Machine Learning

Coursework 3

Santander Value Prediction:

Predict the value of transactions for potential customers

Simone Zanetti

MSc Data Science

Introduction

The following report deals with the attempt to summarise the fundamental passages of the competition proposed By Santander Group on kaggle.com, which I performed in the occasion of the Coursework III of Machine Learning and Statistical Data Mining of the MSc Data Science at Goldsmiths, University of London.

The Competition

The competition proposed by Santander Group relies on the necessity to improve their customer service system. In fact, as stated in the description of the competition, research of Epsilon has observed how 80% of costumers result more likely to do business with a company if it provides personalised services. In many cases, in order to provide them, it is important to be able to anticipate the need of the customer, in order to be ready to interact in a personalised way. In this sense, the possibility of Santander to identify the value of a transaction before this concretely verifies represents a useful added value to move into the above-mentioned direction. In this context, the task regarded the identification of the value of transactions for each potential customer.

The Task

From a technical point of view, the task was represented by the need of performing a Supervised Learning application. As we are dealing with a (continuous) numerical output, this consisted in a Regression problem in which a set of variables/inputs would allow me to obtain/predict the value of the transaction for each client.

Premise

Despite the section in which summarising the results, both in terms of performance and in terms of the way the task has been tackled, usually fits better at the end of the report, I think it is important to premise a few points, before moving into the core of the report. While the data and the challenge have been absolutely an exciting and interesting journey to undertake, I have to admit I did not expect such a particular, and in some sense, extreme situation. This condition has lead to the consequence of spending a huge amount of time in the studying and observation of the data (I should say, contemplation of them). Lots of time has been exhausted trying to figure out a way to tackle the task, and another big part of that has been spent in the pre-processing phase, where the failures have been more than the successes. However, I also think this situation has taught me a lot and has made me more aware of the necessity of being very focused when analysing a dataset. Moreover, it made me realise one time more how true are those who state that 80% of the life of a Data Scientist is spent in the process of Cleaning and processing of Data. The successive sections will allow the reader to have a clearer understanding of how extreme these data are, but so far it is important to bear in mind that a bigger focus has been spent on the cleaning phase, rather than the attempt of applying several and different models.

The Dataset

The first step of my task has regarded the need of understanding the data at my disposal. In this context, I have been surprised by a series of details:

1. With 4459 observation and 4993 variables, the training dataset had more columns than rows. (The first time for me to experience that). While the meaning of the variables was still unknown, the rows seemed to represent an observation per client. In fact, the variable ID was constituted by a unique value per each row.*

2. The number of zeroes in the dataset was negatively surprising to me. (96.9 % of the observations in the dataset were 0s)

3. Every variables seemed to be numeric, with the exception of one (variable ID*), which was removed since it added no meaning to the model.

4. Not very surprising, the test data did not own the label. This is quite normal for competitions, and lead to the necessity of split the training data (which was already quite small in terms of observations) between train and validation set, in order to assess the performance of the different models.

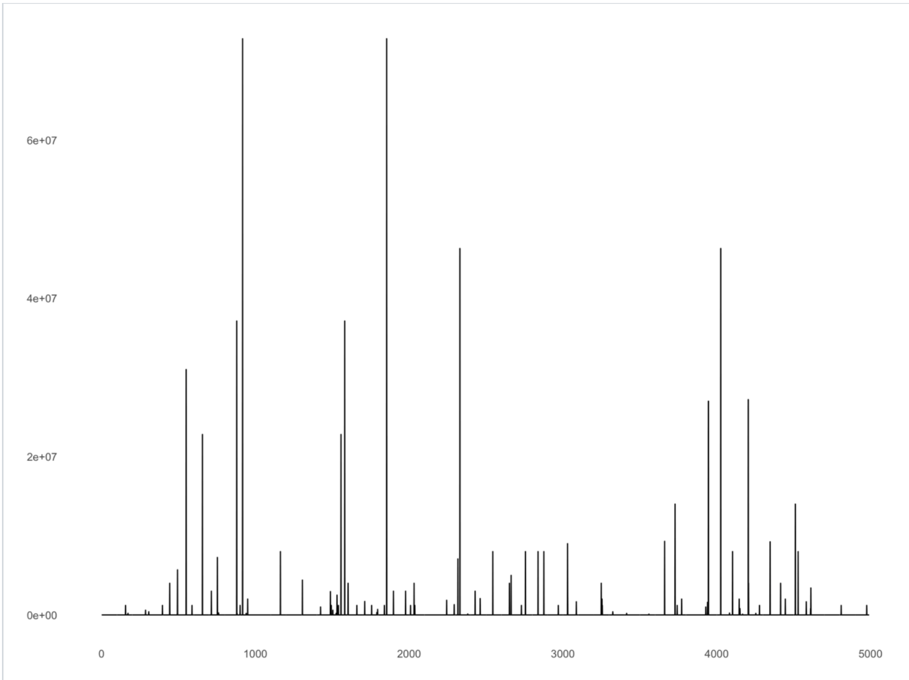
```
> str(train)
Classes 'tbl_df', 'tbl' and 'data.frame':	4459 obs. of  4993 variables:
 $ ID        : chr  "000d6aaf2" "000fbd867" "0027d6b71" "0028cbf45" ...
 $ target    : num  38000000 600000 10000000 2000000 14400000 2800000 164000 60
 $ 48df886f9: num  0 0 0 0 0 0 0 0 0 0 ...
 $ 0deb4b6a8: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 34b15f335: num  0 0 0 0 0 0 0 0 0 0 ...
 $ a8cb14b00: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 2f0771a37: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 30347e683: int  0 0 0 0 0 0 0 0 0 0 ...
 $ d08d1fbe3: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 6ee66e115: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 20aa07010: num  0 2200000 0 0 2000000 ...
 $ dc5a8f1d8: num  0 0 0 0 0 0 0 0 0 0 ...
 $ 11d86fa6a: num  0 0 0 0 0 8000 0 0 0 0 ...
 $ 77c9823f2: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 8d6c2a0b2: int  0 0 0 0 0 0 0 0 0 0 ...
 $ 4681de4fd: int  0 0 0 0 0 0 0 0 22000 0 ...
 $ adf119b9a: int  0 0 0 0 0 0 0 0 0 0 ...
```

```
> summary(train)[,3:13]
   48df886f9          0deb4b6a8          34b15f335          a8cb14b00           2f0771a37        
 Min.   :       0   Min.   :      0   Min.   :       0   Min.   :       0   Min.   :        0  
 1st Qu.:       0   1st Qu.:      0   1st Qu.:       0   1st Qu.:       0   1st Qu.:        0  
 Median :       0   Median :      0   Median :       0   Median :       0   Median :        0  
 Mean   :   14655   Mean   :   1391   Mean   :   26722   Mean   :    4530   Mean   :    26410  
 3rd Qu.:       0   3rd Qu.:      0   3rd Qu.:       0   3rd Qu.:       0   3rd Qu.:        0  
 Max.   :20000000   Max.   :4000000   Max.   :20000000   Max.   :14800000   Max.   :100000000  

   30347e683          d08d1fbe3           6ee66e115           20aa07010            dc5a8f1d8       
 Min.   :       0   Min.   :       0   Min.   :       0   Min.   :        0   Min.   :       0  
 1st Qu.:       0   1st Qu.:       0   1st Qu.:       0   1st Qu.:        0   1st Qu.:       0  
 Median :       0   Median :       0   Median :       0   Median :        0   Median :       0  
 Mean   :   30708   Mean   :   16865   Mean   :    4669   Mean   :  2569407   Mean   :  155216  
 3rd Qu.:       0   3rd Qu.:       0   3rd Qu.:       0   3rd Qu.:   600000   3rd Qu.:       0  
 Max.   :20708000   Max.   :40000000   Max.   :10400000   Max.   :319612000   Max.   :60000000  
```

Analysis of the Data

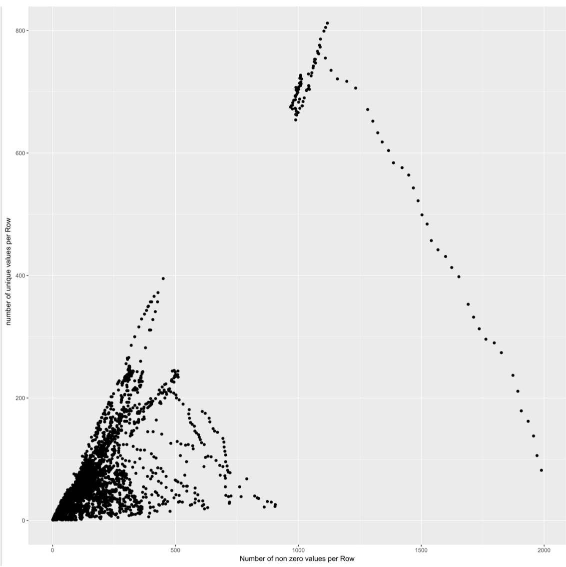
The first step has regarded the attempt of understanding a bit more the dataset, with a particular focus on understanding what the different variables could represent. I assumed that when you have so many zeroes it is important to understand the reason for that and verify if those zeroes could be significant values or a way in which missing values were filled. In this context, I analysed from a graphical point of view few rows/client in order to formulate a hypothesis of what the situation could mean, by plotting a histogram representing the values encountered in the row.

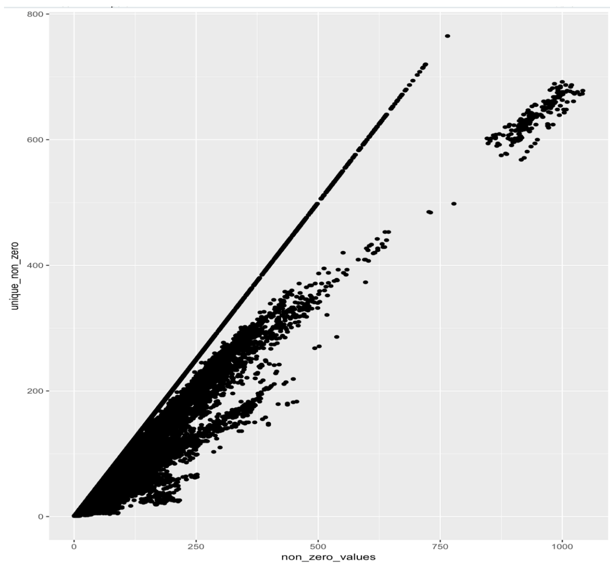


At this point, the hypothesis was that each row could represent some type of chronological order of transaction for a single client. In this sense, zeroes could represent a sort of missing values/no transaction. The second hypothesis has been that the presence of many zeroes could be the consequence of one-hot encoding transformation. However, this did not seem to make full sense since columns were not binary identified (values different from zero were not 1, but different values, various and far from 1). In this perspective, I defined a conservative strategy in which, despite the high amount of variables, I wanted to be very cautious in the phase of pre-processing in relationship with Dimensionality Reduction. This phase will be analysed in the next section.

Summary Statistics

Before that, I decided to deepen the analysis and understanding of the dataset by comparing the different index of summary statistics for each row, such as mean, standard deviation, minimum, maximum, median, skewness, and adding two variables that I considered highly important, such as the sum of the number of zeroes present in the row, as well as the number of unique values. In order to perform this kind of analysis, I transformed each 0 value in a Missing value. This allowed me to obtain values which could be independent by the high amount of zeroes, in the hypothesis that these zeroes were nothing but missing values representing the absence of a transaction.
As shown in the image below, nothing significative has been shown, with the exception of a very particular shape in the scatter plot comparing the number of unique values to the number of zeroes. In here it is immediate to notice a strange perfect linear correlation between them, which seems to be a bit unnatural. This situation gave me the idea of some sort of points artificially added in order to create some noise. Despite the satisfaction for having discovered this element, I did not see any possibility to exploit this information as an advantage. In fact, at first, I thought about excluding these rows from my analysis in situations of Feature selection or when defining my Principal Component Elements. However, all of the results I obtained, such as unsupervised pre-processing, transformation of data or PCA, were based only on the results provided by the Training set (specifically, the portion of the training set already split from the Validation set), with the attempt of avoiding to touch test data. So, I was not able to think about a context in which exclude these data from the test set (which, as previously expressed, does not have label provided) in order to gain an advantage.





Data Cleaning (Unsupervised Pre-Processing)

As previously mentioned, I considered important the necessity of applying a conservative strategy in the cleaning of data and feature reduction. The reason for this was represented by the fact that the data I was handling was surely unusual and tricky, and I did not consider wise to delete columns in this context, so I applied a very conservative strategy. Moreover, it was immediately clear the strategy to apply Principal Components Analysis to the dataset, as well as working with Lasso / Elastic Net feature selection, which would handle the reduction of the variable by themselves. In this context, training data and test data have been joined together in order to perform modification on the entire dataset (which was made possible by the fact that I removed the label from the training set and performed operations which did not require to look at the output). This aspect is very important and it can reduce the risk of creating biases over the task. The first step of the process of Unsupervised data pre-processing (i.d processing of data made without taking into consideration the output of the analysis) regarded the necessity of verifying the presence of Missing Values in the dataset. Luckily the Nan's were only 6 in all the dataset (train + test). In this context, it could have been possible to simply delete the rows, but this would have lead to a longer process, since I had previously split the variables to the output, and filtering the rows could have run the risk to lose the correspondence between the observations and their target. As a consequence, I decided to apply a simple imputation, by replacing the Nan's with the mean of that variable. The second step of the phase regarded the necessity of deleting all those columns with absolute zero variance on it (i.d only a constant value on it). These variables do not have any sense for the algorithm since they mean that for each observation the value (for that specific column) is exactly the same. This allowed me to remove roughly 350 variables. It is important to notice how, starting from this moment, I applied every analysis on a training set (a subset of the Train data) of 3,200 observation. The reason of this choice was due to the fact that with so many zeroes, I wanted to ensure that I would not input any variable with variance different from zero in the entire dataset, while equal to zero in the training set. Moreover, I had to take into consideration that part of the original Train data would have been used as a validation set, and I did not want to run the risk of creating biases on my model. To be fair, those biases would not have been an act of high negligence, cause I was working anyway on a dataset without its output. After this phase, I considered opportune to apply the NearZeroVar function in order to clean the data from extreme cases. In this context, I have been coherent with my goal of being very conservative. In this phase I have been helped by a series of histograms which allowed me to visualise the distribution of the two fundamental parameters which detect Near Zero Variance variables:

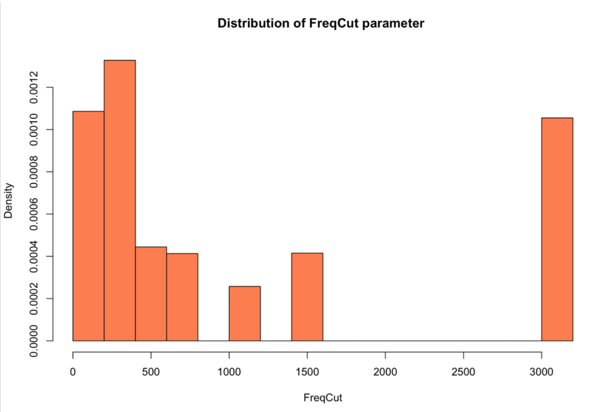
. **FreqCut**: the cutoff for the ratio of the most common value to the second most common value

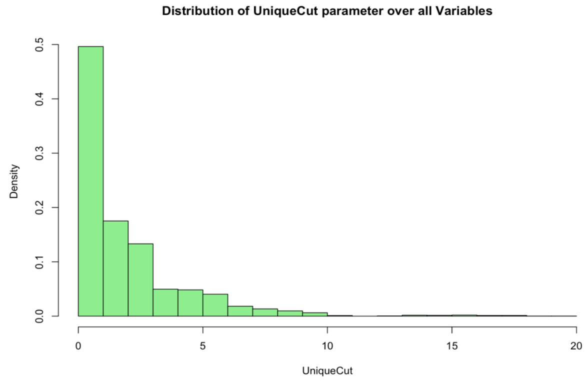
. **UniqueCut**: the cutoff for the percentage of distinct values out of the number of total samples.

Usually, it is possible to define a variable to have variance close to zero if the two following conditions are verified:

.The fraction of unique values over the sample is low (10%)
.The ratio of the frequency of the most prevalent to the second most prevalent is around 20 %.

However, The situation was so that critical that simply applying the function without handling the parameters (by simply applying the rule of thumb) would have deleted all the variables.





	freqRatio	percentUnique	zeroVar	nzv
48df886f9	792.2500	0.87500	FALSE	TRUE
0deb4b6a8	3196.0000	0.15625	FALSE	TRUE
34b15f335	1059.0000	0.68750	FALSE	TRUE
a8cb14b00	3198.0000	0.09375	FALSE	TRUE
2f0771a37	3196.0000	0.15625	FALSE	TRUE
30347e683	453.2857	0.56250	FALSE	TRUE

In this context, I decided to apply a *FreqCut* of 3000. and a *UniqueCut* of 0.01, removing 453 Variables. Moreover, I decided to verify which of the Variables had higher variance. The goal was the one to obtain a list of the 150 variables with higher variance, in order to make an attempt of building a predictive Model using only those columns, later on.

By applying a *FreqCut* of 110 and UniqueCut of 11, I had the possibility to store the list of the columns in a variable, named *highvar* and use it in later stages of my analysis.

Another operation I performed in the context of Unsupervised data Pre-processing regarded the attempt of verifying the variables with a higher correlation into the dataset. The reason for this operation regarded an attempt of avoiding phenomena of collinearity between variables. Regardless of the goal, I have been very conservative again, by only removing those variable with more than 70% of collinearity between each other. In this context, I could not use VIF's method since the number of predictors was superior to the number of observations. Moreover, it is important to the premise that this phase left me with many doubts about its applicability. In fact, from one side, I had such a high number of variables that I moved with the conviction that deleting the highly correlated one would not have created more downsides than the upsides. However, in a context with so many zero values, I also had the impression that the more information I could keep in terms of 'real' numbers (in the hypothesis zero represented simply the absence of transaction), the more this would have been useful for my predictive model. In the end, I decided to remove only those variables with the above-mentioned collinearity, which meant removing 310 variables out of 4400, in the optic that If time had allowed me, I would have tried to move back to this phase and avoid this passage. It is opportune to note that the choice of applying the Spearman correlation coefficient was due to the fact that my data were still highly unbalanced, so not normally distributed (I did not provide any data transformation at the time).

Data Transformation

The first step into the phase of data transformation regarded the necessity of fixing the issue deriving by the high skewness of data. With so many zeroes, and only positive numbers very sparsely distributed, the skewness tests applied to the dataset revealed high skewness for most of the variables.

$$skewness = \frac{\sum(x_i - \overline{x})^3}{(n-1)v^{3/2}} \qquad v = \frac{\sum(x_i - \overline{x})^2}{n-1}$$

```
> skew_tot_pre[1:10]
48df886f9 0deb4b6a8 34b15f335 2f0771a37 30347e683 d08d1fbe3
 51.83234  65.52869  59.99891 102.84484  68.04666 130.35468
20aa07010 dc5a8f1d8 11d86fa6a 8d6c2a0b2
 12.08162  68.01791  75.86533  72.95108
> skew_tot_post[1:10] # It worked
48df886f9 0deb4b6a8 34b15f335 2f0771a37 30347e683 d08d1fbe3
12.749061 16.309319  9.603566 16.964776 12.153252 15.577459
20aa07010 dc5a8f1d8 11d86fa6a 8d6c2a0b2
 1.877433 10.134260  9.896692 14.392824
```

In this context, I applied BoxCox transformation, taking advantage of the fact that I did not have any negative value. However, to make the circumstances feasible to this type of transformation, I decided to add 1 to every value in the dataset. In this way, I did not any 0 in the dataset (which would have imposed me to apply log transformation for them). It is important to bear in mind how adding a constant value to every observation in a column does not modify its distribution. The process worked, as shown in the figure above. However, the skewness was still present and quite high for some variables.
The second phase and the third one regarded the necessity of standardise the data, by obtaining a situation in which each variable had a mean of 0 or a value highly close to it, and a standard deviation of 1 or very close to it, and the attempt of fix the presence of severe outliers in the dataset. In this context, SpatialSign transformation has been applied, which allowed projecting the variables onto a unit sphere, reducing the impact of outliers in some algorithms which may suffer it.

Feature Selection

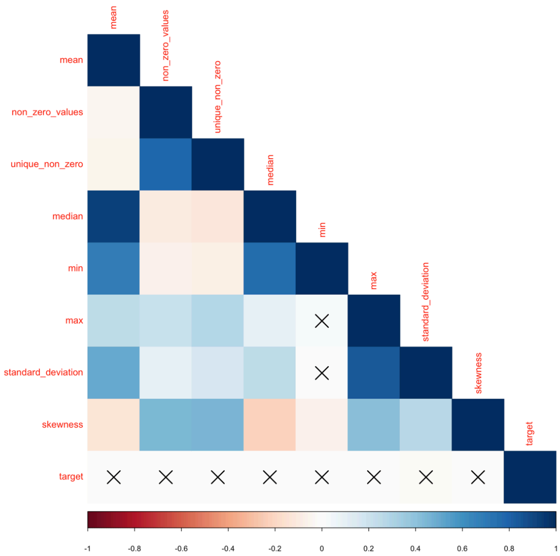
This phase will deal with the attempt of working with the selection of the right Feature to feed into the models. However, since the data was very extreme in terms of characteristics, this phase has represented the most challenging part, as well as the part in which a long series of attempt could have been tried. For this reason, I decided to directly apply a few models for every different strategy of Feature Selection I applied. This could make my task easier to manage, and I believe can be clearer for the reader to follow . Moreover, I regard that the different strategies applied can be considered moving in order of difficulty, from the simplest to the 'hardest'.

1. Summary Statistics Parameters

As a first attempt, I decided to work with the dataset I built in the previous phases with the goal of analysing the variables and the trend. More specifically, this dataset contained summary statistics parameters per row. In particular, as it is possible to observe in the picture below, it contained the mean, standard deviation, sum of all the numbers in the row different from zero (non_zero_values), and the number of unique values per row, the skewness, the median, minimum and maximum. It also contained the Id, but this was deleted immediately.

```
> str(train_describe)
'data.frame':	4459 obs. of  10 variables:
 $ id                : Factor w/ 4459 levels "000d6aaf2","000fbd867",..: 1 2 3 4 5 6 7 8 9 10 ...
 $ mean              : num  7066356 7939801 4233333 1517758 6872846 ...
 $ standard_deviation: num  10020033 10662378 3651269 2185159 9734325 ...
 $ non_zero_values   : int  102 67 18 22 26 761 136 30 223 49 ...
 $ unique_non_zero   : num  52 22 11 12 14 55 54 21 18 29 ...
 $ skewness          : num  1.955 2.317 0.818 1.261 2.336 ...
 $ median            : num  2400000 2225000 4000000 261333 4700000 ...
 $ min               : num  250000 800000 200000 2000 60000 4000 2000 200000 4000 60000 ...
 $ max               : num  40000000 50000000 12000000 6000000 37662000 ...
 $ target            : num  38000000 600000 10000000 2000000 14400000 2800000 164000 600000 979000
```

The fact that I decided to work with another dataset gave me the necessity to operate again the previous phases (pre-processing and data transformation) since the dataset was completely different and no actions of cleaning were taken before. As a consequence, I handled missing values with imputation, fixed the skewness, standardised and applied Spatial Sign Transformation. Moreover, I checked the risk of correlation, which I believed could be high in some cases, since I had the possibility to check this dataset during my exploratory analysis. In this context, I removed variables with a correlation above 0.75, and the final version of the dataset looked in this way:




```
'data.frame':   4459 obs. of  5 variables:
 $ non_zero_values   : int  102 67 18 22 26 761 136 30 223 49 ...
 $ min               : num  250000 800000 200000 2000 60000 4000 2000 200000
 $ standard_deviation: num  10020033 10662378 3651269 2185159 9734325 ...
 $ skewness          : num  1.955 2.317 0.818 1.261 2.336 ...
 $ target            : num  100000 4000000 10909000 800000 28750000 ...
```

After the necessary modification to the dataset, I applied the first algorithm with the goal of predicting the variable Target. It is important to notice how during all the Feature Selection phase, I have had a pretty standard attitude in terms of algorithms application: specifically, I applied a Linear Regression model by using Forward feature selection method, which I modified by helping myself with p-values significance. Despite this model is the simplest in the range of Machine Learning algorithms, this strategy would allow me to verify that everything was running smoothly, and gave me a starting point. Following, I chose one of the most powerful Machine Learning methods, which is Extreme Gradient Boosting. In this context, I decided to directly look for flexible and powerful algorithm, which would not give problems when fed with high dimensional data. However, it is important to notice how in all the choices I have been pretty standard in the parameter tuning, by using the same set for the different methods of Feature selection I applied.

For example, in the first case, I applied the Boosting method, with a set of parameters which has been constant along all the different strategies applied.

```
gbm(target ~., data= describe_train, distribution= "gaussian",
n.trees=5000, interaction.depth = 4,
shrinkage = 0.01, verbose = F)
```

For more details about the parameters used with the XGBoost algorithm, please refer to the Script.

EVALUATION METHOD: RMSLE vs NRMSE

As the evaluation method, I used the **Root Mean Squared Logarithmic Error**, which is the measure of the ratio of predicted and actual. This parameter is very useful when targets have exponential growth, when we care about percentage errors rather than absolute value of errors, and when we want to penalise under estimates more than over estimates (which can be the case of underestimate a transaction)

Moreover, I used the **Normalised Root Mean Squared Error**, which allowed me to have the RMSE parameter, avoiding the high numbers deriving by the fact that the predicted values were not normalised and owned huge values.

Attempt I:

Summary of Statistical Parameters

Methods applied and Results:

Boosting: NRMSE 103.6 / RMSLE 2.075005

XGBoost: NRMSE 112.3 / RMSLE 2.119104

2. 150 Variables With Most Variance

The second attempt I made has been the one of training a model by using only the 150 variables with most variance. In an earlier section, I mentioned the fact that I selected the variables with higher variance by setting the opportune threshold, using NearZeroVar function. In this phase, I filtered the dataset with the aim of maintaining only those variables. Here I decided to first apply Linear Regression, in order to verify that everything worked smoothly, and to have an idea of the parameters in terms of R-squared. Since the number of variables was not too high, I had the possibility of applying the stepforward method of feature selection (the so-called mixed method), which allowed me to define a set of variables to take into consideration. Moreover, I decided to reduce the number of variables by taking into account the results provided by the p-values, which suggested me the necessity of reducing the number of features involved in the linear model.

Attempt II:

150 Variables with Higher Variance

Methods applied and Results:

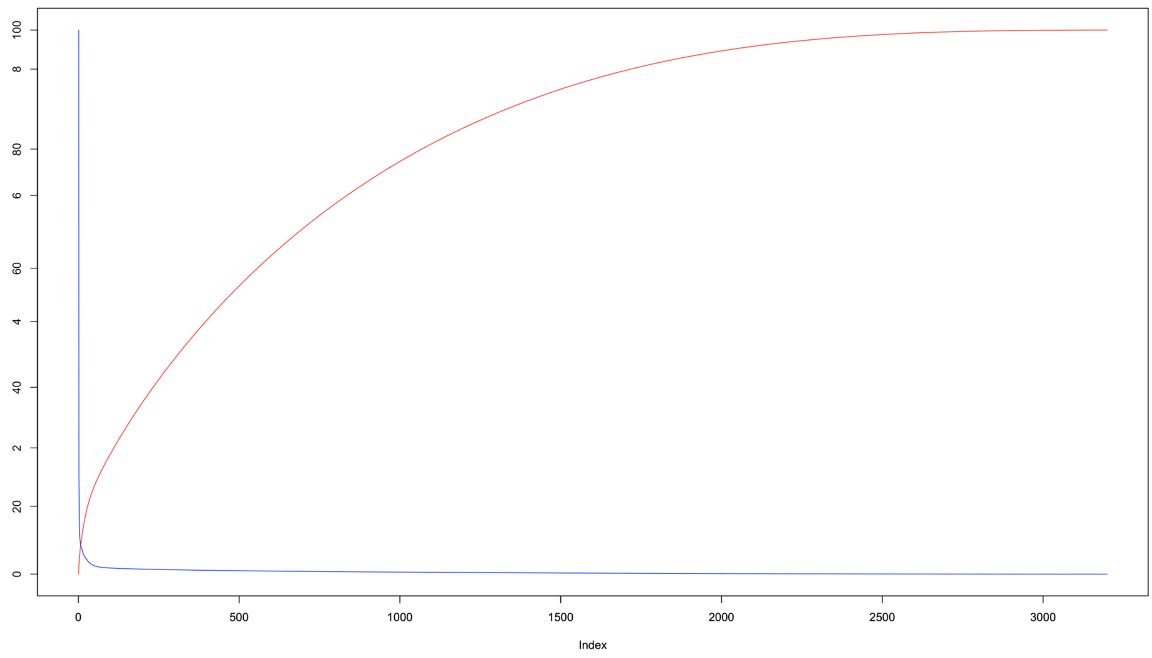
Linear Regression: NRMSE 100.6 RMSLE 2.082953

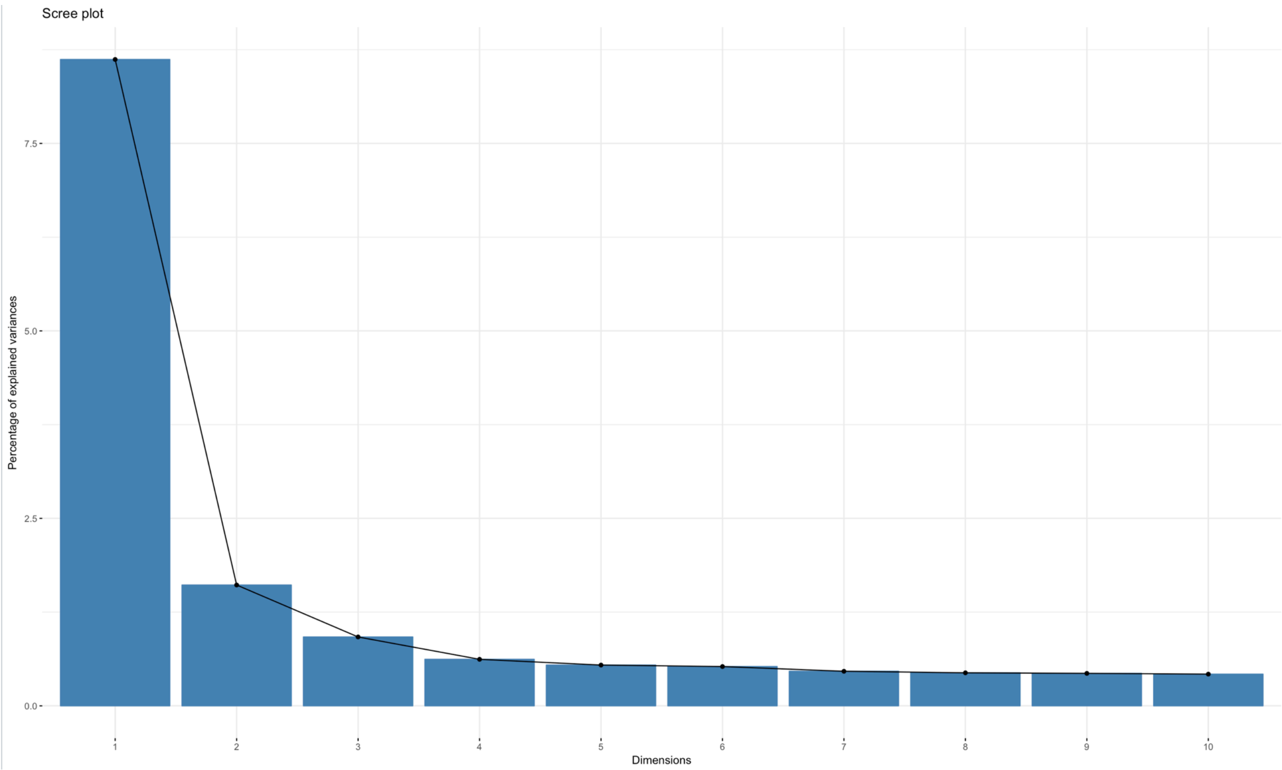
Boosting: NRMSE 95.6 RMSLE 1.97522

XGBoost: NRMSE 96.4 RMSLE 1.94268

3. Principal Component Analysis

After two approaches which I would define introductory, in the sense that they have been useful for me to understand the direction of the task, as well as attempting to verify the limits of the dataset, I introduce the first true method of Dimensionality Reduction (and) Feature Selection. The principal Component analysis represented a necessary attempt to be made in the context of high dimensionality dataset since its function is the one of trying to capture the highest possible amount of variance by converting a set of observations into a set of linearly uncorrelated variables which are called principal components. In this context, I had to first ensure that the variables that I would input into the dataset were numerical, which was the case. The only issue by applying PCA would have been the one of losing the ability to explain the data introduced into the model. However, in this case, this did not represent an issue, since it was already unclear what these variables represent.
After having calculated the Principal Component, the problem regarded the necessity to defining the number of them to keep and to work on. From this point of view, there were different opportunities I could work on. However, the simplest and widely used strategy did not represent a plausible strategy for this case. In fact, the Elbow method (Scree Test) which suggests defining the threshold for the selection of the Principal Component where the single variance of the component starts to become flat, was not a realistic strategy for me to apply. In fact, If I had applied this strategy, I would have ended up having only 3 components, for a total of Variance explained of 11%, as it is possible to observe in the two graphs below.





In this context, two approaches seemed reasonable to me:

. **Heuristic Approach**: a practical approach consisting into looking for the number components which allow working with 85% of variance explained. In this case 1298 dimensions.

. **Keyser-Guttman Rule**: an approach which suggests keeping only those components whose Eigenvalue > 1. In this case 1193 dimensions.

A third approach, probably more complete, would suggest performing **Parallel Analysis**. However, my laptop did not survive at any of the attempt to perform this type of analysis I made.

Moving from the two approaches aforementioned, I decided to roughly average their results and keeping 1240 Principal Components.

As soon as I defined the number of Principal Components to keep, I decided to apply a Linear Regression, as always, in order to verify the process, before moving into more advanced Machine Learning methods, such as Boosting and XGboost.

!!! NOTE: Unfortunately, from now on some of my results will not be available. In fact, two hours before my deadline, my laptop decided to crash, I verified how the code `save.image('results')` gave me prove to not have worked.
This meant I had to run again all my codes, with a huge amount of time lost, as well as some of the objective I had before submitting my coursework, impossible to realise (such as, Deep Learning Neural Network Model).
However, the code are available and highly detailed on my script, ready to be run again.

Attempt III

Principal Component Analysis

Methods applied and Results:

Linear Regression: NRMSE 91.8 RMSLE 1.983927

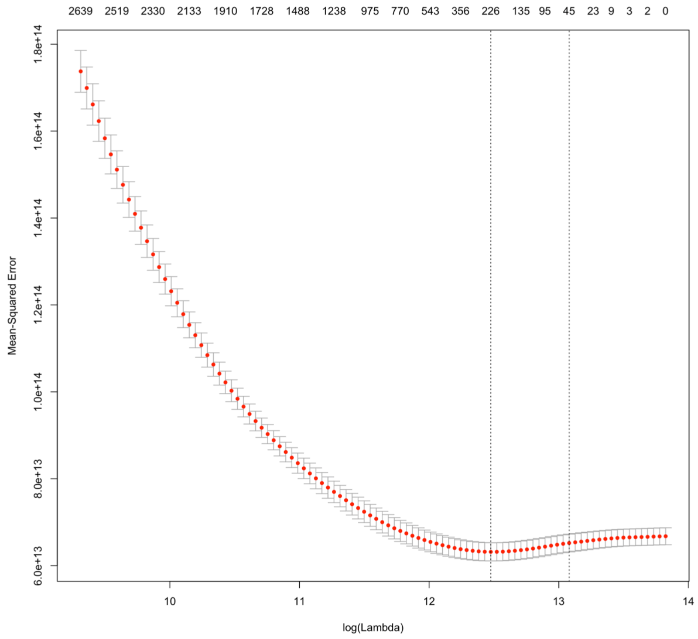
Boosting: NRMSE .. RMSLE ..

XGBoost: NRMSE .. RMSLE ..

4. Lasso Regression

Another necessary attempt in the context of High Dimensionality data and the necessity of performing Feature Selection/ Dimensionality Reduction, is represented by the opportunity granted by the Lasso Regression method. In fact, Lasso penalizes the absolute size of the regression coefficients, based on the value of a tuning parameter λ. When there are many possible predictors, many of which actually exercise zero to little influence on a target variable, the lasso can be especially useful in variable selection, by bringing those coefficient to zero (something which is not possible for Ridge regression, since its way of penalising does not allow any value to be zero, but eventually close to it). Performing a linear regression using Lasso is not very useful for the results in itself, in terms of accuracy. It is more useful because it allows to only keep the variables that Lasso kept while training, and use them into another, and and eventually more powerful, algorithm. By keeping only the variables selected by

Lasso regression, I ended up with 226 variables, which I used to build the Linear Regression, Boosting, and XGBoost models.



Attempt IV: Lasso Feature Selection **Methods applied and Results:** Linear Regression: NRMSE 99.5 RMSLE 2.063868 Boosting: NRMSE .. RMSLE .. XGBoost: NRMSE .. RMSLE ..

5. Elastic Net

An attempt has been made by using Elastic Net, which combines the Lasso and Ridge methods together. However, the results were so that similar in terms of feature selection (with only 3 variables of difference) that I did not move into this, as it would have ended up giving the same results of the models applied after Lasso Feature Selection.

6. Lasso Using P C A

By applying Lasso feature selection to the PCA dataset, I obtained 192 variables, which I fed into different algorithms obtaining the following results:

Attempt VI:

Lasso Feature Selection with PCA

Methods applied and Results:

Boosting: NRMSE 92 RMSLE 1.904634

XGBoost: NRMSE .. RMSLE ..

7. Combine The Results

I tried to add some variables from the statistical summary dataset to the Principal Component dataset, in order to verify whether this would have added some improvement to the model. By performing XGBoost I could have the possibility of verifying the variables in order of importance for the model. In this way, I could have an idea whether the statistical parameters that I added (Non_zero_values, mean, standard deviation, skewness) could be into the first 20 or 30 parameters. However, they turned out to have no great influence on the model, despite this model had an improvement in comparison to the previous ones. This fact gave me the idea to apply another strategy, which will be described in the next paragraph.

Attempt VII:

PCA combined to Descriptive Statistics variables

Methods and Results:

XGBoost: NRMSE 91.8 RMSLE 1.893161

8. Keep The First 50 PCA's Selected By XGBoost

The possibility to consult the results from the XGBoost algorithm in terms of feature importance, allowed me to store into a variable the names of the first most influent PCA's. In this context, I had the possibility to create a dataset which only kept those variables, and to feed them into the different models. This strategy allowed me to obtain the best result so far, with an RMSLE of 1.68.

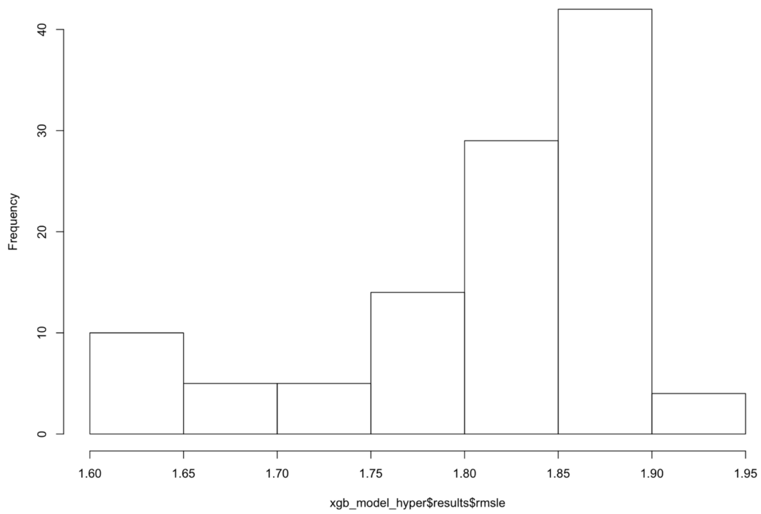
Model Improvements

Before moving into a deeper attention to the model building, I tried to increase the number of PCA's to take in consideration, by keeping 80 of them. However, the results seemed to be worse than before, so I selected as my last dataset the one containing 50 Principal Components and I moved into a phase in which I wanted to give more attention to the models, by applying two strategies:

1) Try to obtain the best hyper-parameters out of XGboost algorithms

```
xgbGrid <- expand.grid(nrounds = c(100,200,300), # max number of trees to build
                       max_depth = c(5, 10, 15, 20),
                       colsample_bytree = seq(0.5, 0.9, length.out = 5),
                       eta = c(0.1,0.01), # learning rate
                       gamma=c(0.1),
                       min_child_weight = c(1),
                       subsample = c(0.8) 
)
```

This grid allowed me to obtain the best result of the analysis, with a RMSLE of 1.62 on the validation set.



The histogram shows the results in terms of RMSLE with the attempts made on all the different parameters in which expand.grid has been set, with a cv set to 10.

2) Try to feed the model into a Deep Neural Network.

Unfortunately, this phase will be performed in the future. Despite it has not been performed, I believe it is very important to mention it, as it seems to represent a useful method due to the characteristics of the data, with a high number of variables.