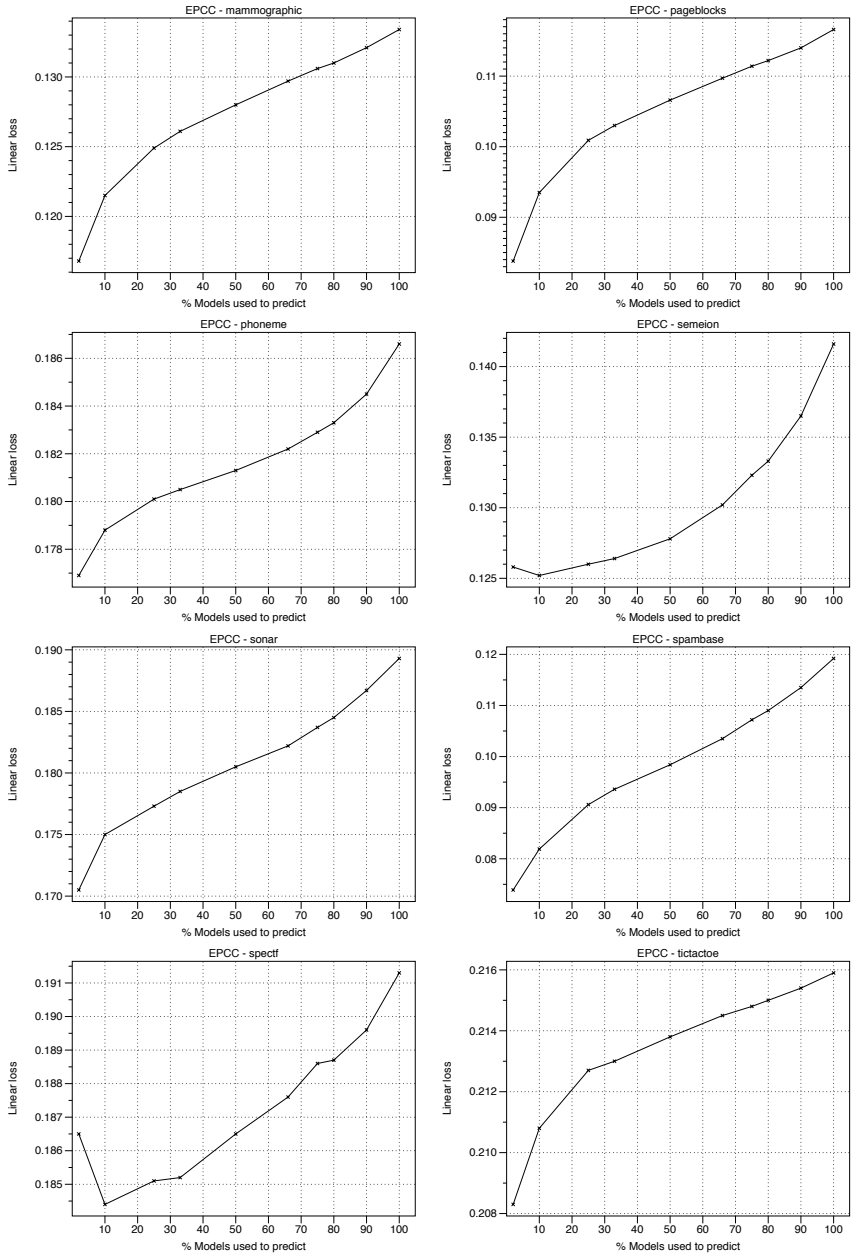

Figure 4.7: MAE scores selecting a different percentage of models using EPCC-ACC (part III)

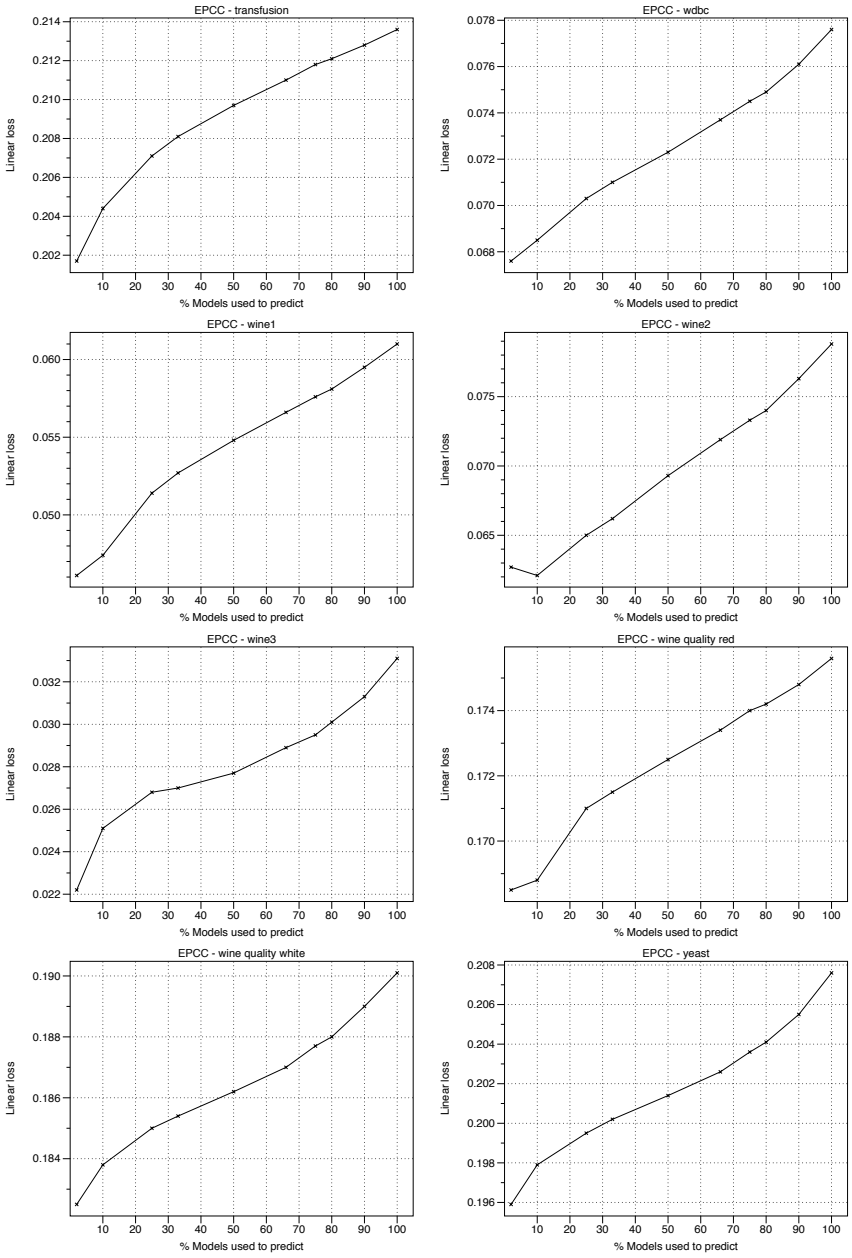

Figure 4.8: MAE scores selecting a different percentage of models using EPCC-ACC (part IV)

pageblocks, *sonar* and *spambase*) and its score is very close to the minimum in another 6 (*ctg2*, *diabetes*, *haberman*, *iris2*, *spectf* and *wine2*). Despite the behavior is somehow unstable (different patterns with no one clearly prevailing), we can identify a pattern that occurs in near half of the cases. We called it the "L" pattern: it starts with a bad result for the best model (2%), then the scores improve rapidly achieving the minimum around 30%, and then the curve remains almost flat from that point. This happens for approximately half of the datasets.

Figures 4.13-4.16 the scores of the EPAC methods using the MAX selection function. The performance of ALL is similar to the case of EAC ensembles: it is the best performer in 4 domains (*balance2*, *german*, *iris2* and *semeion*) and achieves very good scores in another 10 datasets (*balance3*, *cmc3*, *ctg1*, *diabetes*, *iris3*, *phoneme*, *wdbc*, *wine1*, *wine quality red* and *white*). However, here the "L" pattern appears more frequently. This implies that most of the models in an ensemble based on PAC quantifiers should be reasonably good. These results can be explained by the fact that PAC quantifiers should be much better than the corresponding PCC methods due to the correction procedure. Recall that EPCC and EPAC use the same models. Nevertheless, there are also a few domains in which ALL performs quite poorly, noticeably *ctg3*, *pageblocks*, *spampbase*, *tictactoe*, *transfusion* and *wine3*.

Finally, Figures 4.17-4.20 contain the results of the combination of EHDy and DS. Here, the "L" pattern is again present in more than half of the graphs. As we have observed, this kind of behavior tends to appear more often for the ensembles based on sophisticated quantifiers (AC, PAC and HDy) than for those based on the classify and count approach (CC and PCC).

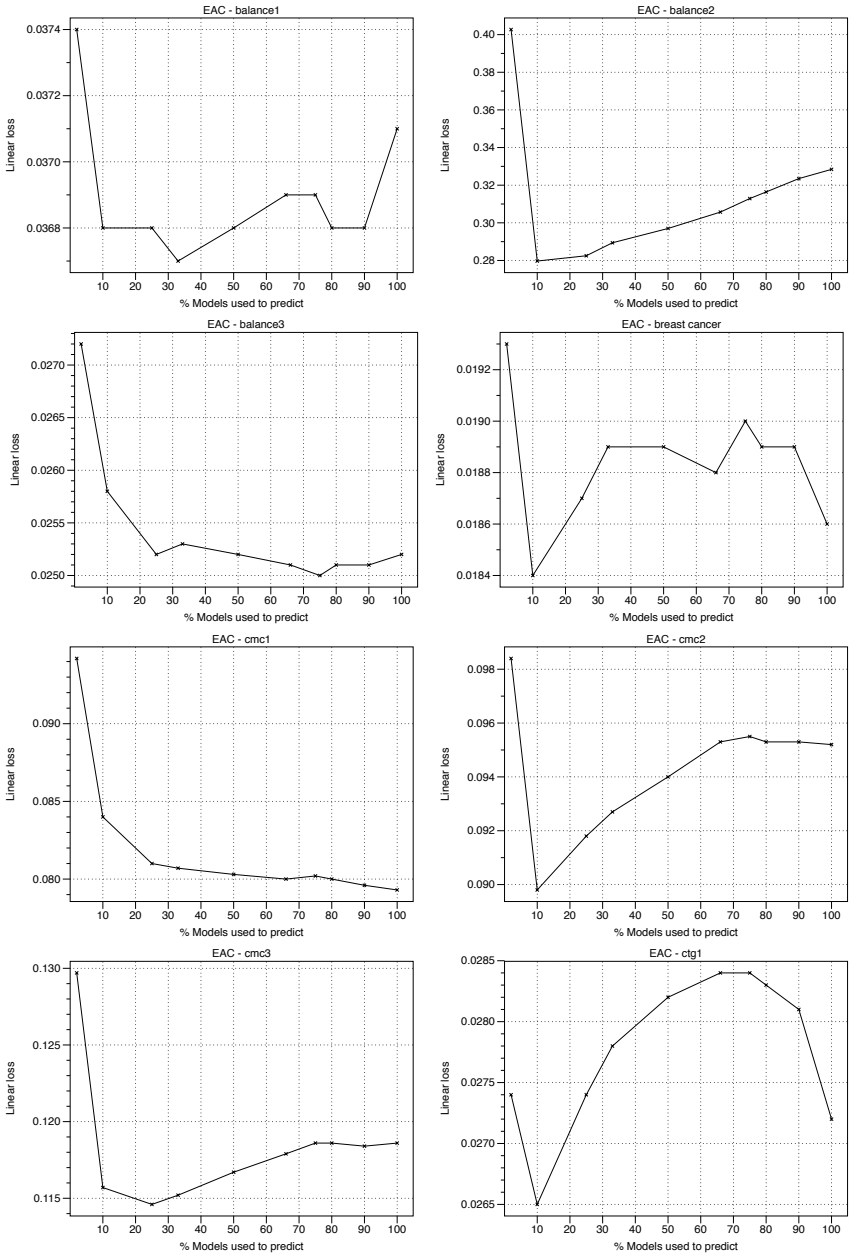

Figure 4.9: MAE scores selecting a different percentage of models using EAC-DS (part I)

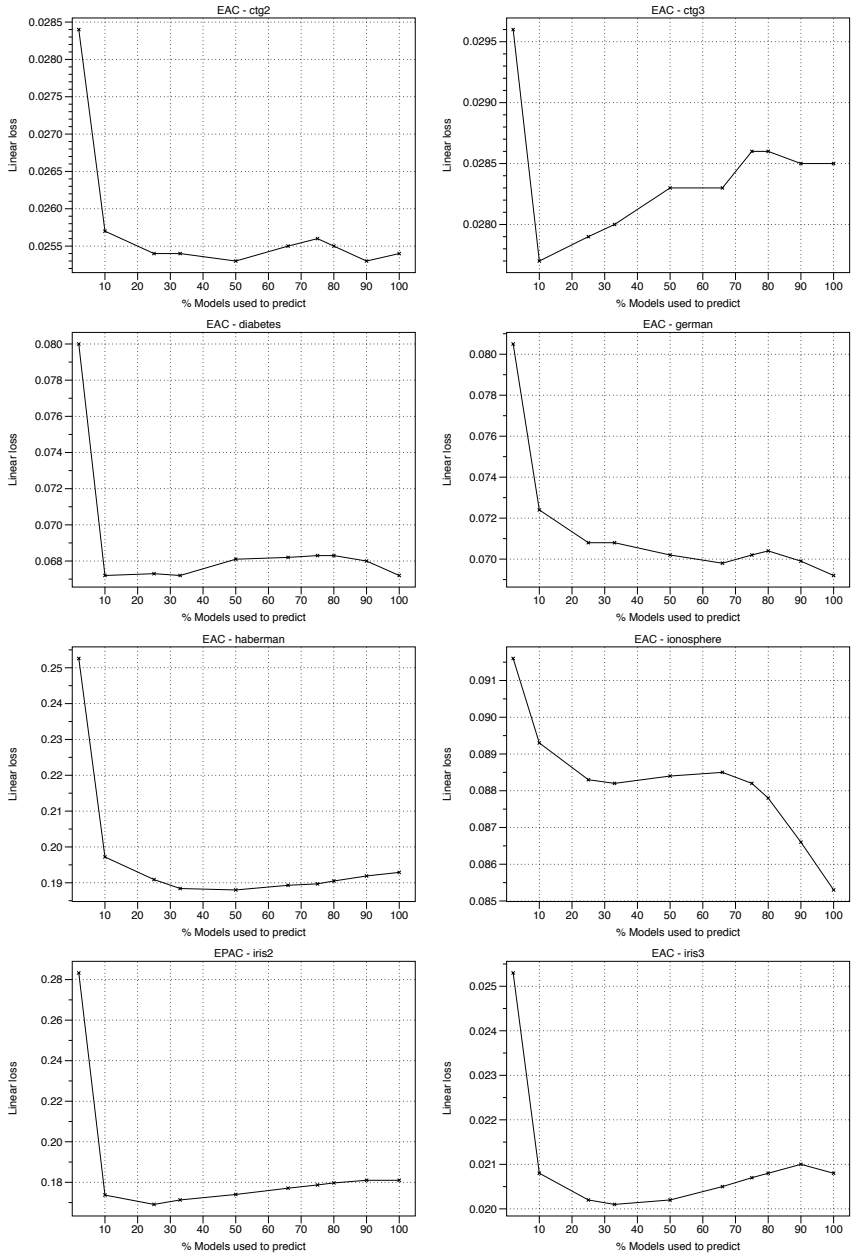

Figure 4.10: MAE scores selecting a different percentage of models using EAC-DS (part II)

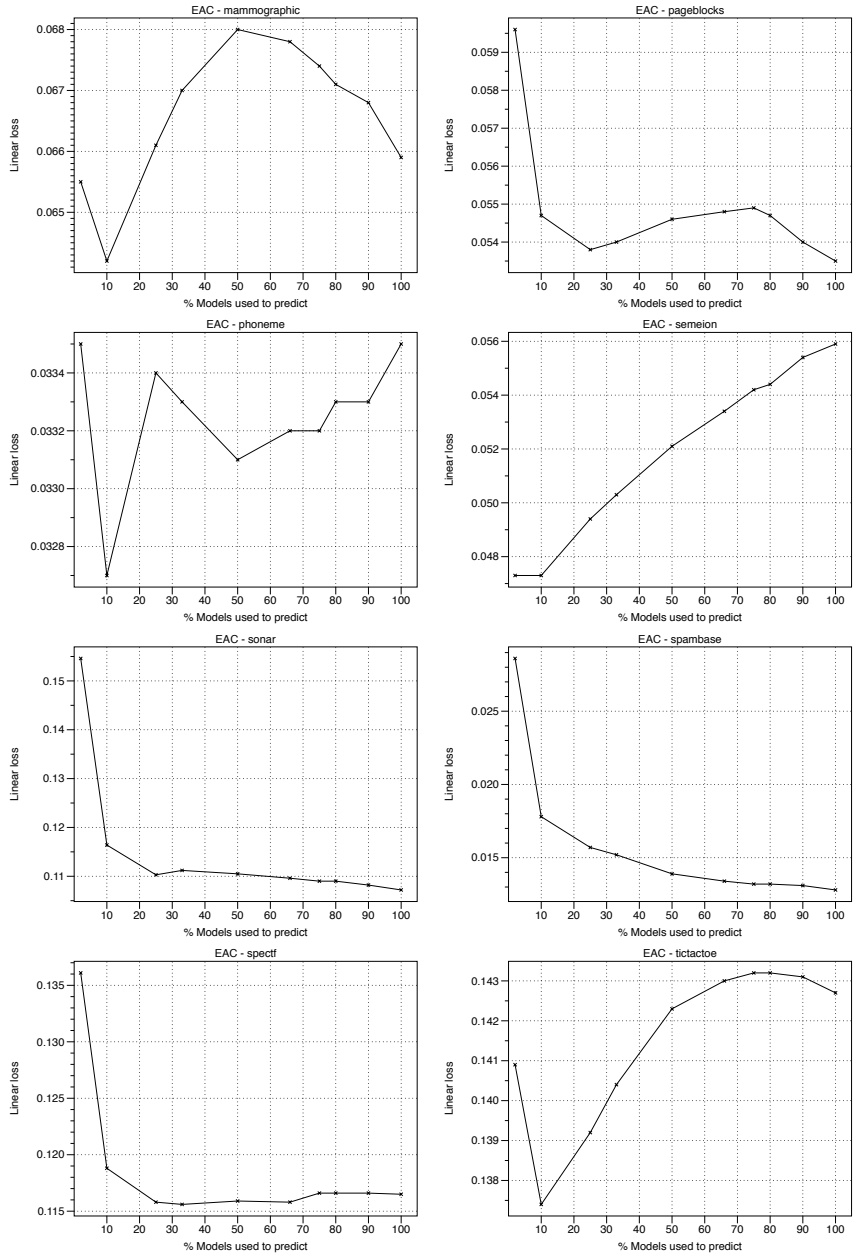

Figure 4.11: MAE scores selecting a different percentage of models using EAC-DS (part III)

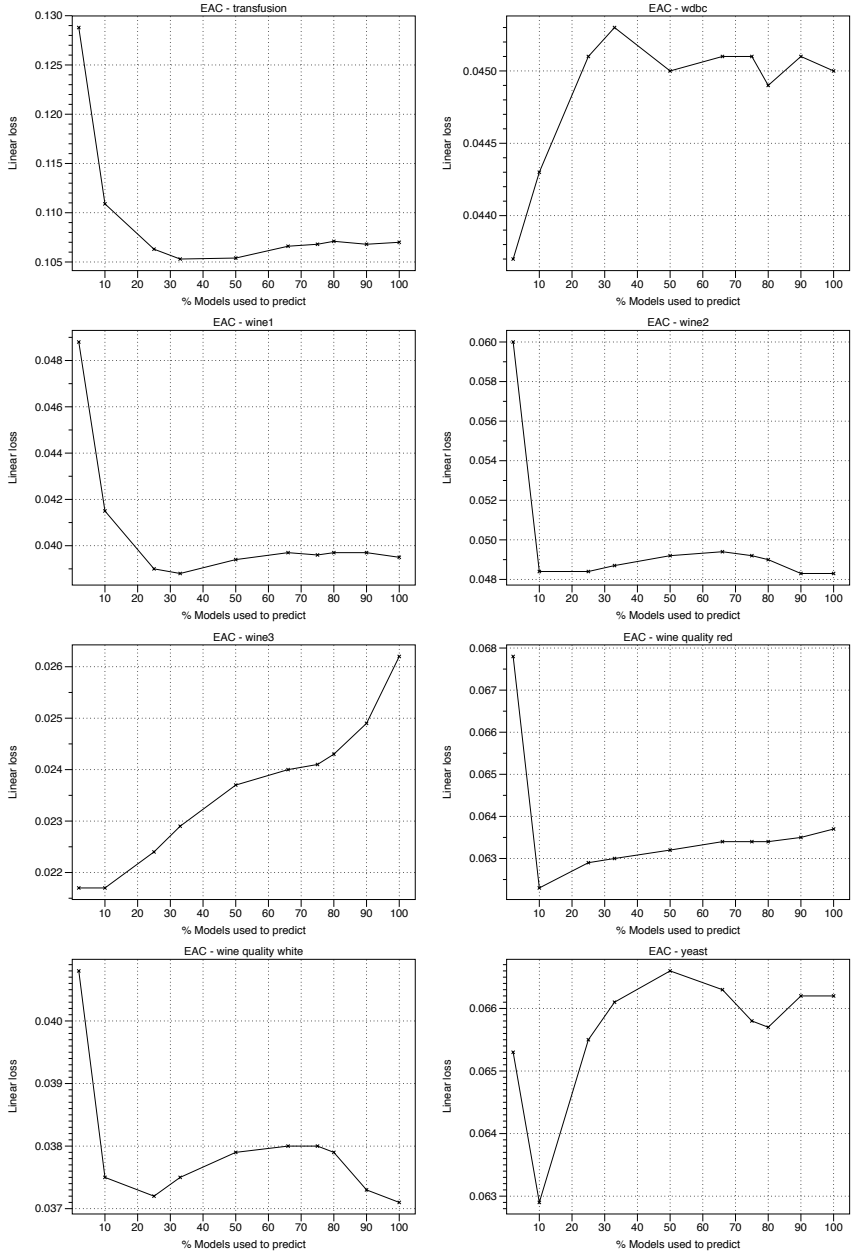

Figure 4.12: MAE scores selecting a different percentage of models using EAC-DS (part IV)

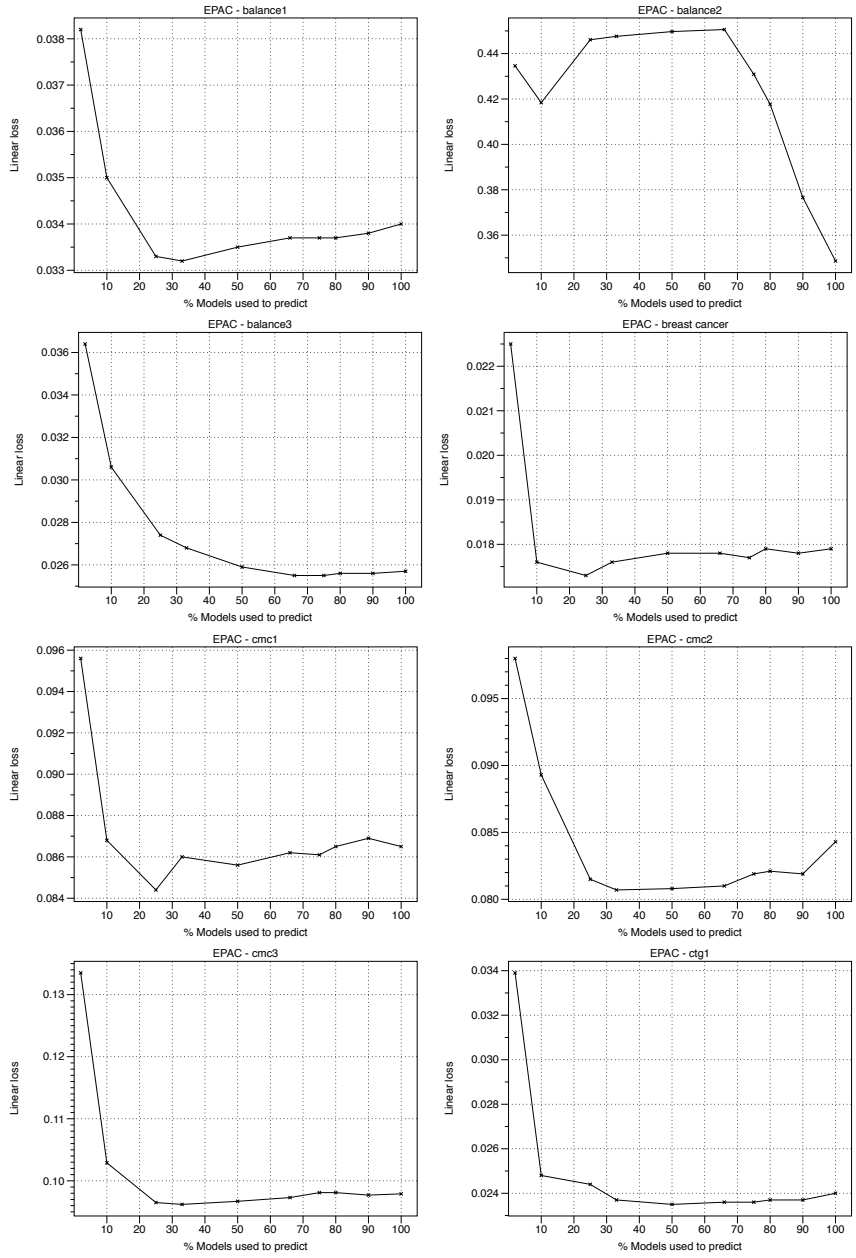

Figure 4.13: MAE scores selecting a different percentage of models using EPAC-MAX (part I)

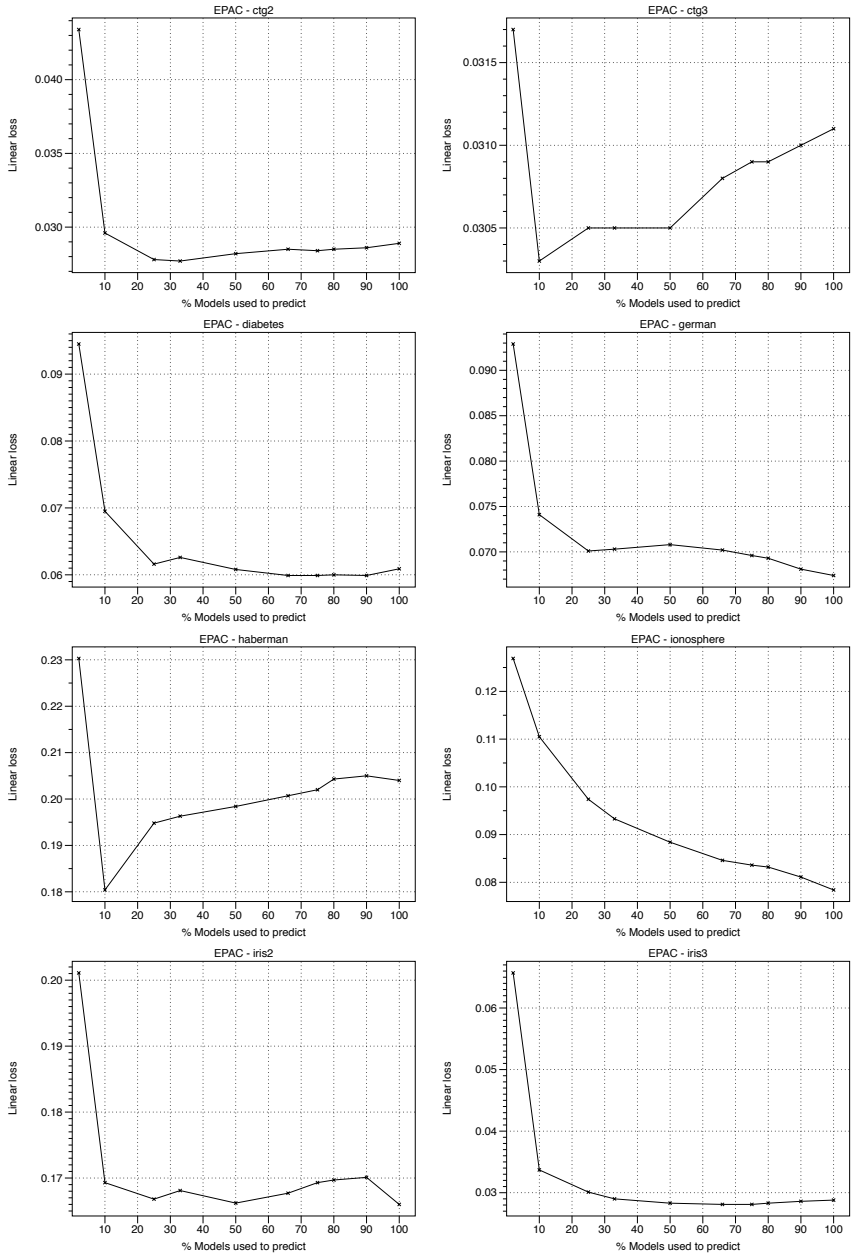

Figure 4.14: MAE scores selecting a different percentage of models using EPAC-MAX (part II)

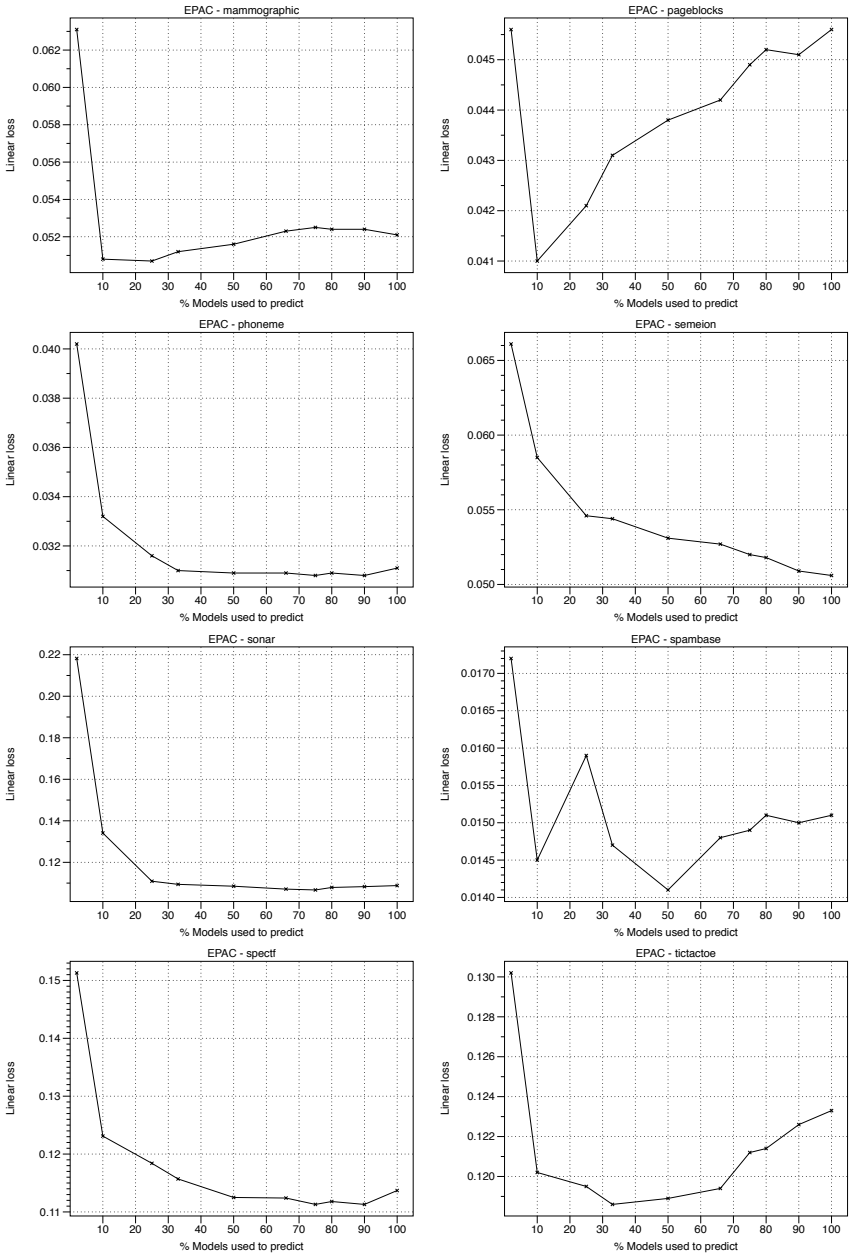

Figure 4.15: MAE scores selecting a different percentage of models using EPAC-MAX (part III)

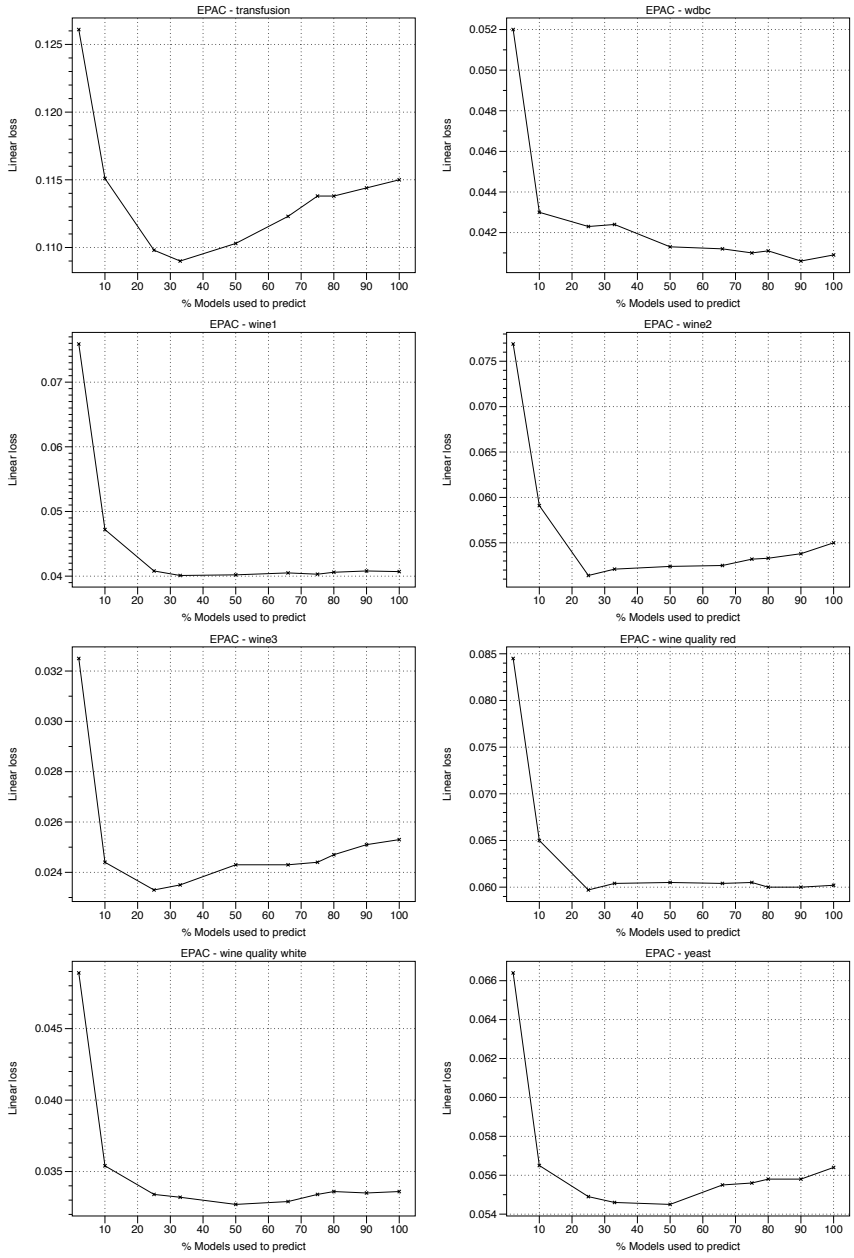

Figure 4.16: MAE scores selecting a different percentage of models using EPAC-MAX (part IV)

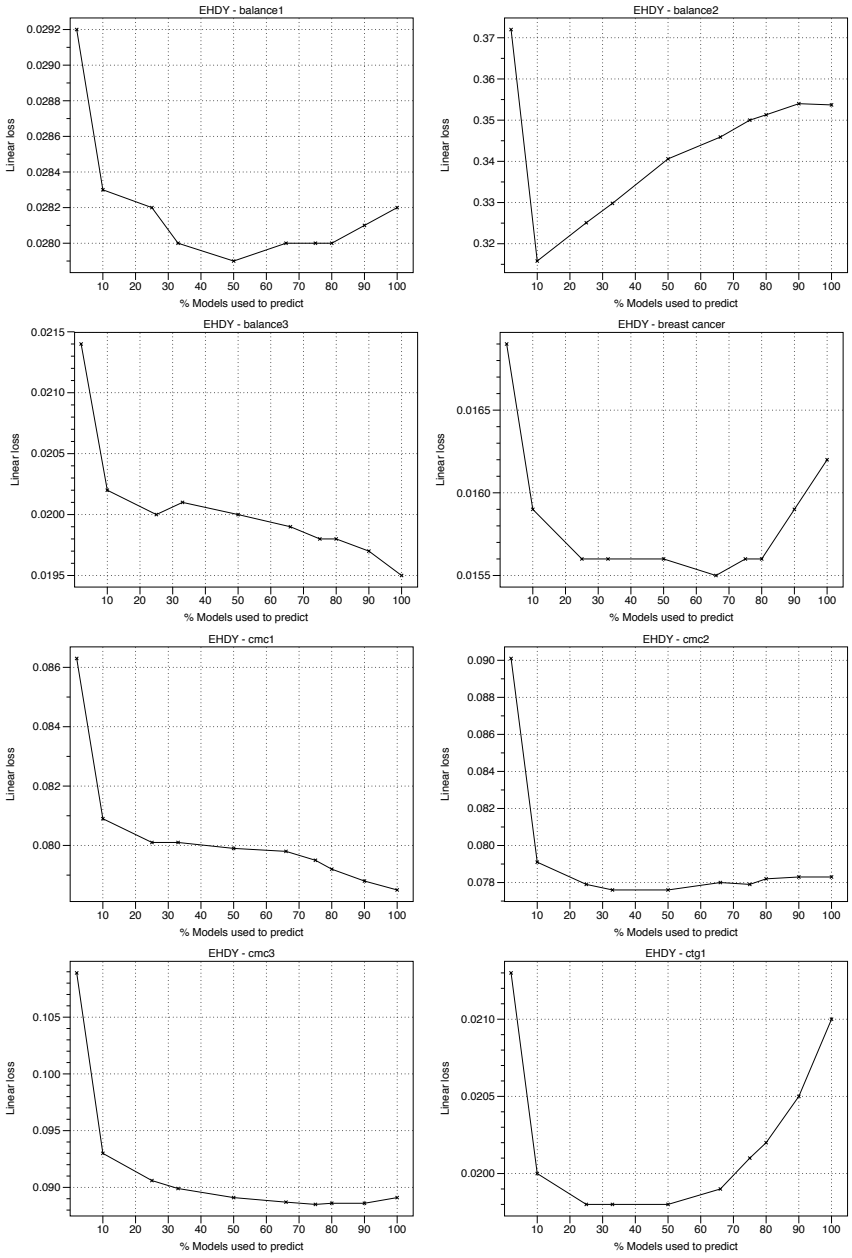

Figure 4.17: MAE scores selecting a different percentage of models using EHDy-DS (part I)

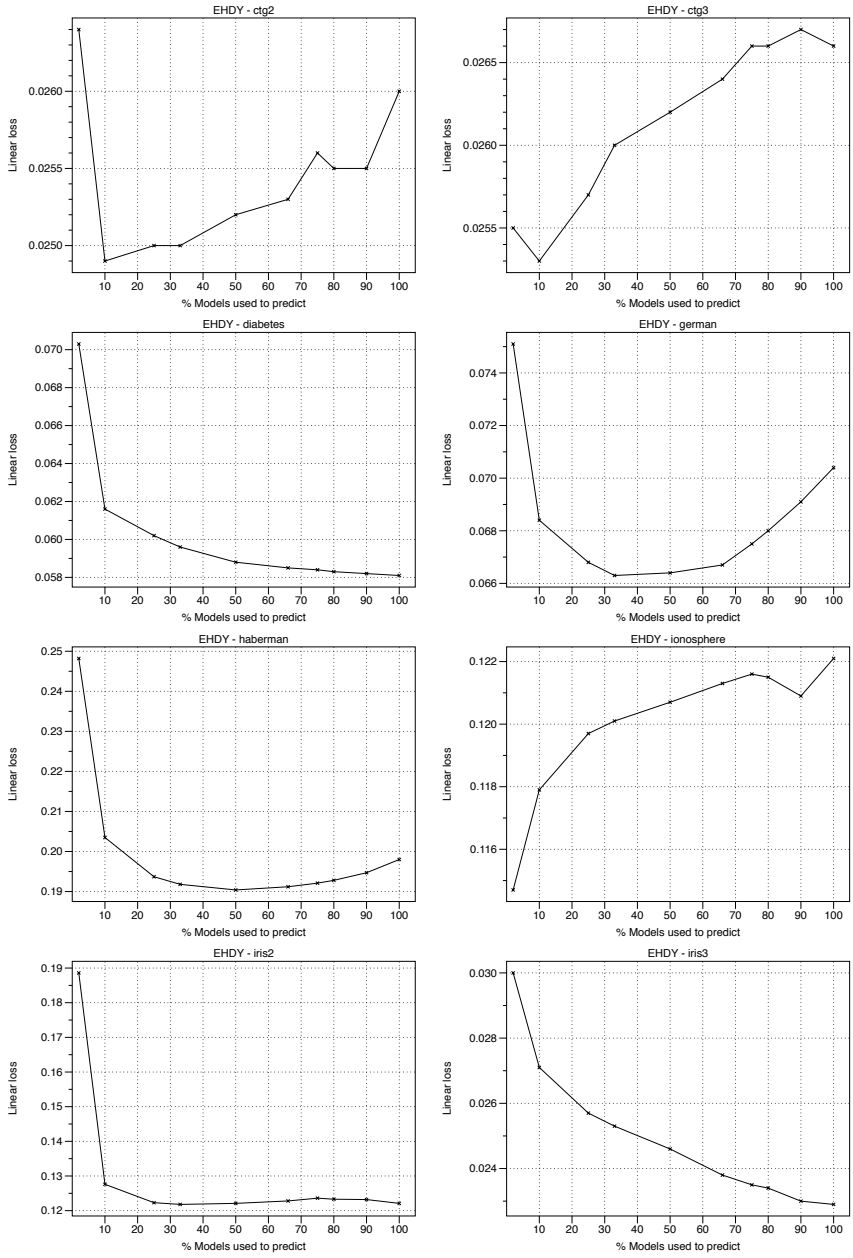

Figure 4.18: MAE scores selecting a different percentage of models using EHDy-DS (part II)

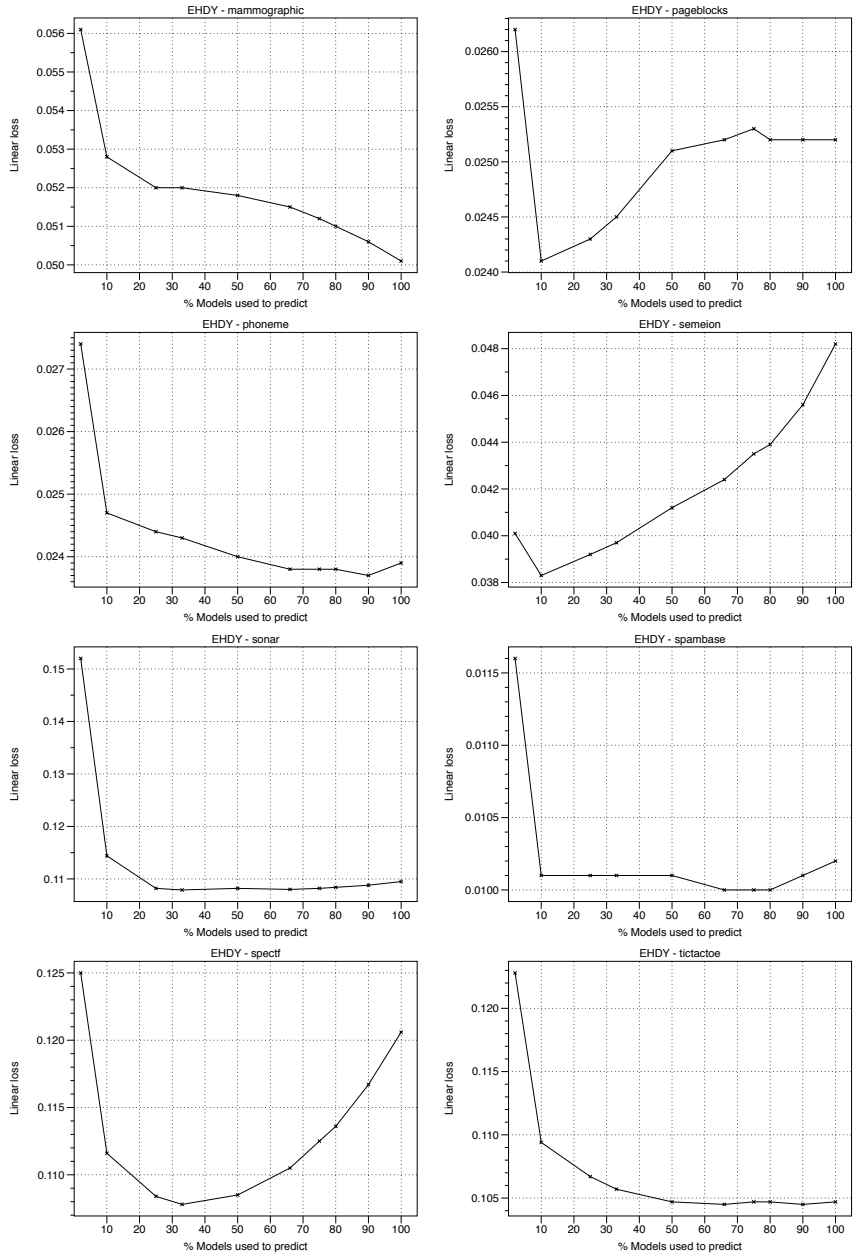

Figure 4.19: MAE scores selecting a different percentage of models using EHDy-DS (part III)

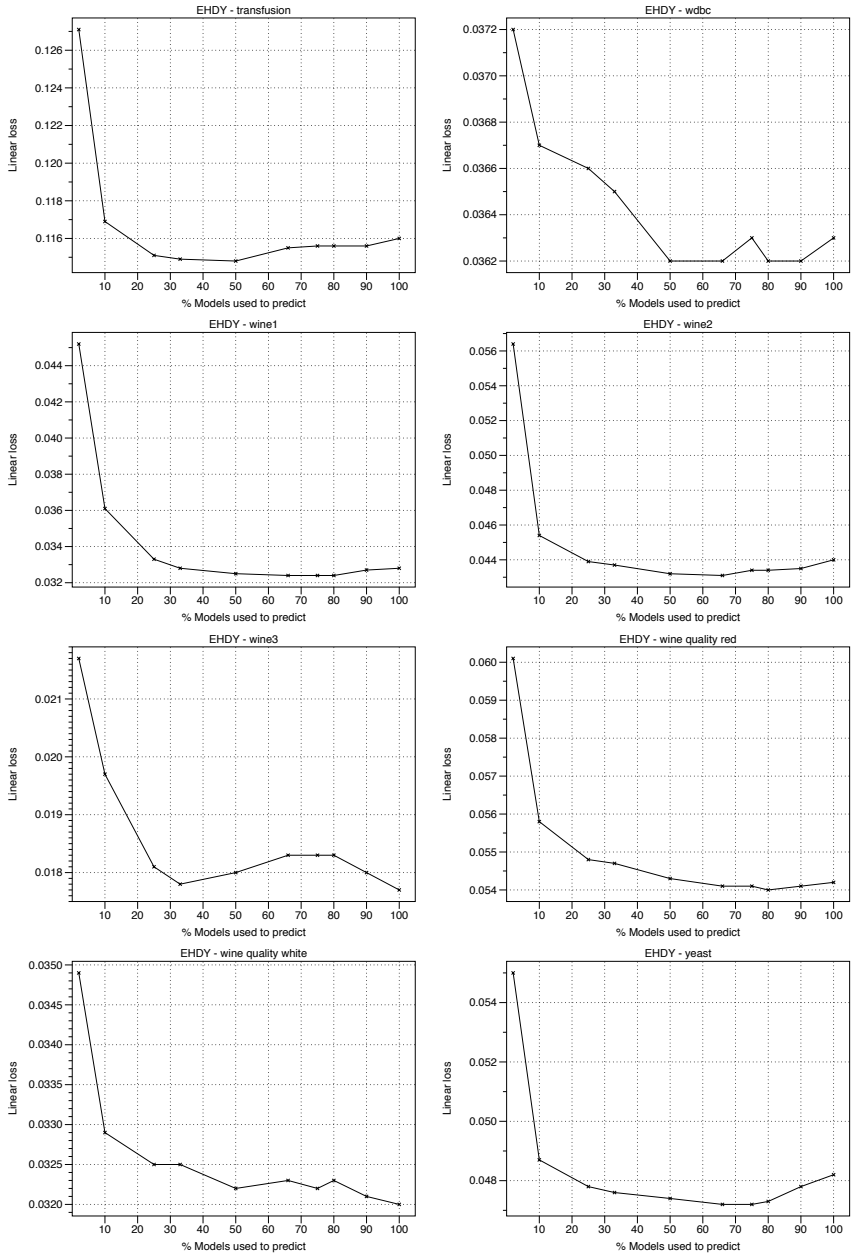

Figure 4.20: MAE scores selecting a different percentage of models using EHDy-DS (part IV)

Chapter 5

Applying ensembles of quantifiers to the sentiment analysis problem

In Chapter 3 we introduced the idea of ensemble learning being an adequate approach to solve those problems with some kind of shift in the distribution, as long as it is possible to determine which factors change in advance. We described our proposal, EoQ, and experimentally showed that in most cases it significantly outperforms other state-of-the-art quantifiers (the own ensemble base quantifiers), over a principally academic group of datasets. Then, in Chapter 4 we improved EoQ by designing specially quantification oriented selection measures. The idea was to select a subset of models according to a defined measure, instead of getting the final estimation by aggregating all the model predictions, like in original EoQ. The next logical step in this dissertation was to experimentally check the effectiveness of our propositions in a real-world problem and application. With that objective, we are applying EoQ to the sentiment analysis quantification problem.

Quantification of user generated opinions in social media constitutes an interesting task, not only in social science related fields, but also to economics and business. Practitioners have been routinely tackling these applications following a suboptimal approach, applying the so-called Classify & Count method, in which the examples are classified using an off-the-shelf classifier and then counted to compute the number of positive and negative opinions. As we have discussed in Chapter 2, this approach is usually outperformed by proper quantification methods.

This chapter is fully dedicated to compare the performance of EoQ, together with the selection measures we defined in the previous chapter, to the one of current state-of-the-art quantifiers in the context of sentiment analysis quantification [Pérez-Gállego et al., 2017b]. Given that it is a multi-class problem, in its more generic form the class variable domain is defined within $\{Positive, Neutral, Negative\}$, we have the opportunity to extend and to evaluate

EoQ in a multi-class environment. We have chosen the one-vs-all decomposition method (see Section 2.1.2) in order to adapt the binary quantification methods to the multi-class requirements, where each comment is labeled as *Positive*, *Negative* or *Neutral*. The last four *SemEval* competitions have been used as benchmarks, and we have found that ensembles behave as good as in other learning problems, combining robustness, stability and a competitive performance.

Next, we are introducing the general framework of sentiment analysis problems, and then we give a more detailed view on tackling this task via quantification. All the details regarding our experimental settings are presented afterwards, including the dataset specifications and our experiment design decisions. We finalize by discussing the obtained results.

5.1 Sentiment Analysis

During the last years we are assisting to an intense web transformation process. It is no longer a mere static information repository but a dynamic system in which users have become the main content contributors. They actively participate in sharing their opinions, thoughts and views about products, events and almost anything in social networks, forums, blogs, etc. With the latest advances in mobile technologies, users can actually interact anytime from anywhere; real time information has become a reality. All these mixture of social networks, discussion groups, forums and blogs are collectively called the *user-generated content* [Liu, 2010a]. It is a new source of information that noticeably represents the word-of-mouth behavior that is present on the Internet.

It has many practical applications and has a potential major value from both the user and business points of view. On one hand, knowing other user opinions is useful when having to take a decision in our daily life. We can learn from other user experiences who have already faced the same situation or decision in the past (*wisdom of crowds*). On the other hand, it is an invaluable information source about user preferences and tastes. A key circumstance is that users on the Internet share their opinions by their own free will, without expecting anything in return and without coercion. Due to the large and diverse number of opinion sources, it appears the necessity of systems able to automatically discover, analyze and summarize the expressed sentiment in the so-called user-generated content. *Sentiment Analysis* grows out of this need.

Sentiment analysis is the computational study of people's *opinions*, *appraisals*, and *emotions* toward entities, events and their properties [Liu, 2010b]. The objective is to determine the text author's attitude, whether (s)he is giving an opinion about an entity/event, expressing an emotional state (s)he is in, or on the other hand trying to cause that state on the text reader. An opinion is considered as a *positive* or *negative* sentiment expressed by a user on a entity or event, by means

of a written commentary. Over the past few years this problem has attracted the attention of both, research in academia, and applications in business and industry. The principal reasons are that on one hand it is a very challenging problem from the research point of view (leading to a potential high number of publications), and on the other it offers a wide range of practical applications.

Sentiment classification can be divided into several specific subtasks [Esuli and Sebastiani, 2005]:

(i) Determining subjectivity, consists in separating texts with factual information from texts with opinions. It is a binary classification task with classes being whether *Objective* or *Subjective* [Wiebe, 2000; Wiebe et al., 2004; Yu and Hatzivassiloglou, 2003].

(ii) Determining polarity, from a subjective text, determine whether the expressed opinion is positive, negative or a mixed one [Pang et al., 2002; Turney, 2002; Dave et al., 2003]. It is a binary or multi-class classification task depending on the consideration of the neutral class. Each instance is labeled as *Positive* or *Negative*, and in some problems it is also considered the *Neutral* class.

(iii) Determining the strength of polarity, consists in quantifying the degree of the sentiment. It is a multi-class problem with classes usually following a range like: *weakly* positive, *mildly* positive or *strongly* positive. It is also called *affective* classification, when trying to predict users mood, such as happiness, sadness, kindness, sureness and so on [Subasic and Huettner, 2001; Grefenstette et al., 2004; Bollen et al., 2011].

5.1.1 Sentiment Quantification

The popularity of social networking and microblogging services, with platforms like Twitter, has made the task of automatically analyzing emotions in user generated content, named as sentiment analysis, a ubiquitous learning application, as we have discussed above. In some cases, sentiment analysis is tackled from the point of view of classifying individual comments. The main task in sentiment classification is to predict the polarity of a given text (e.g. a microblog post), that is, if the comment expresses a positive, a negative or a neutral opinion [Pang et al., 2002]. The interest of this application is that companies may analyze the reasons that make their customers happy, and even more importantly, the causes of their complaints. Despite this approach requires some kind of human intervention at the last part of the process, it has been applied in numerous applications, see for instance [Yu et al., 2013; Schumaker et al., 2012].

However, firms sometimes require another kind of surveys aimed at numerically measuring whether their products meet the consumer expectations or not. In

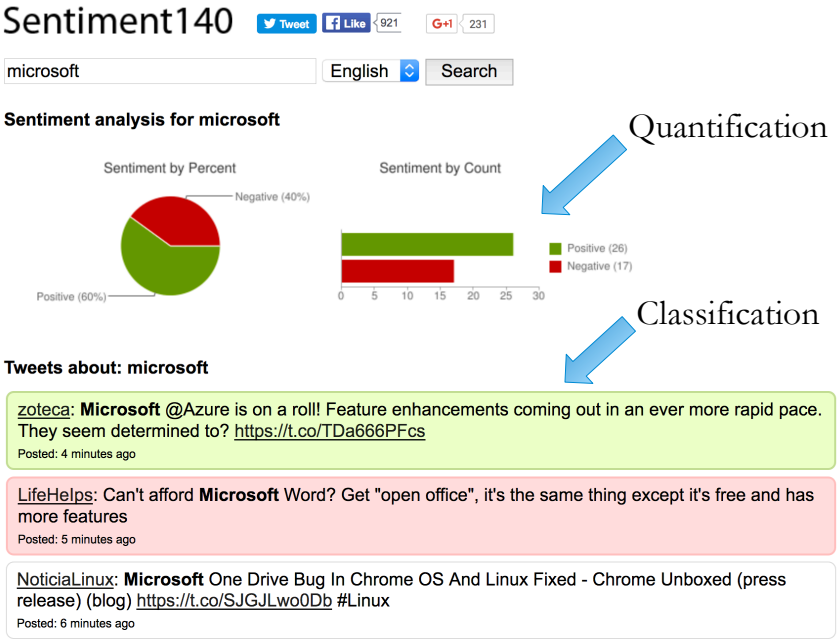

Figure 5.1: Classification vs Quantification: In the top, two graphs show the distribution of positives and negatives. In the bottom, the list of tweets are classified as positives (green), negatives (red) or neutral (white)

marketing, this concept is termed as *consumer satisfaction*, which is defined in the Wikipedia as "the number of customers, or percentage of total customers, whose reported experience with a firm, its products, or its services (ratings) exceeds specified satisfaction goals" [Farris et al., 2010]. Customer satisfaction provides companies with a useful metric allowing them to manage, monitor and, lately, to improve their businesses. Traditionally, this kind of rates have been computed through customer satisfaction surveys. But nowadays, consumer opinions can be obtained online, reducing costs and increasing speed to address consumer issues.

In this context, quantification learning seems the most appropriate approach to obtain models able to automatically measure rates related to consumer satisfaction by the processing of online comments. In fact, the goal of quantification is precisely to learn models that do not return predictions for each individual instance, but an aggregate estimate for a group of such examples. For instance, a company may perform a daily tracking of the acceptance of their products, obtaining a numerical estimate of the proportion of positive and negative comments that users have posted in social networks. The unit, in this case, is the set or a sample of all the comments published each day.

It seems obvious that sentiment quantification [Esuli and Sebastiani, 2010; Gao and Sebastiani, 2015a] fits better than sentiment classification for the requirements of applications that numerically measure consumer satisfaction. Figure 5.1 illustrates the difference between sentiment classification and sentiment quantification using an example taken from `www.sentiment140.com`. In the top, a pie graph and a bar graph show the distribution of positive (green) and negative (red) comments, representing a typical sentiment quantification prediction. The list of tweets related to the entity appears below, and each tweet is labeled as positive, neutral or negative, which is a sentiment classification task. Both tasks should be tackled separately because they require different, and specially devised, learning methods.

One may think that sentiment quantification can be solved using classification, by applying the Classify & Count (CC) approach (Section 2.4.1): comments are classified using a model learned with our favorite classification algorithm, and then counted to compute the percentage of each class. Actually, many practitioners solve sentiment quantification using the CC approach. This is motivated for two main reasons. First, quantification learning is almost unknown yet and many experts and users are unaware that proper quantification algorithms do exist. And secondly, when some practitioners face a quantification application, they have the misleading idea that quantification accuracy depends on the accuracy of a classifier and that both things are related. Or to state it in another way, if they obtain an accurate classifier, then the quantification estimates will be accurate as well. Quantification literature has proven that this conception is incorrect, or at least that genuine quantification algorithms can be more precise than just applying the CC approach.

In this chapter we analyze the applicability of EoQ for multi-class sentiment quantification. The idea is to extend the use of EoQ, discussed in Chapter 3, to the multi-class case using the well-known one-vs-all approach. To increase the exhaustiveness of this study, four different base quantification algorithms and three fusion strategies, introduced in Chapter 4, are considered. The resulting group of methods have been tested over datasets coming form *SemEval* (Semantic Evaluation) competitions (`http://aclweb.org/aclwiki/index.php?title=SemEval_Portal`).

SemEval is an ongoing series of challenges organized by the Association for Computational Linguistics in order to evaluate the improvement of computational semantic analysis systems. In the beginning, *SemEval* competitions were just focused on word sense disambiguation, but the workshops have been evolving and including new tasks each year, like semantic role labeling, coreference and sentiment analysis (*SemEval* 2007). During the last years, *SemEval* evaluations have mainly been centered on Twitter tasks motivated by the popularity of this microblogging service. Noticeably, in 2016 the first quantification challenge was proposed. Here, we employed the datasets from *SemEval* 2013 to *SemEval* 2016 to compare quantification algorithms. This chapter describes the experiments

Table 5.1: Summary of the main characteristics of *SemEval* competition datasets between 2013 and 2016

Competition	Training	Validation	Testing	Training Distribution			Testing Distribution		
				Pos	Neu	Neg	Pos	Neu	Neg
SemEval13	9684	1654	3813	37.1%	47.0%	15.9%	41.2%	43.0%	15.8%
SemEval14	9684	1654	1853	37.1%	47.0%	15.9%	53.0%	36.1%	10.9%
SemEval15	9684	1654	2390	37.1%	47.0%	15.9%	43.4%	41.3%	15.3%
SemEval16	6000	2000	2000	49.2%	35.1%	15.7%	49.7%	34.1%	16.3%

performed using EoQ over such *SemEval* competitions. The main goal is to compare the performance of single quantifiers and their counterpart versions based on EoQ in the context of sentiment quantification.

5.2 Experimental setting

The rest of the chapter is organized as follows. First, we describe the settings used in the experiments. This includes a complete description of: *SemEval* datasets, quantification algorithms compared and the processes to select the learning parameters of such algorithms. Finally, the results are presented in terms of two different quantification performance measures, KLD (2.14) and MAE (2.15), analyzing several aspects.

5.2.1 Datasets

The datasets used in these experiments correspond to *SemEval* competitions, concretely, those celebrated between 2013 and 2016. The set of tweets composing each dataset can be downloaded using Twitter's tools, but probably some of them may be unavailable now, maybe because the user account has been canceled or some tweets have been deleted. For that reason, we preferred those dataset versions that were previously employed in [Gao and Sebastiani, 2015a][1]. These versions have three advantages:

(i) they are complete, no tweet is missing,

(ii) tweets are already represented in a vectorial form, and

(iii) it is easier to reproduce the experiments.

[1]These datasets can be download from `http://alt.qcri.org/~wgao/data/SNAM/tweet_sentiment_quantification.zip`

We used exactly the same representation because our goal is not to propose a new one that could lead to better scores, but to compare quantification approaches. Besides, this vectorial representation, introduced in [Kiritchenko et al., 2014], was used by the winners of *SemEval* 2013 and *SemEval* 2014.

All the four datasets have three classes (positive, negative and neutral comments), and they are split into three subsets: training, validation and testing. The validation part was used to adjust the learning parameters of the algorithms. After that, the model is trained with the combination of the training and validation subsets. Finally, the score for each algorithm is obtained using the testing subset. Table 5.1 contains the main characteristics of the four datasets: the number of examples of each subset and the class probability distribution of the training and testing subsets. Also notice that SemEval13, Semeval14 and Semeval15 share the same training and validation parts, but they differ in the testing set. As we can observe, the distribution drift between the training set and the testing set is reduced, except in the case of SemEval14 dataset. All the details regarding these datasets and the competitions can be respectively found in [Nakov et al., 2013; Rosenthal et al., 2014, 2015; Nakov et al., 2016].

5.2.2 Experimental design

In the experiments reported here we analyzed eight multi-class quantification algorithms, all of them implemented using the one-versus-all approach discussed in Section 2.1.2, Chapter 2. The difference among them is the underlying binary quantifier that was employed. Four binary quantification algorithms were considered: CC, PCC, AC and PAC, and their corresponding ensemble versions, denoted as ECC, EPCC, EAC and EPAC, respectively. Moreover, the ensemble algorithms were implemented using the three different combination strategies described in Section 4.1, Chapter 4:

(i) ALL that combines the predictions of all the models,

(ii) ACC that selects the best models according to their KLD scores, and

(iii) MAX that combines the models that maximize the denominator of the correction equations depending on the base binary quantifier used: $tpr - fpr$ (2.17) for ECC and EAC, and $TP^{pa} - FP^{pa}$ (2.19) for the probabilistic-oriented methods (EPCC and EPAC).

50% of the models were selected for ACC and MAX selection measures, the same value proposed in Section 4.2.1.

The number of models to build the ensembles was set to $m = 30$. In order to generate the training samples for the ensemble models, the prevalence of the

Table 5.2: KLD and MAE scores for all the compared algorithms. Each group contains a single multi-class quantifier and their ensemble versions. Best scores of each group are in bold and * means that this is the overall best score for that dataset

Algorithm	Loss	Semeval13	Semeval14	Semeval15	Semeval16
CC	MAE	0.058865	0.069524	0.102074	0.014175
	KLD	0.017470	0.024149	0.053693	0.001670
ECC-ALL	MAE	0.068942	0.075188	0.100867	0.051114
	KLD	0.021763	0.027255	0.051862	0.020237
ECC-ACC	MAE	0.064791	0.069864	**0.096536**	0.040939
	KLD	0.018968	0.023442	**0.047160**	0.013165
ECC-MAX	MAE	**0.058316**	**0.068850**	0.099722	**0.011442***
	KLD	**0.016759**	**0.023341**	0.051172	**0.001142**
PCC	MAE	0.039744	0.063110	0.074472	**0.016267**
	KLD	0.007992	0.019126	0.027624	**0.002209**
EPCC-ALL	MAE	0.046741	0.065015	0.073954	0.048341
	KLD	0.010016	0.021350	0.026795	0.015448
EPCC-ACC	MAE	**0.034001***	**0.061446**	**0.070910***	0.018432
	KLD	**0.006216***	**0.017619***	**0.025855***	0.002501
EPCC-MAX	MAE	0.037999	0.064884	0.074177	0.019734
	KLD	0.007560	0.019873	0.027803	0.002409
AC	MAE	0.082287	**0.058669**	**0.105278**	0.048955
	KLD	0.031335	**0.023954**	**0.058398**	0.014290
EAC-ALL	MAE	0.079243	0.073523	0.109456	0.021778
	KLD	0.028510	0.026486	0.062021	0.002140
EAC-ACC	MAE	**0.078635**	0.070563	0.109880	0.018944
	KLD	**0.028243**	0.024235	0.062571	0.001910
EAC-MAX	MAE	0.079026	0.074877	0.108600	**0.011512**
	KLD	0.028476	0.028108	0.061039	**0.000718***
PAC	MAE	0.077845	**0.055166***	0.105354	0.054161
	KLD	0.028070	0.028634	0.057657	0.019889
EPAC-ALL	MAE	0.065677	0.068032	0.090424	0.015329
	KLD	0.019701	**0.024476**	0.040986	0.001271
EPAC-ACC	MAE	0.064984	0.068048	0.090240	0.016565
	KLD	0.019338	0.026141	0.040797	0.001413
EPAC-MAX	MAE	**0.064579**	0.069276	**0.089810**	**0.012387**
	KLD	**0.019064**	0.025504	**0.040407**	**0.000793**

positive class was uniformly selected in the range $[p - 0.1, p + 0.1]$, that is $\pm 10\%$ of the actual prevalence p of the training set. The size of each sample was always equal to the size of the training set and the examples were selected using random sampling with replacement, as it was explained in the previous chapters.

Taking into account that PCC and PAC require probabilistic classifiers, we decided to use probabilistic classifiers for all binary quantifiers. This decision ensures that the differences between CC, PCC, AC and PAC are not due to the underlying classifier, because all of them use exactly the same classifier. The same occurs for all ensemble methods. This means that, for instance, EAC-ALL contains exactly the same classifiers as EPAC-ALL, so the differences between them are only due to the use of AC or PAC quantifiers, and in the comparison between EAC-ALL and EAC-MAX the differences are due to the combination strategies.

The probabilistic classifier employed was Logistic Regression [Fan et al., 2008] instead of using Support Vector Machines like in [Gao and Sebastiani, 2015a]. This decision was motivated by the fact that the implementation of LibLinear provided in [Fan et al., 2008], led to faster training times for these datasets than LibSVM [Chang and Lin, 2011], which is an important aspect for training EoQ. The regularization parameter (C) was selected through a search in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$ optimizing the KLD over the validation set. This same policy was employed in [Gao and Sebastiani, 2015a]. Besides, the positive class and the negative class were balanced ($-w$ parameter in LibLinear [Fan et al., 2008]) to obtain good classifiers when dealing with imbalanced cases. The estimates for (tpr, fpr) and (TP^{pa}, FP^{pa}) for AC and PAC-based methods were obtained using CV10x1 (10 folds and 1 repetition) over the training data.

5.3 Experimental results

Table 5.2 shows the KLD and MAE scores of all the algorithms studied. Our main interest is to compare single multi-class quantifiers with their corresponding ensemble versions. In such analysis, we can observe that EoQ algorithm obtain better results than their counterpart single quantifier in 25 out of 32 cases. Only in the cases of PCC for SemEval16, AC for SemEval14 and SemEval15 and PAC for SemEval14 (only in terms of MAE), the single quantifier is better than all the ensemble versions. This result is well known in the literature, several studies conclude that ensembles usually perform better than individual models but not always, see for instance [Fumera and Roli, 2005].

Analyzing each group separately, we see that for CC-based methods, ensemble versions are always better than CC but the differences are small. Besides, the scores of this group are worse than those of the rest of groups in general; this is a common result of quantification papers in which CC is usually outperformed by other quantifiers, see for instance [González et al., 2016b]. However, ECC-MAX

obtains the best overall MAE score in SemEval16 dataset. Actually, this ensemble algorithm seems to be the best one in this group, winning in 3 datasets. Comparing combination strategies, ACC and MAX clearly outperform the baseline approach (ALL).

PCC methods attain the best results. In fact, EPCC-ACC is the overall winner in 5 out of the 8 cases, namely, in SemEval13 and Semeval15 for both measures and also in SemEval14 for MAE. The comparison between ensembles and single quantifiers is also in favor of ensemble algorithms with 3 wins and 1 loss. These results are particularly significant because the single quantifier PCC was the winner in SemEval13 and Semeval15 and ranked second in SemEval14 in the study presented in [Gao and Sebastiani, 2015a]. As it occurs in the previous group, ACC and MAX perform better than ALL.

The group of AC-based quantifiers presents mixed results. On the one hand, the AC method outperforms ensemble algorithms in two of the datasets, SemEval14 and SemEval15. But on the other hand, the opposite result is obtained in SemEval13 and SemEval16. Nevertheless, the scores of this group are clearly worse than those of PCC-based methods, thus it seems that the AC approach does not perform particularly well in these domains. Only in Semeval16, EAC-MAX attains the best overall KLD score. The comparison ALL versus ACC and MAX is usually in favor of the latter approaches but the difference is smaller than in the previous groups.

Finally, PAC-MAX is the best performer in the PAC-based groups being the winner in all datasets except SemEval14, in which PAC (MAE) and EPAC-ALL (KLD) attain the best results, in this last case, it is also the best overall score. As it occurs in the other groups, ALL combination strategy is outperformed by the fusion strategies based on model selection.

In our opinion, one of the main issues of *SemEval* competitions, that should be considered in future editions, is that the adjustment of the learning parameters and the final performance are computed using just one testing set. This procedure is inadequate because quantification learning tasks demand several testing samples to accurately estimate the performance of a given algorithm. Thus, it may occur that a given method achieves worse results than other just due to a bad selection of the optimal values for the learning parameters (C in our experiments). Figures 5.2-5.5 allow to analyze this aspect. The aim of these figures is to compare the scores of the algorithms for the considered values of C, analyzing if the parameter selection process captures the optimal values, and also the robustness of the methods against over-fitting.

Figure 5.2 and Figure 5.3 depict the KLD scores for CC and PCC based methods, respectively. Notice that all the algorithms usually reach the optimal score (the minimum value) with the same value of C and the differences at that point are small. This is logical because they use the same quantification method, and in the case of ensemble methods, they are composed of exactly the same models.

However, the curves still show some interesting facts. First, the combination strategies based on selecting models, ACC and MAX, are more robust than the baseline strategy. This is particularly evident when the value of C is large, increasing the risk of over-fitting. In that cases, the performance of ACC and MAX algorithms are clearly better than that of CC, PCC and the ensembles using ALL strategy. Perhaps the best examples occur for SemEval15 datasets, in which the curves corresponding to ACC and MAX are below for almost all values of C.

Figure 5.4 and Figure 5.5 show the same graphs but for AC and PAC methods. In this case, the pictures are totally different; the behavior is less uniform and the methods present much more variations. The reason is due to the fact that correction equations (2.17) and (2.19) may produce unstable adjustments. The clearest examples are the graphs corresponding to SemEval16 in which AC produces poor scores for small values of C (usually the optimal ones for the rest of the methods). An interesting case happens in this same dataset, according to Table 5.2, EPAC-ALL obtains a better KLD score than EPAC-ACC, however, looking at both graphs the impression is that the latter seems a better method for this dataset. A similar situation occurs for SemEval13, AC is the worst algorithm in terms of KLD, but it is just due to a bad selection of the optimal value for C. Still, overall, it seems that ensembles versions, especially those based on ACC and MAX strategies, are the most stable algorithms, independently of the base quantifier.

Another issue regarding these ensemble methods has to do with the procedure to generate the samples used to train an EoQ. In the previous experiments, the training samples prevalence was uniformly selected in the range $[p - 0.1, p + 0.1]$, where p is the prevalence of the positive class in the training set. This means that, for instance, the range of the prevalences in SemEval13 dataset (see Table 5.1) is $[0.271, 0.471]$ because the prevalence in the training set is $p = 0.371$. Notice that the true prevalence of the testing set falls in that range. The same happens for other classes in Table 5.1. In real-world applications, testing prevalences may be totally unknown, thus EoQ are trained with larger ranges of prevalences, see evaluation experimental design in Sections 3.3.1 and 4.2.1, in an attempt of covering a reasonable number of situations. One may think that the previous experiments were somehow adapted taking into account the class probability distributions of the testing subsets. For that reason, and in order to get more insights about the actual behavior of the quantifiers studied, we performed an additional experiment to analyze the performance of EoQ when they are trained with a larger range of prevalences, concretely $[p - 0.2, p + 0.2]$. In the previous example, the range would be $[0.171, 0.571]$. The results of this experiment are in Figures 5.6-5.9.

Figure 5.6 and Figure 5.7 contains the comparison for CC and PCC based methods. The conclusion is that the optimal scores are similar, maybe a little bit better in same cases for the EoQ generated with $\pm 10\%$ of the actual prevalence. However,

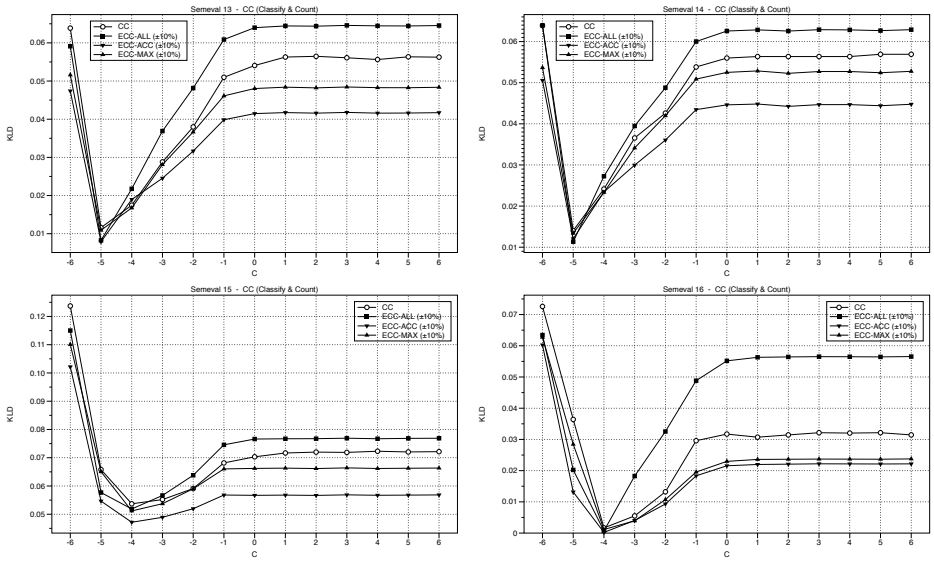

Figure 5.2: KLD scores for all the methods based on CC quantifiers when the training prevalences for the ensembles are uniformly selected in the range $[p - 0.1, p + 0.1]$ and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

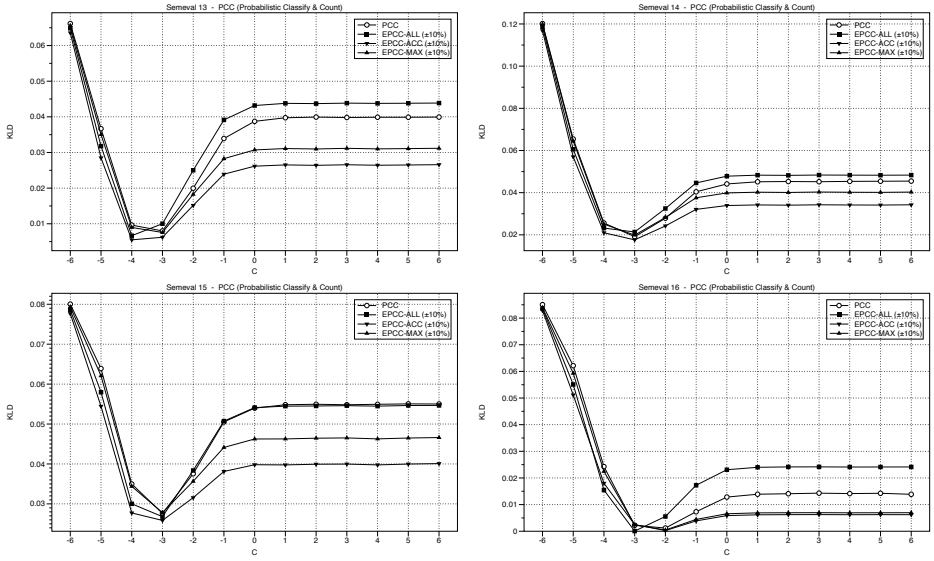

Figure 5.3: KLD scores for all the methods based on PCC quantifiers when the training prevalences for the ensembles are uniformly selected in the range $[p - 0.1, p + 0.1]$ and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

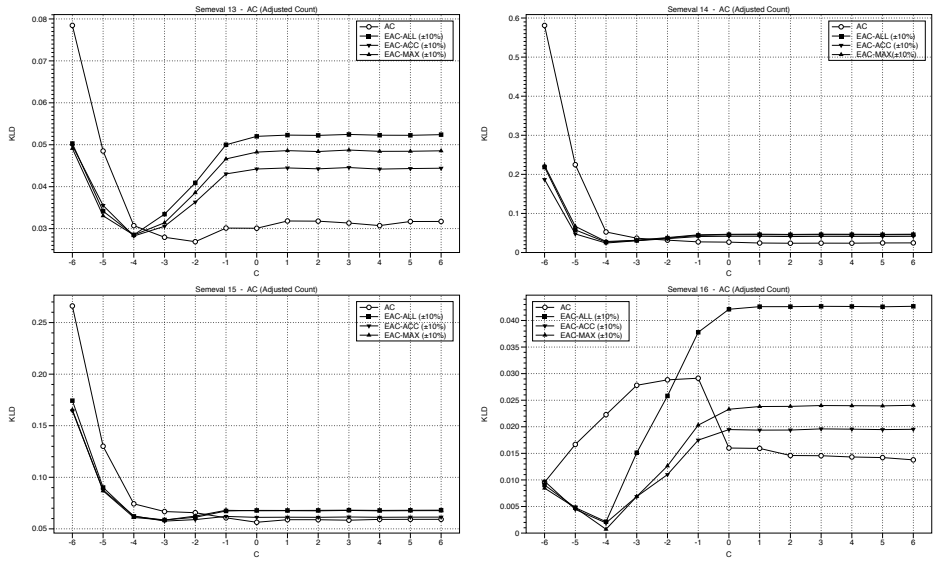

Figure 5.4: KLD scores for all the methods based on AC quantifiers when the training prevalences for the ensembles are uniformly selected in the range $[p - 0.1, p + 0.1]$ and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

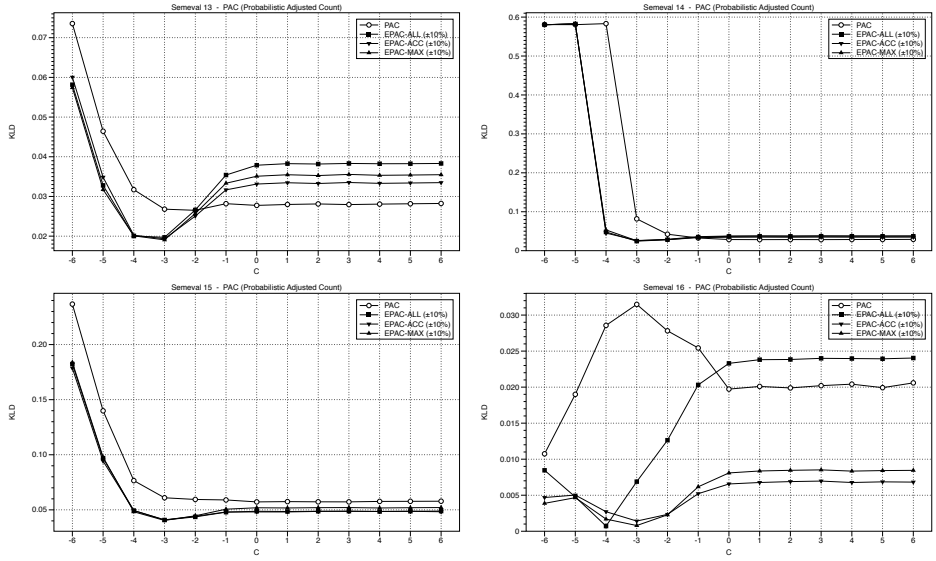

Figure 5.5: KLD scores for all the methods based on PAC quantifiers when the training prevalences for the ensembles are uniformly selected in the range $[p - 0.1, p + 0.1]$ and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

those generated with ±20% seems even more stable. For instance, it is quite remarkable their behavior for SemEval14, SemEval15 and SemEval16 because the curves are almost flat once the optimal value is reached. The larger level of diversity injected into the ensembles seems to imply that in these algorithms, it is easy to tune the learning parameters to obtain a reasonable model. Besides, ACC and MAX strategies also outperform ALL methods in line with the previous experiment.

Figure 5.8 and Figure 5.9 show the same graphs for AC and PAC based methods. The conclusions are quite similar. First, there are very small differences in the optimal results. This is especially true for SemEval14 and SemEval15 datasets, where all methods almost obtain the same scores for all values of C. The differences are larger for SemEval13 and SemEval16. EoQ generated with ±20% seems again more stable than those generated with ±10%, but the curves are not as flat as in Figure 5.6 and Figure 5.7.

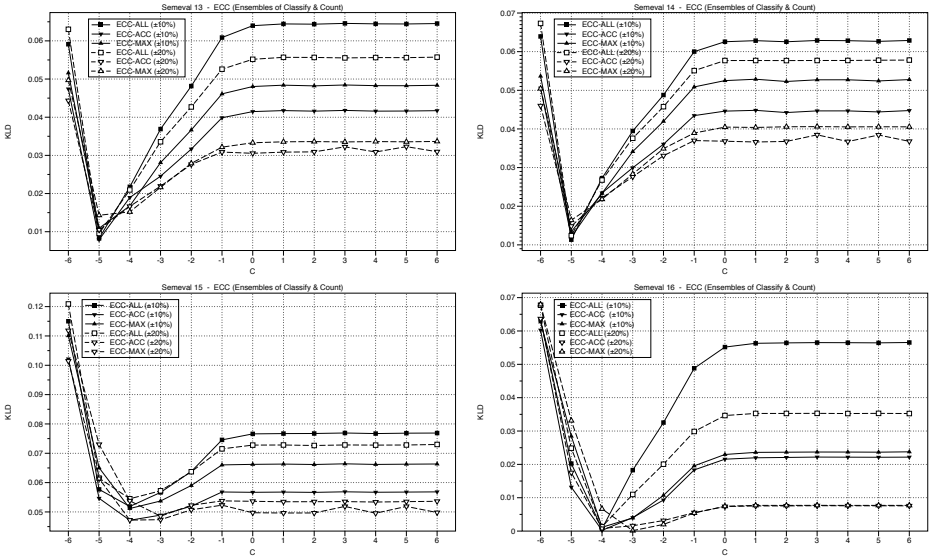

Figure 5.6: Comparison of KLD scores for EoQ based on CC algorithms when the training samples are generated with ±10% and ±20% of the actual prevalence and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

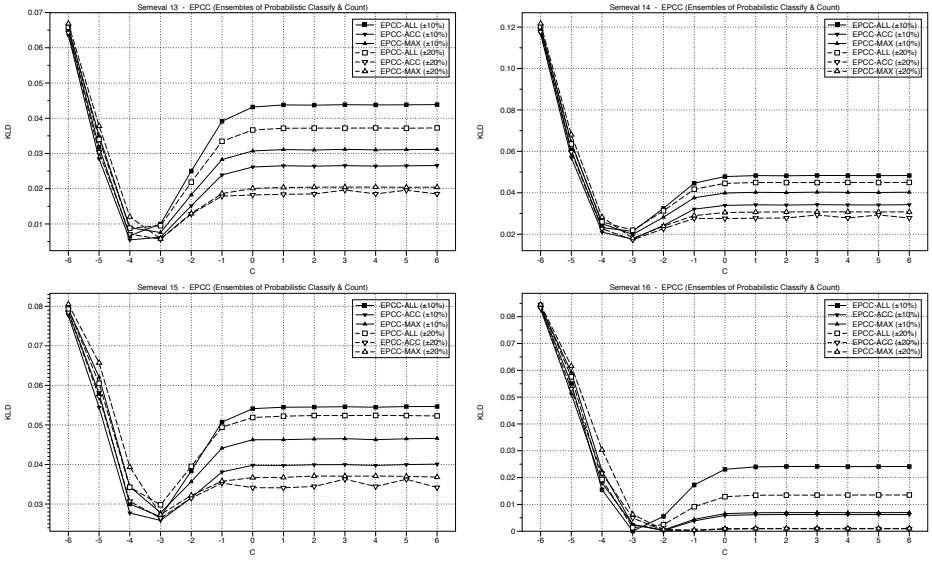

Figure 5.7: Comparison of KLD scores for EoQ based on PCC algorithms when the training samples are generated with ±10% and ±20% of the actual prevalence and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

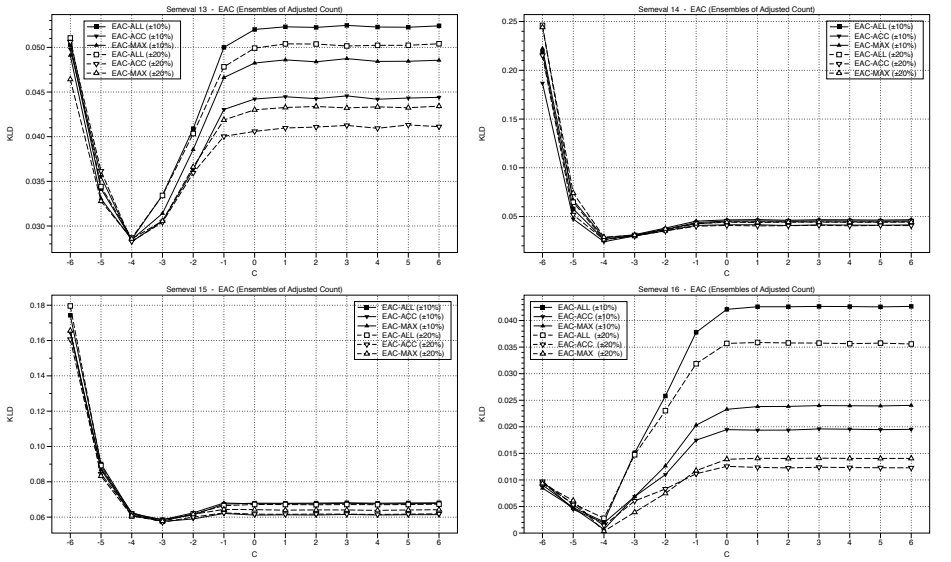

Figure 5.8: Comparison of KLD scores for EoQ based on AC algorithms when the training samples are generated with ±10% and ±20% of the actual prevalence and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

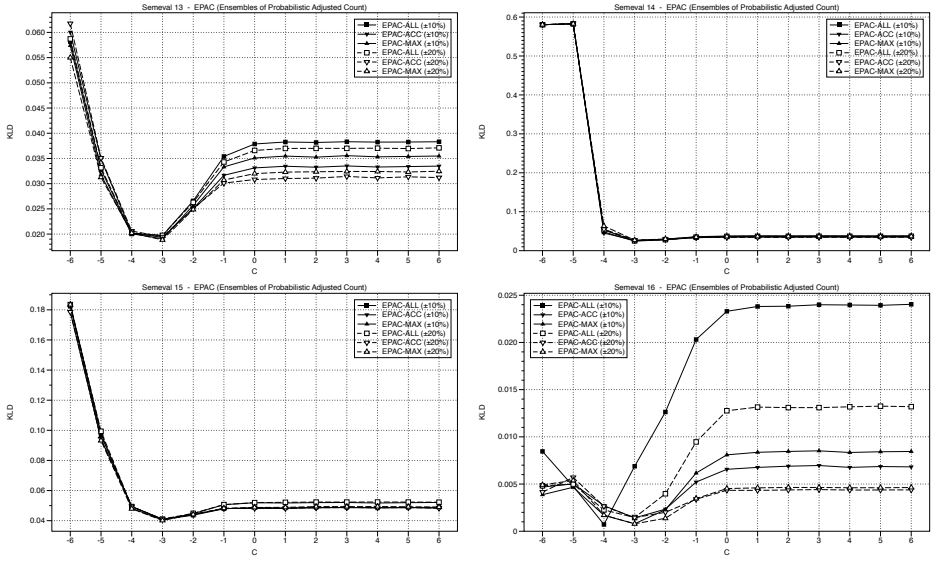

Figure 5.9: Comparison of KLD scores for EoQ based on PAC algorithms when the training samples are generated with ±10% and ±20% of the actual prevalence and C takes the values in $\{10^e : e \in \{-6, -5, \ldots, 5, 6\}\}$

Chapter 6
Conclusions

This chapter summarizes the main contributions of this dissertation and points out some future lines of research.

6.1 Conclusions

In this work we have studied how ensembles behave in a problem characterized by the assumption of data distribution changing between training and test phases. Our core idea is to take advantage of that assumption and to use it in order to appropriately introduce diversity into the ensemble; we generate different training samples with each one representing a particular expected distribution change. First, we have experimentally applied our idea to the binary quantification problem, which is characterized by class prevalence $\mathbf{P}(y)$ changing, but $\mathbf{P}(\boldsymbol{x}|y)$ remaining constant. Training samples are generated under this assumption, and as a result, the ensemble meta model is better suited for dealing with unseen prevalence test sets. Experimental results over benchmark datasets in Chapter 3 demonstrate that the ensemble quantifier adapted versions outperform its original counterparts.

One of the major contributions of this work is to propose a reasonable approach for using ensembles in quantification learning that performs better than state-of-the-art quantifiers. However, we also claim that the significance of this approach is not only limited to presenting the first ensemble-based quantifiers, but to introduce an idea that is applicable to other learning problems in which it is possible to define how the data distribution changes throughout the time. Moreover, we think that this work opens new lines of research within this type of learning tasks, in aspects like diversity inclusion in function of the expected changes.

The second contribution of this work is to propose three new selection criteria for ensembles of quantifiers that can be used to implement both static and dynamic ensemble selection methods. Moreover, all the proposed criteria are devised for tackling quantification problems, exploiting their peculiarities. In particular, two of them are especially designed to work in combination with some state-of-the-art quantifiers, namely AC, PAC and HDy.

The experiments reported in Chapter 4 show that using these criteria with a simple selection scheme based just on ranking improves the performance of the ensembles when all the models are averaged. Our results are in line with those reported in Britto et al. [2014] in the sense that most of the times ensemble selection performs better, but not always. Interestingly, these new criteria outperform in some cases the well-know competence measure based on accuracy. Another important conclusion is that the performance strongly depends on an appropriate combination between the selection criterion used and the base quantifier. In this sense, dynamic ensemble selection using both the HDy algorithm and the HDy selection function seems the overall best method.

The fact that quantification learning needs to make estimations for sets of examples instead for individual instances, requires to extend the work that has been done in the field of ensemble selection for other learning tasks, like classification or regression. According to our results, the most promising approach is the one that has into account the similarities between the training and testing distribution. Besides, adequate selection methods, more sophisticated that the ones used here will boost the performance of this kind of ensembles.

Finally, in this work we have analyzed the applicability of multi-class EoQ in the context of sentiment quantification, where a user opinion can be considered positives, negatives or neutrals. These ensembles return the probability distribution of the three classes given unlabeled samples. The study in this case was aimed at analyzing whether these EoQ perform as well as those ensembles used in other supervised tasks, like classification and regression. Our experiments over SemEval datasets allow us to conclude that the performance of EoQ is at least competitive with respect to single quantifiers and sometimes better. In fact, ensembles algorithms are the winners in most of the datasets in terms of KLD and MAE scores. Moreover, they show a nice behavior regarding robustness against over-fitting and stability with respect to some of their components, namely the procedure to generate the training samples.

6.2 Future work

There are several possible lines for future research. Among them, we would like to highlight the following:

(i) To develop ensembles of quantifiers for multi-class quantification. Here, we have just applied a straightforward solution –the one-vs-all approach– but other methods could be applicable. Additionally, some elements of our proposal, namely the process to generate training samples, should be refined for the multi-class case.

(ii) To study new methods to combine ensemble models. In addition to the selection measures introduced in this work, there are a bunch of alternatives in the literature that have been proposed for ensembles in other kind of learning tasks, such as classification or regression. Some of them could be applicable for ensembles in quantification learning. Besides, new approaches specially devised for quantification could also be developed.

(iii) To apply ensembles of quantifiers for other real-world applications. Our main interest in this case would be to find a problem where the training data is composed of several training samples, obtained at different time for instance. In this scenario, each weak ensemble model could be trained using one of these training samples. We could compare then the performance of such approach to the one presented here, analyzing whether actual samples are more appropriate than artificially generated ones or if both could be combined for boosting ensemble performance.

Bibliography

Rocío Alaiz-Rodríguez, Enrique Alegre-Gutiérrez, Víctor González-Castro, and Lidia Sánchez. Quantifying the proportion of damaged sperm cells based on image analysis and neural networks. In *Proceedings of SMO'08*, pages 383–388. World Scientific and Engineering Academy and Society (WSEAS), WSEAS Press, 2008.

Robert E Banfield, Lawrence O Hall, Kevin W Bowyer, and W Philip Kegelmeyer. Ensemble diversity measures and their application to thinning. *Information Fusion*, 6(1):49–62, 2005.

R. Barandela, J.S. Sánchez, V. García, and E. Rangel. Strategies for learning in class imbalance problems. *Pattern Recognition*, 36(3):849–851, 2003.

Jose Barranquero, Pablo González, Jorge Díez, and Juan José Del Coz. On the study of nearest neighbor algorithms for prevalence estimation in binary problems. *Pattern Recognition*, 46(2):472–482, 2013.

Jose Barranquero, Jorge Díez, and Juan José del Coz. Quantification-oriented learning based on reliable classifiers. *Pattern Recognition*, 48(2):591–604, 2015.

Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

A. Bella, C. Ferri, J. Hernández-Orallo, and M.J. Ramírez-Quintana. Aggregative quantification for regression. *Data Mining and Knowledge Discovery*, pages 1–44, 2013.

Antonio Bella, Cesar Ferri, José Hernández-Orallo, and Maria Jose Ramirez-Quintana. Quantification via probability estimators. In *IEEE International Conference on Data Mining (ICDM'10)*, pages 737–742, 2010.

Alessio Benavoli, Giorgio Corani, and Francesca Mangili. Should we really use post-hoc tests based on mean-ranks? *Journal of Machine Learning Research*, 17(5):1–10, 2016.

J. Bollen, H. Mao, and X.J. Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2011.

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

Alceu S Britto, Robert Sabourin, and Luiz ES Oliveira. Dynamic selection of classifiers—a comprehensive review. *Pattern Recognition*, 47(11):3665–3680, 2014.

Gavin Brown, Jeremy Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

Rich Caruana, Alexandru Niculescu-Mizil, Geoff Crew, and Alex Ksikes. Ensemble selection from libraries of models. In *Proceedings of the twenty-first international conference on Machine learning*, page 18. ACM, 2004.

Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2:27:1–27:27, May 2011. ISSN 2157-6904.

Giovanni Da San Martino, Wei Gao, and Fabrizio Sebastiani. Ordinal text quantification. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 937–940. ACM, 2016.

K. Dave, S. Lawrence, and D.M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of WWW'03*. ACM, 2003.

Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15. Springer, 2000a.

Thomas G Dietterich. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. *Machine Learning*, 40(2):139–157, 2000b.

A. Esuli and F. Sebastiani. Determining the semantic orientation of terms through gloss classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 617–624. ACM, 2005.

Andrea Esuli and Fabrizio Sebastiani. Sentiment quantification. *IEEE Intelligent Systems*, 25(4):72–75, 2010.

R.E. Fan, K.W. Chang, C.J. Hsieh, X.R. Wang, and C.J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9: 1871–1874, 2008.

Paul W Farris, Neil Bendle, Phillip Pfeifer, and David Reibstein. *Marketing metrics: The definitive guide to measuring marketing performance*. Pearson Education, 2010.

Tom Fawcett and Peter Flach. A response to webb and ting's on the application of roc analysis to predict classification performance under varying class distributions. *Machine Learning*, 58(1):33–38, 2005.

G. Forman. Quantifying trends accurately despite classifier error and class imbalance. In *Proceedings of ACM SIGKDD'06*, pages 157–166, 2006.

G. Forman, E. Kirshenbaum, and J. Suermondt. Pragmatic text mining: minimizing human effort to quantify many issues in call logs. In *Proceedings of ACM SIGKDD'06*, pages 852–861. ACM, 2006.

George Forman. Counting positives accurately despite inaccurate classification. In *Machine Learning: ECML 2005*, pages 564–575. Springer, 2005.

George Forman. Quantifying counts and costs via classification. *Data Mining and Knowledge Discovery*, 17(2):164–206, 2008.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156, 1996.

Giorgio Fumera and Fabio Roli. A theoretical and experimental analysis of linear combiners for multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):942–956, 2005.

Wei Gao and Fabrizio Sebastiani. Tweet sentiment: From classification to quantification. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 97–104. ACM, 2015a.

Wei Gao and Fabrizio Sebastiani. Tweet sentiment: From classification to quantification. In *International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2015)*, 2015b.

Wei Gao and Fabrizio Sebastiani. From classification to quantification in tweet sentiment analysis. *Social Network Analysis and Mining*, 6(1):1–22, 2016.

Santiago García and Francisco Herrera. An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. *Journal of Machine Learning Research*, 9:2677–2694, 2008.

Pablo González, Eva Álvarez, Jose Barranquero, Jorge Díez, Rafael González-Quirós, Enrique Nogueira, Angel López-Urrutia, and Juan José del Coz. Multiclass support vector machines with example-dependent costs applied to plankton biomass estimation. *IEEE Transactions on Neural Networks and Learning Systems*, 24(11):1901–1905, 2013.

Pablo González, Eva Álvarez, Jorge Díez, Ángel López-Urrutia, and Juan José del Coz. Validation methods for plankton image classification systems. *Limnology and Oceanography: Methods*, pages 1–15, 2016a.

Pablo González, Jorge Díez, Nitesh Chawla, and Juan José del Coz. Why is quantification an interesting learning problem? *Progress in Artificial Intelligence*, pages 1–6, 2016b.

Pablo González, Alberto Castaño, Chawla Nitesh, and Juan José del Coz. A review on quantification learning. Technical report, Artificial Intelligence Center, Gijón, Spain., `http://www.aic.uniovi.es/~juanjo/quant-review-grAcc.pdf`, 2017.

Víctor. González-Castro, Rocio Alaiz-Rodríguez, and Enrique Alegre. Class distribution estimation based on the hellinger distance. *Information Sciences*, 218:146–164, 2013.

G. Grefenstette, Y. Qu, J.G. Shanahan, and D.A. Evans. Coupling niche browsers and affect analysis for an opinion mining. In *In Proceedings of RIAO*. Citeseer, 2004.

D.J. Hand. Classifier technology and the illusion of progress. *Statistical Science*, 21(1):1–14, 2006.

Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 12(10):993–1001, 1990.

Mohammad Javad Hosseini, Zahra Ahmadi, and Hamid Beigy. Using a classifier pool in accuracy based tracking of recurring concepts in data stream classification. *Evolving Systems*, 4(1):43–60, 2013.

Matthew Karnick, Metin Ahiskali, Michael D Muhlbaier, and Robi Polikar. Learning concept drift in nonstationary environments using an ensemble of classifiers based approach. In *Neural Networks, 2008. IJCNN 2008.(IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, pages 3455–3462. IEEE, 2008.

Gary King and Ying Lu. Verbal autopsy methods with multiple causes of death. *Statistical Science*, 23(1):78–91, 2008.

Svetlana Kiritchenko, Xiaodan Zhu, and Saif M Mohammad. Sentiment analysis of short informal texts. *Journal of Artificial Intelligence Research*, 50:723–762, 2014.

AH-R Ko, Robert Sabourin, and A de Souza Britto. Combining diversity and classification accuracy for ensemble selection in random subspaces. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 2144–2151. IEEE, 2006.

J Zico Kolter and Marcus A Maloof. Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research*, 8:2755–2790, 2007.

Bartosz Krawczyk and Michał Woźniak. Untrained weighted classifier combination with embedded ensemble pruning. *Neurocomputing*, 196:14–22, 2016.

Meelis Kull and Peter Flach. Patterns of dataset shift. In *First International Workshop on Learning over Multiple Contexts (LMCE) at ECML-PKDD*, 2014.

Ludmila Kuncheva. Classifier ensembles for changing environments. In *Multiple Classifier Systems*, pages 1–15. Springer, 2004.

Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine Learning*, 51(2):181–207, 2003.

Chen Lin, Wenqiang Chen, Cheng Qiu, Yunfeng Wu, Sridhar Krishnan, and Quan Zou. Libd3c: ensemble classifiers with a clustering and dynamic selection strategy. *Neurocomputing*, 123:424–435, 2014.

B. Liu. *Handbook of Natural Language Processing*, chapter Sentiment Analysis and Subjectivity, pages 1–38. CRC Press, Taylor and Francis Group, 2010a.

B. Liu. Sentiment analysis: A multi-faceted problem. *IEEE Intelligent Systems*, 25(3), 2010b.

Letizia Milli, Anna Monreale, Giulio Rossetti, Fosca Giannotti, Dino Pedreschi, and Fabrizio Sebastiani. Quantification trees. In *IEEE International Conference on Data Mining (ICDM'13)*, pages 528–536, 2013.

J.G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N.V. Chawla, and F. Herrera. A unifying view on dataset shift in classification. *Pattern Recognition*, 45(1): 521–530, 2012.

Preslav Nakov, Zornitsa Kozareva, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Theresa Wilson. Semeval-2013 task 2: Sentiment analysis in Twitter. In *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval-2013)*, 2013.

Preslav Nakov, Alan Ritter, Sara Rosenthal, Fabrizio Sebastiani, and Veselin Stoyanov. Semeval-2016 task 4: Sentiment analysis in Twitter. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1–18, San Diego, California, June 2016. Association for Computational Linguistics.

David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, pages 169–198, 1999.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

Ioannis Partalas, Grigorios Tsoumakas, and Ioannis P Vlahavas. Focused ensemble selection: A diversity-based method for greedy ensemble selection. In *ECAI*, pages 117–121, 2008.

Ioannis Partalas, Grigorios Tsoumakas, and Ioannis Vlahavas. Pruning an ensemble of classifiers via reinforcement learning. *Neurocomputing*, 72(7):1900–1909, 2009.

Pablo Pérez-Gállego, José Ramón Quevedo Pérez, and Juan José del Coz Velasco. Uso de ensembles en problemas con cambios de distribución caracterizables. In *Actas de la XVI Conferencia de la Asociación Española para la Inteligencia Artificial, CAEPIA 2015*, 2015.

Pablo Pérez-Gállego, Alberto Castaño, José Ramón Quevedo, and Juan José del Coz. New static and dynamic quantifier selection measures. *Under Review. Neurocomputing*, 2017a.

Pablo Pérez-Gállego, Alberto Castaño, José Ramón Quevedo, and Juan José del Coz. Ensembles of quantifiers for sentiment quantification. *Under Review. Decision Support Systems*, 2017b.

Pablo Pérez-Gállego, José Ramón Quevedo, and Juan José del Coz. Using ensembles for problems with characterizable changes in data distribution: A case study on quantification. *Information Fusion*, 34:87–100, 2017c.

John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A.J. Smola, P.L. Bartlett, B. Schölkopf, B. lkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 2000.

Joaquin Quiñonero Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. *Dataset shift in machine learning*. The MIT Press, 2009.

T. Rakthanmanon, E.J. Keogh, S. Lonardi, and S. Evans. MDL-based time series clustering. *Knowledge and Information Systems*, 33(2):371–399, 2012.

Sara Rosenthal, Alan Ritter, Preslav Nakov, and Veselin Stoyanov. Semeval-2014 task 9: Sentiment analysis in Twitter. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*, pages 73–80. Dublin, Ireland, 2014.

Sara Rosenthal, Preslav Nakov, Svetlana Kiritchenko, Saif M Mohammad, Alan Ritter, and Veselin Stoyanov. Semeval-2015 task 10: Sentiment analysis in Twitter. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 451–463, 2015.

Dymitr Ruta and Bogdan Gabrys. Classifier selection for majority voting. *Information fusion*, 6(1):63–81, 2005.

M. Saerens, P. Latinne, and C. Decaestecker. Adjusting the outputs of a classifier to new a priori probabilities: A simple procedure. *Neural Computation*, 14(1): 21–41, 2002.

L. Sánchez, V. González-Castro, E. Alegre-Gutiérrez, and R. Alaiz-Rodríguez. Classification and quantification based on image analysis for sperm samples with uncertain damaged/intact cell proportions. In *Proceedings of the 5th International Conference on Image Analysis and Recognition, ICIAR'08*, pages 827–836, 2008.

Martin Scholz and Ralf Klinkenberg. Boosting classifiers for drifting concepts. *Intelligent Data Analysis*, 11(1):3–28, 2007.

Robert P Schumaker, Yulei Zhang, Chun-Neng Huang, and Hsinchun Chen. Evaluating sentiment in financial news articles. *Decision Support Systems*, 53 (3):458–464, 2012.

David W Scott. *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

Bernard W Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

Andrew Solow, Cabell Davis, and Qiao Hu. Estimating the taxonomic composition of a sample when individuals are classified with error. *Mar. Ecol.: Prog. Ser.*, 216:309–311, 2001.

Kenneth O Stanley. Learning concept drift with a committee of decision trees. *Technical Report: UT-AI-TR-03-302, Department of Computer Sciences, University of Texas at Austin, USA*, 2003.

Amos Storkey. When training and test sets are different: characterizing learning transfer. *Dataset Shift in Machine Learning*, pages 3–28, 2009.

P. Subasic and A. Huettner. Affect analysis of text using fuzzy semantic typing. *Fuzzy Systems, IEEE Transactions on*, 9(4):483–496, 2001.

Christino Tamon and Jie Xiang. On the boosting pruning problem. In *European Conference on Machine Learning*, pages 404–412. Springer, 2000.

L. Tang, H. Gao, and H. Liu. Network quantification despite biased labels. In *Proceedings of the 8th Workshop on Mining and Learning with Graphs*, pages 147–154. ACM, 2010.

Grigorios Tsoumakas, Ioannis Partalas, and Ioannis Vlahavas. A taxonomy and short review of ensemble selection. In *Workshop on Supervised and Unsupervised Ensemble Methods and Their Applications*, 2008.

Grigorios Tsoumakas, Ioannis Partalas, and Ioannis Vlahavas. An ensemble pruning primer. In *Applications of supervised and unsupervised ensemble methods*, pages 1–13. Springer, 2009.

Alexey Tsymbal, Mykola Pechenizkiy, Pádraig Cunningham, and Seppo Puuronen. Dynamic integration of classifiers for handling concept drift. *Information fusion*, 9(1):56–68, 2008.

P.D. Turney. Thumbs up or thumbs down?: Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of ACL'02 Annual Meeting*, pages 417–424. Association for Computational Linguistics, 2002.

Haixun Wang, Wei Fan, Philip S Yu, and Jiawei Han. Mining concept-drifting data streams using ensemble classifiers. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 226–235. ACM, 2003.

S. Wang and X. Yao. Diversity analysis on imbalanced data sets by using ensemble models. In *Computational Intelligence and Data Mining, 2009. CIDM '09. IEEE Symposium on*, pages 324–331, 2009.

Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

J.M. Wiebe. Learning subjective adjectives from corpora. In *Proceedings of AAAI'00 National Conference*, page 735, 2000.

J.M. Wiebe, T. Wilson, R. Bruce, M. Bell, and M. Martin. Learning subjective language. *Computational linguistics*, 30(3):277–308, 2004.

H. Yu and V. Hatzivassiloglou. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of EMNLP'03*, pages 129–136. Association for Computational Linguistics, 2003.

Yang Yu, Wenjing Duan, and Qing Cao. The impact of social and conventional media on firm equity value: A sentiment analysis approach. *Decision Support Systems*, 55(4):919–926, 2013.

Huaxiang Zhang and Linlin Cao. A spectral clustering based ensemble pruning approach. *Neurocomputing*, 139:289–297, 2014.

Yi Zhang, Samuel Burer, and W Nick Street. Ensemble pruning via semi-definite programming. *Journal of Machine Learning Research*, 7(Jul):1315–1338, 2006.

Zhi-Hua Zhou, Jianxin Wu, and Wei Tang. Ensembling neural networks: many could be better than all. *Artificial intelligence*, 137(1):239–263, 2002.

I. Žliobaitė. Learning under concept drift: an overview. Technical report, Faculty of Mathematics and Informatics, Vilnius University, Lithuania, 2010.