Binary Classifier for Direct Marketing Client Behavioural Predicition

Johannes Bogensperger, Miguel Alvarez Bordils

June 2019

Abstract

This project aims to apply the theoretical knowledge of the UPC Master in Research and Innovation Informatics course Machine Learning on a real problem. We strive to gain experience solving the problems caused by a real world project. We chose to use the bank marketing dataset. The goal of the initial project was to predict whether a customer is likely to purchase a deposit, once targeted by direct marketing techniques. Prediction outcome is based on personal and marketing features. We will compare the results achieved on previous work on this problem and evaluate applicable machine learning techniques on the problem. Finally we will build our own prediction model.

Contents

---

1 Introduction

This project will use the Direct Banking dataset Repository [2012] to find a model which enables us to predict customer behaviour. Based on the features of an individual, we want to find out if this person is more likely to buy a bank deposit upon recieving a phonecall from a marketing call center. This dataset is based on a real phone direct marketing campaign from a Portuguese banking institution. In some cases, clients were adressed several times to try to sell a bank deposit. The output variable for the dataset is a "yes" or "no" for this particular product.

Our goal is to test different machine learning techniques in order to evaluate if they are applicable for trying to predict the clients decision based on our previous knowledge of his features. For this, we will first make a brief summary of previous related studies. Next we will process and clean our data to be able to train the different models upon it. The models used will be LDA, QDA, RDA, KNN, Naive Bayes classifier, different SVMs and ANNs. After that, we will choose a final model based on Cross-Validation and discuss the obtained results.

The set includes 17 variables which are mixed both numerical and categorical. The feature set includes variables regarding client personal information such as age, job or marital status and data related to the marketing characteristics such as number of contacts or length of the call. This data was acquired between May 2008 and November 2010. We chose this dataset because investigating on human behavior is interesting from both economical and sociological point of view. In case we observe a significant result, besides the success in our machine learning algorithm we would achieve some interesting results that would provide insights on the problem domain and other fields of studies such as Marketing or Psychology.

---

2 Previous Work

Regarding on previous research performed on this data set, there are a few relevant papers that should be highlighted before getting into a deeper level of detail. We want to remark that the results on all papers reflect the high level of complexity of the data.

The first machine learning paper that used this Direct Marketing dataset [Sergio Moro a, 2013] did a first thorough exploration of the data. Originally, the data consisted of 59 attributes. After three iterations and problem redefinition, the dataset was reduced to the 17 attributes we have now. The metrics used to measure performance are AUC, ALIFT and true positive rate. The best result was achieved by a SVM. Personally we think they achieved wrong conclussions as the paper identifies length of the call as a relevant factor on achieving a positive outcome, which is something that cannot be chosen prior to the call and thus not a viable predictive variable.

Following Moro, Hany [Hany. A. Elsalamony, 2013] used the refined dataset to test an MLPNN and a Ross-Quinlan decision tree model (C5.0). To measure their performance they use classification accuracy (TP + TN / N), sensitivity (TP / (TP + FN)), and specificity (TN / (TN + FP)) achieving 90%, 60% and 94% respectively. Also again they use AUC in ROC space (The ROC curve is the sensitivity as a function of fall-out). Again this paper uses call duration as a prediction feature which in our opinion may lead to wrong results. Similarly, [Ta, 2014] achieved similar results training a MLPNN: accuracy of 88%, sensitivity of 40% and specificity of 95%. They also used call duration as a predictive feature.

The last publication we would like to remark is the cost-sensitive decision tree presented by Correa [Alejandro Correa Bahnsen, 2015]. The cost function used is based on estimation of return on investment for a bank on a one year deposit and the cost of a call center. They do not state wheter or not they use duration of the call. This time, to measure performance they use F1 score and obtain between 0.3 and 0.36 for cost-sensitive decision trees or decision trees on their own. I think it is important to remark that we cannot really benchmark with the first papers as their feature selection is flawed due to last call duration.

---

3 Data Exploration

For this project, we are using a dataset called "Bank Marketing Dataset", publicly available in the Machine Learning Repository Repository [2012]. The dataset consists on fourty five thousand rows with the features below stated.

This dataset is quite unique as there are not many available big datasets on bank clients information. As before mentioned, this data was acquired by a Portuguese Banking Institution upon adressing their clients via telemarketing agency. Some of the clients were adressed several times. The features of the table are the following:

Table 1: Variable description

Items	Features
age	age of the client (numerical).
job	Type of job of the client (categorical).
marital_status	Marital status of the client (categorical).
education	Level of education reached by the client (categorical).
credit_default	Wheter the client has credit default (categorical)
avg_balance	average monthly balance of the client (numerical)
housing_loan	Wheter the client has a loan for a house (categorical).
personal_loan	wheter the client has a personal loan (categorical).
contact_type	How was lastly adressed the client by the telemarketing agency (categorical).
month_lc	Month in which last contact was made (categorical)
day_lc	Day of the month in which last contact was made(categorical)
duration_lc	Duration in seconds of the last call (numerical)
amt_contact_campaign	Number of times the client was lastly adressed in the last campaign (numerical)
days_since_lc	Number of days since the last call was performed to the client (numerical)
amt_preCampaign	Number of times the client has been adressed in the current campaign (numerical)
outcome_lastCampaign	Outcome of the previous direct marketing campaign (categorical)
subscription_target	Output variable: wheter the client has bought the product or not (binary)

3.1 Missing, anomalous, incoherent and incorrect values.

As this dataset has been previously refined by [Sergio Moro a, 2013]. This predefined refinement is the same used in all the referenced papers. We do not find any missing or anomalous values. In accordance with Professor Belanche, we don't exclude univariate Outliers, since we see them still as relevant information. In order to find incoherent or incorrect values we are going to look for multivariate outliers using the Mahalanobis distance. As it can be seen in our code, the number of outliers was too big, around 26% (classical + robust Mahalanobis outliers.) of our rows were identified as outliers. Thus, we tried to find the outliers amongst the outliers. Still, this amount was too large (6.7% of the dataset). We decided to keep all the outliers in our data.

A relevant note here is that variable "days_since_lc" which states the number of days since the last marketing call is "-1"" for those clients who hadn't been contacted before. This remark is only made to highlight this is not a wrong value. However we categorized this variable. A nominal representation of this variable is more suitable, since the numeric representation is flawed with this construct of -1.

3.2 elimination of irrelevant or redundant variables

We decided to discard the **duration of last call** (duration_lc) variable for our models, which states the duration of the last call performed to the customer. First of all, because this value cannot be known prior actually doing the call and this means we are using a result of the marketing campaign. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model. As before stated, most of the papers fail to exclude this feature, which in our opinion is affecting the predicition and thus we cannot benchmark against them.

3.3 coding of non-continuous or non-ordered variables (nominal or binary)

In the following step, defined our categorical features as factors. This applies almost half of our features so this will determine most of the techniques that we will use to make our models. In big numbers, we have 5 numerical variables: age, avg_balance, duration_lc (which we decided to discard), amt_current_campaign and amt_preCampaign. Doing some exploratory plots, (see fig. 1 below and

fig. 5 in appendix 1) we cannot infer any relation or any gaussian distribution amongst the numerical variables.

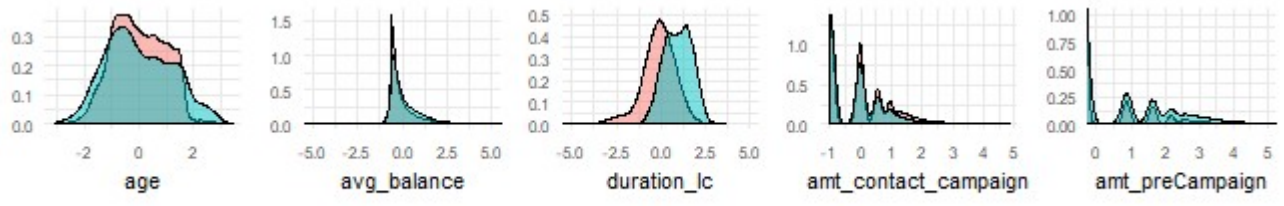

Figure 1: Density Distributions

Regarding the categorical values, we cannot find any clear tendency from the visual representation (see figures 5 and 6 in appendix 1). Amongst those that we can actually highlight information, it is not adding novelty to what intution may tell us once we know the dataset. Customers in a more concerning financial situation are less likely to buy a deposit (those having a personal loan or credit default). Clients who had bought deposits in previous campaigns are also likely to buy this type of products again.

3.4 Standarization and Logaritmic scaling of the numeric variables.

Most models work best with standardization (mean=0 and sd=1) and gaussian distributions. Our raw data is highly skewed and not normally distributed (see figures 2 and 3 below). In order to achieve better performance, we applied standarization and logaritmic scaling to achieve a closer distribution to that target.

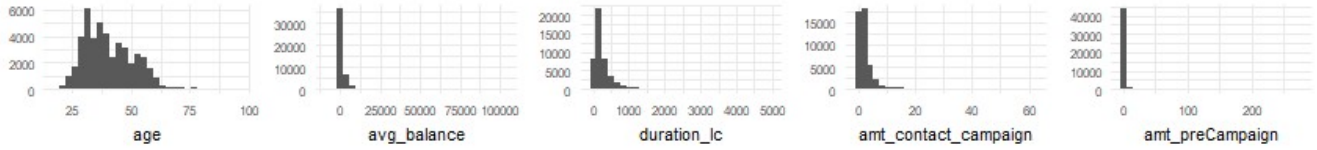

Figure 2: Histograms prior standarization and applying log

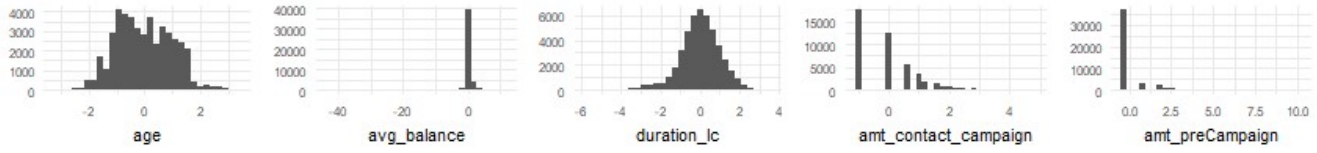

Figure 3: Histograms after standarization and applying log

It can be seen in **figure 3** that we didn't achieve the desired gaussian distribution. Especially due to univariate e.g. in the case of avg_balance. The method applied works in usual cases, as can be seen on the variable duraction_lc. Unfortunately this variable is discarded prior to the modelling process, as described in section 3.2.

3.5 Multiple Corresponce Analysis (MCA)

Since the majority of our variables is categorical we performed a MCA. Unfortunately the scattered and highly unstructured appearance of the factor map and variable map did not reveal useful exploitable insights into the data. A factor map of our MCA can be seen in Figure 7 in the appendix.

3.6 Data Clustering

Due to the mostly categorical data clustering our data requires extra steps. We chose to use the gowers distance to represent categorical and continuous data in the same distance matrix. Since the calculation of this gowers distances with the **daisy** function requires a lot of RAM we were only able to cluster a subset. Therefore we clustered a subset of 5000 observations. To be able to provide a visual interpretation of this data we apply multidimensional scaling upon the mentioned distance matrix to bring it into the twodimensional space. Since our data mostly comes from the categorical side, we prefer medoid clustering with the PAM method which is very similiar to k-means in terms of the process. We have to provide the amount of clusters and the algorithm derives the clusters with minimal distances to its members. Therefore we evaluate the solutions for two to ten clusters and evaluate the silhouette scores. The silhouette score of a measure is defined how similiar it is to all other cluster members. We found that **4 clusters** has the optimal silhouette. The optical representation in 2 dimensions is shown in Figure 4.

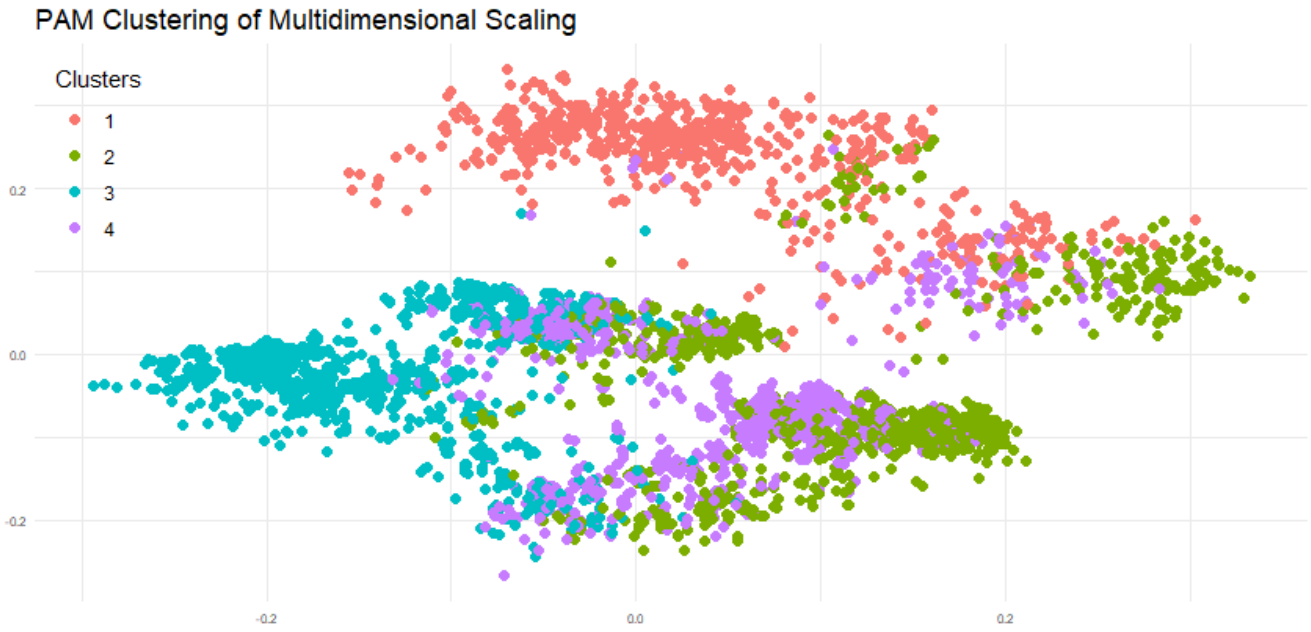

Figure 4: Clustering

The clusters summary shows us that none of the clusters have a particularly high “yes” percentage. The primary information inference is done with Cluster 2, since it represents Individuals which are very likely to not take credit. The yes:no ratio is about 1:20 in there, compared to 1:5/1:6 in the other clusters. Cluster 2 is mainly defined by blue-collar workers with a lower age, which are more likely to have housing loans. Pensionists, Managers and others professions are underrepresented. The variable for the outcome of the last campaign in 99,8% of the cases “unkown”. Which leads us to the conclusion that these Individuals were already not targeted by the bank beforehand, since they are the wrong target group.

4 Validation Protocol

In our attempt to predict customer behaviour, **14** methods were used. We first attempted using generalized linear models with logistic regression, discriminant analysis algorithms (linear, quadratic and regularized), K-nearest neighbour algorithm, Naive Bayesian classifier, artificial neural networks with multilayer perceptron and radial basis function kernels. Also Random forests and support vector machines with linear kernels, polynomical kernls and radial basis function kernels. Since we should benchmark these models to decide which one performs best, we need to define a validation protocol.

We split our data into a train and testset with a ratio of 2/3 and 1/3. The validation and model selection will be done using cross validation. The final test and generalization error will be determined with the test set.

Since our dataset is highly unbalanced and there are only about 11% of "yes" targets we needed to upsample our data, so our models will consider these observations equally relevant as the "no" observations. We upsampled our training Data with the **SMOTE** method to reach an equally balanced dataset.

The crossvalidation used in our project differs for the different models. The standard method used was 10 fold crossvalidation. Unfortunately due to resource restrictions, we applied 5 fold crossvalidation for RBF networks and the support vector machines. Furthermore we used for their training only a small datasample of 7000 observations (3500 per class).

5 Models used

As we saw in during the lecture, there is not one model, which outperforms all others. Therefore we implemented a variety of models seen in the lecture. In this sections we will briefly discuss the basics, advantages and disadvantages of the models used.

5.1 Generalized Linear Models (GLM)

We used a generalized linear model with logistic regression. This means that we train a linear model with a "logit" (sigmoidal) link function using the maximum likelihood optimization. This type of link function is used for binary target variables, as our subscription_target. We evaluate this model since in opposition to the discriminat models it does not make any assumptions on the distribution of the data. Since half of our numerical values are highly skewed this model could possibly provide better results than Discriminant Models. Furthermore it should be quite robust against outliers compared to discriminant models.

5.2 Discriminant Models

In general discriminant models like Naive Bayes, LDA, QDA, RDA are designed to classify categorical data. All of these models assume gaussianity on the data, which is in practice and as we can see in our histograms (see figures 2 and 3) is not the case for our numeric variables in the train sample.

5.2.1 Naive Bayes (NB)

The Naive Bayes Classifier would be the preferred option if we would know or would be able to precisely estimate the class distribution and prior/posterior probabilites. In this case, this classifier would have the lowest error rate of all possible methods. Therefore we will try to estimate them based on the given training set and evaluate its performance.

5.2.2 Linear Discriminant Analysis (LDA)

This type of model assumes equal covariance matrices between the different classes and will therefore most likely not provide superior accuracy (Since we assume this is unlikely). In case the covariance matrices are different it will suffer from high variance. We will train this model anyways for the sake of completeness and learning purposes. Furthermore we could theoretically use LDA as dimensionality reduction method, since we can work in a subspace with less dimensions and work with less parameters. The decision boundaries of this model are linear, as indicated by the name.

5.2.3 Quadratic Discriminant Analysis (QDA)

This version of the discriminant models works with quadratic decision boundaries and has way more parameters to train (due to different covariance matrices for each class), which makes them more flexible in terms of class separation. The downside is that they cannot provide dimensionality reduction and the amount of parameters rise quadratically to the amount of variables.

5.2.4 Regularized Discriminant Analysis (RDA)

The RDA technique is kind of a mixture between the LDA and QDA models. Depending on the parametrization of its **alpha** (regularization parameter) it can be reduced to each of the two previous techniques or perform a mixture of them. The regularization parameter determines to which extent the Covariance matrices are mixed. This model could therefore outperform the pure forms LDA and QDA. While doing CV the optimal hyperparameter was found to be 0, therefore basically QDA was applied (which can be also seen in the equal CV error which are equal, except for tiny differences which occur due to the random sample selection at CV)

5.3 K-Nearest Neighbours (KNN)

The KNN classiier with its theoretical infinite VC-Dimension should be tried as well. It usually assigns the class of the majority of its closest k neighbours. Ties are broken at random. Our final model for KNN chose k=7, so there won't be ties actually.

5.4 Artificial Neural Networks

We are going to use two types of neural networks from the lectures. In general Articificial Neural Networks are able to handle non-linear relationships and don't assume gaussianity of our input data. Furthermore their retraining can be done efficiently once we received more data, so we don't have to retrain the whole model. We will consider Multilayer Perceptrons and Radial Basis Function Networks both from the RPackage "RSNNS". In general these methods are able to handle the training of big amounts of data faster than SVMs (we trained our MLP on the whole train dataset of 56k observations) and once trained the predictions are quite fast (e.g. compared to big Random forests).

5.4.1 Multilayer Perceptron (MLP)

The training algorithm of this model is the backpropagation algorithm showed in the lecture. However a big open challenge of the MLP is to distinguish the amount of hidden layers and nodes. This estimation is still subject to research, therefore we used a simple CV approach. Finally we used an implementation with a single layer and the optimal amount of nodes was found to be 5.

5.4.2 Radial Basis Function Network (RBF Network)

The RBF network uses, as the name indicates, radial basis functions as activation function and always only use a single hidden layer. They are not trained by the backpropagation algorithm, instead they are trained by a two step algorithm. In this iterative process the centers are derived (initially e.g. by k-means) and in the next step the weights are calculated with respect to some objective function. This model estimates how close a given input is to each neuron in terms of euclidean distance. Therefore numeric input is needed. Since the conversion from categorical to numeric dummy variables was not done by the implementation we had to do that manually before it. RBFs seem to have a troublesome implementations in R package "RSNNS", furthermore they didn't perform very well, so they are definitely not a good candidate for our final model.

5.5 Support Vector Machines (SVM)

Support Vector Machines are a model which excels at a fast prediction once computed and handling non-linear relations due to its kernels. A SVM will always try to maximize the margin between data and the decisionboundary. Furthmore these decision boundaries (hyperplanes) are handled in a Hillbert space which can be of higher dimension than the input space. The kernel trick lets us exploit this in a computational efficient manner, where we only have to calculate the scalar product of two observations. The downside is that training the model is still very expensive. Therefore we were not able to train it on the full dataset. We decide to downsample the training dataset and use an balanced dataset with 7000 observations.

Using the correct kernel function is therefore essential for the performance of the model. All our SVM models are chosen from the R Package kernlab. All of these models must also be regularized to not overfit the data, therefore we need to estimate the Cost Parameter **C** which defines how much the decision boundary is allowed to be violated.

5.6 SVM linear

This type of model doesn't exploit the kernel trick to separate the data in the higher dimensional space, but it can benefit from the objective to separate the data with maximal margin and that only a subset of the observations is used to derive the decision boundary. The optimal Cost parameter according to CV is 0,25.

5.7 SVM poly

The SVM's with a polynomial kernel exploit the kernel trick and separate the data in a higher dimensional space. So they are more flexible than the linear SVMs. But the degree of the polynomial needs to be evaluated as well as the regularization parameter. We found that the best degree is 3 and the optimal Cost parameter is 0,25.

5.8 SVM RBF

The RBF SVMs exploit the kernel as well and map the observations to a dimensional space which has much more dimensions than the orignial one. Furthermore since its based on the radial basis function it is even more flexible than the polynomial kernelized SVMs usually. With Crossvalidation we found a small sigma of 0,00812 to be optimal and a C of 1.

5.9 Random Forests (RF)

Random Forests are an old but well performing model. It consists out of bagged Decision trees and uses the idea that each model adds a bit of information to the table. For this fact they can perform quite well on various range of problems. The final prediction is made according to a majority vote of all decision trees. These trees vary in the variables they use for the nodes as well as in the values chosen as decision boundary. Random Forests are known for their good generalization properties, which are less vulnerable to overfitting compared to other models.

Random Forests have expensive prediction costs since they always need to check all trees (which can be a lot) for making a prediction. Their training is not especially expensive, but as most methods they need to be retrained completely once new data shall be included.

Random Forests can furthermore give information about if a variable is important to the model or not. This is based on the amount of occurances of the variable in the forest.

---

6 Results

In this section we will present the results of the executed crossvalidation of the evaluated models. Furthermore the relvant hyperparameters can be found in the second column.

6.1 Evaluation Metric

Since we have a highly unbalanced datastructure, the accuracy is not able to represent the quality of the trained models. We therefore evaluate the F1-Score. The F1-Score is calculated by the harmonic average of precision and recall. This score will not present a good value when only the “no” targets, which are the majority of the data, are classified well and the small percentage of “yes” is not predicted well.

Table 2: Results with parameters

Models	F1_Score	parameters
GLM	0.7703498	
LDA	0.7639154	
QDA	0.8278807	
RDA	0.8283334	
KNN	0.8392244	K=7
NB	0.7030176	
ANN.MLP	0.9203540	size=5
ANN.RBF	0.6711823	size=30
SVM.LIN	0.6686989	cost=0.25
SVM.POLY	0.6865434	cost=0.25;scale=0.01;degree=3
SVM.RBF	0.6931638	sigma=0.00828;cost=1
RF	0.9255557	ntree=631;mtry=3;sampleSize=1000/1000

Our validation is completely based on CV and since we didn't hold out a separate validation sample. We can see that the Random Forest outperforms the other Models in terms of CV F1-Score. Thus we choose the Random Forest with this parametrization as our final model.

We are not aware of any performance restrictions which would exclude any kind of model. Since a random forest cannot be trained on the fly, this might become relevant if the solution should be used with very big amounts of data, with changing patterns. Therefore a continuous retraining would be necessary and we would be tempted to use an MLP. Since the MLP performs nearly equally well, but with more efficient retraining possibilites. On the other hand Random Forests should be quite resistent against overfitting which made us finally choose the Random Forest.

---

7 Final Model

Due to the achieved results, Random Forests was found to be the most applicable model to our problem. We will now retrain the RF with the train dataset and the chosen hyperparameter.

Once we retrained the Forest, we use it to predict the untouched test sample to evaluate its performance and the generalization error.

Table 3: Final Confusion Matrix

Actual Value	Pred No	Pred Yes
No	12388	908
yes	1090	685

Table 4: General performance indicators

Accuracy	Kappa	AccuracyLower	AccuracyUpper	AccuracyNull	AccuracyPValue	McnemarPValue
0.8674275	0.3323902	0.8619093	0.8728034	0.8822241	1	5.14e-05

Table 5: By Class performance indicators

	x
Sensitivity	0.3859155
Specificity	0.9317088
Pos Pred Value	0.4300063
Neg Pred Value	0.9191275
Precision	0.4300063
Recall	0.3859155
F1	0.4067696

The F1-Score of the prediction of the test dataset can be seen above. Compared to the only paper which provides information about the F1-Score (Correa [Alejandro Correa Bahnsen, 2015]), we achive even better results. The paper from Correa states a F1-Score of 0.3 to 0.36 where we achieve 0.407 in our final prediction. These results makes us believe that our model can be seen as quite good.

The only other paper which allows us to calculate the F1 score is from Hany [Hany. A. Elsalamony, 2013]. The calculated F1-Score of 0.7325 is way higher, but it should be noted that this was likely to occur. They are using the duration of the call as feature in their model. Of course the likelihood of making an deposit is highly correlated with the length of the last promotion call. But, this variable is not known

before a new marketing campaign starts. Therefore a benchmark against projects which use the last call duration seems not applicable.

However the final F1-Score of the testset is about 0.5187 (generalization Error) smaller than the training F1-Score. This is highly disappointing and leads us to the conclusion that the high generalization error was caused by an overfitted model. Since the Bias in training was quite low and the final prediction is quite poor. Therefore we assume high variance due to overfitting is the cause.

The overall accuracy seems to be OK with 0.8674, but a sensitivity of 0.3859 is quite poor.

---

8 Scientific and personal conclusions

The F1-Score we used to benchmark our solutions is more suitable than the accuracy of the model. Using Accuracy as benchmark for the quality of a model facing imbalanced data is not a good idea, as can be seen in the results section. We achieve sufficient accuracy but the sensitivity is not really relevant in unbalanced problem.

It seems to us that we overfit the data. Therefore we conclude that some of the models with a lower F1-Score might have actually been the better choice, since the might have higher bias but less Variance. We will explore this option in the next section about future work.

The depending on the cost of a call and the earned money per sold credit, this model might add value to the domain. The Sensitivity of our model is quite low and we will miss out half of the possible sold deposits. On the other hand we saved the vast majority of calls since only a small percentage of the calls would actually be executed.

In terms of the important variables of the final model, we can say that the individual properties of the individuals were not the most important ones. The properties of the interaction e.g. date of the last contact were far more important than the martial status or if the person defaulted on a credit before. Especially the small importance of the credit default can be seen as interesting, since this indicates usually a complicated relationship to banks or about the persons financial situation. Therefore we assume that other personal indicators might be a better fit for predicting those sales and are the true factors which cause the high amount of variance we cannot explain with our model. So the collected data is most probably not capturing the ongoing process of deciding if a deposit is likely to be sold very well. Especially since most we didn't find papers with a good Sensitivity or F1-Score which don't use the duration of the last call.

The clustering provided a brief insight of which group is not likely to be in the target focus. Since they all had been not contacted previously, we assume that there were already other models in place. These models caused that basically none of the individuals in cluster two had been contacted before.

---

9 Future Work and Limitations

Our final model has a high variance and generalization-error. Therfore we conclude we overfitted our model. This might be caused by the oversampling of 700 percent of the yes class. We took a sneak peak on how our model would behave with different sampling techniques. More due to curiousity, since this

approach is not scientifically sound. With downsampling the data to a 1:1 dataset or only oversampling the "yes" class to 350 percent and discarding half of the no-class. In both cases the training F1-Score was lower and in the second case the Multilayer perceptron outperformed the Random Forest. Together wit a higher F1-Score on the test set, this leads to the conclusion, that using a extra validation datasample would be a good idea to prevent this. We could have tried multiple versions on how to under/oversample our data and benchmark it against the validation sample to prevent or reduce the high gap between training and test F1-Score.

In terms of the general approach we would see the urgent need to define a customized loss function in the future. Based on approximations of the domain e.g. average deposit amount, interest rates etc. we could define a model which analyzes the real benefit / loss of the prediction outcome. This would add a lot of value to our model since the cost of a not-sold loan will probably be way higher than of a single wasted advertising call.

Furthermore in reality we would usually have more datasources / variables to include in our model. With initiatives as know-your-customer (KYC) and comprehensive Customer Relationship Management systems (CRM) in every bank, it is not a very likely scenario to execute predictions on such a little dataset (especially in terms of amount of variables).
