June 14, 2020

QFRM Assignment 4

Abdullah Jakoby (2548344)
Achraf Guetbach (2542045)

Supervisor:
Prof. Borovkova

1 Introduction

For this case we created a credit scoring model that predicts the probability of credit default. For this we used a dataset from kaggle.com. First we cleaned the data and plotted some basic visualizations. Second we created a simple logistic regression model and analysed the residuals and other statics. Third we created an ensemble model of multiple logistic regression models. Finally we compared the results of the different models. In this report we describe each step in detail. We provide the results in a a table along with a short description of the metrics we used. Finally, we summarize our findings in the conclusion. For this case we used Python 3.7.

2 Data Analysis

In this part we describe the properties of the dataset. We show plots to visualize some variable and we describe the cleaning process of the data. The whole data cleaning process in visually represented in a flowchart in Appendix C.

2.1 Data description

For this project we used the Home Equity dataset from Kaggle [1]. This dataset contains 5960 observations of loan performance ionformation for home equities. The dataset contains twelve explanatory variables and one dependent variable "BAD" to denote a credit default. The "BAD" variable contains binary values where a 0 means the loan was repaid and a 1 means the loan was defaulted on.

Variable name	Description
BAD	1 = client defaulted on loan 0 = loan repaid
LOAN	Amount of the loan request
MORTDUE	Amount due on existing mortgage
VALUE	Value of current property
REASON	DebtCon = debt consolidation HomeImp = home improvement
JOB	Six occupational categories
YOJ	Years at present job
DEROG	Number of major derogatory reports
DELINQ	Number of delinquent credit lines
CLAGE	Age of oldest trade line in months
NINQ	Number of recent credit lines
CLNO	Number of credit lines
DEBTINC	Debt-to-income ratio

Table 1: Variable descriptions

2.1.1 Descriptive statistics

The data consists of 12 variable, ten are continuous variables and two are discrete variables (See appendix C for the summary statics). There are two main issues with the data. First, there are two discrete variables which contain strings representing each category in the data. These variables are "REASON" and "JOB", describing the reason of default and the occupation of the loanee, respectively. Second, there are a number of missing values in the data, which cannot be handled by the logistic regression directly.

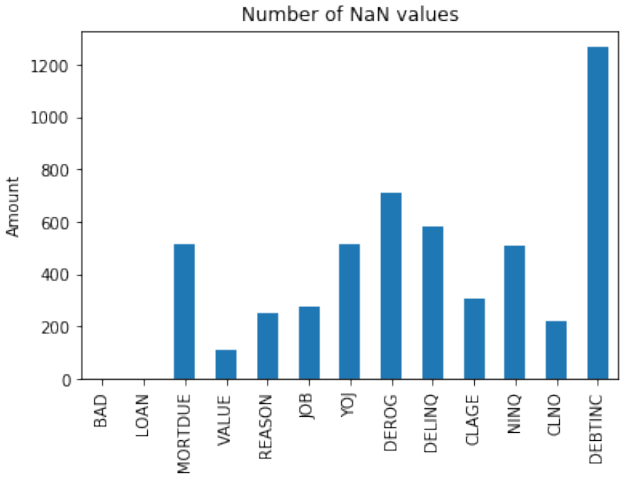

Figure 1: Missing values per variable

2.2 Data cleaning

To solve the first issue we converted the discrete variables to dummy variables. For the missing missing values in the discrete variables we created separate dummy variables. For the missing values in the continouos variables we took took another approach. At first we just replaces all missing values with the mean value of the columns. However, for some variables this seemed not the right approach. When we looked at the distribution of some of the variables we found that they were heavily skewed to one side. Taking the median in this case would automatically result in a value of zero. And given the fact the these two variable contain a large number of missing values, filling them up with the median would alter the distribution too much.

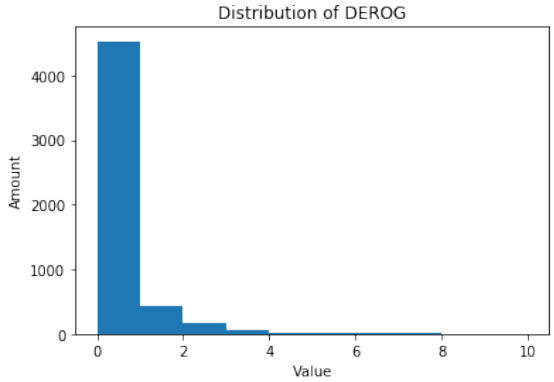

Table 2: Distribution DEROG

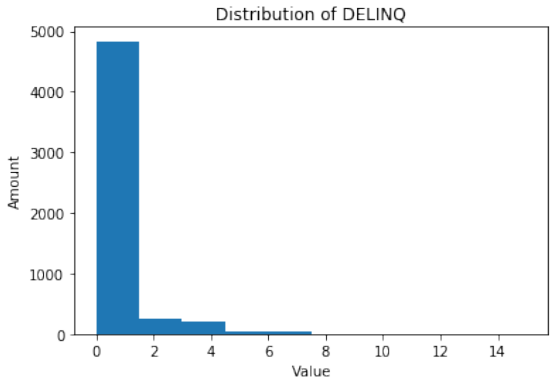

Table 3: Distribution DELINQ

Instead we filled these variables with the mean value and we also experimented with trying to create a model to predict these missing values.

2.2.1 Missing value model

Taking the mean of a variable is a very imprecise way of imputing missing values. Especially when we are dealing with very skewed distributions. To counter this we came up with a model that predicts the missing values for DEROG and DELINQ since these two variables by this problem.

To do this we took all variables in the dataset, with the exception of DEROG, DELINQ and BAD, and we estimated a regression model to predict the missing values. For this we used the GradientBoostingRegressor from the Scikit-learn package in python [3]. At first we used a standard linear regression model, but that lead to very poor performance of our eventual model. We found that the the performance of our credit scoring model increased by using this approach. See the results section for more detail.

2.3 Data transformations

We also found that some features contained a large outlier values or a skewed distribution. To improve the performance of our model we transformed some of these features with mathematical operations. In the plots below we see the distributions of the variables we transformed.

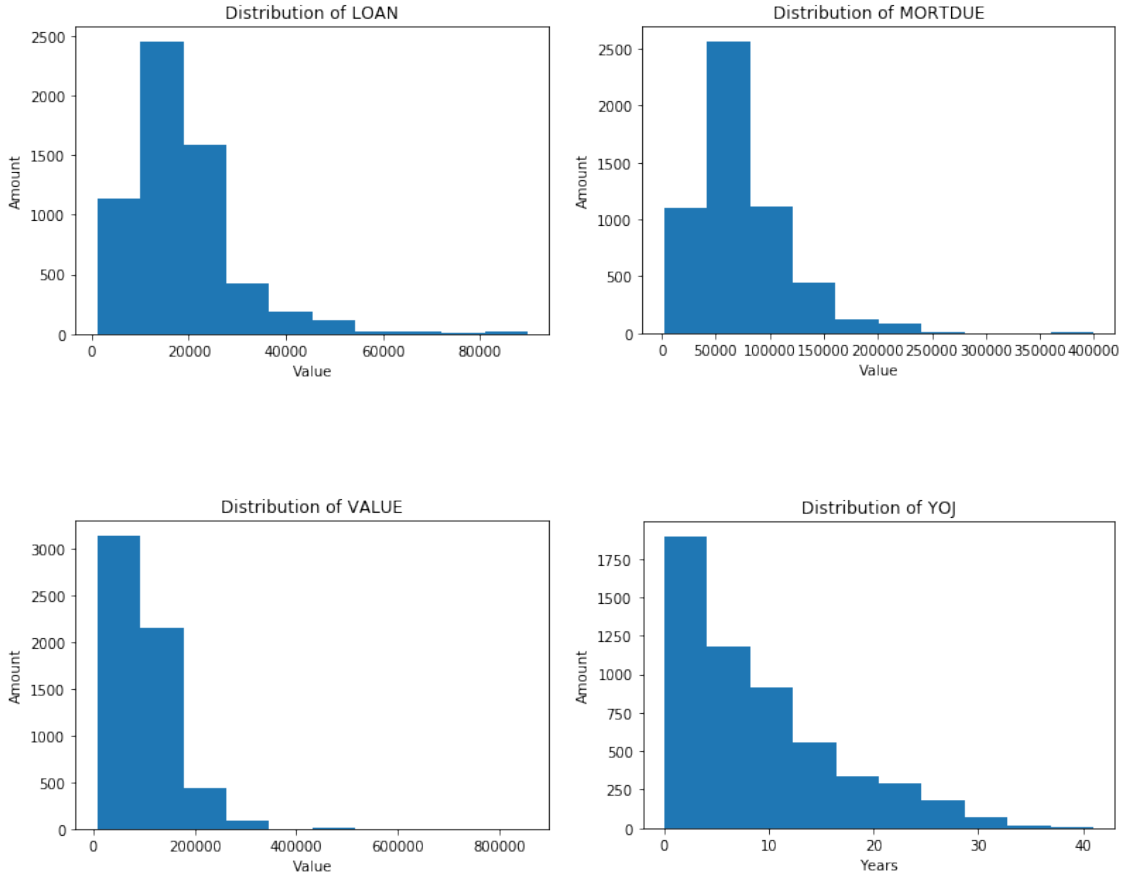

Table 4: Distributions before transformation

By taking to log for LOAN, MORTDUE and VALUE and by taking the square root for the YOJ value we were able to transform these variables into bell shaped distributions. This eliminated large outlier values and improved the performance of our model. The distributions of the variables after transformation can be found in Appendix B.

3 Models

In this section we will decribe the two models that we used. We started with a simple logistic regression model. We then create an ensemble of multiple logistic regression and we compared the performance of both models. We made use of both the Scikit-learn package[1] and the Statsmodels package[2] for our logistic regression model. For the simple model we used the stats models version to generate the descriptive statistics table where all the coefficients are presented. However, for the Ensemble model we used te Scikit learn version of the model. Also when we compare the performance of both models we use the Sckit-learn version. We did this because the Scikit-learn version of the model is easier to implement in a ensemble model, but unfortunately it lacks a summary table functionality thereby making ik hard to evaluate the coefficients and their p-values.

3.1 Simple Logistic Regression

We used all the features as described in the previous section and used them as input for our logistic regression. We used the logistic regression from the statsmodels package in Python [2]. This package represents the results of the model in a very neat manner.

3.2 Ensemble Model

Using a single model to make predictions can often be problematic. A single model can always overfit, especially when there is not a lot of data available. To counter this we created an ensemble model that combines the predictions of multiple logistic regression models together. This is done as follows. Fist we split the data into a training set and a test set. The training set is used to estimate our parameters. The test set is used to make predictions in order to evaluate our model's performance. Second, randomly sample a number of columns from the data without replacements and we sample rows with replacements. We repeat this process N times, thereby creating N different datasets. Third, we estimate a different logistic regression model on each sampled dataset. Finally, use the test set to make predictions for each model. We stored the predictions in a matrix where each column represented a prediction and each row represented an observation. The predictions are then averaged by row to create the final prediction of our ensemble model.

To further increase the performance of our model we used another logistic regression model to predict the final model prediction. We did this by using the matrix with the predictions of our ensemble model as explanatory variables in the final model. The figure in Appendix D depicts the whole process schematically.

4 Results

In the plot below we see that our simple model has an pseudo R squared of 0.2302. The table in the picture below also shows that, at a significance level of 0.05, there is no relation between credit default and taking a loan for debt consolidation or home improvement. This also goes for a loanee with that works an office job or as a manager. Another remarkable finding is that the coefficients for the variables LOAN, MORTDUE and VALUE are very close to zero and also significant at a significance level of 0.05. This means that in our case the amount of money involved in the loan is not indicative of whether a credit default.

Logit Regression Results

Dep. Variable:	BAD	No. Observations:	3993
Model:	Logit	Df Residuals:	3974
Method:	MLE	Df Model:	18
Date:	Sat, 13 Jun 2020	Pseudo R-squ.:	0.2302
Time:	09:35:55	Log-Likelihood:	-1492.7
converged:	True	LL-Null:	-1939.1
Covariance Type:	nonrobust	LLR p-value:	5.643e-178

	coef	std err	z	P>|z|	[0.025	0.975]
LOAN	-1.84e-05	5.15e-06	-3.573	0.000	-2.85e-05	-8.31e-06
MORTDUE	-4.297e-06	1.78e-06	-2.408	0.016	-7.79e-06	-7.99e-07
VALUE	3.947e-06	1.28e-06	3.083	0.002	1.44e-06	6.46e-06
YOJ	-0.0131	0.007	-1.827	0.068	-0.027	0.001
DEROG	0.6134	0.060	10.261	0.000	0.496	0.731
DELINQ	0.7542	0.049	15.424	0.000	0.658	0.850
CLAGE	-0.0054	0.001	-7.852	0.000	-0.007	-0.004
NINQ	0.1781	0.025	7.017	0.000	0.128	0.228
CLNO	-0.0191	0.005	-3.579	0.000	-0.030	-0.009
DEBTINC	0.0658	0.008	8.258	0.000	0.050	0.081
Reason_DebtCon	-0.4233	0.289	-1.466	0.143	-0.989	0.143
Reason_HomeImp	-0.1061	0.295	-0.360	0.719	-0.683	0.471
Job_Mgr	0.6336	0.328	1.935	0.053	-0.008	1.276
Job_Office	0.1501	0.331	0.454	0.650	-0.498	0.798
Job_Other	0.9319	0.308	3.026	0.002	0.328	1.536
Job_ProfExe	0.8180	0.321	2.552	0.011	0.190	1.446
Job_Sales	2.3080	0.408	5.659	0.000	1.509	3.107
Job_Self	1.2661	0.380	3.332	0.001	0.521	2.011
Intercept	-3.4051	0.424	-8.027	0.000	-4.237	-2.574

Figure 2: Logistic regression model summary

4.0.1 Metrics

To compare all variations of our model we a we will be using different metrics, namely accuracy, precision, recall and the F1 score. These metrics are defined as follows.

The Accuracy is a metric that reflects the amount of True Positives out of the total number of observations.

$$Accuracy = \frac{TruePositives}{TotalNumberOfObservations} \tag{1}$$

Precision is defined as the number of times the model predicted a certain class correctly divided by

the number of times it predicted that class in total.

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{2}$$

Recall is defined as the number of times the model predicted a particular class correctly to be "true" divided by the total number of true observations of that class.

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{3}$$

The F1 score is a metric of model performance. It is the harmonic mean of both both the precision and the recall.

$$F1score = 2 \cdot \frac{Precicion \cdot Recall}{Precicion + Recall} \tag{4}$$

4.0.2 Performance table

In the table below we see the performance for both the simple model and the ensemble model, according to the metrics we defined before. We made a distinction between an Ensemble model where we use the mean for the final prediction and an Ensemble where we use a model to make a final prediction (see Figure 2). Furthermore, we look at the effect of NaN prediction in for the missing values and at the effect of row sampling. Since row sampling is only used for the Ensemble model, it was not shown for the simple model.

We see that NaN prediction has a positive effect on the accuracy and F1 score for all models. For all models it seems to cause a small increase of the recall at the cost of a slightly lower precision precision. For the Ensemble model with mean, the row sampling seems the have the same effect as the NaN prediction. On the contrary, for the Ensemble model with prediction the row sampling increases both precision and recall, aside from also increasing accuracy and the F1. When we combine both the NaN prediction and the row sampling, we see a small increase precision and a decrease in recall for both model. In both cases the accuracy and F1 score increase.

When we compare the simple model with the Ensemble with mean model, we see that the simple logit has a higher accuracy and precision and a lower recall and F1 score. The Ensemble with prediction perform better that both models on all metrics.

Model	NaN prediction	Row sampling	Accuracy	Precision	Recall	F1
Simple Logit	No	-	0,823	0,7187	0,3194	0,4423
Simple Logit	Yes	-	0,8464	0,6961	0,3378	0,4548
Ensemble with Mean	No	No	0,8098	0,5833	0,4699	0,5205
Ensemble with Mean	Yes	No	0,8235	0,5369	0,5067	0,5213
Ensemble with Mean	No	Yes	0,7966	0,5394	0,5069	0,5226
Ensemble with Mean	Yes	Yes	0,8149	0,5107	0,571	0,5392
Ensemble with Prediction	No	No	0,8586	0,923	0,3888	0,5472
Ensemble with Prediction	Yes	No	0,884	0,8795	0,4504	0,5957
Ensemble with Prediction	No	Yes	0,8734	0,9463	0,449	0,6091
Ensemble with Prediction	Yes	Yes	0,8937	0,9226	0,4798	0,6313

Table 5: Performance for all model variations

5 Conclusion

When we evaluated the simple model we noticed that for this dataset the money involved in the loan was not indicative of whether the loanee would default on their debt. This is most likely due to the fact that loans are issues based on the risk profile of the loanee. This means that the loanee would only be able to borrow an amount of money that it could pay back in principle. We also found that there was no significat relation between the reason of taking the loan and the ability to pay it back.

During our research we found that using a model to predict NaN values can definitely increase the performance of a model. It ensures that the accuracy, recall and precision of a model increases, at the cost of a small decrease in precision. We also found that an ensemble model has significantly better performance than a simple model. However, it is important to take into account that a prediction model on top of the many predictions works better than using the mean of all predictions.

5.1 Limitations

We made use of two packages for the logistic regression. In the statsmodels version we used a constant, wheras in the Scikit-learn version we did not. The reason for this is that we did not want to further increase the performance table that we created. However, we do recognize that adding a constant might increase the performance of our model.

Furthermore, when we filled the missing values, with both the mean method and the prediction method, we did not take into account that we should apply the filling of the NaN values separately for the train and test set, because not doing that causes data leakage. This means that some information of the test set is present in the training set, which might cause the model to seemingly perform better by incorporating information of the test set.

References

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

[2] Seabold, S. and Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. *Proceedings of the 9th Python in Science Conference*

[3] Vallala, Ajay (2018). HMEQ_DataPredict clients who default on their loan *Kaggle: https://www.kaggle.com/ajay1735/hmeq-data*

6 Appendix

A Summary statistics

	BAD	LOAN	MORTDUE	VALUE	YOJ	DEROG	DELINQ	CLAGE	NINQ	CLNO	DEBTINC
count	5960.000000	5960.000000	5442.000000	5848.000000	5445.000000	5252.000000	5380.000000	5652.000000	5450.000000	5738.000000	4693.000000
mean	0.199497	18607.969799	73760.817200	101776.048741	8.922268	0.254570	0.449442	179.766275	1.186055	21.296096	33.779915
std	0.399656	11207.480417	44457.609458	57385.775334	7.573982	0.846047	1.127266	85.810092	1.728675	10.138933	8.601746
min	0.000000	1100.000000	2063.000000	8000.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.524499
25%	0.000000	11100.000000	46276.000000	66075.500000	3.000000	0.000000	0.000000	115.116702	0.000000	15.000000	29.140031
50%	0.000000	16300.000000	65019.000000	89235.500000	7.000000	0.000000	0.000000	173.466667	1.000000	20.000000	34.818262
75%	0.000000	23300.000000	91488.000000	119824.250000	13.000000	0.000000	0.000000	231.562278	2.000000	26.000000	39.003141
max	1.000000	89900.000000	399550.000000	855909.000000	41.000000	10.000000	15.000000	1168.233561	17.000000	71.000000	203.312149

Figure 3: Summary statistics continuos variables

B Data transformation

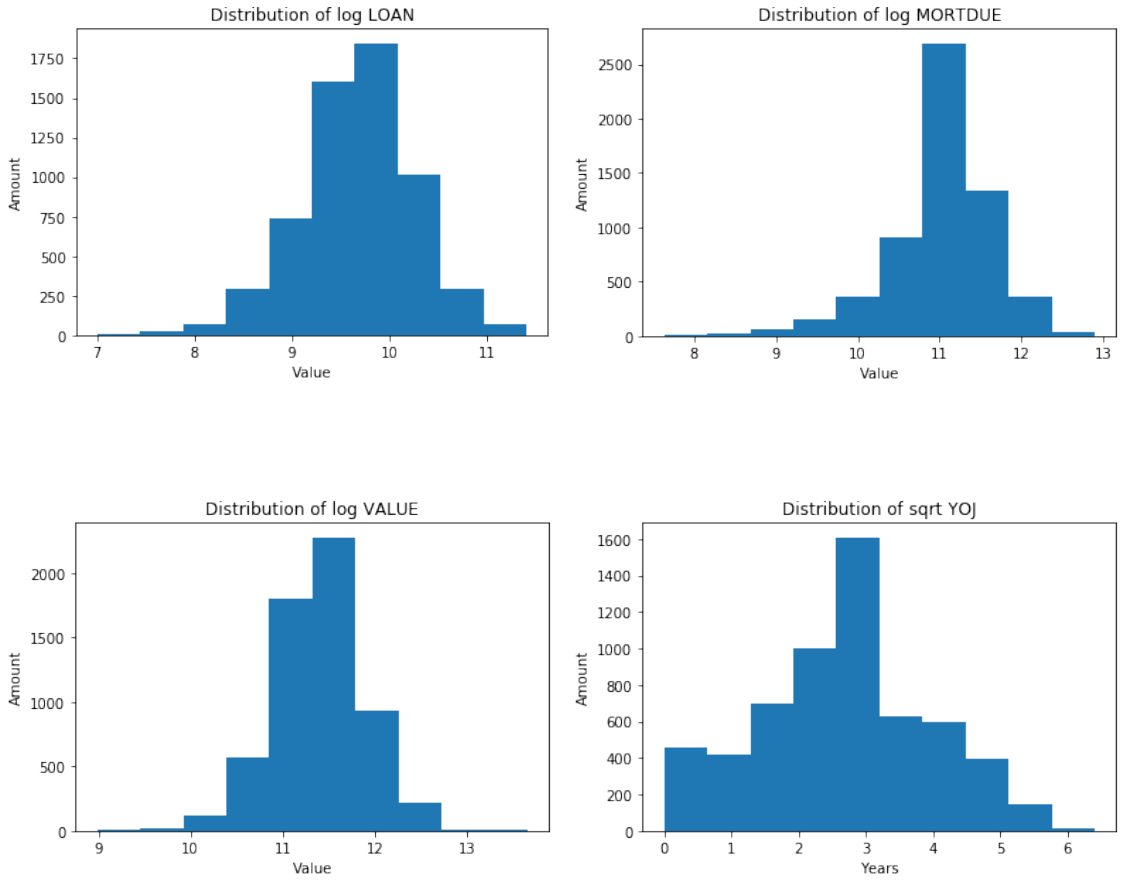

Table 6: Distributions after transformation

C Data Cleaning Process

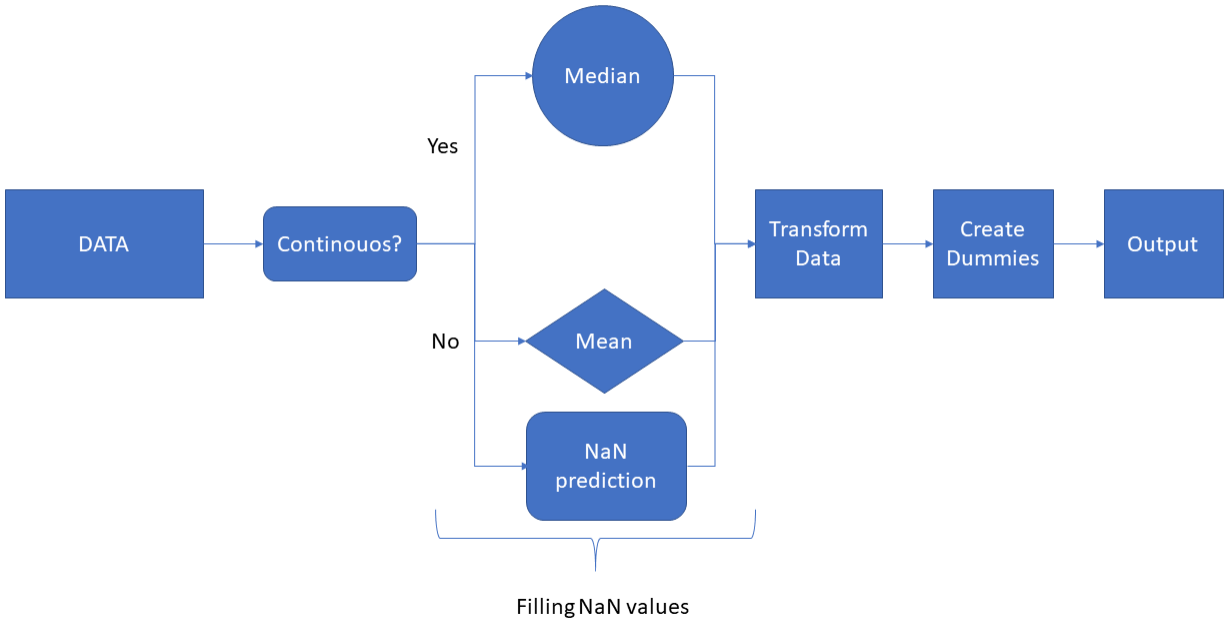

Figure 4: Flowchart data cleaning process

D Ensemble model

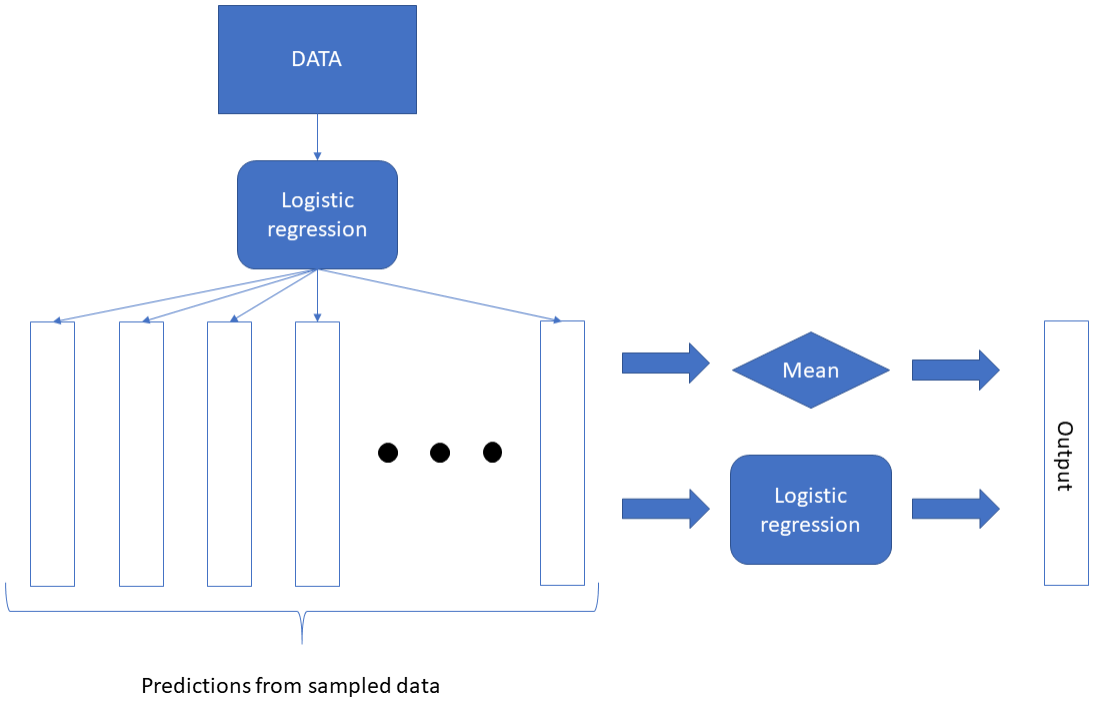

Figure 5: Flowchart Ensemble model