Advanced Machine Learning
Kaggle - BNP Paribas Cardif Claims

Antonios Andronis*, Pengfei GAO†, Napoleon Koskinas‡,
Syed Muhammad Ali Shah§ and Nikolaos Perrakis¶
University of Southampton
School of Electronics and Computer Science
Data Science Postgraduate Students
Email: *aa3e15@soton.ac.uk , †pg6g15@soton.ac.uk , ‡nk5g15@soton.ac.uk ,
§smas1c15@soton.ac.uk , ¶np4g15@soton.ac.uk

Abstract

In this report we present the approach from Team Gauss on the Advanced Machine Learning project. We worked on the BNP Paribas Cardif Claim dataset, a 2 pattern classification problem. The dataset had nominal categorical features and many missing values. We applied several supervised learning algorithms and evaluated their predictive accuracy. Feature construction played a key part in creating an accurate model, especially because of the anonymization, and the missing values of the dataset. Overfitting was a prevalent problem on many methods and generalisation ability was a close second. We overcame them by choosing appropriate parameters for our algorithms but did not fully avoid them.

I. Introduction

Machine Learning is a field that merges Computer Science and Statistical Learning. It has been described as the "field of study which gives computers the ability to learn without being explicitly programmed"[1]. Of course computers have not yet become completely independent, but the evolution of Machine learning methods greatly contributed to the progress made in the filed of Artificial Intelligence. In recent years, when huge amounts of data concerning every sector are produced and stored worldwide, Machine Learning techniques have been flourishing as they enable computers to learn from available data and create models which enhance prediction and decision making. Data scientists, analysts and researchers develop and consequently apply these analytical modelling methods building on knowledge coming from pattern recognition and computational learning theory. In addition, they need to have a deep understanding of statistics, mathematical techniques and probabilistic modelling in order to apply the right method on every case.

In Machine Learning there is not a method which can be considered as "panacea", so the concerned Machine Learning practitioner should originally identify the kind of problem he has to face and analyse its nature to decide the process that needs to be applied. Firstly, he needs to note whether the problem belongs to supervised or unsupervised learning. Thereafter, comes the first contact with the available data which leads to the appropriate preprocessing techniques to be applied so that the data transforms into a format on which machine learning algorithms can perform. The actual analysis takes place when one or more machine learning methods are applied on training data and a prediction model is trained. This model is later applied on the test data in order to evaluate its efficiency on predicting unseen data by learning from data coming from the same source or distribution. In the end comes the evaluation of the model which can lead either to feedback to the whole analysis process in case the one applied did not work efficiently enough, or to the final conclusions and insights which can be extracted by modelling data.

The fore-mentioned authors, for the needs of Advanced Machine Learning module in the University of Southampton in academic year 2015-2016, decided to form a five-member group which undertook

a real-world Machine Learning challenge and enabled them to apply the pipeline described above and obtain useful experience and skills.

II. Project Background

Insurance companies are working very hard to facilitate their clients for providing the best services especially at the time when their clients are facing sudden events in their life. In order to exceed the client expectation specifically in terms of claim approval and imbursements, insurance companies are using complex machine learning and data mining techniques to perform classification of true and false claims, risk assessment and payment process optimization.

False claims are increasing at the accelerated rate in UK[2] People are using very creative ideas about submitting false claims to insurance companies that apparently look a straightforward case for approval. It is a big challenge for the insurance company to timely classify the fraud claim and process the genuine case. Insurance companies uses the claim history and data from loss indicators to perform predictive analysis on people who can potentially do false claim.

The project team decided to research on insurance claim management subject by participating in Kaggle competition where a French insurance company named BNP Paribas Cardif has provided an opportunity to Data Scientists to work on the dataset the concerned company provided. Within this competition, Kagglers are challenged with assessing the validity of insurance claims.

The problem is a two-class classification problem, so the project team needed to apply supervised learning techniques in order to classify the claims as:

1) Claims which have enough information to process and decision could be taken straight away for approval and payments.
2) Claims which have not enough information to process and need more information for decision.

The requirement of the competition from BNP Paribas Cardif was to predict a probability of classifying a claim to a class. Each submission is evaluated with a logarithmic loss metric which punishes both incorrect classification but also high confidence on misclassification.

III. Machine Learning and Synchronization Tools

The project was implemented in Python (version 2.7). To implement supervised learning algorithms we relied on scikit-learn[3], an efficient tool for data analysis that contains a wealth of algorithms to use. Additionally we used pandas and numpy, two powerful libraries providing easy-to-use data manipulation and analysis tools. We also used matplotlib library to visualize the results of our exploratory data analysis and learning outcomes. Very useful for the progress of the project was also Jupyter IPython Notebook, that helped team collaboration by sharing visualised results conveniently among team members.

Furthermore, we used Microsoft's OneNote for the purpose of synchronization and keeping track of meetings. Finally, we used the LaTeX editor and Powerpoint to create our report and presentation respectively.

The source code was placed in a GitHub[4] repository. This enabled us to work separately on the code. This way we could divide the work between the team members and handle version control efficiently.

IV. Dataset Exploration

The training and test sets provided by the competition are roughly 115 Mb each. The data has been anonymized and contains 131 features. As we have already mentioned we have to face a classification problem. The portion of each class in the training data is shown in figure 1.

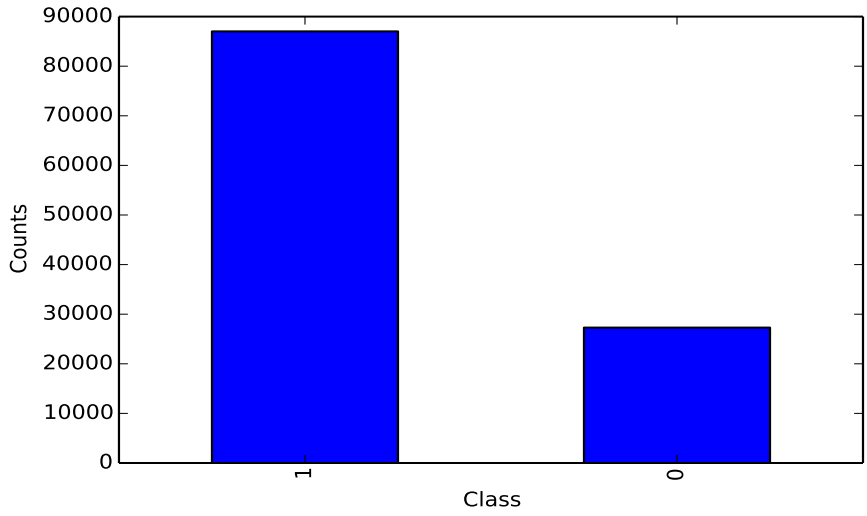

Fig. 1. Class counts in training dataset.

Because of the anonymization we do not know information about the nature and meaning of the features. Hence we are led to treat categorical features as nominal features, which creates challenges in the section of preprocessing as they are described later. The number of distinct values in categorical features in the training set is shown in the table below.

Features	v3	v22	v24	v30	v31	v47	v52	v56	v66	v71
Training Set	4	18211	5	8	4	10	13	123	3	9
Features	v74	v75	v79	v91	v107	v110	v112	v113	v125	
Training Set	3	4	18	8	8	3	23	37	91	

The same is shown for the test set. We observed that we have more distinct values on feature v22 on the test set rather than on the training set. Consequently, we had to take that into consideration for feature construction during preprocessing.

Features	v3	v22	v24	v30	v31	v47	v52	v56	v66	v71
Test Set	4	18253	5	8	4	9	13	117	3	9
Features	v74	v75	v79	v91	v107	v110	v112	v113	v125	
Test Set	3	4	17	8	8	3	23	37	91	

Moreover our dataset contains a lot of missing values. We are plotting the percentages of missing values per feature on the following figures 2 - 4. We see that there is a pattern among the percentages of missing values among the features. This is likely an artefact of the way the data were collected from BNP Paribas but because of the anonymization we cannot infer more from this.

V. Preprocessing

In the previous section were mentioned the types of variables which the given dataset consists of. Numerical values are in appropriate form to be processed by machine learning algorithms as most of the times they represent ordinal measures. On the other hand, categorical values need to be preprocessed and transformed into numerical so that the algorithms can be applied on them as well. In addition, the project team had to overcome the obstacle of missing values in the dataset as they constituted a considerable

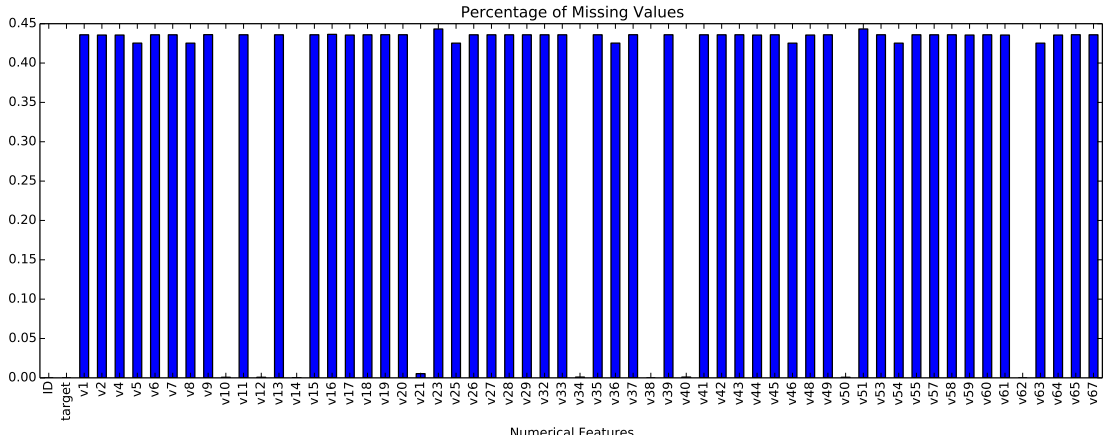

Fig. 2. Numerical Missing Values Percentages on the training set.

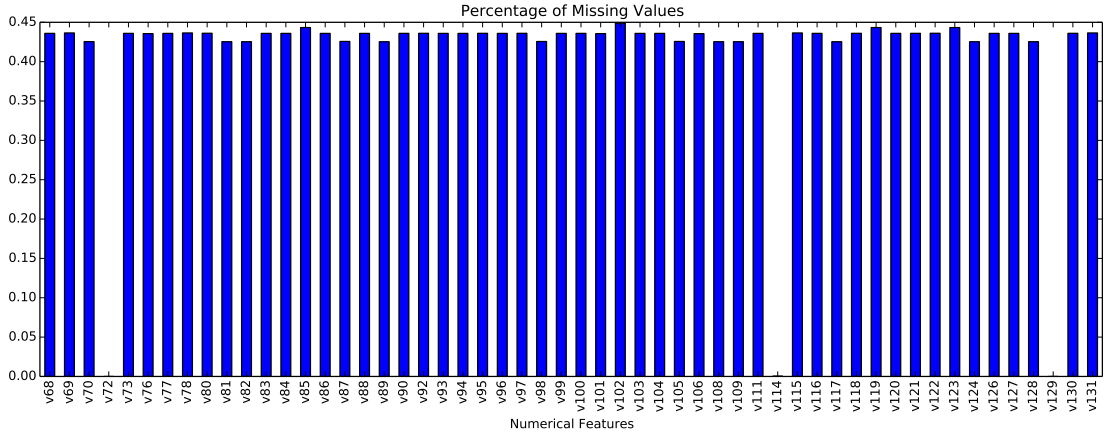

Fig. 3. Numerical Missing Values Percentages on the training set.

percentile of the available data. Lastly, the standardization of all variable values was decided as the scale of values was different for every variable.

A. Categorical features conversion

The approach followed by the team for the preprocessing of features with categorical values is a process that can greatly influence the final outcome of the analysis. The most common and successful approach in this case is an implemented method called One-Hot-Encoder, which expands the dataset by as many new features as the possible values of every categorical feature, having zeros in new features if an observation does not have a categorical value or one in the new feature that corresponds to the value of this observation. This transformation creates a very sparse matrix.

Unfortunately, in our case this method could not applied because one of the features with categorical variables ('V22') had 18,000 distinct values. The application of OneHotEncoder for such an approach would lead to a matrix with several thousands of features, and the final matrix would be extremely sparse. Additionally, the main problem would be that matrices with so large dimensions in features, as well as in observations need computers with much higher processing power than the conventional computers available to the members of the team, as the computational complexity increases exponentially.

As a result of the above observation, the team decided to deploy an alternative approach which was

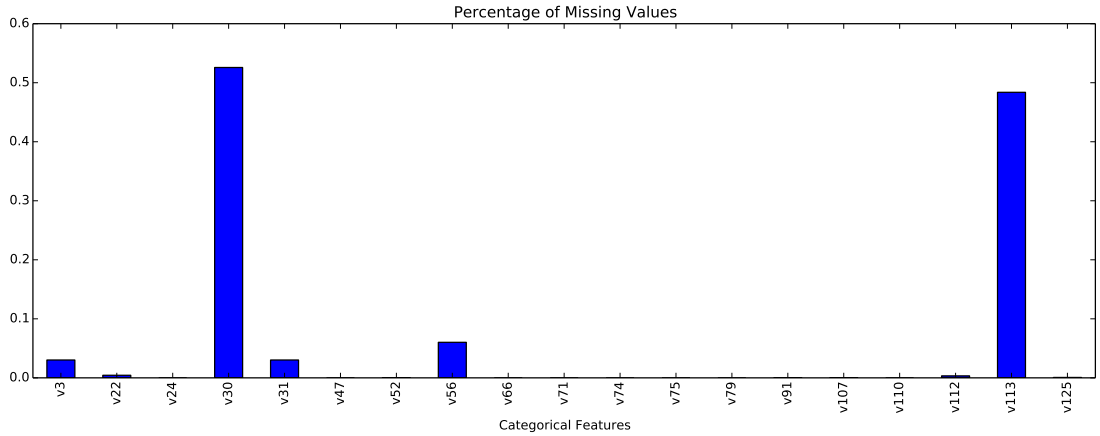

Fig. 4. Categorical Missing Values Percentages on the training set.

based on the idea of replacing each categorical value a probability value. For every categorical distinct value of each variable we calculated the conditional probability of getting a response ('target') value of one (1) when this particular categorical value appears. We then created a transformation matrix that mapped - per feature - all the categorical values into their respective probabilities. Then, we replaced the categorical values of both the training and the test sets with using the same mapping transformation matrix, in order to get consistent results.

For the values of the categorical features of the test set that were not existing on the training set, we decided to give them the value NaN (Not-A-Number) and handled them on the next step.

B. Filling missing values

As mentioned before, a considerable part of the dataset matrix is filled with missing values, a fact which incommodes the analysis process and the application of algorithms. The team needed to find a way to avoid using the missing values in calculations.

The first attempted method was Non-negative Matrix Factorization, aiming to represent the data as the dot product of two matrices of lower dimensions, which would encode the important information, ignoring the missing values. The method performed well, with low approximation error in low dimensional matrices, but when we attempted to apply the same method the actual training dataset, it failed to converge in an acceptable time period. For that reason, this method was dismissed, although it could work properly in smaller matrices, as it did in a part of the training set consisting of one thousand observations.

The chosen approach to fill the missing values was to use the median of each feature where a NaN value occurred. Other approaches suggested the use of the mean value of each column, but the median was preferred as the median is not affected by outliers of the feature values, so the model will be trained without being affected by long tails in the distribution of some variables.

C. Standardization

Machine Learning estimators are inefficient in producing accurate models when there is difference in the magnitude of values between the features of the dataset, like in our case. Therefore, the transformation of the values in similar scales was decided. We had to choose between normalization and standardization of the data.

Normalization rescales the values to a range of [0, 1]. This can be useful in cases where the data are of similar positive scale, but normalizing the data results in not representing the outliers. On the other hand, standardization transforms the distribution to a new one with zero (0) mean and standard deviation

of value one (1). As it addresses the above issue more effectively, its application is usually preferred over normalization in cases of data transformation.

VI. Supervised Learning Algorithms

We will now talk about the algorithms we used to create our models from the data we have. Prior to handling the problem there is no correct approach on addressing the problem. This is why we tried several methods.

A. Logistic Regression

We started with logistic regression as our first algorithm. We did that because we wanted an elementary benchmark estimator to compare it with our next results. Logistic regression is a well behaved linear classifier and, naturally, works better if the classes are linearly separable in the feature space. Logistics regression does not usually overfit but this can be further enhanced by using a regularizer. We used a standard option of an L2 regularizer. The results we obtained are shown in the table below:

Error on Training Set	0.41070
Error on Validation Set	0.40328
Error on Test Set	0.56916

By comparing the errors between the training and the validation set we saw that our results hint on good generalising ability. But when we submitted our results at kaggle we got significantly worse evaluation score. This told us that our classifier had poor generalisation ability. The reason for this conflict on generalisation performance comes from the way we created our validation set. It was randomly sampled, with 20% size from the training set we were provided from kaggle. Hence it behaved similar to the training set we used to train our classifier. This resulted in our model having good generalisation ability on the validation set but not on the test set because it did not resemble so closely the behaviour of the training set.

B. Support Vector Machines

Moving on we tried support vector machines. They are able to classify problems where the decision boundary is more complex than linear. They are reliant on proper feature space construction and there are many available methods, through different kernels, to use. We got our best results by using a radial basis function kernel. We present those results in the following table:

Error on Training Set	0.55027
Error on Validation Set	0.54168
Error on Test Set	0.55214

As we see our support vector machine model was far more consistent than the logistic regression model. The errors on all three sets are very similar. Actually our support vector machine model was the most consistent in displaying similar accuracy on all three sets.

C. Decision Trees

Decision trees is a suitable method to face classification and prediction problems. The advantages of decision trees are that they perform automatically selection of the most important features. In addition, decision trees do not suppose linear relationship between input values and the response value.It can also be mentioned that Decision Trees do not need much preprocessing effort by the researcher, as they can handle missing values and take the same decisions either on normalized data or not.

On the other hand, if the trees are not properly pruned and the tree growth limited, they tend to overfit the training data and as a result their prediction ability becomes poor.

On our application we chose a forest with 20 trees, max depth of a tree equal to 5 and we tried to reduce the features used for best split to 60. The criterion which provided the best results was Gini as it minimizes the misclassification in contrast to Entropy which is destined for use in exploratory analysis.

Error on Training Set	0.44590
Error on Validation Set	0.43875
Error on Test Set	0.51726

D. Random Forest

Last but not least we used random forest. It is an ensemble method that is very efficient in it's accuracy and easy in implementation because it is quite tolerant on how the feature space is constructed. The results we got with random forest classification are:

Error on Training Set	0.43844
Error on Validation Set	0.45468
Error on Test Set	0.49881

By comparing the errors between the training and the validation set we see that we have a small occurence of overfitting. This explains why we get slightly worse results on the test set where overfitting affects our accuracy even more. One advantage of random forest is that it can tell us how important our features are in classifying a dataset. We plot the most important features in figure 5

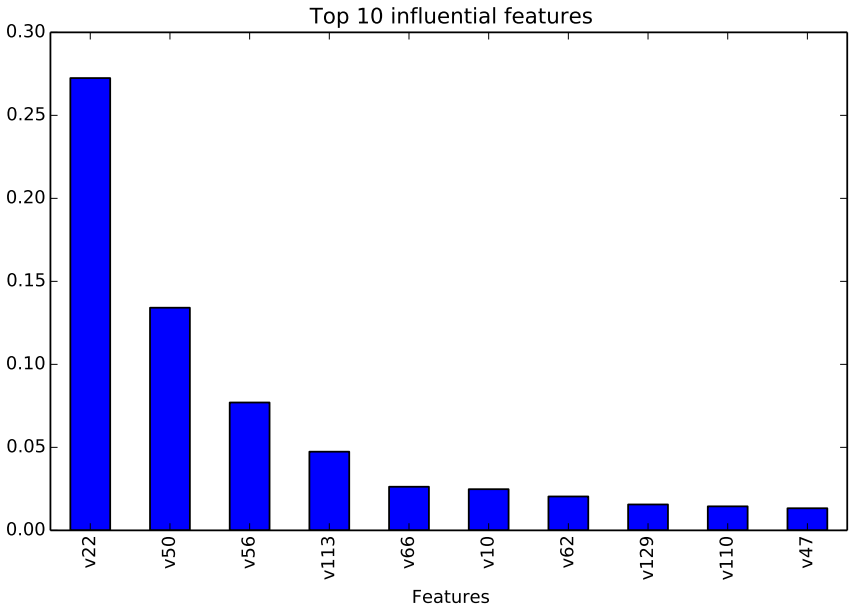

Fig. 5. Feature Importance from Random Forest Classifier.

VII. The Kaggle's winning solution

The winners of the competition made a forum post explaining their The winning team's approach[5]. They stated the most important thing for the competition was feature construction. Their breakthrough

came when they figured out how the dataset was anonymized and used a minor deanonymization process to create more appropriate feature space. They identified feature v22 as the customer and features v56 and v113 as the product types. Even though they got their best result with extreme gradient boosting they also tried other methods such as neural networks and support vector machines.

VIII. Conclusion

For this competition we tried various approaches. Our analysis began with logistic regression. It was a model relatively easy to implement but it had poor generalisation ability. This suggests that the classes are not linearly separated. Support vector machines taught us not to underestimate a model with bad prediction accuracy. If implemented properly it can be consistent enough that we can rely with more certainty on it's predictions in new data. Moving forward to ensemble methods we saw that they can be very accurate. Because of the way their algorithms work though they have overfitting problems and one should be very cautious of their generalization ability. In our case, where we picked our best results for each method, this problem was not prevalent enough to prevent them from outperforming the other methods. This however may not always be the case.

Acknowledgment

The authors would like to thank Dr. Adam Prugel-Bennett and Dr. Srinandan Dasmahapatra about the useful conversations held weekly. Their suggested approaches and insightful suggestions contributed in understanding the problem and exploring the power of appropriate machine learning techniques which effectively tackled the challenges which arose.

References

[1] Phil Simon (March 18, 2013). *Too Big to Ignore: The Business Case for Big Data.* Wiley. p. 89. ISBN 978-1-118-63817-0.

[2] Kate Palmer (July 13, 2015) *Insurance cheats cost households 90 a year as bogus claims reach record high* http://www.telegraph.co.uk/journalists/kate-palmer/11736121/Insurance-cheats-cost-households-90-a-year-as-bogus-claims-reach-record-high.html

[3] Pedregosa et al., *Scikit-learn: Machine Learning in Python*, JMLR 12, pp. 2825-2830, 2011.

[4] Team GitHub repository: https://github.com/Iolaum/Gauss

[5] Kaggle Winning team forum post, taken at 3 May 2016. https://www.kaggle.com/c/bnp-paribas-cardif-claims-management/forums/t/20247/1-dexter-s-lab-winning-solution