

School of Computer Science

Data Mining in Fulfilment of

DATA9910

Maksymilian Drzezdzon

C15311966

Degree: TU060/1

Module Coordinator: Brendan Tierney

Declaration of Ownership: I declare that the attached work is entirely my own and that all sources have been acknowledged: 🗹

**Date: 2020/11/09**

Definition of problem - Business Understanding

The purpose of this data mining project is to identify customers who are most likely to subscribe to a term deposit account based on previous marketing campaigns.

https://www.datascience-pm.com/crisp-dm-2/

[x] Definition of problem

[ ] Data exploration and descriptive analytics

[ ] Identification of data insights from previous step

[x] Details of any data prep

[ ] Details of each data mining algorithm used; config used etc

[ ] Details of evaluation and performance of measures from your data mining models, which one preformed best, why this might have been the case and how the results compare across all the models.

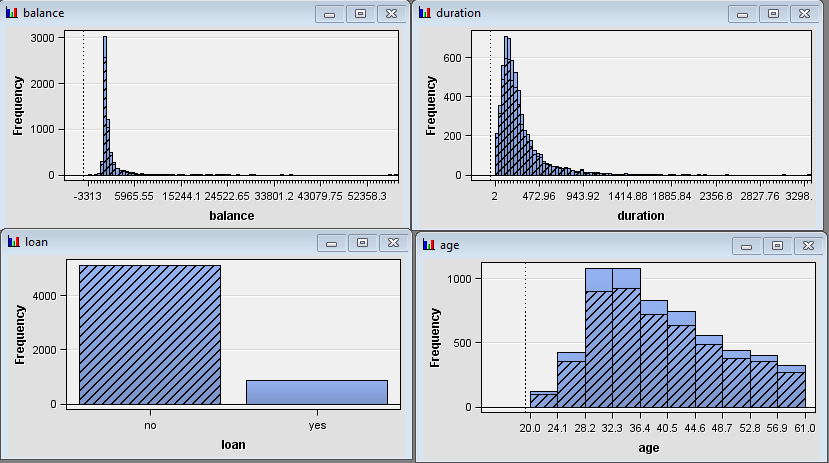
[ ] Discussion of how your results compared to the research paper and any conclusions that you can draw from this comparison

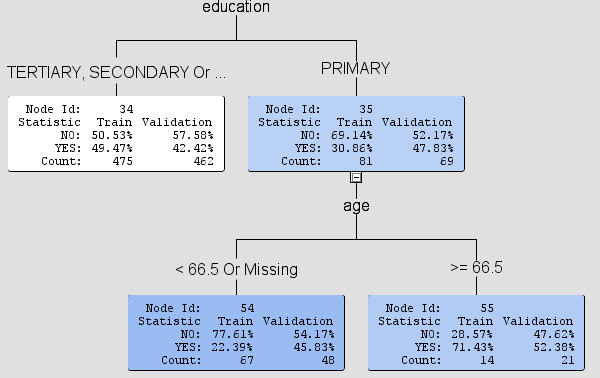
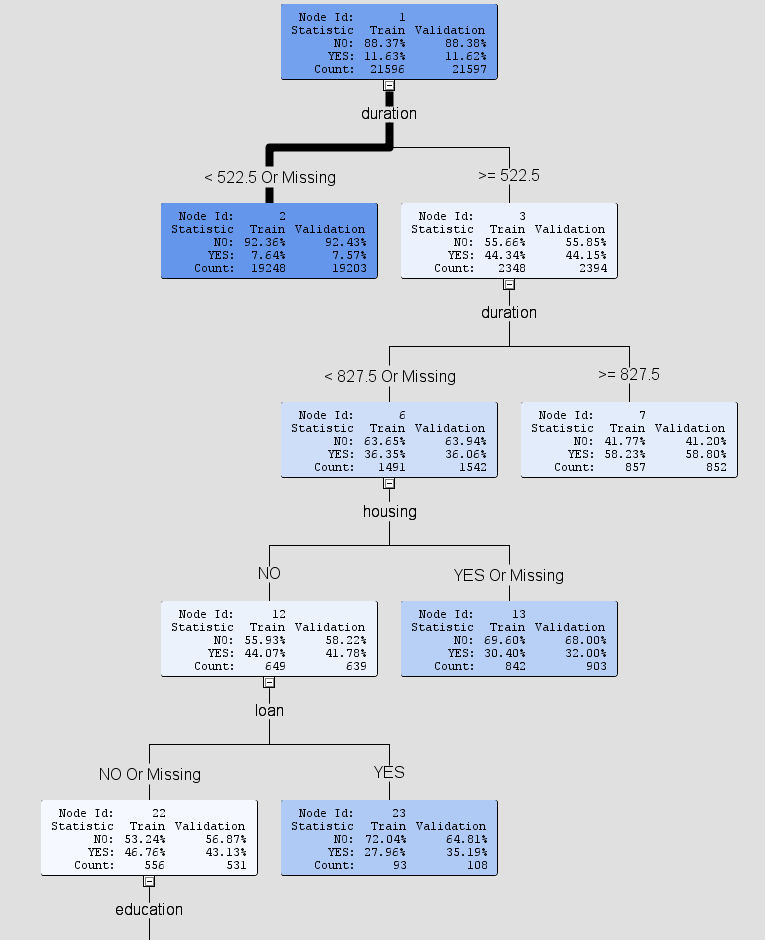
[ ] Send as PDF

[ ] You will be required to document your approach to solving and evaluating this classification problem, based on the CRISP-DM process and documentation template guide.

Data Exploration/Understanding

Data was explored using the **StatExplore** node. The workflow is Data Node – StatExplore Node.



You can see the representation of all other participants that don’t have a loan across the age, balance and duration bar charts. Another augmentation that was made to what SaS already provides was the target variable **Y** was renamed to “subscribed\_term\_deposit” as its more descriptive in the R code and all yes or no values were converted to 1 or 0 as neural networks perform better on binary data. Bin sizes were increased for duration and balance to get a better read for the representation of clumped up values. After that the data was passed through a decision tree results are as follows 

The thicker the line the more data has been passed in that direction/leaf. The lighter the shade the more impure/bad the data is at predicting the target variable, in this case it being **Y** which is whether the customer accepted the thing. According to the decision tree, the best predicators are a person doesn’t have a house, doesn’t have a loan has a higher education status of above primary school, with longer conversations not materialising into good costumers as one would expect.

**More exploring on each column?**

costumers as on

<https://learning.oreilly.com/videos/getting-started-with/9781492028406/9781492028406-video318749?autoplay=false>

<https://learning.oreilly.com/videos/ibm-spss-modeler/9781787286924/9781787286924-video2_1?autoplay=false>

<https://towardsdatascience.com/machine-learning-interpretability-techniques-662c723454f3>

<https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5>

<https://community.dataiku.com/t5/General-Discussion/White-Box-vs-Black-Box-Models/m-p/4536>

<https://medium.com/sciforce/introduction-to-the-white-box-ai-the-concept-of-interpretability-5a31e1058611>

<https://blog.dataiku.com/white-box-vs-black-box-models-balancing-interpretability-and-accuracy#:~:text=On%20the%20other%20hand%2C%20white,accuracy%2C%20but%20higher%20explainability>).

https://www.siliconrepublic.com/enterprise/white-box-machine-learning

Data Preparations

First data was explored in R and SaS enterprise miner. Values that can be dropped are **poutcome** because it’s all unknown, **previous** all values are 0, **pdays** are all -1, **months** is always may, **contact** is mostly unknown, this was done with the code below.

df = read.table("bank/bank-full.csv", header = T, sep = ";")

drop = c("poutcome", "previous", "pdays", "months", "contact")

df = df[,!(names(df) %in% drop)]

unique(df$job , incomparables = FALSE)

unique(df$education , incomparables = FALSE)

names(df)[names(df) == "y"] = "subscribed\_term\_deposit"

df$subscribed\_term\_deposit[df$subscribed\_term\_deposit == "no"] = 0

df$subscribed\_term\_deposit[df$subscribed\_term\_deposit == "yes"] = 1

df$job[df$job == "unknown"] = NA

df$education[df$education == "unknown"] = NA

df = na.omit(df)

write.csv(df, "bank-full-updated.csv", row.names = F)

 This decision which variables to keep was made by selecting all variables in SaS and clicking explore which generates visualisations, when you select a column all other graphs reflect this action and show the representation of that column in other graphs, **no** in the loan bar chart was selected.

Data Mining Models Used - Modelling

**Data Models Setup**:

Data Node – Holds Dataset.

Data Unchanged – Holds the original Dataset with 0 changes apart from setting the y variable as target and type as binary for data mining models.

Data Partition Node – Holds 50% of data for training and 50% for test.

Import Data Node – Renamed to Data and changed variable y to **binary** type from nominal and set it as the target., likewise for the unmodified dataset.

Model Comparison Node – Selection Statistic is set to Misclassification Rate with the HP selection Statistic also being set to the same as all models are configured this way unless specified in the LIFT or AUC-ROC sections.

These are the settings used for all evaluations unless specified otherwise in said evaluation.

**Logistic Regression**

Configuration used: All defaults except in the model selection the selection model was changed to stepwise, the default setting is logistic regression, linear regression has to be configured in the class targets.

This was done to have SaS enterprise miner determine the most useful inputs for the best result. [1] Another change was in the same table the selection criterion was set to validation misclassification similar to the decision tree above to get the best result possible.

Model workflow used: Usually, the workflow for a regression model would look like this Data Node – Replacement Node – Data Partition Node – Impute Node – Regression Node. This process was not followed in this case because there is no data to be replaced, instead the workflow was Data Node – Data Partition Node – Regression Node.

**Neural Network**

Configuration used: All defaults are in place; in the Train table the model selection criterion is set to misclassification to pick out the best model.

Model workflow used: Data Node – Data Partition Node – Neural Network Node – Model Comparison Node.

**Decision Tree**

Configuration used: Selection statistic is set to Misclassification Rate. All other fields are left as they were.

Model workflow used: Data Node – Data Partition Node – Decision Tree Node – Model Comparison Node.

**Gradient boost**

Configuration used: Selection statistic is set to Misclassification Rate. All other fields are left as they were.

Model workflow used: Data Node – Data Partition Node – Gradient Boost Node – Model Comparison Node.

**HP SVM**

Configuration used: Everything is left in default as it was created.

Model workflow used: Data Node – Data Partition Node – HP SVM Node – Model Comparison Node.

**Auto Neural Network**

Configuration used: Selection statistic is set to Misclassification Rate. All other fields are left as they were.

Model workflow used: Data Node – Data Partition Node – Auto Neural Network Node – Model Comparison Node.

**MBR**

Configuration used: Selection statistic is set to Misclassification Rate. All other fields are left as they were.

Model workflow used: Data Node – Data Partition Node – MBR Node – Model Comparison Node.

Model Performance – Evaluation

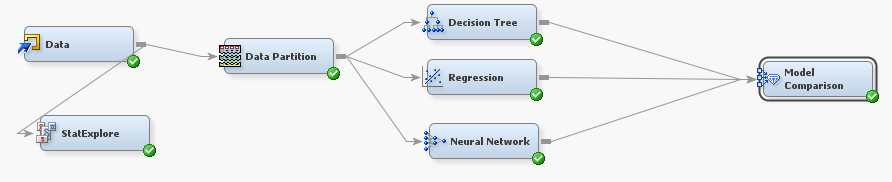
**Comparing Models**

Configuration: All settings left on default except just like before selection statistic is set to misclassification rate so the comparison node will also have the same setting and it will use the validation data in the selection table to compare the results of each model.

The data type of the target variable y was changed to binary to be compatible with the SVM algorithm.

This general layout was used for both modified and unmodified datasets, all algorithms listed in the results appended below were attached in a similar fashion and run concurrently to be compared in the model comparison node.

An educated guess before these experiments are run is that the neural network should perform the best, albeit it being a black box model, it wouldn’t be preferred for deployment especially if a white box model performs close enough to it. [100]

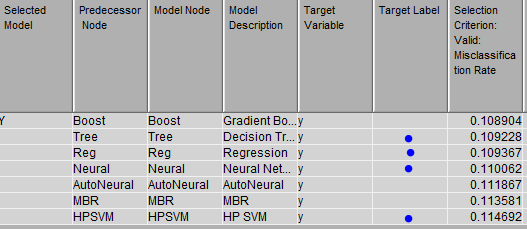


This is workflow for how models were assessed. A Data Node – Data Partition Node – Models being compared connected to the Model Comparison Node. With a result as follows for data with dropped values as explained in the data preparation section.

Models that overlap with the ones used in the original research paper are marked with a blue dot.

Results for modified bank dataset.

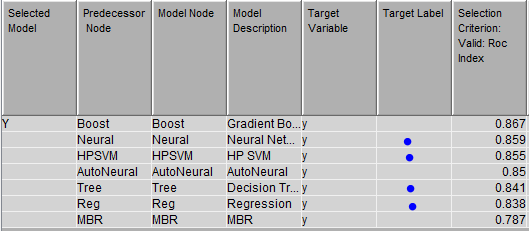
Assessed using Misclassification rate.



The best performing algorithm is the gradient boost and from the algorithms used in the research paper there is a close result between the decision tree and linear regression models of 0.1092 and 0.1093. According to these results my original hypothesis is wrong, the neural network did not perform well in comparison to the other models.

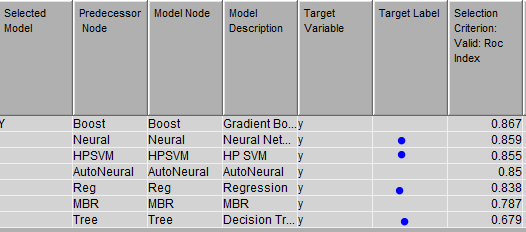
Assessed using LIFT.

An adjustment made for this run of the models was that the subtree assessment was set to LIFT from misclassification, this greatly increased the performance of the decision subtree model.



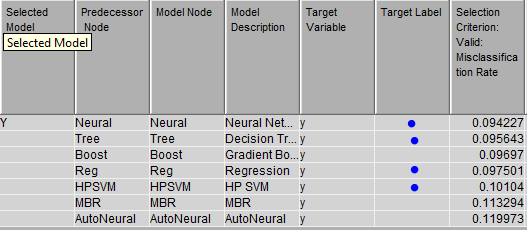
Assessed using AUC-ROC.

The model selection statistic was changed to ROC, however fort the HP Selection Statistic for the HP SVM model no such option exists; therefore, it was left as misclassification rate.



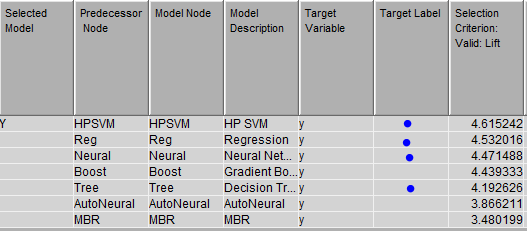
Results for the unmodified bank dataset.

Assessed using Misclassification rate.



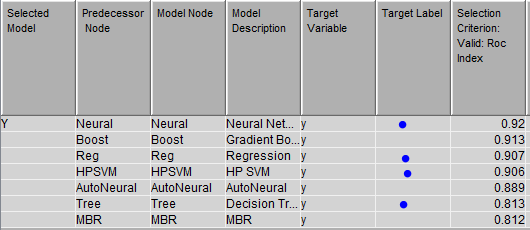
In this case the neural network did perform the best with a score of 0.0942 with the decision tree model behind it with 0.0956.

Assessed using LIFT.



Assessed using AUC-ROC. [2]

The model selection statistic was changed to ROC, however fort the HP Selection Statistic for the HP SVM model no such option exists; therefore, it was left as misclassification rate



When assessing these models on the unchanged data.

According to these results the best preforming models are: (if an algorithm used in the paper is not first in a category then the first model from the paper on the results table will be added for the sake of comparison with its place on the list)

Modified Dataset:

Misclassification – Gradient Boost with the Decision Tree on 2nd place.

LIFT - Gradient Boost with the Neural Network on 2nd place.

AUC-ROC – Gradient Boost with the Neural Network on 2nd place.

Unmodified Dataset:

Misclassification – Neural Network and Decision Tree.

LIFT – Support Vector Machine, Logistic Regression and Neural Network in that order preform very close to each other.

AUC-ROC – Neural Network and Logistic Regression on 3rd place.

How Did Results Compare To Original Research?

In the original paper titled A Data-Driven Approach to Predict the Success of Bank Telemarketing, logistic regression, decision trees, neural networks and support vector machine were used.

Note original findings

Appendix

<https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5> [2]

References

<https://documentation.sas.com/?docsetId=emref&docsetTarget=n1jqzz8cssr9m2n1ktx2iyv87q56.htm&docsetVersion=14.3&locale=en#n1m5w9deopojaqn1jykkfp0x5fcy> [1]