POLITECNICO DI TORINO

Mathematical Engineering
Master's degree

Master's Thesis

Classification of imbalanced data applied to the insurance market

Supervisor
Prof. Luca Cagliero
Company Tutor
Dott. Riccardo Rinaldi

Author
Miriam Dessì
s244168

A.A 2019-2020

Summary

Classification is a typical task in the field of data mining. Classifying an observation means assigning it to a category of examples labeled data by means of a learning algorithm.
Classification models are often developed using data where the event associated to one of the classes is rare. We can think about a clinical study, where the data collected by a screening program usually include few patients with the disease, rarity class, and many healthy people. Such models tend to achieve a poor accuracy in classifying observations belonging to the rarity class. We defined this kind of problems as class imbalanced.
The main purpose of this master thesis is the investigation of the problem of classification in imbalanced data sets. First, a theoretical study about the nature of the problem and the state of art solution has been presented. Then, a real application of the problem has been described through a case of study coming from the insurance market.

The class imbalanced problem can be considered one of the challenging problem in data mining, as it is present in many real-world domains such as computer science, epidemiology, finance and so on. This has brought along a growth attention from both academia and industry. It is very meaningful in theory and in practice to investigate the classification of imbalanced data sets.
To this end, this thesis work can be divided into two parts. In the first part, a discussion about the problem of data imbalanced itself has been presented. It has been studied why the skewed distribution negatively affects the accuracy of the standard classifiers. The reason behind it has to be found inside the classifier learning process structure, that is often built for balanced training examples. It means that equally misclassification costs are assigned inside the classes. This leads to a biased towards the majority class.
However the imbalanced distribution of the data is not the only factor that hinder the learning task. Several data intrinsic characteristics have been analyzed such as: the presence of small disjuncts, the overlapping between classes, the presence of noise and borderline examples. It has been shown how they affected the learning process, exasperating the imbalanced problem.
To conclude this first part of the work, the state of art solutions have been described. They can be divided into four groups: data level, algorithm level, cost-sensitive and ensembles methods. Data level approaches act directly on the training set, employing resampling methods to balance the class distribution. Resampling techniques can be categorized into three groups: undersampling, oversampling and hybrids. More in detail, several

undersampling and oversampling techniques have been illustrated. Among the oversampling approaches SMOTE and its variations (Boorderline SMOTE, Safe-Level SMOTE, ADASYN) have been described. It has been identified the main drawback of these methods, that is the blind choice of the k nearest neighbors considered to generate the new synthetic examples. A new novel method called "Automatic determination of neighborhood size", able to solve this problem, has been explained. Among the undersampling techniques four data cleaning approaches have been described: Edited nearest neighbors rule, Tomek Link, One side selection and Neighborhood cleaning rule. Their objective is to clean the majority class examples from noisy and borderline examples in order to have a smoother decision boundary. Lastly, a cluster based sample approach has been described.
Algorithm level or internal approaches aim to improve the existing learning process in favour to the minority class. This kind of method requires specific knowledge about the classifier and the data domains application. Among these techniques we have the Kernel based methods, the Active learning methods and One class learning methods. Cost sensitive approaches include data level, algorithm level or both mixed. The objective of this kind of solutions is to assign different misclassification cost to each class. There are two main ways to implement cost sensitive methods: Direct methods and Meta-learning methods. The direct methods act inside the learning process adding cost sensitive elements, while the meta algorithms act on the training data. The latter can modify the data on pre-processing step, resampling the examples according with cost sensitive decision matrix, or at the end of the process acting to the classification threshold. As a combination of all these approaches there are the ensembles. They are built with the aim of improving the performance of a single classifier by training several classifiers and then aggregate their prediction. The two most famous ensemble techniques are Bagging and Boosting. Bagging methods generate different bootstrapped training dataset. The model is trained on each boostrapped training set and then all the predictions are average. Boosting procedure on the other hand, combines weak classifiers in other to produce produce a powerful "committee". The ensembles methods do not need to modify the base classifiers. In imbalance context, they generally act on the ensembles learning algorithm, inserting data level approaches and cost sensitive frameworks.

In the second part of the thesis it has been presented a case of study developed during an internship in Reale Mutua Assicurazioni. The company is interested in exploiting information about data regarding RCA quotations. The business aim is to detect important information about the client's price sensitivity. The collected data coming from an online sales channel "Segugio". Among all the quotations recorded, only 1% have been converted into a policy. This means that the number of positive class examples (converted policies) is very imbalanced compared with the number of negative examples (not converted policies). In order to handle the imbalanced class problems two set of experiments have been proposed. In the first ones, three different undersampling and three different oversampling techniques are applied to balance the class distributions. The performances of these approaches have been compared in terms of F-measure, employing four classifiers : Decision tree classifier, Random Forest classifier, Logistic Regression, Support Vector Classifier. Also the performances in the original imbalanced dataset have been taken into

account. From these experimental results, it is emerged the incapability of all the classifier to better recognize element of the minority class, that corresponds to low values of recall measures. When an increase of this value has been recorded, such as in the Decision tree (80%) and Random forest (84%), applying RUS techniques, a drop in the precision, (4.5% and 5% respectively) has been registered resulting in low F-measure scores (8%, 10%). Tomek Link and Neighborhood cleaning rule, on the other side, were not been able to delete a high number of examples from the majority class, leaving the training dataset imbalanced as in the original scenario. When the oversampling techniques are employed, a number of positive elements equals to the difference with the number of the examples in the majority class are generated. As our imbalanced was about 1%, a very high number of new samples are generated inside the training set. When we test the training model on each validation fold, that is not affected by oversampling, the results are a degradation of all the classifiers performances. Despite this, Decision Tree classifier without the application of any re-sampling techniques resulted as the best model, but with a recall only about 64%. This is compensated by a high precision, about 80%, that lead the F-measure about 74%, that is best value among all the tested approaches.
Once it has been identified the best classifier and the best re-sampling techniques the first part of the experiments is concluded.
In the second part of the experiments, an algorithm level approach is employed to improve the performance of the best control algorithm. To this end, a study of the performances employing Gradient boosting decision tree classifier has been carried out. The Gradient boosting classifier has been shown to be a very powerful classifier able not only to reach the highest performance in terms of F-measure (84%), precision (93%) and recall (74%), but also to give us information about the importance of the variables in our model. Understand what happens inside the "black-box" is of primary relevance in business decision contests. The results suggested that the gross amount is the most relevant features and it plays a fundamental role in the discrimination between the classes. Finally, it has been carried out an example of the limitation of this model. It has been observed a drop in the model precision (43%), when the extreme values of gross amount are excluded from the training dataset, and so we have more or less the same values of this variable among both the classes. In this scenario it would be interesting to investigate the interaction between the others most informative variables, in order to understand which factors influence the client's price sensitivity.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This master thesis aim to investigate the classification in imbalanced dataset. To this end and theoretical and practical investigation of the problem is carried out. This work, is divided into two parts. In the first four Chapters theoretical arguments to explain the imbalanced class problems will be carry out. In the last two Chapters a real application of the problems will be presented through a case of study. The structure of this work, is the following.
In the Chapter 2, an investigation about the nature of the problem is presented. The study of some intrinsic characteristics of the data such as the problem of the small disjuncts, the overlapping between classes, the presence of noise and borderline examples, shows in which way they affected the learning process when the imbalanced between classes is present.
In Chapther 3 the state of art solution are analyzed. In deep we will examined some data level approaches. Among the oversampling method we will analysed: SMOTE, Borderline-SMOTE, Safe-Level SMOTE, ADASYN, AND-SMOTE, CBO. [[4],[8],[3],[11],[34],[15]]. Between the undersampling will be analyzed: Edited Nearest neighbor, Tomek link, One side selection, Neighborhood cleaning rule, and a cluster besed undersampling techniques [31],[27],[16],[18], [33]
. In Chapter 4 will be found a description of three well known classifiers together with a brief explanation of the reasons that take these classifiers to be biased towars the majority class. The classifiers will be described are : the support vector machine, the logistic regression and the decision tree.
In the Chapter 5 the case of study will be presented. It will be described the contextualization but also the structure on the data collected by the company and that wil e analyzed in Capther 6. Here, we can find all the experimental results.

Chapter 2

Problem statement

2.1 The nature of problem

In the last decade many machine learning approaches have been developed to cope with the imbalance learning problem. The awareness about the latter is growing since it finds application in many domains such as computer science, f.i. face recognition, epidemiology, f.i. medical diagnosis in rare disease, finance, f.i. detection of fraud detection, and so on.

In all these cases the imbalance is a direct result of the dataset nature, this type of imbalance is called *intrinsic*.

In contrast with this definition there is the *extrinsic* imbalance, when the imbalance is caused by the biasing in the data acquisition process. In order to clarify this definition we can think about a dataset that is generated by a stream of balanced data over a specific time interval. If during the acquisition interval the transmission of the data has occasional interruption, then, the results dataset, can be imbalanced [10]. So we have obtained an imbalance dataset from a balanced data space.

Class imbalance can furtherly be cathegorized in absolute or relative [29]. The former class is based on samples associated to exceptional event that are generally difficult to observe, so they are rare in an absolute sense. On the other hand objects may not be rare in an absolute sense but are rare relative to other objects. An imbalance can be defined *relative* when many positive or negative example can be observed but the imbalance ratio is high [[29]].

In the next section the main problems that arise in mining imbalanced data are discussed and analyzed.

2.2 The problem of skewed distribution

By definition a dataset is "imbalanced" if its imbalance ratio (IR) is > 1.

$$IR = \frac{N^+}{N^-} \tag{2.1}$$

where N^- and N^+ are the number of examples belonging to the minority and the majority class respectively.

The minority examples are usually the most important instances to be learnt but, as we said in the previous section, they are difficult to acquire because their are associated with extraordinary or costly events. As a result their representation inside the dataset is weakened by the strong presence of the majority class instances.

However, the main problem that has to be faced with imbalanced dataset, is that standard classifier are biased towards the majority class. This behaviour is influenced by different factors: for example the classifier learning process would be designed with the objective of optimize global metrics such as prediction accuracy. This results in a good coverage around the majority class while the minority examples are misclassified.

The skewed class distribution is not the only responsible of the performance degradation. There are several data intrinsic characteristics that must be taken in account in order to achieve a better performance and a complete understanding of the problems.

2.3 The problem of small disjuncts

The minority concepts may additionally contain sub-concepts with limited instances, surrounded by majority class examples [Fig 2.1].

This scenario causes degradation in the classifier and it is known as *problem with small disjuncts*. A great number of studies demonstrated that small disjuncts have a higher error rate than large disjuncts [[30], [15],[28],[12]].

More in detail, when a classifier tries to learn instances, it creates several disjuncts, which are joined with a subconcept of the original concept. The coverage of a disjuncts corresponds to the number of training examples correctly classified. If that coverage is low, a disjuncts is considered small. The classifier bias, the presence of noise attributes and the size of the training set make small disjuncts more error prone. Different approaches has been used to face this problem such as the deletion of all small disjuncts or the use of statistical significant testing to ensure the importance of small disjuncts in order to remove only the subconcepts that are not significant. All these strategies lead to poor

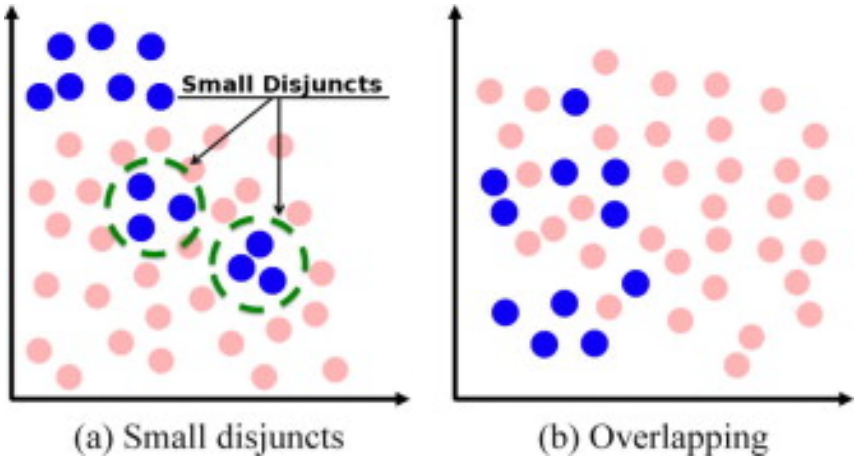


Figure 2.1: Illustrazion of small disjunct (a) and overlapping (b) [[25]]

performances [[12]].

Recently, [15] proposed a cluster-based oversampling method to deal with class imbalance and small disjuncts simultaneously. This method is able to identify rare cases and to re-sample them as well as to avoid the creation of small disjuncts in the learned hypothesis.

2.4 The problem of overlapping between the classes

Overlap appears when instances of two different classes coexist in the same region of the data space. As a consequence, the training examples belonging in that region have equal prior probability estimations, making them very difficult to be learnt.

In [7] the authors showed that overlap could play a major role in determining classifier performance w.r.t. imbalance.

In [24] several experiments are conducted with different synthetic dataset with the purpose of testing various imbalance ratio and degree of overlap. It has been inferred that the class probability distribution is not the only factor that results in a loss of performance of the learning system but it is also related to the level of overlapping among the classes.

Another important contribution is made by [5] that investigated the relationship between imbalance and overlap. In particular, they demonstrated that trying to solve overlap problem separately lead to a more complex learning system w.r.t. the imbalance one. When the two factors are acting in concert, they cause difficulties that are more severe than one would expect by examining their effects in isolation.

In other to validate their hypothesis, the authors employed different synthetic dataset varying the imbalance ratio, the degree of overlap, both of them jointly and testing the

performances through SVM classifier. Their results showed that, for small training size and an elevate imbalance level, the classifier performance worsen, the complexity of SVM problems grows proportional to the overlap level and the training size. Furthermore SVMs reach a breaking point when imbalance and overlap level are very high, this point is correlated with the peak of complexity.

2.5 The problem of noisy examples

Noisy examples are instances of one class located deep inside the region of other class, these samples are corrupted in the attribute value or in class labels.

In the case of imbalance dataset, the presence of noisy examples have a huge impact in the learning of the subconcepts or minority class instances. The learner should be modified in order to cover noisy examples taking to the undesirable effect of having small-disjuncts.

In order to avoid this scenario, some overfitting techniques, such as pruning, are commonly used, but in this way some important minority instances will be neglected.

In [26] an empirical study on the effect of class imbalance and class noise is designed on eleven different learning algorithm and seven data sampling techniques. The authors identify which approaches are more robust in imbalance and noise dataset. Their results showed that many classification algorithm are more sensitive to noise than imbalance, but as imbalance increases in severity, it plays a larger role in the performance of classifiers and sampling techniques.

2.6 The problem of borderline examples

Another problem of high interest is the high or low presence of instances located in the area surrounding class boundaries, named *borderline* examples. The latter is strictly related with the overlapping between classes.

As we can see (Fig 2.2) these examples have a huge impact in the determination of the boundary's shape, and the presence in concert of noise instances will take them to move to the wrong side of the decision boundary.

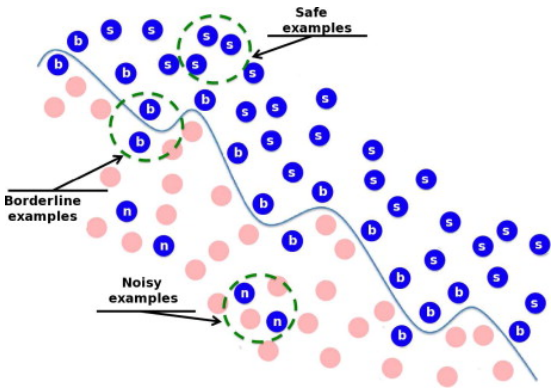


Figure 2.2: Illustration of the decision boundary in presence of borderline examples [in[25]]

In classification problems the better the definition of borderline area, the more accurate identification of the different classes will be. In [21] the authors presented an experimental study in which they investigate the impact of noise and borderline examples in the minority class on classifier performances. Their results showed that the borderline examples affect performance of a classifier leading to degradation.

Chapter 3

State of art solutions

A large number of approaches have been proposed to deal with the class imbalance problem. These proposal can be organized into three categories depending on how it face the class imbalance.

1. Algorithm level.
   The algorithm level (*internal*) approaches aim to improve the existing classifier learning process in favour of minority. This kind of method demands specif knowledge about the classifier and the data domain applications. Consequently, they have low generality.

2. Data level.
   Data level (*external*) approaches introduce preprocessing step where the training instances class distribution is altered getting more balanced samples, in order to decrease the skewed class distribution during the learning process that allow classifier to perform in conditions similar to the standard classification. A dataset can be resampling through three ways: over-sampling the minority class, under-sampling the majority class or with a combination of the two previous approaches. Although data level approaches have the advantage to be independent from the underlying classifier they present some drawbacks. Undersampling the majority class we can exclude potentially useful data while oversampling the minority class can increase the likelihood of overfitting. Additionaly will be difficult to maintain the same data distribution after oversampling because the presence of overlapping between classes is more probably.

3. Cost sensitive.
   Cost sensitive include data level, algorithm level or both levels mixed. The aim of this of solutions is to assign different misclassification cost to each class and therefore to minimize higher cost error. The standard public dataset do not contain costs, that are very hard to set because their strictly dependence on the dataset characteristics. As a consequence overfitting is highly likely when searching to find the most probable costs.

4. Ensembles of classifier.
   Another group of techniques that have been very popular in the last decade are Ensembles of classifier. The basic idea behind this approach is to train several classifier and then aggregate their prediction to output a single class label. Not only multiple classifier could have better performance than a single one, but also convey diversity for avoiding the overfitting of some algorithms. The two main techniques are Bagging and Boosting.

3.1 Data level techniques

3.1.1 Oversampling methods

The simplest oversampling method is *random oversampling*. Through this method exact copies of existing instances are made, this increases the likelihood of overfitting. In order to deal with this problem several approaches have been proposed.

SMOTE [[4]] :

For each minority class samples the k nearest neighbours belonging to the same class are found, then new synthetic examples are generated by linearly interpolating some or all of them. During the process the number of positive nearest neighbours selected to generate the new examples is chosen randomly. This method, compared to the random oversampling with replacement, improves the performances. The reason of this behavior is linked with the decision region built in the two scenarios. When random oversampling is used, the decision region of the minority class, as the the number of positive examples grows, can shrink. On the other hand, SMOTE acts in order to enlarge the decision region of minority class generating correlated points, increasing the coverage inside this class. The greatest drawback of SMOTE is that it generates data samples from each minority class examples, neglecting majority class distribution.

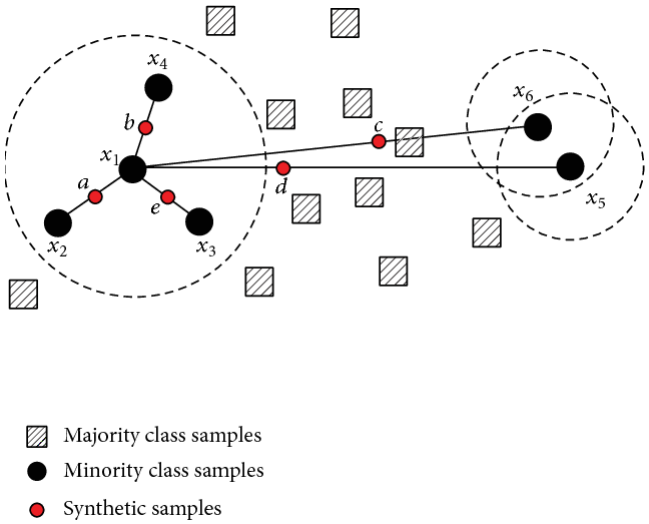


Figure 3.1: Illustration of SMOTE technique [[13]]

As we can see in the figure 3.1 the synthetic minority examples c and d are generated very close to majority examples, this increases the probability to face with the overlapping between classes problem.

In order to solve this problem, several adaptive sampling methods have been proposed in the last years, including SMOTE variants and cluster-based methods. Some representative works are next described.

Borderline SMOTE [[8]]:

In this method only examples near to the borderline of the positive class are over-sampled. The borderline examples are of main importance in the determination of a robust decision boundary, as they are frequently misclassified.

In borderline SMOTE a new parameter m is introduced, the nearest neighbors size of a minority example. Employing this parameter, the ratio r_i is computed, as the number of majority examples finding among the m nearest neighbors of each minority example. Successively this number is used to categorize examples as: "noise" ($r_i = 1$), "danger" ($0.5 < r_i < 1$), and "safe" ($r_i < 0.5$). The "danger" examples are the easiest misclassified inside the positive class, so only this kind of sample is taken into account inside the model.

For each danger examples, a number s of its positive nearest neighbors are selected, and for each of them a new synthetic sample is generated according with SMOTE technique. The approach described below is called B-SMOTE1.
The authors proposed also B-SMOTE2. In B-SMOTE2 also the negative nearest neighbors of "danger" examples are over-sampled. The generated negative examples are as close as possible to the positive ones. The described methods outperform the Random oversampling and SMOTE in terms of true positive rate and F-measure.

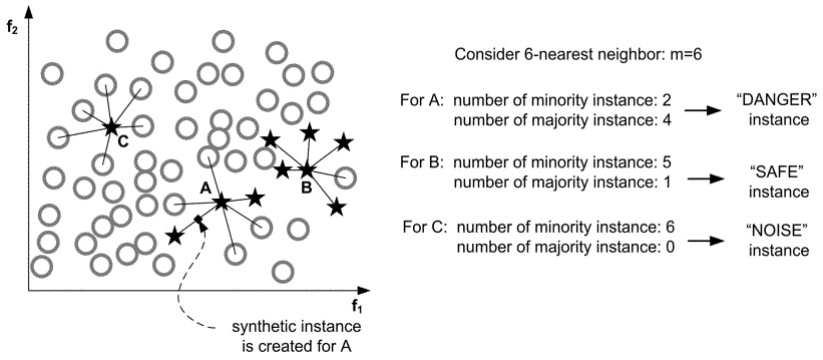


Figure 3.2: Examples of safe, noise and danger instances [[10]]

Sefe-Level SMOTE (SL-SMOTE) [[3]]:

In this work each positive instances is assigned to a "safe-level". The "safe-level" of a positive instance is defined as number of its k nearest neighbors belonging to the same class.
Another important parameter to take into account is the "safe-level ratio". It is the ratio between safe-level of a positive instances and the safe-level of one of its nearest neighbors.

The SL-SMOTE algorithm starts selecting a positive instance p and one of its positive nearest neighbors n, then the safe level of p and n and their safe-level ratio are computed. Five scenarios can be identified according with the different values safe-level ratio (s_{lr}).

1. $s_{lr}= \infty$ and $s_l(p) = 0$.
   This means that both p and n are noise examples, so they are not considered for the generation of new synthetic instances.

2. $s_{lr}= \infty$ and $s_l(p) > 0$.
   A new instance is generated, duplicating p.

3. $s_{lr}= 1$
   In this case both p and n are safe examples, so new synthetic instances are generated interpolating these points.

4. $s_{lr}>1$ and $s_{lr}<1$.
   In the first case p is safer than n and so new synthetic instances are created closer to p. The second case is the inverse.

Through this algorithm the new synthetic examples are collocated in a safe region, avoiding the overlapped and noisy ones. The authors have shown that the proposed method reaches higher values of F-measure and precision compared to those of SMOTE and Borderline SMOTE, using C4.5 decision tree.

ADASYN [[11]]:

The key idea in ADASYN algorithm is to adaptively assigns to each minority class examples the number of synthetic samples that must be generated using a weight distribution. For each examples x_i belonging to the minority class the k nearest neighbors are found. Then, the ratio ri between the number of majority examples present in the k nearest neighbors and the number of k is computed and normalized ($\hat{r}_i$). After that, the number of synthetic examples to generate for each x_i is expressed by the total number of synthetic samples to be generated for the minority class multiplied by $\hat{r}_i$. In this way more synthetics instances are created for those samples that are more difficult to learn, leading to an adaptively decision boundary shift towards these instances. For performances assessment the authors employed decision tree as classifier and G-mean as metric. Their results have shown that the new method outperforms SMOTE.

All the previous methods that have been illustrated, used a common parameter k, the number of nearest neighbors. An improper value of this parameter might lead to poor decision boundary and overlap between classes. In particular we can identify two common scenarios ([34]):

1. Minority noise instances in majority cluster.
   If we consider the rectangular noisy examples in Fig 3.3 (a), techniques such as SMOTE, ADASYN and Borderline-SMOTE with a k equal to 5, admit the generation of synthetic samples from these points. This results in a increment of false positive rate.

2. Presence of small disjuncts.
   Small disjunct stays for sub-cluster made up minority class instances. In this case if the value of k is higher than the size of any minority cluster, it is likely the generation of synthetic instances inside the majority region. This scenario is illustrated in Fig 3.3 (b) when an instance of B cluster is selected. Moreover, the analogous problems could show when the minority class is characterized by a complex distribution [Fig 3.3 (c)].

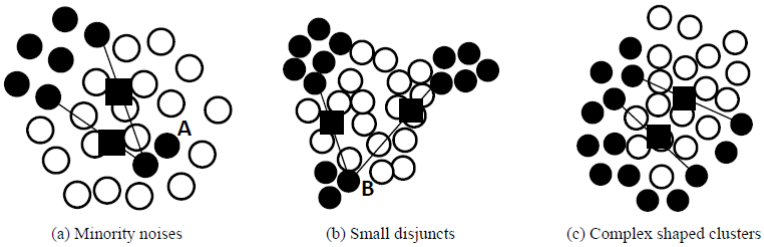


Figure 3.3: Problems observed with incorrect k value [[34]]

These examples suggest that the parameter k must be set carefully. It means that a different number of nearest neighbors have to be selected for each minority class examples, according with their characteristics. To this end the follow method is proposed.

Automatic determination of neighborhood size in SMOTE (AND) ([34]):
AND method aims to determine the k size of each minority class examples. For that purpose, for each example x_i belonging to the minority class, hyper-rectangular regions are built. These regions are constructed in the following way:

$$region_{i,k} = \cap_{j=1}^{p}[\min(x_{i,j}, n_{k,j}), \max(x_{i,j}, n_{k,j})]$$

where n is the kth nearest neighbors, and p is the features dimensional space. Next the union on K (total number of neighbors) is performed. Each time a new neighbor is taken into account, classification is performed according the following rule: the objects inside the region are classified as positive while those outside as negative. Accumulating the regions, more majority objects are included at each iterations, as a results the precision decreases.
The drop point of this measure is employed to set the best k parameter, and it is used as in SMOTE techniques. The only variant is that the new synthetic points are not generated interpolating a generic positive instance with one or all its k nearest neighbors, but are

generated inside the region. The main advantages of this method is that the synthetic samples are generated in order to preserve the original distribution of minority class.
The authors have compared the CART classifier performances applying SMOTE, AND-SMOTE, ADASYN, AND-ADASYN in terms of AUC-ROC and AUC-PR.
The results have shown that the application of this new approach to SMOTE and ADASYN lead to a classification improvement.

All the previous analyzed techniques are named "distance based". The main drawbacks of this kind of methods is that they do not suffer the similarity between data. As a consequence one of the main problem of imbalanced class, "the small disjuncts", has not been handled with accuracy.
A cluster based oversampling approach is now proposed.

Cluster based oversampling (CBO)[[15]]

The CBO aims to solve within and between class imbalanced in parallel. First, the train instances belonging to both the classes are clustered employing K-means algorithm. Then, the majority and minority class examples are randomly oversampled. The majority class examples are oversampled in order to reach the same size of the largest majority cluster. The instances of each minority class clusters are oversampled until every cluster that contains a number of samples equal to the ratio between majority class size and the number of minority class clusters made. This approach inherits from K-means, a main property. The clusters are built to have a within cluster variation as small as possible. As a result, CBO method is able to easily identify rare cases and re-sample them individually. The author have been demonstrated that this new approach performs better than random undersampling and oversampling, and An's oversampling.

Based on this approach others cluster based oversampling methods are developed such as the Kmeans SMOTE, DBSMOTE, A-SUWO [[6],[2],[22]].

3.1.2 Undersampling methods

Random undersampling belances the minority class through the random elimination of sampling belonging to the majority class. Through this blind approach potentially useful data might be discard leading to an increase of the learner loss. To overcome these issues several approaches based on cleaning techniques are proposed such as:

Edited Nearest Neighbor (ENN) [31]:
The basic rule's of this techniques is to remove each majority class examples misclassified by at half of its k nearest neighbors. If we named M the majority class training set and with S the set of the saved instances, then the EEN algorithm can be described as follows:

1. let $S = M$.
2. for each x_i in M, drops x_i from S if it is misclassified by K-nearest neighbors classifier.
3. return S.

This approach cleans up the majority class from noisy and borderline examples, leading to a smoother decision boundary and a greater classifier accuracy.

Tomek Links [27]:
Tome links, finds a pair of minimally distanced nearest neighbors belonging to opposite classes. More precisely given an instances pair x_i and x_j where $x_i \in P$ (set of positive samples) and $x_j \in N$ (set of negative samples) and $d(x_i, x_j)$ the distance between x_i and x_j, we said that they make a Tomek link if not exist x_k such that $d(x_i, x_k) < d(x_j, x_j)$ or $d(x_j, x_k) < d(x_i, x_j)$. Therefore when two instances that participate to a Tomek link, are either noisy or borderline. A good practise is to remove from training set majority class examples that realise a Tomek link. In this way are preserved all minimally distanced neighbor belongs to the same class, leading to well defined clusters in the training set which turn improves classification performance.

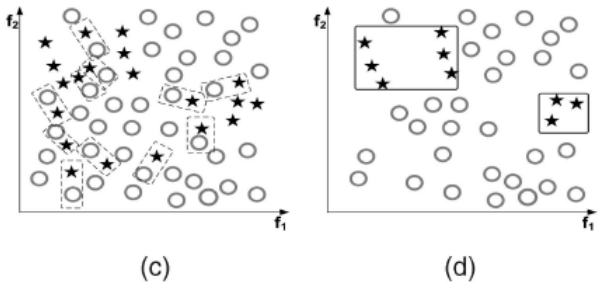


Figure 3.4: Tomek link illustration ([10])

One sided selection (OSS) [16]:
The aim of this method is to shrink the majority class set deleting noise, borderline and

redundant examples in order to soften the sensitivity of imbalanced classes distribution. The OSS procedure starts with the creation of a set C called *consistent subset*. A subset C of the training set T is consistent if correctly classified all the element in T employed 1-NN. Next all the examples that participate to a Tomek Link are remove to C. The authors compare the performances of two classifiers training with the training set without changing, the training set without noisy and borderline examples, and that make by OSS. The last case lead to improvement of the G-means in all the scenarios.
The major drawbacks of OSS is its extremely sensitivity to the noisy examples as they are removed from the training set.We remember that noisy examples are proud-misclassified. Keeping some noisy examples in the training set, might improve the capability of the learner to recognize them. To this end the following method is proposed.

Neighborhood cleaning rule [18] :
It is based on the same principles of OSS but to remove the noisy examples is used Edited nearest neighbors rule. The ENN approach preserve more data than Tomek Link. The authors employed the last method basing on the idea that is difficult to maintain a global good accuracy while the dataset is reduced, and so including in the dataset characteristic of the data such as noise might improve classification. The resulting method give more emphasis to data cleaning than data reduction.
The proposed algorithms follows the following steps:

1. The training set T is divided in two set: P that contains the instances of the minority class and M that contains the others instances.
2. In M are found the noisy elements throught ENN and put in a new set$A$.
3. Next the three nearest neighbors that miscalssified element in C e that lies in M are moved in a set B. To avoid the elimination of an excessive number of element in C, in the B construction only 0,5$|C|$ elements are take into account.
4. Finally the new training set $S = T - (\{A\} \cup \{B\})$

Experimental results have shown that NCN outperfoms RUS and OSS in terms of True Positive rates when 3-NN and C4.5 classifiers are employed.

Cluster based sample (SBC) [33] :
SBC undersampling approach aim to preserve majority class distribution. The method starts with the clusterization of all the training examples. The clusters including more majority class objects respect to the minority ones, are the most meaningful, as they represent faithfully the characteristics of the majority class objects. The majority instances of these kind of clusters are then undersampling in random way, taking into account the ratio between majority and minority objects inside each cluster.
The authors proposed others five approaches based on SBC, changing the way in which the majority object are selected in inside each cluster. For examples SBCNM-1, SBCNM-2, SBCNM-3, SBCMD employ NearMiss family methods to identify majority instances. NearMiss family methods have the goal to remove majority class objects nearest or farthest to the minority ones, using K-NN algorithm.

The authors have demonstrated that the classical SBC method reaches a high performance in terms of positive class accuracy compare to RUS, NearMiss2, SBCNM-1, SBCNM-2, SBCNM-3, and SBCMD, using back propagation neural network.

Chapter 4

Classification models

Classification is a typical task in the field of data mining. In classification setting we have a set of training observations $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$ where x express the set of features that lies in a m-dimensional space X, and y is the response *qualitative* variable that takes values in k-dimensional space Y, representing the categorical domains. The training set T_n is employed to build a classifier, through a rule $R_{T_n} : X \to Y$. The rule R_{T_n} divided the input space X in subspaces, according to the corresponding labels such that:

$$\frac{P(Y_i|x)}{P(Y_j|x)} > t \quad \forall i \neq j \tag{4.1}$$

where $P(Y_i|x)$ is the conditional probability for Y to belong to the class i given the information about the variable x, while t stays for the threshold over that the observation x is associated the class i. In order to estimate the probabilities 4.1 several methods have been proposed, which can be categorized into two main groups: Frequentist and Bayesian approaches. The first ones assumes that the $P(x|Y_i)$ comes from a parametric distribution for example a Gaussian distribution (Linear discriminant analysis) while the seconds employed a non parametric estimation for this probability (Decision Tree). We will focused only to binary classification, in this context the response variable can assume only two values, that is $Y = \{Y_0, Y_1\}$ [[20]].
The classification models that will be described are: Logistic regression, Support vector machines, Decision tree, Random Forest and finally Gradient Boosting.

4.1 Logistic Regression

The logistic regression models the probability that Y belongs to a particular category via linear function in x, that gives outputs between 0 and 1, called *logistic function*:

$$P(Y = 1|X) = \frac{e^{\beta_0 + + \beta 1 X_1 + \cdots + \beta p X_p}}{1 + e^{\beta_0 + + \beta 1 X_1 + \cdots + \beta p X_p}} \tag{4.2}$$

where $X = (X_1, \ldots, X_p)$ are the p predictors.
After simple calculation we find:

$$\log \frac{P(Y = 1|X)}{P(Y = 2|X)} = \beta_0 + \beta 1 X_1 + \cdots + \beta p X_p \tag{4.3}$$

The left side of 4.3 is named *logit*, while its argument is called *odds*.
The coefficients $\theta = \{\beta_0, \ldots, \beta_p\}$ are unknown and must be estimated, based on the training data. Logistic regression models are generally fitted employing *maximum likelihood* method. The conditional likelihood of Y given X can be expressed:

$$p_k(x_i, \theta) = P(Y = k|X = x_i, \theta) \tag{4.4}$$

The two class problems is generally encoded with 0/1 outputs, $y_i = 1$ when $k = 1$ and $y_i = 0$ when $k = 2$.
The likelihood for N observations can be formulated as:

$$l(\theta) = \prod_{i=1}^{N} p(x_i, \theta)^{y_i} (1 - p(x_i, \theta))^{1-y_i} \tag{4.5}$$

taking the log on the both sides and after some manipulations we can write the log-likelihood :

$$l(\theta) = \sum_{i=1}^{N} y_i(\beta_0 + \beta^T x_i) - \log(1 + e^{\beta_0 + \beta^T x_i}) \tag{4.6}$$

Learning the models means to estimate the parameters θ that maximize the likelihood. To this end the gradient's computed ignoring β_0 :

$$\nabla_\beta l(\beta) = \sum_{i=1}^{N} (y_i - p(x_i, \beta)) x_i \tag{4.7}$$

and the Hessian :

$$\nabla^2_\beta l(\beta) = -\sum_{i=1}^{N} x_i x_i^T p(x_i, \beta)(1 - p(x_i, \beta)) \tag{4.8}$$

as $p(x_i, \beta)(1 - p(x_i, \beta)) \geq 0$, the Hessian of l_β is semidefinite negative so l_β can be represented by a concave function. For a concave function we can always find maximum points through numerical methods such as Newton-Raphson. [9] [14]

In [23] a study on the effects of class imbalanced on maximum likelihood logistic regression has been conducted, showing the tendency of this model to under-predict the conditional probability of the minority class.

4.2 Support vector machine

SVM finds a optimal separate hyperplane between separable e non separable classes.
Given training data $\{x_i, y_i\}_{i=1}^N$ with $y_i = \pm 1$, a hyperplane is defined as:

$$\{x : f(x) = \beta_0 + \beta^T x\} \tag{4.9}$$

where $\|\beta\|$ =1.
If the classes are separable exist infinite function $f(x)$ such that $y_i f(x_i) > 0 \ \ \forall i$.
Therefore we have to use a criteria to decide which separate hyperplane to use.
In SVM the maximal margin hyperplane is chosen. As the term suggest the maximal margin hyperplane is the hyperplane with the farthest minimum distance to the training observations. Therefore construct the maximum margin hyperplane implies the resolution of the following optimization problem :

$$\max_{\beta_0,\ldots,\beta_p \ \|\beta\|=1} \quad M \tag{4.10}$$

$$\text{s.t.} \quad y_i(\beta_0 + \beta^T x_i) \geq M \tag{4.11}$$

According with the constraint each observation must lie on the correct side of the hyperplane and at with a minimum distance M from it.
If overlap between classes is allowed, the margin is called *soft*.
The hyperplane in this case is built with the objective of correctly separete most of training observations, but also to permit the existence of some misclassified examples.
To this end the optimization problem can be rewrite :

$$\max_{\beta_0,\ldots,\beta_p \ \|\beta\|=1} \quad M \tag{4.12}$$

$$\text{s.t.} \quad y_i(\beta_0 + \beta^T x_i) \geq M(1 - \epsilon_i) \tag{4.13}$$

$$\text{s.t.} \quad \epsilon_i \geq 0 \quad \sum_{i=1}^{N} \epsilon_i \leq C \tag{4.14}$$

In 4.14 ϵ_i are named *slack variables*.
This variables tell us the collocation of the *ith* observation respect to the hyperplane and the margin, while C parameter indicates the amount of observations that can violate the margin. The instances that lie on the margin or on the wrong side of the margin are known as *support vector*.
As we can observed form the optimization problem formulation only the support vectors influenced the classification. Therefore the parameter C control the bias-variance trade-off of the support vector.
In imbalanced data context if the parameter C is not high the SVM builds a margin that is the largest as possible with low cumulative error, resulting in a classifier that learn to

classify all the examples as negative.
The cumulative error about the positive class don't lead a significant contribution to the classification. Moreover as the degree of imbalanced increase also the proportion between negative and positive support vector becomes imbalanced [9] [14].
As a consequence a test observation near to the boundary is more likely classified as negative as its neighborhood consist of only negative support vector [[1],[32]].

4.3 Decision Tree

Tree based method segments predictors space into rectangular regions, and then fit a sample model inside each regions.
Given training observations $\{x_i, y_i\}_{i=1}^{N}$, with $x_i \in R^p$, we have to split the predictor space into distinct boxes $R_1, \ldots, R_J$.

Firstly a splitting variables X_j and a split point s are considered, that determine the pair of half-planes:

$$R_1(j,s) = \{X|X_j \leq s\} \quad R_2(j,s) = \{X|X_j > s\} \tag{4.15}$$

the value of j and s are chosen to minimize:

$$Q_1 + Q_2 \tag{4.16}$$

where Q_1 and Q_2 stay for measure of node impurity, inside R_1 and R_2 respectively.

The most popular measure of node impurity are : the classification error rate, the Gini index, the cross-entropy.

In a generic node j belong to the region R_J with N_j observations, we can defined:

$$\hat{p}_{jk} = \frac{1}{N_j} \sum_{x_i \in R_j} I(y_i = k) \tag{4.17}$$

it represents the proportion of observation in the node j that belong to the class k.

The classification error rate express the fraction of training observation in the region R_j do not classified in the most common class:

$$E = 1 - \max_k \hat{p}_{jk} \tag{4.18}$$

The Gini index's a measure of the total variance inside a class K in the region R_j:

$$\sum_{k=1}^{K} \hat{p}_{jk}(1 - \hat{p}_{jk}) \tag{4.19}$$

As small values of the Gini index show that the node is composed by observations of one class, it is also see as a measure of the node purity.

The cross-entropy deviance is quite similar to the Gini index and it is defined as :

$$-\sum_{k=1}^{K} \hat{p}_{jk} \log \hat{p}_{jk} \tag{4.20}$$

After the first split the process is replicated for all the regions up to a stopping criteria is not reach.

During the classifier building process we have used measure of node impurity to determine the best splitting criterion.
All these measures are minimized when 4.17 reach values high values.
In the binary classification task this means that the most frequent occurring class is chosen at each iteration.
For imbalanced dataset this results in splitting rules that promotes the majority class.
Therefore inside each region are found association rules with high confidence but not with high significant and rules with low confidence, i.e. those involve the minority class, are discarded. [[19]] [9] [14].

Chapter 5

Case study from the insurance domain

The insurance market is all about the concepts of risk and management of risk. Nowadays insurance companies have compiled huge volume of data and extract meaningful information is the new challenge task. Machine learning techniques are employed by insurers to device strategy for customer acquisition and retention, to risk assessment, to claims prediction but also to develop procedure for prices optimisation. The last is the focus of this case study.

Reale Mutua Assicurazioni is interested to exploit information about RCA quotation, in order to test client's price sensitivity and consequently, to adjust the policy price dynamically. The price optimisation procedure give the opportunity to increase the client loyalty in the long period but also to maximize the profit of the company.
In the next section a detailed description of the case study is provided.

5.1 Case study description

The insurance company collected every day thousand of RCA quotation. Of all these quotation only a 1% are converted namely are signed. This means that we are going to handle high imbalanced data, with a positive class (converted policies) that have a greater importance respect to the majority ones for the domain experts.
In order to understand the nature of the imbalanced problem it is necessary to describe the procedure through which a client can request a quotation, and the way in which the data that will analyzed are collected.
In this case of study are considered only the online quotation and in particular those coming from the most used sale channels: "Segugio".
The customer that would request a quotation for a RCA policy through Segugio, must insert the demanding personal data in the channel. Based on these data the channel displayed a policy price. The client can decide to sign the policy and so to fix an appointment in the agency or to not sign the policy.

An main information is that the policy price is not static, but it changes during the time. As a results the clients demand several quotations before accept the proposed price. This is the main factor that influenced the imbalanced class distribution in this case of study.

5.2 Features enrichment

In this section are described the available dataset and then the process of features extraction.

There are two types of dataset: those coming from Segugio and those that coming directly from the company. The data coming from Segugio can be divided into three main areas:

1. Registry dataset.
   Inside this category there are both contractor than owner registry data (if they are not the same person) i.e : name and surname, city of residence, date of birth, profession.

2. Vehicle characteristics dataset.
   These are the data includes in the car booklet i.e set-up, model, date of enrolment, covered kilometers.

3. Insurance client history dataset.
   The risk certificate contains the information about the insurance client history i.e number of accidents, distance from the last accidents,the effective data, the risk class, but overall the merit category.

The company dataset contains all the parameters employed by the company to assign the policy price but also includes the information about the conversion (or not) of the policy.

All these dataset are not ready for the analysis, some preprocessing steps are needed to built our final dataset. First of all they have been merged in a unique dataset according with the primary keys that were in common. The others operation are following described

5.2.1 Data preprocessing

Data preprocessing is a fundamental step in Machine Learning as the quality of our data have a huge impact on the model learning. Therefore is really important analyze our data before training our model. The following steps are applied in this context:

1. Handling Nan values

2. Standardization

3. Elimination of features with low variance

4. Correlation analysis

1. **Handling Nan value**
   In any real word dataset we face with null values. It might means that this type of information or is not available or it could be the results of some previous elaboration of the dataset. As no model can deal these values we have to manage it. To this end two main approach are possible:

   (a) Drop
   We can decided to drop the rows or columns that contains null values, but it is not the best approach as it could lead to loss of information, especially in little dataset. Frequently could happen that a high number of Nan's value are associated to a particular features, so if it don't give a lot of information into the model we can deleted it.

   (b) Imputation.
   We can substitute missing value through: customized function, mean or median along the column, or a constant.

   Firstly is used the *isnull()* function presents in *pandas*, in order to have the number of null value for each fields, then the columns with more then 0.70 % for Nan's value are removed. For the others cases according with the domain experts is decided to substitute the Nan's values with a great number out of range of each sample belonging to the dataset. This choice is taken with the prospective to preserve important information.

2. **Standardization**
   Standardization is process to put variables on the same scale. This process helps to compare samples belonging to different scale. For example *Percentuale sconto aggregatore* range values(0,1) while *importo premio lordo annuo* range values (500,3000) so the learning model usually give more weight to the second variable. Through standardization we can compare scores between different types of variables.
   The score is defined in this way: $z_{i,j} = \frac{x_{i,j} - \mu_j}{\sigma_j}$.
   Therefore for each sample $x_{i,j}$ we subtract the column mean e dived for the column standard deviation.

3. **Elimination of features with low variance**.
   It is decided to delete features with variance equal to zero, so those that have the same value for all samples. This type of variable doesn't give contribution to the model explanation. For example *codice canale estrazione* is a code that identify sales channels of the policy and it is always the same for all samples (Fig 5.1)

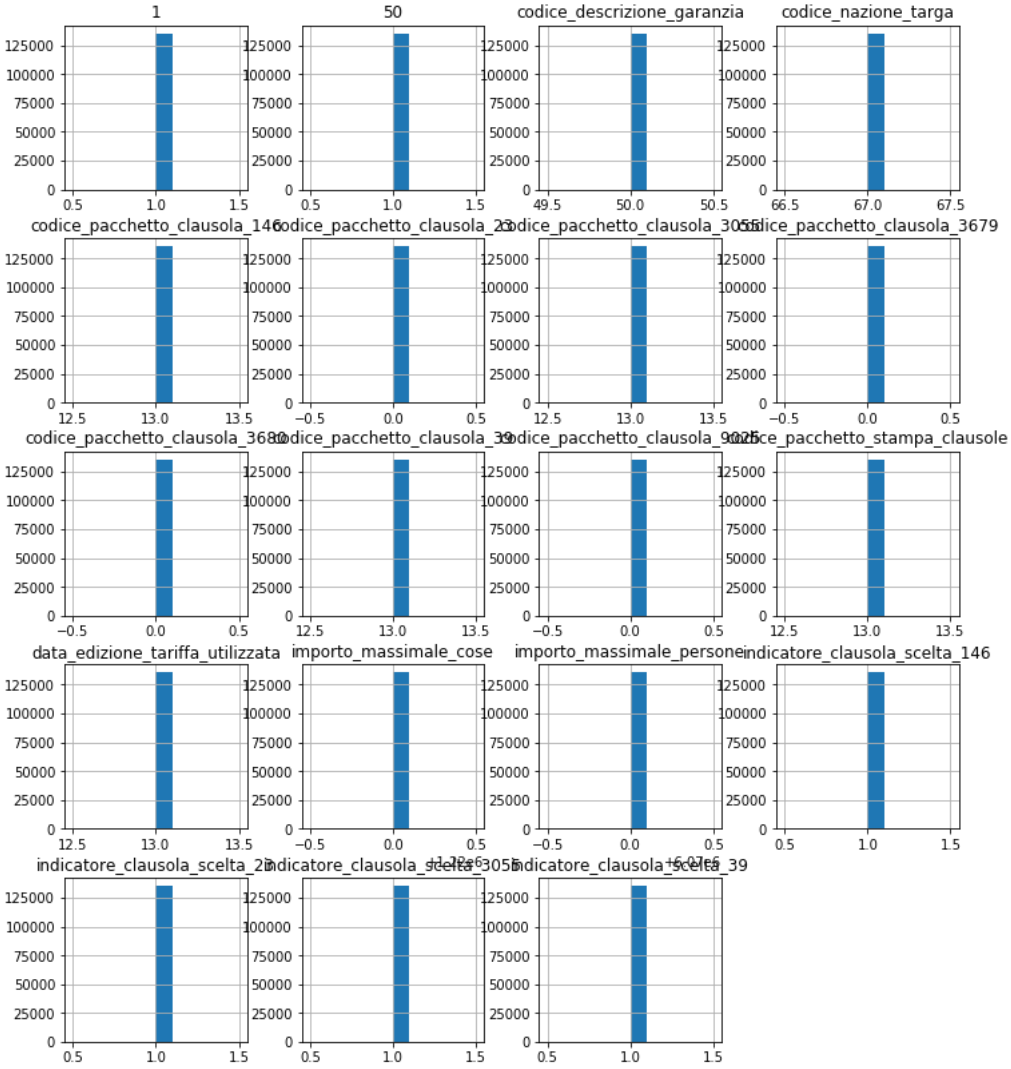


Figure 5.1: Example of features with variance equal to zero

4. **Correlation analysis**
   Correlation analysis is perform to identify the strength of relationship between a pair of variables.
   The Person correlation index coefficient between two random variables X and Y is defined as:

$$\rho = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2 (y_i - \bar{y})^2}} = \frac{Cov(X,Y)}{Var(X)Var(Y)} \quad -1 \leq \rho \leq 1$$

   A high correlation means that two variables have a strong relationship.

   As we can see from the Fig **??** there are some group of variables highly correlated. An

	importo_premio_lordo_annuo	importo_ssn	importo_premio_netto_annuo	importo_imposte	01/08/2017	importo_premio_rca
importo_premio_lordo_annuo	1	0.999	1	0.997	-0.0234	0.953
importo_ssn	0.999	1	0.999	0.997	-0.0243	0.953
importo_premio_netto_annuo	1	0.999	1	0.996	-0.0242	0.952
importo_imposte	0.997	0.997	0.996	1	-0.0239	0.951
01/08/2017	-0.0234	-0.0243	-0.0242	-0.0239	1	-0.0223
importo_premio_rca	0.953	0.953	0.952	0.951	-0.0223	1
conto_preventivi	0.0296	0.0295	0.0295	0.0294	0.00922	0.0289
importo_premio_cvt	-0.0176	-0.0183	-0.0182	-0.0179	0.843	-0.0167
percentuale_sconto_prezzo_base	-0.45	-0.451	-0.45	-0.447	0.0189	-0.247
longitudine_residenza_x	0.105	0.105	0.105	0.103	-0.00356	0.121
latitudine_residenza_x	-0.0678	-0.0663	-0.0661	-0.0762	-0.00117	-0.0948

Figure 5.2: Some examples of strong correlation between variables

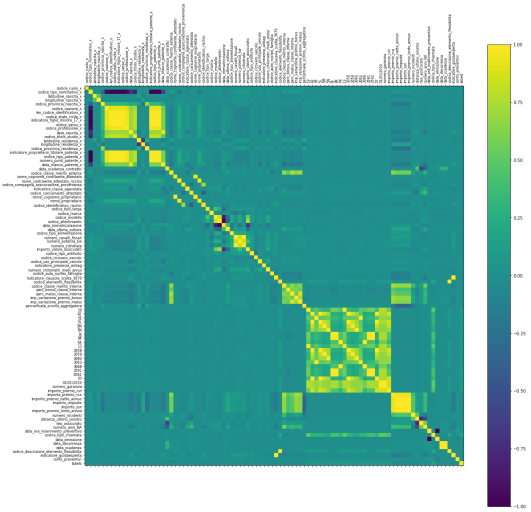
important examples is provided by the features that include the "importo" denomination (Fig 5.2). The variable *importo premio lordo annuo* is strongly correlated with *importo premio netto annuo*, *importo ssn*, *importo imposte*, *importo premio rca*. This means that if we try to predict the value of *importo premio lordo annuo* using the correlated variable through a linear regression we have a low mean square error and high variance explained (Fig 5.3).

```
Coefficients:
 [0.14127876 0.12769117 0.73139957]
Mean squared error: 0.00
Variance score: 1.00
```

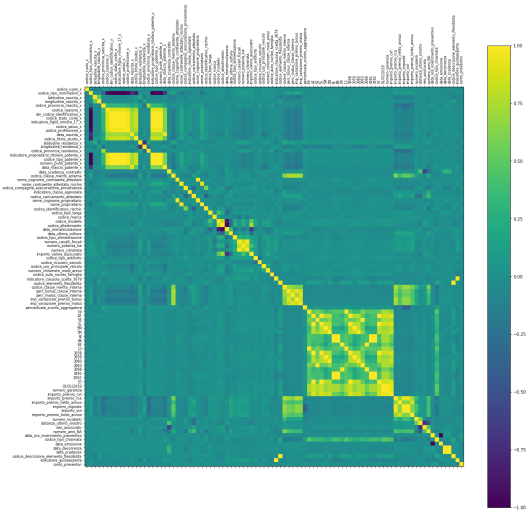
Figure 5.3: Output of *importo premio lordo annuo*'s prediction through its most correlated variables

All the information included in *importo premio lordo annuo* can be explained through the others variables.
Therefore take in account correlated features means to have redundant information in our dataset.
A good practice is to remove the features with correlation coefficient greater than a threshold.
In our case we have chose threshold equal to 0.80.
Before the elimination of these features is necessary to make a point.
The correlation coefficient is more influenced by the negative class, as this represents 99% of our samples.
Therefore it is useful to analyze separately, the behavior of the variables inside the minority class.

Analyzing the correlation inside the positive class (Fig 5.4 (b)) we can point out that the high correlation between the variables that express an amount is not preserved at all.
As an examples $'importo\_premio\_rca'$ and $'importo\_premio\_netto\_annuo'$ in the all dataset had a correlation coefficient equals to 0.95, while inside the positive class this value is decrease up to 0.66.



(a) Correlation matrix all dataset



(b) Correlation matrix positive class

Figure 5.4: Correlation matrices

	importo_premio_cvt	importo_premio_rca	importo_premio_netto_annuo	importo_imposte	importo_ssn	importo_premio_lordo_annuo
importo_premio_cvt	1.000000	-0.069728	-0.030424	-0.077234	-0.078404	-0.058716
importo_premio_rca	-0.069728	1.000000	0.662689	0.939952	0.944741	0.693836
importo_premio_netto_annuo	-0.030424	0.662689	1.000000	0.615569	0.614209	0.886228
importo_imposte	-0.077234	0.939952	0.615569	1.000000	0.996238	0.626246
importo_ssn	-0.078404	0.944741	0.614209	0.996238	1.000000	0.627471
importo_premio_lordo_annuo	-0.058716	0.693836	0.886228	0.626246	0.627471	1.000000

Figure 5.5: Amounts correlation positive class

In view of this are eliminated from our features set only the variables that have strong correlation in both scenarios.

The final datset contains 190K samples and 50 features. The imbalance between the two classes is about 1 %. Now we are ready to process our data, set up configuration and experiments are described in the next chapter.

Chapter 6

Experiments

The experiments can be divided into two part.
In the first Decision tree classifier, Random Forest classifier, Logistic Regression, Support Vector Classifier are used with the parameters in table 6.1. Each classifiers performances is evaluated in terms of F-measure. This measure is obtained after a stratified five-folds cross validation. The final F-measure is the average of the values obtained in the five validation sets. The classifiers are applied on the original set and on the sets obtained employing several re-sampling techniques (table 6.2). From these experimental results it is emerged the incapability of all the classifier to better recognize element of the minority class, that corresponds to low values of recall measures. When this value sightly increase such as in decision tree and random forest, applying RUS techniques, a drop in the precision is registered resulting in low F-measure scores. Despite this, Decision Tree classifier without the application of any re-sampling techniques results the best model, but with a recall only about of 64%. This is compensated by a high precision of % 80 that lead the F-measure about 74%, that is best value among all the tested approaches. A very similar results in terms of F-measure is obtained employing Tomek Link and Neighborhood cleaning rule as preprocessing techniques and decision tree as classifiers. As these two cleaning techniques are not able to delete a significant amount of majority examples from the training set, the training set in the original scenario and in this two cases are quite the same. Once it has been identified the best classifier and the best re-sampling techniques the first part of the experiments is concluded.

In the second part of the experiments an algorithm level approach is employed to improve the performance of the best control algorithm. To this end a study of the Gradient boosting decision tree classifier will be carried out. In the Gradient boosting classifier the imbalanced classes is handle by the learning process itself and we will see that there is not the need to employ any data level techniques. It is able to reach the highest value of F-measure 84%, including a recall of 78%. A great advantage of the gradient boosting classifier is the easy interpretability of its results. In the last part of this chapter an analysis of the variables importance inside this model will be presented but also an example of the model limitations.

6.1 Experimental design

All the experiments have been carried out using *Jupiter Notebook* in which it is installed *Python 3.7* version. Jupiter notebook sever was executed remotely on an Azure Databricks cluster with 14.0 GB Memory, 4 Cores, 0.75 DBU.

Before testing any models it is important to split our dataset into training and test set (sometimes we used also a validation set) in order to avoid over or under fitting.
This may happen when we built a model that performances very well on the training set but it will be not so accurate in front of new data.
This means that the model is not generalizable so we can not extend our prediction to others data.

In **Sklearn** there is a sublibrary *model selection* that contain the "*train_test_split*" function which require:

- the training size
- the test size
- if we want to shuffle the data before splitting
- if we want to split the data in a stratified way, using the class label for example.
- if we want to shuffle the data of instances to put in the training and test set

Our dataset consists in 196K samples and 50 features. It is decided to put the test size equal to 0.20, to shuffle the data and especially to stratified respect to the class label in order to train a model with the same criticality of the original dataset. The training set is next divided into five stratified folds. Four of these folds are used for training the classifiers and the last block as validation set. This further splitting is made in order to validate the best parameters configuration for a classifier or the best performances towards different model. In this way, once the control method (or the best configuration of parameters) is identified it can be tested on a new set of data, that we have called test set. This procedure lead to a great reduction of the chance of overfitting problems.

The classifiers employed are:

- Support vector machine
- Logistic Regression
- Random forest
- Decision tree

Algorithm	Parameter	Value
SVC	C (penalty)	1
	Kernel	Linear
Logistic Regression	penalty	l_2
Random Forest	max_features	$2\sqrt{n\_features}$
	number_estimators	100
	criterior	Gini
Decision Tree	min_samples_leaf	5
	Criterior	Gini

Table 6.1: Experimental setting for classical classification algorithm

In order to evaluate the classification performances, the following measure of performances are employed:

- Precision (P)
- Recall (R)
- F1 score (F1)
- Specificity (TN)
- The area under the Receiver Operating Characteristic (AUC)

One way to solve the imbalanced learning problems is to modify the class distribution in the training data by under or over sampling.
In the following experiments several re-sampling techniques are compered (Table 6.2), in order to investigate their contribution on the classifiers performances.

Re-sampling strategy	Technique	Reference
Undersampling	RUS	
	Tomek link	[27]
	Neighborhood cleaning rule	[18]
Oversampling	ROS	
	SMOTE	[4]
	Kmeans SMOTE	[17]

Table 6.2: Re-sampling tested techniques

6.2 Experiments results on the resampling techniques

This section aim is twofold. Firstly we want to analyze the impact of several resampling strategy on our imbalanced dataset. To this end several under and oversampling performances are compered in terms of F-measure metric including also the performance in the original dataset. According with this study we want to establish if the data level approach is able to solve the imbalance problem and which are the best strategy and best classifiers performances.
The average and standard deviation F-measure results on the five test sets of the cross validation process for each classifiers and each preprocessing techniques are shown in Fig 6.4. From this graph it is excluded the Logistic Regression as its performances are resulted very low and unchanged in all the problems and so it is not considered of interest for this analysis. In tables 6.1,6.4,6.5 are reported more detailed information including the precision, recall,specificity and AUC mean values for each problem.
As one might be expect the performances applying random undersample is the worst for all classifiers. The random undersampling delete from the majority class elements in a random way up to a balance with the number of minority class elements is reached. This lead to the elimination of 99% of our training data, and consequently to a high loss of information resulting in very poor performances. On the opposite scenario when over-sampling techniques are employed, a number of positive elements equals to the difference with the number of the examples in the majority class are generated. In this scenario we have an extreme number of new samples. The training set size in each folds doubles respect to the natural set of data. When we test the training model on each validation folds, that is not subject to oversampling, the results is a degradation of all the classifiers performances. Only the average F-measure value obtained with random oversampling when the decision tree is used as classifier is not so corrupted. In other to better understand this results, we remember that the F-measure metric is a weighted average of precision and recall. If we look at the results in table 6.5 we can observe that all the precision values, compares to the values of the underlying performances in the imbalanced dataset, in table 6.3, are interested by a huge drop. This means that including a great number of new positive examples into the training sets, lead to an increase in the false positive rate. Classifiers tends to overestimate the elements of the majority class.

The best performances are obtained: in the original scenario, applying Tomek Link and Neighborhood cleaning rule. The the last two are two data cleaning techniques. Tomek Link and Neighborhood cleaning rule act on the original dataset deleting from the majority class all the noise and borderline examples. In Figure 6.5 we can observe that only applying the Neighborhood rules we can observe a slight reduction on the majority class examples. On mean are deleted 567 majority examples on each training folds sets applying Tomek link, and 3432 with NCR techniques.

From this preliminary analysis is revealed that the decision tree classifier is the most robust classifiers. While these data-level approaches are not able to handle effectively the imbalanced problems. In order to ascertain this statements a statistical analysis is following carry out.

One-way ANOVA is a practical statistical technique to evaluate if two or more population means are different or not, or better, to understand if different treatments are able to produce a concrete effect on the control variable. In this case it is tested if the F-measure values obtained by the decision tree are influenced by three types of treatments: the Tomek Link, Neighborhooh cleaning rule and any resampling.
Firstly, it is necessary to fit a linear regression model to predict the F-measure value in function of the treatments and to check from this previous analysis if this model explained a significant amount of variance or not. The results in Fig 6.1 shows that the F-statistic of the model is about 2.007 and the p-values 0.154, this tells us that the there is not a significant difference in groups means. It is pointed out by the p-values of the coefficients in the output, that are all greater than 0.05. This p-values are the results of the t-test between each mean groups and the intercept, that corresponds to the model without resampling. In order to ascertain the linear regression outputs we can see from the figure 6.2 and the figure 6.3 that normality and homoscedasticity of the residuals are verified.

OLS Regression Results

Dep. Variable:	f_measure	**R-squared:**	0.273
Model:	OLS	**Adj. R-squared:**	0.137
Method:	Least Squares	**F-statistic:**	2.007
Date:	Fri, 22 Nov 2019	**Prob (F-statistic):**	0.154
Time:	19:08:11	**Log-Likelihood:**	34.751
No. Observations:	20	**AIC:**	-61.50
Df Residuals:	16	**BIC:**	-57.52
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.7418	0.021	34.848	0.000	0.697	0.787
C(resampling)[T.ncr]	-0.0171	0.030	-0.567	0.578	-0.081	0.047
C(resampling)[T.ros]	-0.0586	0.030	-1.947	0.069	-0.122	0.005
C(resampling)[T.tl]	0.0096	0.030	0.319	0.754	-0.054	0.073

Figure 6.1: Linear regression results

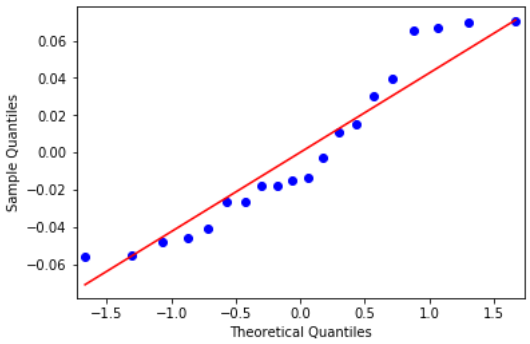


Figure 6.2: Residuals normality distribution

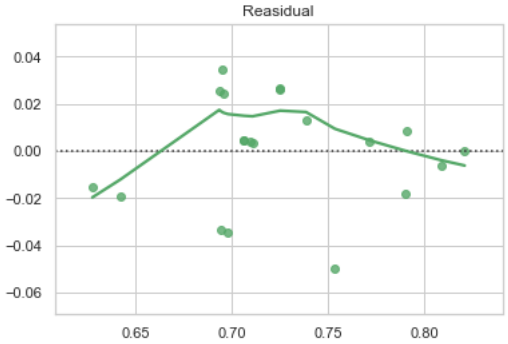
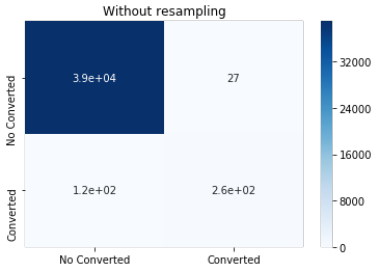


Figure 6.3: Residuals homoschedasticity

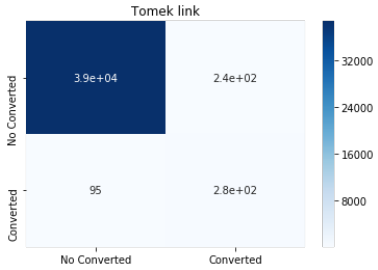
In conclusion the performance of the Decision tree applying Tomek Link, Neighborhood cleaning rules and in the natural scenario are statistically equivalent. Now that we have identify the best models we can test them on the test set.

The results on the test set shows that the Decision tree better performs when we don't employed any resampling techniques. The corresponding F-measure are 77.6% , 62.5%, 64.1%.

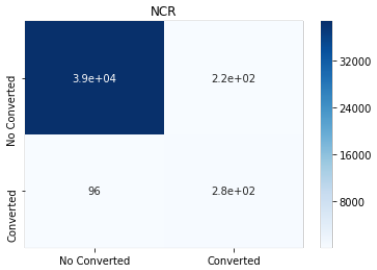
It is relevant now, to reintroduce our objective of this study. On one side we want to analyze the impact of imbalanced data on classifiers performances, on the others we want also to find a models that give us meaningful information about the positive class and high accuracy on the underlying class. The decision tree recall is only about 64% . In the next section it is applied a new technique to improve this value.



(a) Test set confusion matrix natural scenario



(b) Test set confusion matrix applying Tomek Link



(c) Test set confusion matrix applying NCL

Algorithm	P	R	F1	TNR	AUC
SVC	0.809	0.454	0.474	0.97	0.716
LR	0.011	0.383	0.022	0.674	0.528
RF	0.975	0.419	0.585	0.99	0.70
DT	0.808	0.642	0.741	0.99	0.82

Table 6.3: Experimental results with the natural sets

Classifier	Technique	P	R	F1	TNR	AUC
SVC	RUS	0.056	0.50	0.08	0.75	0.55
LR	RUS	0.011	0.383	0.02	0.675	0.52
RF	RUS	0.05	0.869	0.108	0.85	0.85
DT	RUS	0.045	0.803	0.086	0.834	0.818
SVC	Tomek link	0.214	0.614	0.168	0.69	0.65
LR	Tomek link	0.011	0.38	0.02	0.675	0.528
RF	Tomek link	0.980	0.428	0.591	0.99	0.711
DT	Tomek link	0.895	0.654	0.775	0.99	0.826
SVC	NCR	0.605	0.653	0.450	0.783	0.718
LR	NCR	0.011	0.393	0.022	0.671	0.532
RF	NCR	0.976	0.416	0.583	0.99	0.708
DT	NCR	0.863	0.624	0.724	0.99	0.811

Table 6.4: Under-sampling experimental results

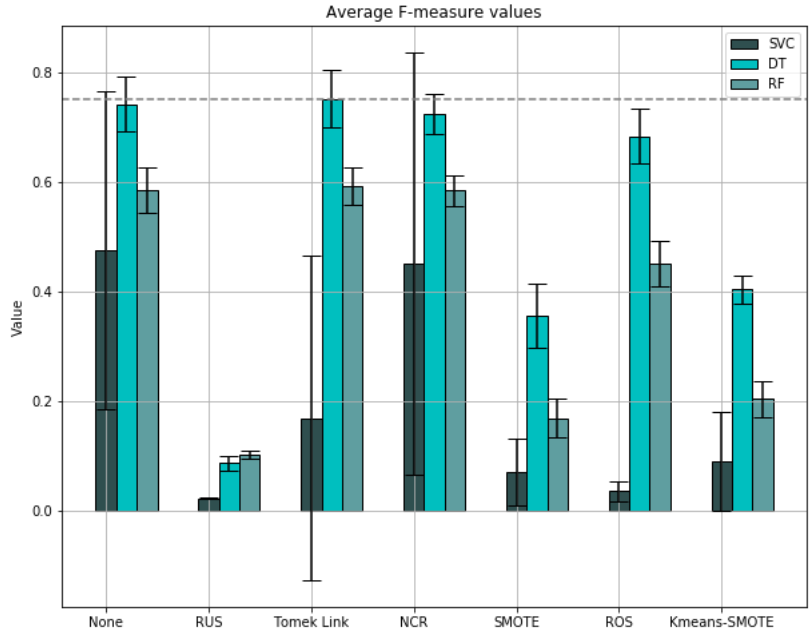


Figure 6.4: Average and standard deviation F-measure values results on the resampling experiments

Figure 6.5: Training classes distribution applying undersampling techniques

Classifier	Technique	P	R	F1	TNR	AUC
SVC	ROS	0.0118	0.686	0.034	0.553	0.619
LR	ROS	0.010	0.353	0.020	0.675	0.513
RF	ROS	1	0.291	0.450	1	0.645
DT	ROS	0.677	0.690	0.683	0.99	0.843
SVC	SMOTE	0.038	0.728	0.567	0.069	0.673
LR	SMOTE	0.010	0.352	0.02	0.675	0.513
RF	SMOTE	0.909	0.092	0.167	0.99	0.546
DT	SMOTE	0.365	0.346	0.354	0.994	0.67
SVC	KMeans	0.054	0.713	0.09	0.594	0.65
LR	KMeans	0.010	0.352	0.020	0.675	0.513
RF	KMeans	0.471	0.352	0.271	0.996	0.674
DT	KMeans	0.471	0.352	0.271	0.996	0.674

Table 6.5: Over-sampling experimental results

6.3 Experiments results on the gradient boosting classifier

In this section it is employed a boosting ensemble method to handle with the imbalanced data problem: the Gradient boosting decision tree classifiers. As we have see in the description of this model the gradient boosting tree classifiers is built using small decision tree as "weak learner". The boosted tree is the sum of each tree. The revolutionary idea inside the gradient boosting algorithm is that each tree is fitted on the negative gradient of the loss function computed on boosted tree obtained in the previous iteration. At each iteration, the estimated constants assigned to each nodes in the new boosted tree, are optimized on the errors committed by their predecessors. We will see that this strategy lead to a very powerful model not only able to handle the imbalanced class problems, but also to give us interpretative results, that will have a main role in the business decision context.

Firstly it will be presented the experiments results of the classifier applied on our imbalanced dataset. Then an explanation of the variables importance is carried out. Lastly is presented an example of the model limitations.

Tuning Hyper-parameters

In the Gradient boosting classifier as many complex algorithm is subject to overfitting. In order to handle this issue it is of main importance the role of two hyperparameters:

- Learning rate
  The learning rate controls the residuals impact on the final output in each decision tree. Lower values are generally preferred as they give high robustness to the model

- Number of estimators
  Number of sequential tree to be model. There's a trade-off between the learning rate and the number of trees needed, so it is necessary to jointly cross validate these parameters.

To this end a grid search is implemented using a five folds cross validation and F-measure criterion to evaluate the performances.

Parameters	Values
Num_estimators	[500,600,700,800,1000]
Learning rate	[0.01,0.05,0.10]
CV	5
scoring	F1

Table 6.6: Hyperparameter cross validated

The output of this grid search 6.6 points out that the best parameters are: 1000 iterations and a learning rate of 0.10.

```
1 print(gd_sr_gbc.best_estimator_)
```

```
GradientBoostingClassifier(criterion='friedman_mse', init=None, learning_rate=0.1, loss='deviance', max_depth=3, max_features=None,
max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1, min_samples_split=2, min_weight_fraction_leaf=0.0,
n_estimators=1000, presort='auto', random_state=0, subsample=1.0, verbose=0, warm_start=False)
```

Figure 6.6: Grid search best model

With the same procedure seen for the resampling method are computed the metrics measures for the gradient boosting classifiers on the imbalanced dataset. In fig 6.7 are reported the average values of recall, precision and f-measure. The gradient boosting classifier reach a recall of 78% compare with the 64% of the decision tree and a f-measure about 84% against the 74% of the other classifier.

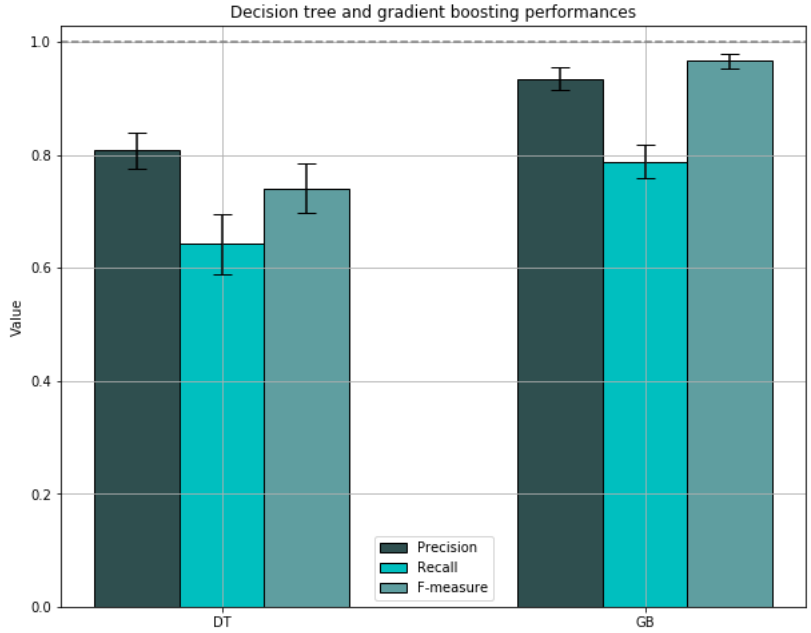


Figure 6.7: Comparison of the performance measures on DT and GB

Gradient boosting classifier with this parameter configuration outperform all the others methods. It is able to recognized the element of the positive class preserving a high precision rate.

In order to understand if this model is overfitting our data, the trend between training and test loss function it is analyzed.
The configuration of the Gradient boosting classifier that it is choose in this test, employed the binomial negative likelihood loss. The plot in Fig 6.9 shows that the training and test error scores follow the same trend. As one can be attend the test loss function lies above the training ones. Moreover this graph demonstrate also that after a number of iteration equal to 500 more or less the complexity of model increases but there is no an improvement in the error function.

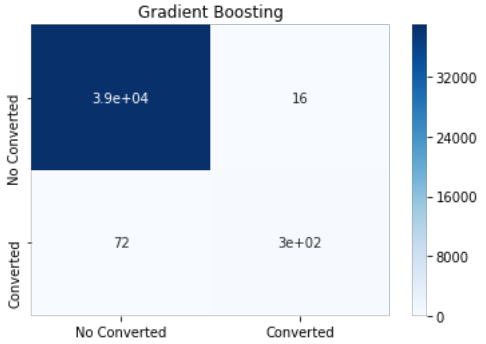


Figure 6.8: Confusion matrix

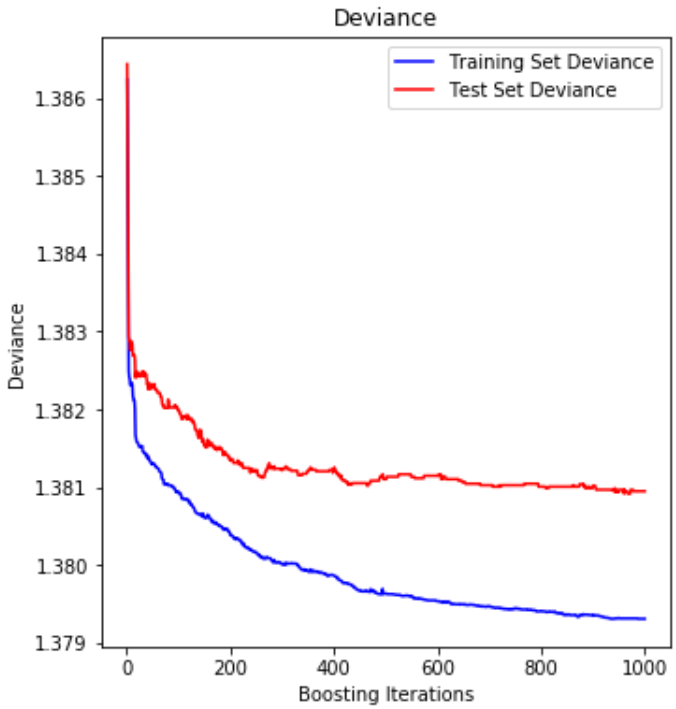


Figure 6.9: Comparison of loss deviance in the training and in the test set

Once establish the robustness of our model we can employed it to extract important interpretable information about the relationship between the target response and the predictors.

6.3.1 Results interpretation

The Gradient boosting classifiers is not only able to achieve the highest accuracy on the positive class, but it gives also interpretable results. In the business cases it is of fundamental importance as the interest of what happen inside the "black-box" might help the decision makers.
In this subsection's presented an investigation about the relative importance of the features on the output variables.

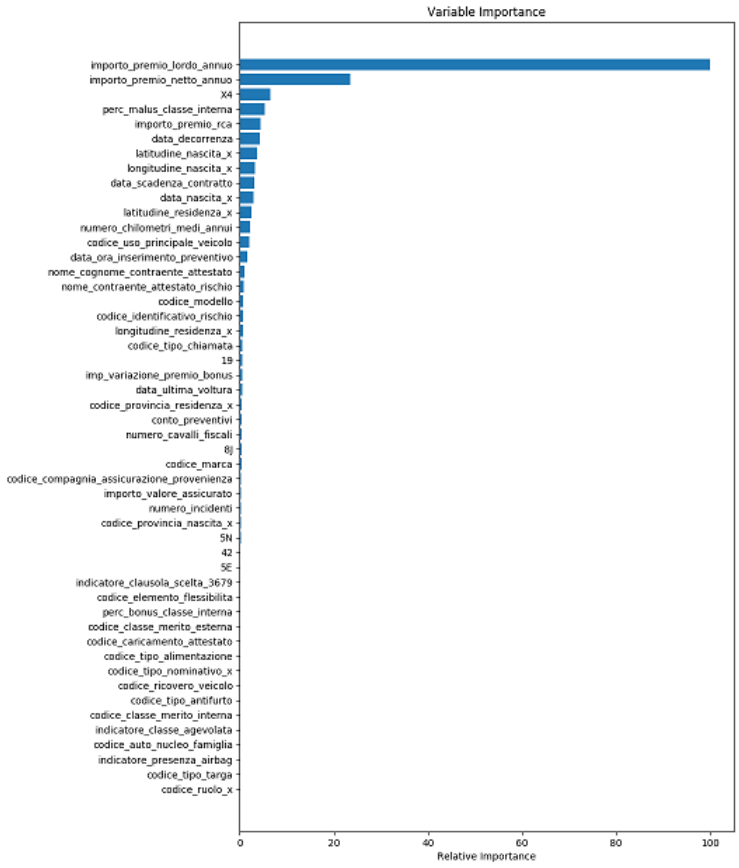


Figure 6.10: Features relative importance gradient boosting classifier

The gradient boosting classifier gives us an important metric to evaluate the contribution of each features in the target output: the relative features importance. We remember that the relative variables importance quantifies the improvements that of each predictor gives in the squared-error when it is used as splitting variables on the decision tree. In the Figure 6.10 is displayed the relative variables importance for each features in dataset. As one might be expect the features that are discriminant for classification are the "amounts", followed by geographical information. This means that the customer is clearly influenced by the policy price when he have to decide if sign or not the contract. This statement will be latter explained.

Now that it is known that only a limited group of variables are relevant for the classification, the features space we will be built according with their importance scores.
For each value of the features importance scores a subset of variables is introduced into the model, then the new model with the subset of features selected is validated through five folds cross validation and the mean F-measure is collected.

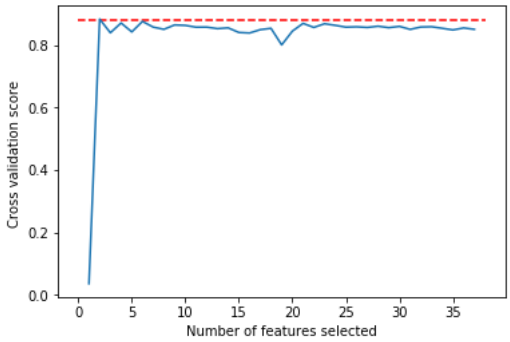


Figure 6.11: Features validation selection F-measures scores

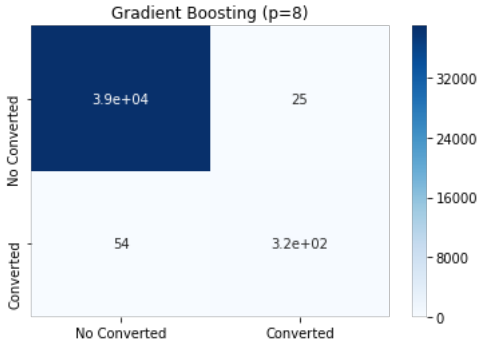


Figure 6.12: Confusion matrix performance with the eight most important features

The graph (6.11) shows that only the features with the eight highest scores gives an improvements on the models in terms of F-measure, then it reaches a platau. Accordingly with these results the model is retrained and tested (6.12)

6.3.2 An example of the model limitations

In this subsection some limitation of our models are carried out.
From the previous section is carried out that the " gross amounts" is the most important variable in our model. This variables is the most influence in the discrimination between the two classes in the classification task.
An analysis of its distribution inside the target classes and in the whole dataset might be

of interest to understand its behavior.

If we look at the distribution of this variable inside the dataset (Figure 6.13) we can observe the presence of a long right tail, or better of values that are very far from the mean. Through the standardization of this distribution we can easily detect these extreme values. To this end it is chosen a *z_score* grater than 2.5 (red line in 6.13). Before delete this examples from our training set we can observe from the scatter plot in Figure 6.14 that they are discriminating between the classes. Or, raher, all the quotation with a gross amount greater than this *z_score* belongs to the class of the not converted.

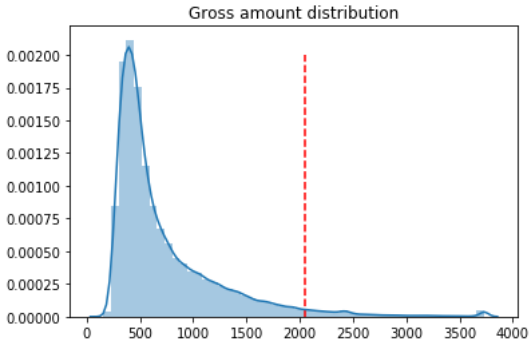


Figure 6.13: Gross amount distribution

It is natural to wonder what happens in our classification process if these extreme values are put out from our training set. The confusion matrix in Figure 6.15 shows a drop in the precision measure in this scenario. This means that the classifier is less accurate when it faces very similar gross amounts inside both the classes.

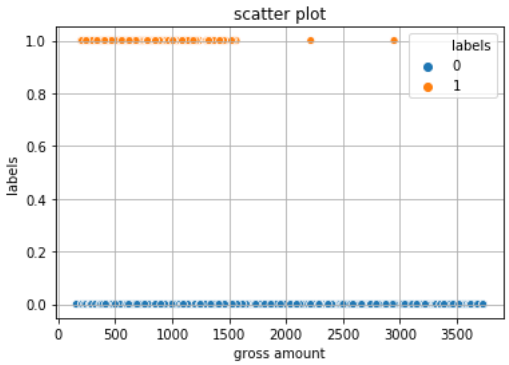


Figure 6.14: Scatter plot between the gross amount and labels

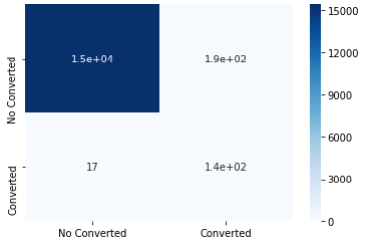


Figure 6.15: Confusion matrix performance after the elimination of gross amount extreme values

In conclusion the gradient boosting classifier is a very powerful classifier able not only to reach a high performances in an extreme imbalance scenario, but also to give us very useful information about the importance of the predictor inside the models. But in business decision context it is important to take into account the phenomena observed at last, and to investigate the others factors that will be determinant or more discriminant in this scenario.

Chapter 7

Conclusion

The main purpose of this thesis concerns the investigation of the problem of classification in imbalanced data. First, a theoretical study about the nature of the problem and the state of art solutions has been presented. Through this study, has been pointed out that there are two main ways to handle the issues of imbalanced data. We can act directly on the training data applying several resampling strategies in order to balanced the classes distributions, or we can modify the existing learning algorithms for biasing the process towards minority class. In the case of study examined in this thesis both the strategies are experimented on a real imbalanced data. Three undersampling and three oversampling techniques were applied on the trainind dataset and their performances among three different classifiers were compared . It has been demonstrated that every data level algorithm were not capable of solving the imbalanced data problem. Random undersampling was the one that showed the worst performances among all the classifiers, as it deleted a huge amount of sampling in other to balance the classes taking to the loss of information. In case of SMOTE and K-means SMOTE it has been observed the opposite scenario. In the training over-sampled folds are introduced on mean 118k new synthetic examples belonging to the positive class and when the classifiers are tested in the imbalanced validation folds their performances is very poor independent from the classifier. Tomek Link and Neighborhood cleaning rule, on the other side, are not able to delete a sufficient number of examples from the majority class, their performance in the case of random forest and decision tree is the same of that in the original dataset. From this first analysis emerged that the decision tree classifier is the more robust classifier and its best performance is reach in the base case without any resampling approach. It reached a F-measure about 74%, a precision of 80% and a recall only about 64%. In order to increase the recall value the performance of the gradient boosting decision tree classifiers is tested. In the gradient boosting classifiers each decision tree is fitted on the residuals, that minimizing the loss function. The boosted tree is the sum of each tree. It has been shown to be a very powerful classifiers able not only to reach the highest performance in terms of F-measure, precision and recall (that is about 74%) but also to gives us information about the importance of the variables in our model. Understand what happens inside the "black-box" is of primary relevance in business decision contest. The results has been suggest that the gross amount is the most relevant features. It plays

a fundamental role in the discrimination between the classes. It has also been an example of the limitation of this model. When the extreme values of gross amount are excluded from the training dataset, and so we have more or less the same value of this variable among both the classes , a drop in the model precision has been observed.

7.1 Future works

The future developments of this work are twofold. On one side one could think to improve the gradient boosting model performance on the critical scenario observed. In this particular cases it will be studied the interaction effects between the others important variables in order to understand if there are some others factors that might influence customer decision in this context. On the other side integrating new data in the model will be studied the customers behaviors on a larger temporal space.

Bibliography

[1] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. "Applying support vector machines to imbalanced datasets". In: *European conference on machine learning*. Springer. 2004, pp. 39–50.

[2] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. "DBSMOTE: density-based synthetic minority over-sampling technique". In: *Applied Intelligence* 36.3 (2012), pp. 664–684.

[3] Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. "Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem". In: *Pacific-Asia conference on knowledge discovery and data mining*. Springer. 2009, pp. 475–482.

[4] Nitesh V Chawla et al. "SMOTE: synthetic minority over-sampling technique". In: *Journal of artificial intelligence research* 16 (2002), pp. 321–357.

[5] Misha Denil and Thomas Trappenberg. "Overlap versus imbalance". In: *Canadian Conference on Artificial Intelligence*. Springer. 2010, pp. 220–231.

[6] Georgios Douzas, Fernando Bacao, and Felix Last. "Improving imbalanced learning through a heuristic oversampling method based on k-means and SMOTE". In: *Information Sciences* 465 (2018), pp. 1–20.

[7] Vicente Garcıa, Jose Sánchez, and Ramon Mollineda. "An empirical study of the behavior of classifiers on imbalanced and overlapped data sets". In: *Iberoamerican Congress on Pattern Recognition*. Springer. 2007, pp. 397–406.

[8] Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. "Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning". In: *International conference on intelligent computing*. Springer. 2005, pp. 878–887.

[9] T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, 2009. ISBN: 9780387848846.

[10] Haibo He and Edwardo A Garcia. "Learning from imbalanced data". In: *IEEE Transactions on knowledge and data engineering* 21.9 (2009), pp. 1263–1284.

[11] Haibo He et al. "ADASYN: Adaptive synthetic sampling approach for imbalanced learning". In: *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*. IEEE. 2008, pp. 1322–1328.

[12] Robert C Holte, Liane Acker, Bruce W Porter, et al. "Concept Learning and the Problem of Small Disjuncts." In: *IJCAI*. Vol. 89. Citeseer. 1989, pp. 813–818.

[13] Feng Hu and Hang Li. "A novel boundary oversampling algorithm based on neighborhood rough set model: NRSBoundary-SMOTE". In: *Mathematical Problems in Engineering* 2013 (2013).

[14] Gareth James et al. *An introduction to statistical learning*. Vol. 112. Springer, 2013.

[15] Taeho Jo and Nathalie Japkowicz. "Class imbalances versus small disjuncts". In: *ACM Sigkdd Explorations Newsletter* 6.1 (2004), pp. 40–49.

[16] Miroslav Kubat, Stan Matwin, et al. "Addressing the curse of imbalanced training sets: one-sided selection". In: *Icml*. Vol. 97. Nashville, USA. 1997, pp. 179–186.

[17] Felix Last, Georgios Douzas, and Fernando Bacao. "Oversampling for Imbalanced Learning Based on K-Means and SMOTE". In: *arXiv preprint arXiv:1711.00837* (2017).

[18] Jorma Laurikkala. "Improving identification of difficult small classes by balancing class distribution". In: *Conference on Artificial Intelligence in Medicine in Europe*. Springer. 2001, pp. 63–66.

[19] Wei Liu et al. "A robust decision tree algorithm for imbalanced data sets". In: *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM. 2010, pp. 766–777.

[20] Giovanna Menardi and Nicola Torelli. "Training and assessing classification rules with imbalanced data". In: *Data Mining and Knowledge Discovery* 28.1 (2014), pp. 92–122.

[21] Krystyna Napierała, Jerzy Stefanowski, and Szymon Wilk. "Learning from imbalanced data in presence of noisy and borderline examples". In: *International Conference on Rough Sets and Current Trends in Computing*. Springer. 2010, pp. 158–167.

[22] Iman Nekooeimehr and Susana K Lai-Yuen. "Adaptive semi-unsupervised weighted oversampling (A-SUWO) for imbalanced datasets". In: *Expert Systems with Applications* 46 (2016), pp. 405–416.

[23] Thomas Oommen, Laurie G Baise, and Richard M Vogel. "Sampling bias and class imbalance in maximum-likelihood logistic regression". In: *Mathematical Geosciences* 43.1 (2011), pp. 99–120.

[24] Ronaldo C Prati, Gustavo EAPA Batista, and Maria Carolina Monard. "Class imbalances versus class overlapping: an analysis of a learning system behavior". In: *Mexican international conference on artificial intelligence*. Springer. 2004, pp. 312–321.

[25] José A Sáez et al. "SMOTE–IPF: Addressing the noisy and borderline examples problem in imbalanced classification by a re-sampling method with filtering". In: *Information Sciences* 291 (2015), pp. 184–203.

[26] Chris Seiffert et al. “An empirical study of the classification performance of learners on imbalanced and noisy software quality data”. In: *Information Sciences* 259 (2014), pp. 571–595.

[27] Ivan Tomek. “Two modifications of CNN”. In: *IEEE Trans. Systems, Man and Cybernetics* 6 (1976), pp. 769–772.

[28] Gary M Weiss. “Learning with rare cases and small disjuncts”. In: *Machine Learning Proceedings 1995*. Elsevier, 1995, pp. 558–565.

[29] Gary M Weiss. “Mining with rarity: a unifying framework”. In: *ACM Sigkdd Explorations Newsletter* 6.1 (2004), pp. 7–19.

[30] Gary M Weiss and Haym Hirsh. “A quantitative study of small disjuncts”. In: *AAAI/IAAI* 2000 (2000), pp. 665–670.

[31] Dennis L Wilson. “Asymptotic properties of nearest neighbor rules using edited data”. In: *IEEE Transactions on Systems, Man, and Cybernetics* 3 (1972), pp. 408–421.

[32] Gang Wu and Edward Y Chang. “Class-boundary alignment for imbalanced dataset learning”. In: *ICML 2003 workshop on learning from imbalanced data sets II, Washington, DC*. 2003, pp. 49–56.

[33] Show-Jane Yen and Yue-Shi Lee. “Cluster-based under-sampling approaches for imbalanced data distributions”. In: *Expert Systems with Applications* 36.3 (2009), pp. 5718–5727.

[34] Jaesub Yun, Jihyun Ha, and Jong-Seok Lee. “Automatic determination of neighborhood size in SMOTE”. In: *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication*. ACM. 2016, p. 100.