CHURN PREDICTION

Marketing Data Science

Prof: Repoussis Panagiotis

Student: Chalkiopoulos Georgios

TABLE OF CONTENTS

PROBLEM DESCRIPTION

01

Dataset

Each row represents a customer, each column contains customer's attributes described on the column Metadata.

The data set includes information about:

- Customers who left within the last month – the column is called Churn
- Services that each customer has signed up for
- Customer account information

The Dataset is available here.

Goal & Approach

Our target is to build a classification model which will predict customer's churn.

We will use RapidMiner: *"a powerful data mining tool that enables everything from data mining to model deployment, and model operations"*

PROCESS

02

Software

- RapidMiner for academics was downloaded which offers a free license for students
- The software uses Operators which have to be downloaded from the extension menu
- The data, mentioned previously, was downloaded and saved locally
- Processes for each step are presented in the Appendix while the process file is available in the submission files

Steps

The process is split into the steps presented below

- Data Import
- Data Exploration
- First approach – AutoModel
    - We will get a quick result of the most famous models which will guide the development
- Model Train and Optimization
    - Feature engineering/Selection
    - Model Comparison/ Model selection
    - Grid Search/CrossValidation
    - Model Combination

Data Import

- Data was imported using the Read CSV Operator, which was piped to the Statistics Operator

- This allowed to get a first view of the data, and investigate, potential, issues presented in the following slides

Data Exploration

- Target label (Churn) is unbalanced

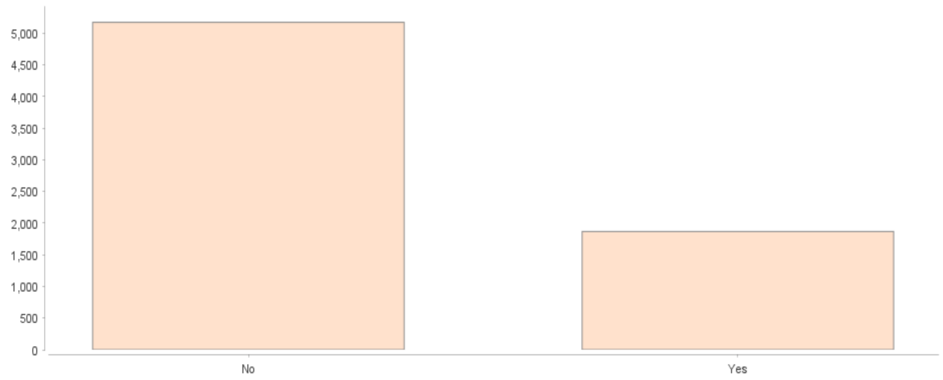
Churn

Summary

Category

Missing: 0.00%
Infinite: 0.00%
ID-ness: 0.03%
Stability: 73.46%
Valid: 26.51%

Top Values



2 Distinct Values:

Value	Count	Percentage
No	5,174	73.46%
Yes	1,869	26.54%

Data Exploration

- The only variable with missing values is TotalCharges, having 11 missing values (0.16%)

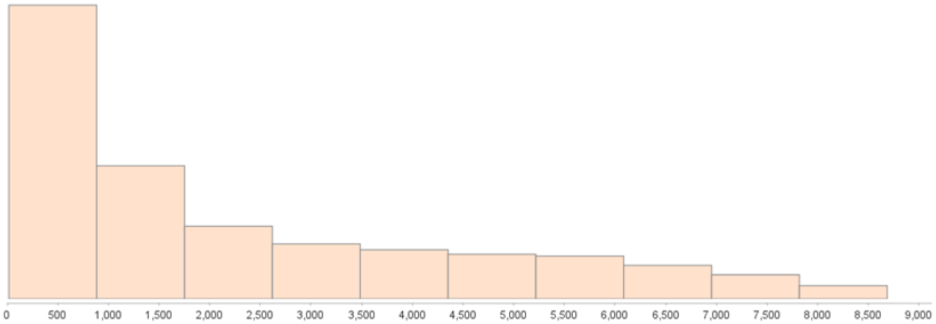
TotalCharges

Summary

Number

Missing: 0.16%
Infinite: 0.00%
ID-ness: 4.39%
Stability: 0.16%
Valid: 95.30%

Distribution



Statistics

Name	Value
Minimum	18.800
Maximum	8684.800
Average	2283.300
Standard Deviation	2266.771

Data Exploration

- The missing TotalCharges values have 0 tenure. We will impute the missing value with the MonthlyCharges one

tenure	TotalCharges	Churn
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No
0	?	No

Data Exploration

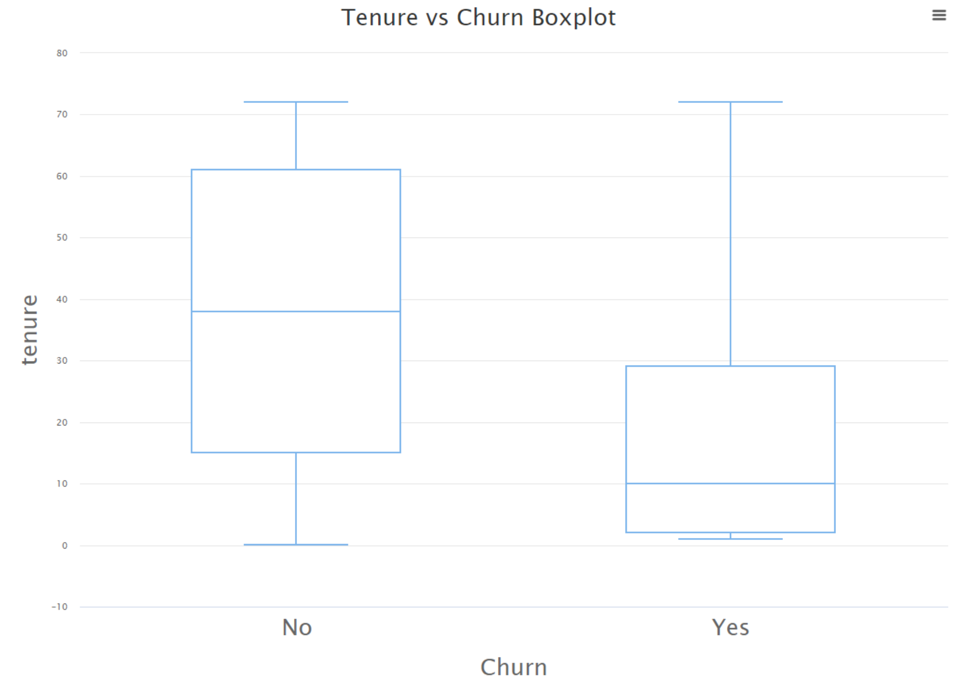
- Correlation Matrix: There seems to be a strong correlation between TotalCharges and Tenure with a smaller one between TotalCharges and MonthlyCharges

Attributes	SeniorCitizen	MonthlyCharges	tenure	TotalCharges ↑
SeniorCitizen	1	0.220	0.017	0.102
MonthlyCharges	0.220	1	0.248	0.651
tenure	0.017	0.248	1	0.826
TotalCharges	0.102	0.651	0.826	1

Data Exploration

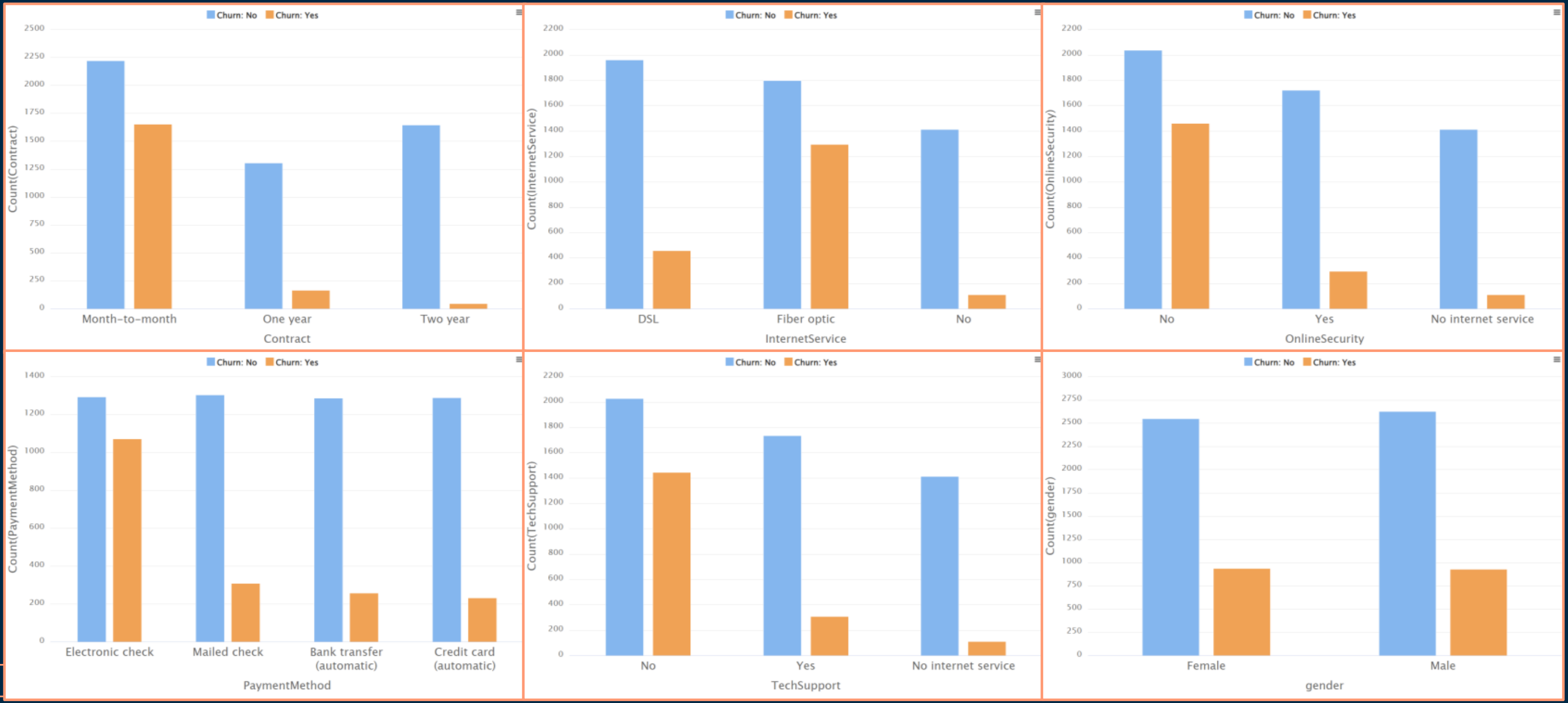
The Results Tab gives the option of generating various visualizations, to get insight of the data.

- It seems that Churn is related to Tenure:
    - New customers tend to leave early



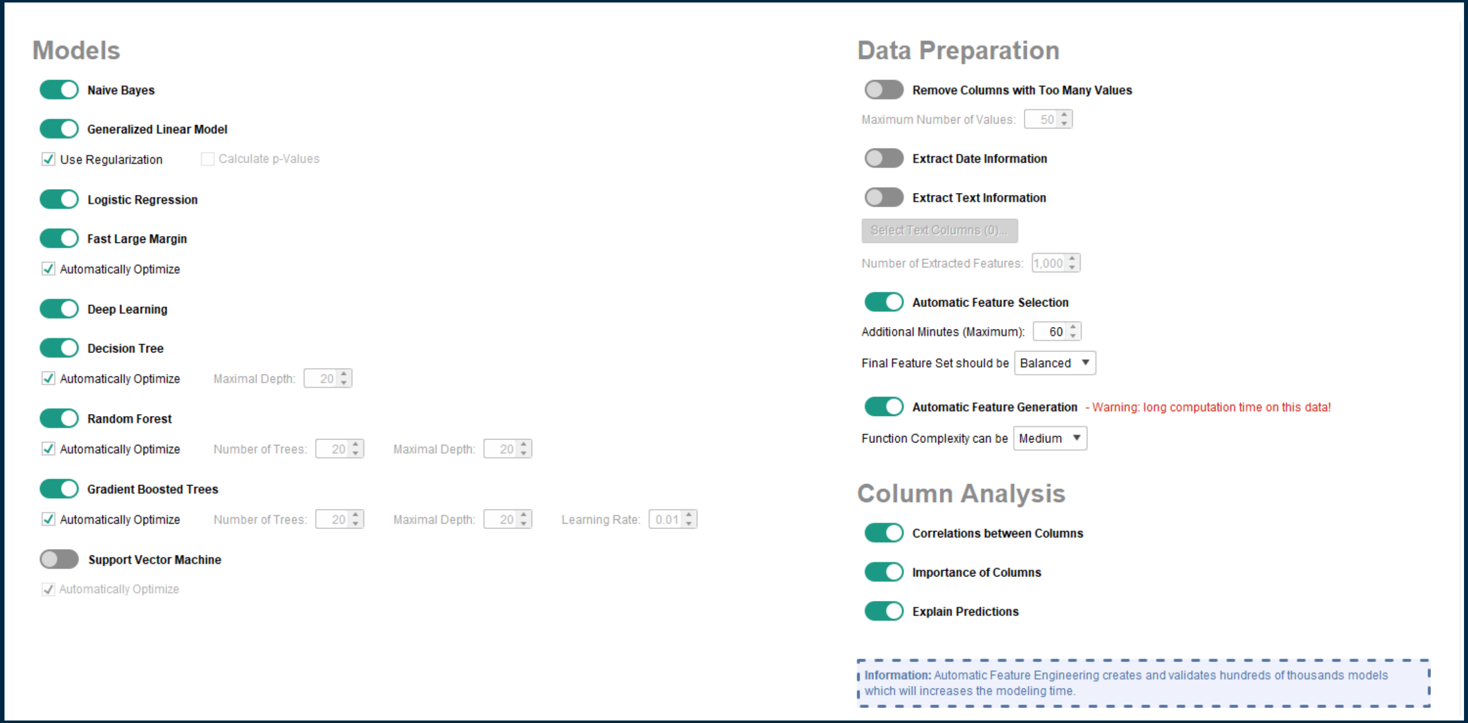
Data Exploration

Some interesting plots are presented below.



First approach – AutoModel

Using the AutoModel feature, we performed a run with the parameters set as below:

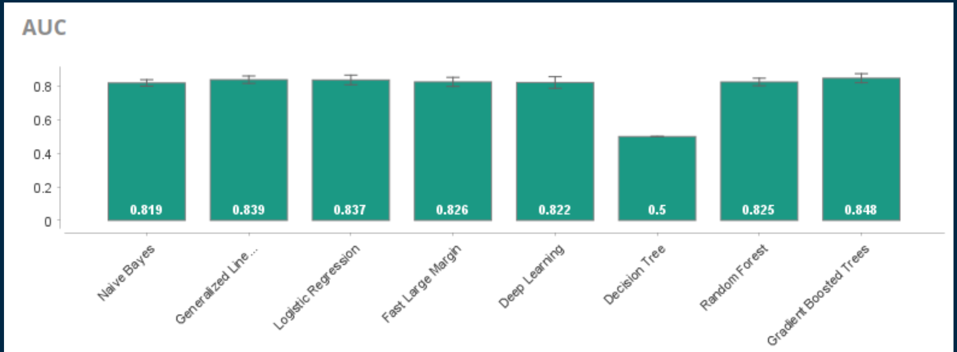


First approach – AutoModel

The Results Tab gives the option of generating various visualizations, to get insight of the data.

- We can see that the Gradient Boosted Trees, Generalized Linear Model and Logistic Regression Model performed well

- Detailed results are included in the submission file



Model	AUC	Standard Deviation	Accuracy
Naive Bayes	0.819	± 0.019	79.5%
Generalized Linear Model	0.839	± 0.022	80.7%
Logistic Regression	0.837	± 0.029	78.2%
Fast Large Margin	0.826	± 0.027	79.9%
Deep Learning	0.822	± 0.035	80.4%
Decision Tree	0.5	± 0	73.5%
Random Forest	0.825	± 0.022	79.2%
Gradient Boosted Trees	0.848	± 0.027	81.2%

First approach – AutoModel

The Auto Model process did not take into consideration the unbalanced data which, generally, resulted in low recall scores

- We build a process in order to validate and/or improve the results provided by the Auto Model Pipeline

- Auto Model did not generate any new features, during feature engineering

Criterion	Model		
	Gradient Boosted Trees	**Generalized Linear Model**	**Logistic Regression**
Accuracy	**81.21%**	80.72%	78.23%
Classification Error	**18.79%**	19.28%	21.77%
Auc	**84.77%**	83.89%	83.74%
Precision	69.14%	70.48%	**80.52%**
Recall	**49.30%**	44.75%	22.65%
F Measure	**57.49%**	54.69%	35.29%
Sensitivity	**49.30%**	44.75%	22.65%
Specificity	92.30%	93.35%	**98.11%**

Model Training & Optimization

We tried various pre-processing steps and executed runs to see if the change would improve the results. An example of such a run is shown below

- As shown in the Two tables, by converting the nominal values to Numerical (One Hot Encoding) the results were better

Description	Nominal (Base model)		
Accuracy	**76.97%**		
	true No	true Yes	class precision
pred. No	4042	490	**89.19%**
pred. Yes	1132	1379	**54.92%**
class recall	**78.12%**	**73.78%**	

Description	Numerical (Base model)		
Accuracy	**77.51%**		
	true No	true Yes	class precision
pred. No	4115	525	**88.69%**
pred. Yes	1059	1344	**55.93%**
class recall	**79.53%**	**71.91%**	

Model Training & Optimization

Various additional steps were tested, which are mentioned below. By using breakpoints and tables, likes the one presented in the previous slide, we build the final model.

- Feature Engineering:
    - All metrics were worse when we used the automatic feature engineering
    - Results were better by manually generating features
- Correlated values: All metrics were worse when we removed the TotalCharges variable (highly correlated with Tenure)
- Upsampling: Better results were achieved using upsampling of the *"yes"* label.
- Normalization: Better results were achieved using normalization (0-1 norm and SeniorCitizen column not included)

Model Training & Optimization

RapidMiner provides two types of parameter search, the GridSearch and Evolutionary rearch

- Grid Search: By using the GridSearch operator we were able to further improve the results
- Evolutionary Search: the results were worse compared to Grid Search

RESULTS

03

Auto Model – Gradient Boosted Trees

Accuracy: 81.2%

	true No	true Yes	class precision
pred. No	1377	263	83.96%
pred. Yes	115	257	69.09%
class recall	92.29%	49.42%	

Model Parameters:

- maximal_depth = 2
- Learning_rate = 0.1
- Number_of_trees = 90

Best Model – Gradient Boosted Trees

Accuracy: 84.24%*

accuracy: 84.24% +/- 1.15% (micro average: 84.24%)

	true No	true Yes	class precision
pred. No	4348	805	84.38%
pred. Yes	826	4369	84.10%
class recall	84.04%	84.44%	

Model Parameters:

- maximal_depth = 15
- Learning_rate = 0.034
- Number_of_trees = 510

*cross validation score with upsampling

Best Model – Gradient Boosted Trees

Accuracy: 77.35%*

accuracy: 77.36%

	true No	true Yes	class precision
pred. No	830	114	87.92%
pred. Yes	205	260	55.91%
class recall	80.19%	69.52%	

Model Parameters:

- maximal_depth = 15
- Learning_rate = 0.034
- Number_of_trees = 510

*20% test set, no upsampling

Best Model – Gradient Boosted Trees

Generated features:

Attribute Name	Function Expression
Tenure*charges	tenure*TotalCharges
TotalCharges/MonthlyCharges	TotalCharges/if(MonthlyCharges > 0, MonthlyCharges, 1)
TotalCharges/tenure	TotalCharges/if(tenure > 0, tenure, 1)

Best Model – Gradient Boosted Trees

Feature importance:

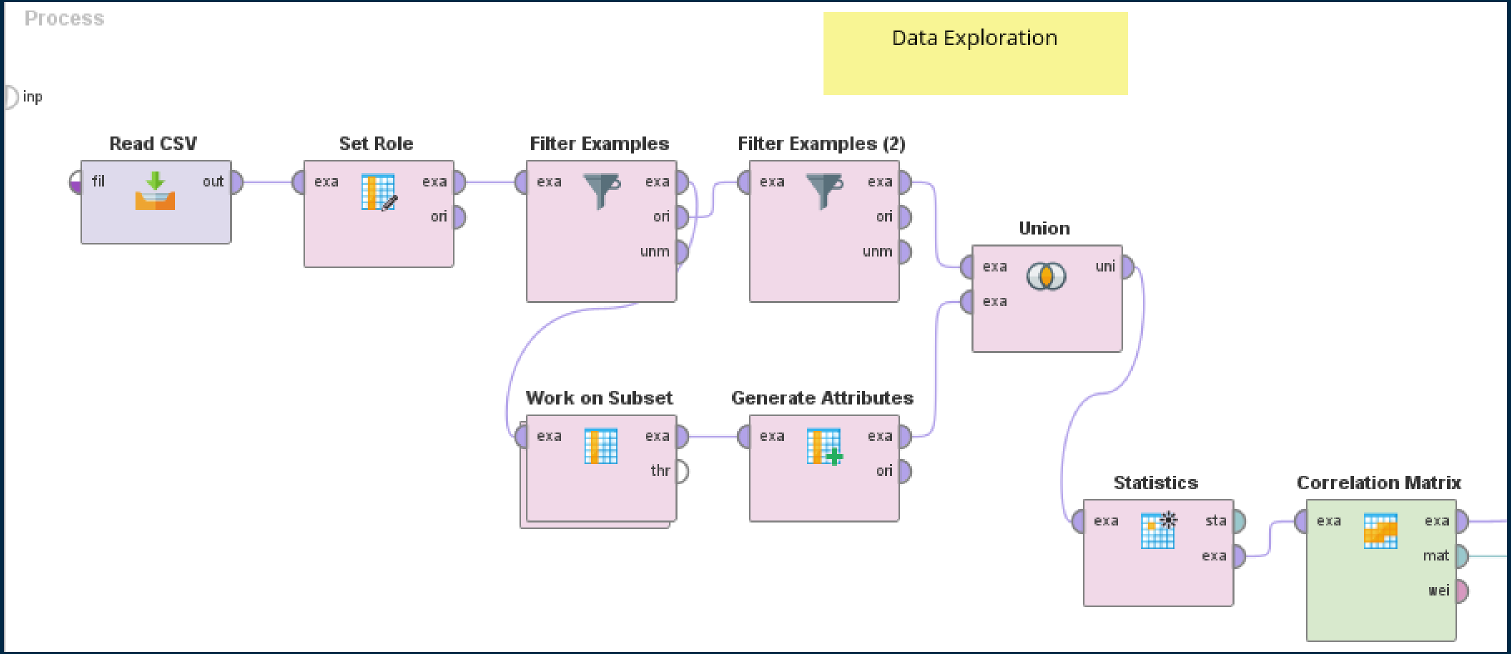
Attribute	weight
Contract = Month-to-month	0.193
Contract = Two year	0.129
OnlineSecurity = No	0.124
TechSupport = No	0.119
TotalCharges/MonthlyCharges	0.107
tenure	0.106
InternetService = Fiber optic	0.091
PaymentMethod = Electronic check	0.087
tenure*charges	0.075

Appendix

04

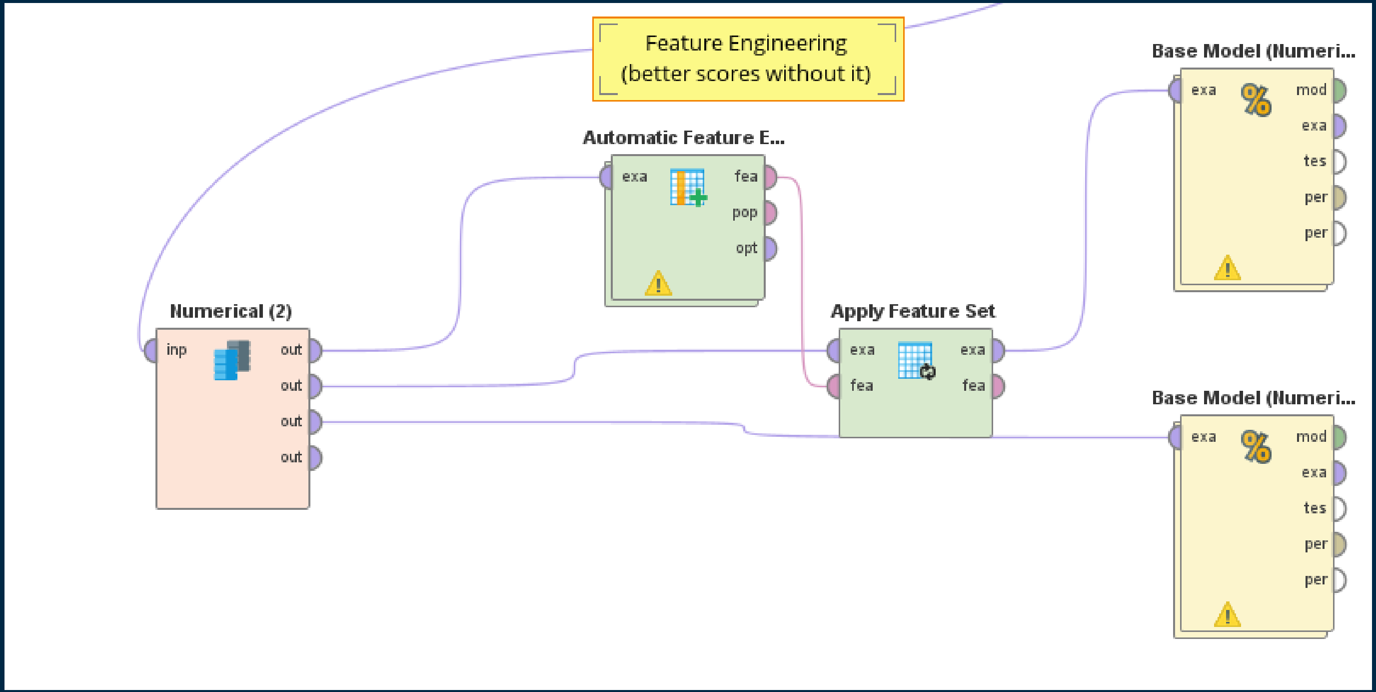
Data Exploration

The following setup was used in RapidMiner for the Data Exploration part.



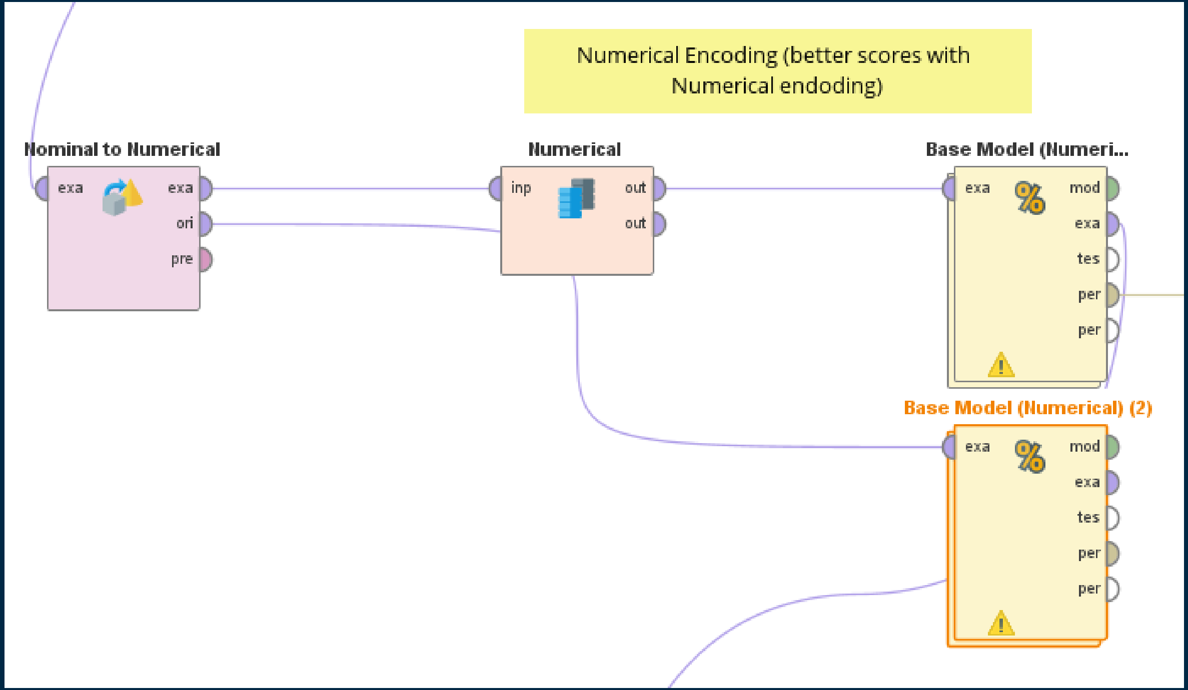
Model Training & Optimization

The following setup was used in RapidMiner for the Feature Engineering part.
The results were better with the original data.



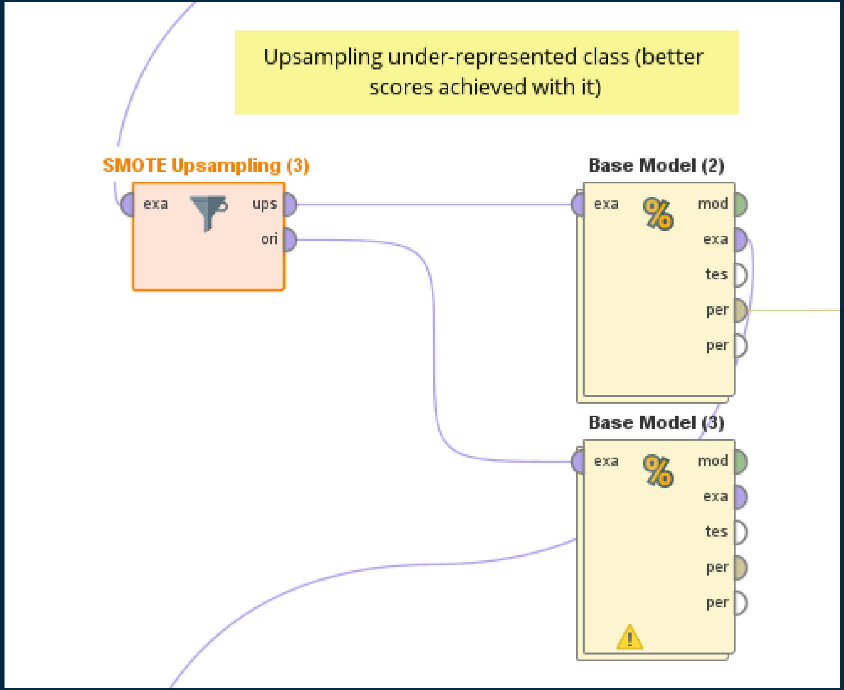
Model Training & Optimization

The following setup was used in RapidMiner to compare the Nominal vs Numerical dataset. The One-Hot Encoded Dataset was better.



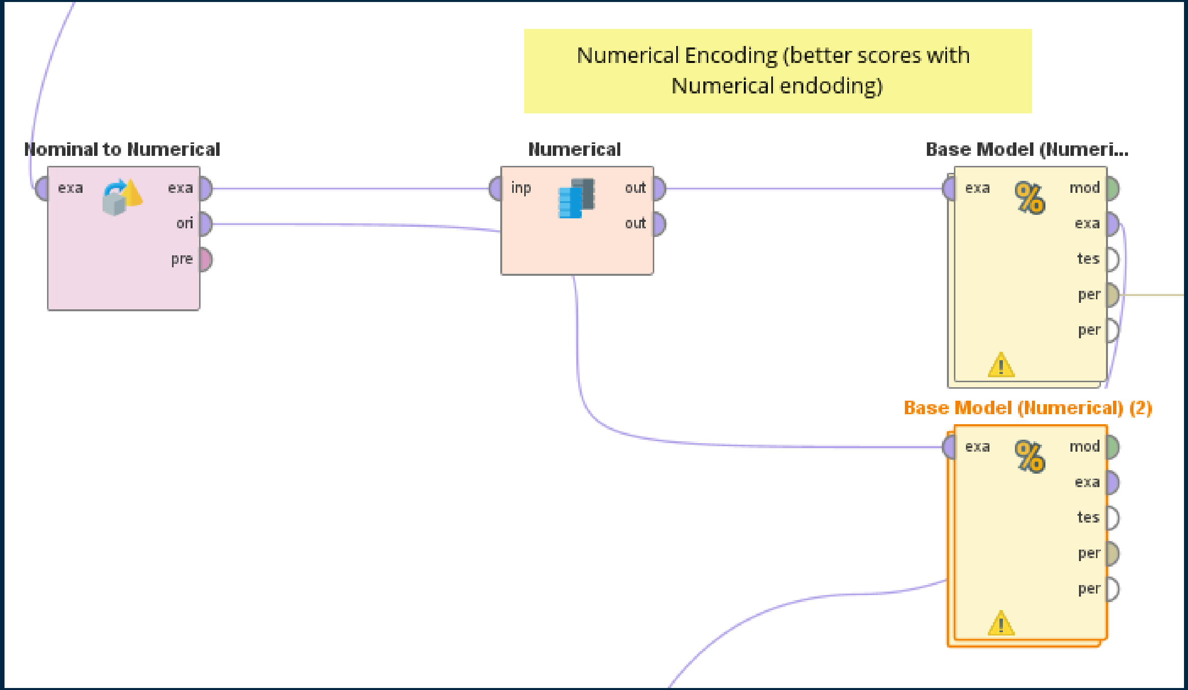
Model Training & Optimization

The following setup was used in RapidMiner to evaluate the up-sampling performance dataset. Upsampling resulted in higher scores.



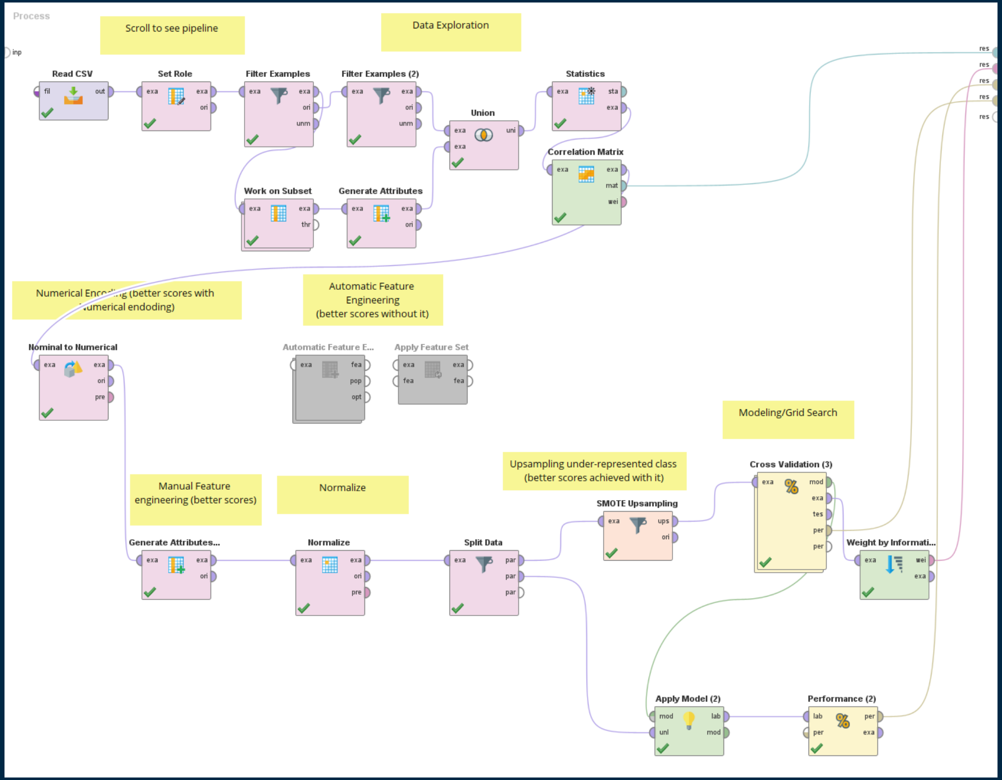
Model Training & Optimization

The following setup was used in RapidMiner to compare the Nominal vs Numerical dataset. The One-Hot Encoded Dataset was better.



Model Training & Optimization

Final Pipeline



THANK YOU

Questions?

Contact: gchalkiopoulos@aueb.gr