University of Antwerp

AXA Data Science Challenge

AXA & University of Antwerp

AXA Data Challenge 2023-2024
UA_2103TEWDAS_2324
Machine learning
Prof. Dr. David Martens

Team members:
Hans Gevers
Juline De Molenaer
Milán Péter

Table of Contents

1. Business understanding

AXA Belgium, part of the AXA Group headquartered in Paris, is, among other business units, a leading insurance company in Belgium. As part of a strategic initiative, AXA wants to maintain its position as a market leader in the field of car insurance. For its car insurance business, AXA mainly works through exclusive brokers (like Jean Verheyen) or other banks (like ING), which limits AXA's direct interaction with customers. The initiative aims to increase the conversion rate of car insurance offers to retail customers. By leveraging their data, AXA envisions accomplishing this conversion rate increase by gaining insights into its current offerings process, by better understanding the customer profiles that are more likely to accept an offer and understanding which profiles are missing out, by spotting as soon as possible which offers are more susceptible to being converted, and by improving the offers to better meet the expectations of the customer. In this project, we are tasked with identifying the characteristics of offers that are most likely to be converted into contracts. We will do so by analysing a dataset containing car insurance offers from 2023. The variables included in this dataset refer to customer, car, and broker characteristics. Through predictive modelling, we will develop a model that learns patterns in the car insurance offering data, allowing the model to predict the probability of a car insurance offer converting into a contract. By building a reliable and performing model, AXA can gain insights into the characteristics that influence the probability of conversion. By deploying the model, AXA could not only accurately predict the probability of a future car insurance offer being converted into a contract, but AXA could also use these insights to tweak some of the adjustable characteristics of the offer and to provide advice and guidance for brokers to arrange the best contractual details for potential customers. In short, leveraging the model will prove to be useful for AXA in multiple ways: increasing the conversion rate, better customer insights, reinforcing AXA's current leading position, and improving AXA's car insurance offering.

2. Data understanding

The purpose of data understanding is to gain an understanding of the data at hand. It involves gaining relevant insights into the structure of the dataset and individual variables and identifying potential data quality issues. The dataset received consists of 44.928 observations and 22 variables. We divided the dataset into nominal, ordinal, discrete, continuous, and binary variables, making it easier to make visualization. When checking the data types, we discovered that some data types were not correct as stated by the documentation provided. An overview of the different visualizations (pie charts, bar charts, and box plots) and calculations (number of unique values, percentage missing values, and descriptive statistics) per data type can be found in Appendix 2. For some variables with a significant number of unique values, no visuals were created as no clear insights could be derived from the visuals. From the visualizations and the calculations, we found that for some variables, a significant amount (between 40% and 80%) of observations was missing (*nbYearsAttest, profession, license_year_5y*, and *catalogValue*). In terms of outliers, *catalogValue* and *powerKW* were found to contain many (valid) outliers. Two invalid outliers were found: *birthday_5y* ('1695') and *vhConstructionYear* ('1014'). Correlation matrices were calculated and there seems to be a significant correlation between *license_year_5y* and *birthday_5y*, and *catalogValue* and *powerKW*. These correlations seem reasonable. The older someone is, the longer they have their driver's license and powerful cars are often more expensive. No duplicates were found. Upon further inspection, we saw that the missing values could be due to an option 'other' in the questionnaire, or a missing category (e.g. the fuel types 1, 2, 3, 4, 6, 7, and 8 were mentioned, so it seems reasonable to conclude that missing values should be fuel type 5). Data understanding was only performed on the training data, to avoid insights obtained from the test data influencing our thought process in the data preparation step.

3. Data Preparation

The third step in the CRISP-DM framework is data preparation. In essence, the purpose of data preparation is to prepare or transform the data in a form that is suitable for modelling. In our project, several steps were taken. First of all, irrelevant variables and variables that were subject to data leakage, were deleted. Next, the dataset was split into training, validation, and test. Subsequently, the missing values are handled. In the fourth step, data types that did not match the description in the guidelines were converted. After that, outliers were handled. Afterwards, variable modifications were made as needed. Lastly, the necessary variables were encoded to make them suitable for analysis or standardized, if needed. Note that for data preparation and subsequent steps, different preparation and modelling approaches were conducted, and performance of models was compared (see Appendix 7). Only the approach of the best performing code (code 1) is reported and discussed. The differences between the data preparation in the three different coding approaches can be found in Appendix 9.

Removing Variables

Some variables were removed, for several reasons. Due to data leakage, *availableActions* was removed. The variable *isSecondHandVehicle* was removed as it was no longer measured in the test data. Hence, we specified only 22 variables in the data understanding, instead of the original 24. The variable model was removed since there was a high association between make and model, as formally proven by a Chi2 test.

Data Split

Next, the dataset was split into 60% training data, 20% validation data, and 20% test data. The choice to split before preprocessing was a deliberate choice. Preprocessing should be done on the three datasets separately, to prevent data spillover from the training set to the validation set or the test set. For this reason, if specific values were replaced by another specified value in further preprocessing, the replacing value was always calculated based on the training set. Especially for handling missing values, this is an important consideration. The training dataset will be used to train the machine-learning models. The validation dataset was used for hyperparameter tuning through a grid search combined with 5-fold cross-validation, and the test dataset was used to evaluate the performance of the different models.

Missing Values

Since some variables contained a lot of missing values, careful handling was needed. Appendix 3 contains an overview of the missing values replacements that we performed. Broadly speaking, missing values were either replaced by a statistic calculated for the training set (mode for discrete variables and median for continuous variables) or by a value that made sense based on our interpretation from the data understanding.

Changing Data Types

As some data types of some variables did not match the data type from the guidelines, the data types were converted to the appropriate type. More specifically, discrete variables were converted to integers, nominal variables were converted to strings, continuous variables were converted to floats, and binary variables were converted to Booleans. Furthermore, *postal_code_XX* was converted to a nominal variable, as backed by literature (I. POPULATIONS, VARIABLES, and DATA, n.d.).

Outliers

The invalid outliers (*birthday_5y* = 1695 and *vhConstructionYear* = 1014) were replaced by the mode. The valid outliers were concentrated in two variables: *catalogValue* and *powerKW*. There were a lot of outliers, the most extreme outliers being 930 for *powerKW* and 10.190.000 for *catalogValue*. Although these are not very likely, cars with such specifications do exist. The variables were standardized first and subsequently, the outliers were clipped to the range -3 to 3.

Variable Modification

The variable *Year_Month* was split into *Year* and *Month*. As all observations are from 2023, the year does not add any explanatory value, hence the *Year* was removed and only the *Month* was kept. For the sake of dimensionality, the variable make was reduced to the top 10 most occurring car brands and an 11th category 'Other'. This ensures that in the encoding step, only 11 variables are related to make, rather than 266 if this transformation was not applied. This approach proved to be beneficial for the performance of the models. For *postal_code_XX*, the values '33-34' and '64-65' were replaced by 33 and 64 respectively, in order to reach uniformity among the *postal_code_XX* values.

Encoding

In this last step in the data preparation phase, the necessary variables were encoded. The nominal variables (*profession*, *fuelType*, make, and *postal_code_XX*) were encoded using dummy encoding. The ordinal variables were encoded using dummy encoding as well. Thermometer encoding was performed initially, however, the performance of dummy encoding for ordinal variables led to a higher AUC compared to thermometer encoding.

4. Modelling and Evaluation

Modelling

The next step in the CRISP-DM framework is modelling. In the modelling step, appropriate techniques are selected. The techniques are applied to the preprocessed training dataset to identify patterns and relationships in base models. Subsequently, the hyperparameters of the models are optimized using the preprocessed validation dataset and, ultimately the predictions are made on the test dataset. Several classification models were built, such as k-Nearest Neighbor, Decision Tree, Logistic Regression, Random Forest, Linear Support Vector Machine, and Non-Linear Support Vector Machine. All models used are classifiers for supervised learning and by theory usable for the given classification task. Short descriptions of the models are provided in Appendix 4.

In essence, building the baseline models involves three different steps. First, the model is initialized. For the base models, this implies just selecting the algorithm with standard hyperparameter values. In the next step, the model is fitted to the training data. Finally, in the third step, the probabilities are predicted on the test set. With these predictions, the performance of the model can be assessed. This assessment is thoroughly described in the evaluation.

In the first modelling iteration, the base versions of the models were trained. This implies that no values for hyperparameters were specified, but rather the standard values were used. For the hyperparameter tuning, an extra step is added before the three steps described above. A hyperparameter grid is defined, which includes all potential values we want to check for the selected

hyperparameters. An overview of the optimized hyperparameters for preferred models can be found in Appendix 5. Through a grid search, combined with cross-validation, any of the possible combinations of specified hyperparameter values is fed to the model. This implies that the model goes through the three steps described above for any of the combinations. The performance of all combinations is saved, and, in the end, the best-performing model is selected, and the best-performing hyperparameter values are shown, together with the specified performance metrics. Note that hyperparameter tuning is not a one-shot solution. Multiple iterations might be needed if one of the hyperparameter values of the best-performing model happens to be the minimum or maximum of the specified range. This implies that it might be desired to redefine the hyperparameter grid with adjusted (i.e. higher or lower) values for the specific hyperparameter. Also, if the optimal hyperparameter value is not one of the extremes, it might be insightful to try out some other values that are near the optimal hyperparameter value and see if the optimal value changes, together with the performance of the model.

Besides that, the modelling phase is a highly iterative process, not only in terms of hyperparameter values but also in terms of variables included. Through consideration, we dropped some variables that we suspected not to be very performing, to see if it boosts performance. One might argue that just leaving all variables in the model might be a good approach to capture as much as possible, however, more variables (that potentially do not add value to the model) might also imply more noise and complexity, hence making it more difficult for the algorithm to find patterns in the data. We approached the data preparation and modelling phase separately. Each of us tried out different data preparation and modelling steps (hyperparameter grids and exclusion of different variables) to see which approach yielded the best performance. Only the best-performing approach was specified in the report. However, for the other approaches, please refer to Appendix 7 and 9 and the *.py* file provided.

Evaluation

Evaluation is the next step in the CRISP-DM framework. In the evaluation step, the performance of the model is assessed. Obviously, the performance of a model can be assessed in many ways. For this project, we opted for AUC, accuracy, confusion matrix, cost, and lift as performance metrics. In Appendix 6, an overview of the best-performing models before and after hyperparameter tuning, together with their optimal hyperparameter values, can be found.

From the performance comparison tables, it is clear that the best-performing model is a Random Forest Classifier with the following values for hyperparameters: criterion = entropy, max_depth = 50, max_features = sqrt, min_samples_leaf = 2, min_samples_split = 20, n_estimators = 1000.

The choice for the model described above is based on a thorough analysis involving the AUC, accuracy, cost, and lift curve. The following values for the performance metrics were found:

- AUC = 0,7444
- Accuracy = 0,7013
- Confusion Matrix = [[5164 750]; [1934 1138]]
- Cost = 5684
- Lift (10%) = 1,2602

In terms of AUC and accuracy, the Random Forest Classifier proved to be the best model. In terms of lift, the Decision Tree Classifier had a slightly higher lift. In terms of costs, the performance of the

Random Forest Classifier is average. The AUC score indicates that the model is able to distinguish between positive and negative classes reasonably well, compared to a random model (AUC = 0,5). In terms of accuracy, the model is able to make the correct prediction in 70,13% of the cases. As the classes in the dataset are quite well-balanced, accuracy can be seen as a reliable metric for this specific case. In terms of cost, the costs indicate the total cost or penalty for misclassifications. We argued that false negatives are 5 times as costly as false positives, people that would actually convert but that are predicted to be non-converters are missed opportunities. Depending on the value of a car insurance, this might be a significant amount of money. In terms of lift, a lift score (10%) of 1,2602 indicates that our model is 26% better than random guessing at identifying positive instances when targeting 10% of the instances of the test set.

5. Business Insights and Recommendations

Finally, in the business insights and recommendations stage, we explain the implications that a modelling approach as described above can have when deployed in a real-life situation. First of all, it's wise to consider the effects of the model in terms of explainable AI. Next, we discuss what AXA can eventually achieve by deploying a model for car insurance conversion, together with clear recommendations for AXA to fully leverage the potential of the provided model.

Explainable AI

Explainable AI (XAI) can be defined as the ability of AI systems to explain their behaviour by clarifying the underlying mechanisms of their decision-making processes in such a way that it becomes comprehensible for humans (Bernardo, 2023). Of course, the implications of Explainable AI are case-specific. For some business cases, explainability is considered more important than in other business cases. For example, when deploying a machine learning model for credit scoring or deciding if people should receive a loan or not. When someone is predicted as not applicable for a loan, the owner of the AI model needs to be able to explain why that person is predicted as not applicable. "Because the model said so" is not a valid explanation. However, in cases like predicting conversion for car insurances, the need for explainable AI for external communication is limited but should not be neglected. For internal use, insights into the decision rules of the model can be useful. A model like a Random Forest Classifier is more of a black box compared to a decision tree. However, due to the fact that decisions made by the model do not directly affect individuals in a harmful way, the Random Forest Classifier is a reasonable choice as a model. It is important to consider that in terms of insights into variables that influence the conversion probabilities, Random Forest is more complex to interpret compared to simpler models like Decision Tree, this is an important consideration to make.

Recommendations for AXA

In terms of recommendations for AXA, several points can be made. First of all, with regard to the performance of the model, AXA should investigate if an accuracy of 70% is considered acceptable. This investigation depends on different factors. First, the intended use of the model has to be specified. Is the model for internal guidance only, or will it be used to make (investment) decisions in real life? In the case of the latter, AXA might want to aim for a higher accuracy. Next, the investigation also depends on the cost of misclassification. In the report, we argued a cost of five for false negatives and a cost of one for false positives. This translates to a second recommendation for AXA: quantify the actual cost matrix. The outcome of this exercise might influence the need for a higher accuracy. Suppose that the cost of misclassification is more than the values that we specified, AXA might want to aim for a higher accuracy.

Given the preliminary correlation matrices constructed, no (very) strong correlations were found, which might indicate that the explanatory variables do not have the desired explanatory power required. AXA might want to consider collecting additional data to have more variables that could add value to the model. Also, in order to limit the data preparation work for future projects, special attention should be devoted to data quality. The dataset provided was quite dirty. AXA should investigate data quality optimization measures to realize quick wins in terms of improving data quality. Better data quality will probably lead to better model performance.

If AXA decides that the model performance is sufficient, either with or without taking the recommendations into account, AXA could use this model to optimize its conversion strategies. In additional data analysis, useful insights can be derived into the effects of specific variables on the conversion probabilities. Business rules can be defined based on this additional analysis. For example, if birthday tends to have an impact on conversion probabilities (e.g. the older, the more likely to convert), AXA might want to use this information to pay special attention to older people or invest more resources in trying to convince older people to convert, as they would be easier to convince, compared to younger people. Similarly, additional business rules can be defined. For future work, we would consider building target-specific models which each orient towards specific customer profiles or personas in order to optimize data-usage and training time.

From ethical point of view, we explicitly condemn usage of our model for discriminating customers of any kind based on unethical grounds. The model has been developed to primarily predict the conversion potential of AXA's customers.

6. Conclusion

In conclusion, the project focused on enhancing AXA Belgium's position as a market leader in car insurance by increasing the conversion rate of offers to retail customers. The comprehensive analysis involved understanding the business context, exploring and preparing the dataset, and developing predictive models. The data understanding phase revealed key insights into the dataset's structure and identified potential data quality issues. Notable observations included missing values, outliers, and variable associations. In response, the data preparation phase addressed these issues through careful handling of missing values, data type corrections, outlier management, and variable modifications.

The modelling and evaluation stages employed various classification models, including k-Nearest Neighbour, Decision Tree, Logistic Regression, Random Forest, Linear Support Vector Machine, and Non-Linear Support Vector Machine. Through an iterative process, the Random Forest Classifier emerged as the best-performing model based on metrics such as AUC, accuracy, cost, and lift. The selected model demonstrated an *AUC of 0.7444*, an *accuracy of 70.13%,* and a *lift (10%) of 1.2602*. Moreover, it has a relatively low cost of possible wrong predictions.

Overall, the project succeeded in developing a robust predictive model that can effectively identify the characteristics of car insurance offers most likely to be converted into contracts. Implementation of the model is anticipated to contribute to AXA Belgium's goals of increasing the conversion rate, gaining better customer insights, reinforcing its market leadership, and improving the overall car insurance offering.

Additional background information is available in the appendices that were not referred so far.

7. References

Bernardo, V. (2023, November 16). *TechDispatch #2/2023 - Explainable Artificial Intelligence*. European Data Protection Supervisor. Retrieved December 9, 2023, from https://edps.europa.eu/data-protection/our-work/publications/techdispatch/2023-11-16-techdispatch-22023-explainable-artificial-intelligence_en

Nerad, J.R. (2020, December 29). Buying A Car At The End Of The Year Can Save You Thousands…If You Do It Right. Forbes. Retrieved December 7, 2023 from https://www.forbes.com/sites/jacknerad2/2021/12/29/buying-a-car-at-the-end-of-the-year-can-save-you-thousandsif-you-do-it-right/

ScikitLearn. (2019, June 24). *Decision Trees*. ScikitLearn. Retrieved December 9, 2023, from https://scikit-learn.org/stable/modules/tree.html?fbclid=IwAR1ICvVBhvY7ZGUU-vaqx8PLUJy5_4yFCUbk2Mdooe-P6locdNDaKtLDmLE

Scikit-Learn. (2019, June 24). *Support Vector Machines*. Scikit-Learn. Retrieved December 9, 2023, from https://scikit-learn.org/stable/modules/svm.html?fbclid=IwAR20hSiPlK1IX4I5SR0VUeR8cvJFEyivTXSDwpDXWZoslaRa7TPyaYY34dk#classification

Scikit-Learn. (2023, March 2). *Ensembles: Gradient boosting, random forests, bagging, voting, stacking*. Scikit-Learn. Retrieved December 9, 2023, from https://scikit-learn.org/stable/modules/ensemble.html?fbclid=IwAR3Xp2whmeNGCWke672yn2_n4NY_s_u0xDwYZIRAY3beCm8B3BwdXn0joHA#random-forests-and-other-randomized-tree-ensembles

Scikit-Learn. (2023, March 2). *Logistic regression*. Scikit-Learn. Retrieved December 9, 2023, from https://scikit-learn.org/stable/modules/linear_model.html?fbclid=IwAR1Nk1bezACyQSUcroydeVseRbc-T9w97KKKIW6JG87tblgukW16RMQMZ08#logistic-regression

University of Houston. (n.d.). *I. POPULATIONS, VARIABLES, and DATA*. Retrieved December 9, 2023, from https://online.math.uh.edu/MiddleSchool/Modules/Module_5_Prob_Stat/Content/Stat/Ch4_1.pdf

8. Appendices

Appendix 1: ChatGTP

Throughout the project, in some specific cases, guidance from ChatGPT was used. Note that we handled the output of ChatGPT with regards to interpretations with care. For code debugging purposes, ChatGPT was consulted to get insights into code errors and potential corrections. As we experienced some problems with the provided functions (e.g. lift and cost), we asked ChatGPT for another possible block to reach the same purpose. For aesthetic reasons, some visualizations were made, based on code of ChatGPT.

Moreover, the questions below were asked as well.

- **Can you correct values in a test set for classification modelling?**
  "Yes, you can correct values in a test set for classification modeling. However, it is important to note that the test set should not be modified in any way during the modeling process. …"
  This answer confirmed our prior knowledge.
  Asked at Copilot (Microsoft Explorer AI, consulted on 26/10/2023)

- **Which hyperparameter range would you use for following classifier ?**
  Ct = trial.suggest_int("C", 1, 20, log=True) maxi = trial.suggest_int("max_iter", 1, 5000, log=True) tolt = trial.suggest_float("tol", 1, 5, log=True) solvert = trial.suggest_categorical("solver", ["lbfgs", "newton-cg", "sag", "saga"]) classifier=LogisticRegression(C=Ct, tol=tolt, penalty='l2', dual=False, fit_intercept=True, class_weight='balanced', solver=solvert, max_iter=maxi, multi_class='multinomial', verbose=0)
  For the given logistic regression classifier, here are the suggested hyperparameter ranges
  C (Regularization Parameter): Suggested range is from 1 to 20 with a logarithmic scale.
  max_iter (Maximum Number of Iterations): Suggested range is from 1 to 5000 with a logarithmic scale.
  tol (Tolerance for Stopping Criteria): Suggested range is from 1e-5 to 5 with a logarithmic scale)
  solver (Optimization Algorithm): Suggested options are ["lbfgs", "newton-cg", "sag", "saga"].
  So, you can use these ranges when setting up your Optuna study for hyperparameter optimization.
  This ensures that you explore a diverse range of hyperparameters during the optimization process.
  Adjust the ranges based on your specific use case and requirements.
  This answer is rather limited, as no explicit indication of common used ranges for the specific parameters was made. The use of the optuna package was abandoned given specific integration problems.
  Asked at chatGTP (chat.openai.com, consulted on 30/11/2023)

- **Can postal code be considered as an ordinal variable?**
  No, postal codes are typically considered categorical variables, not ordinal variables.... They do not have a inherent order or ranking. ...
  Nevertheless code 2 integrate it as such.
  Asked at chatGTP (chat.openai.com, consulted on 07/12/2023)

- **Can value added tax be considered as an ordinal variable?**
  No, value-added tax (VAT) is not considered an ordinal variable. ... The VAT rate represents a quantitative value, but it does not have a natural order or ranking. …
  Nevertheless code 1 integrates it as such, motivated by the intuition of a preference for the lowest possible tax level. The equidistant categories are interpreted as low, rather low, medium, rather high and high.
  Asked at chatGTP (chat.openai.com, consulted on 07/12/2023)

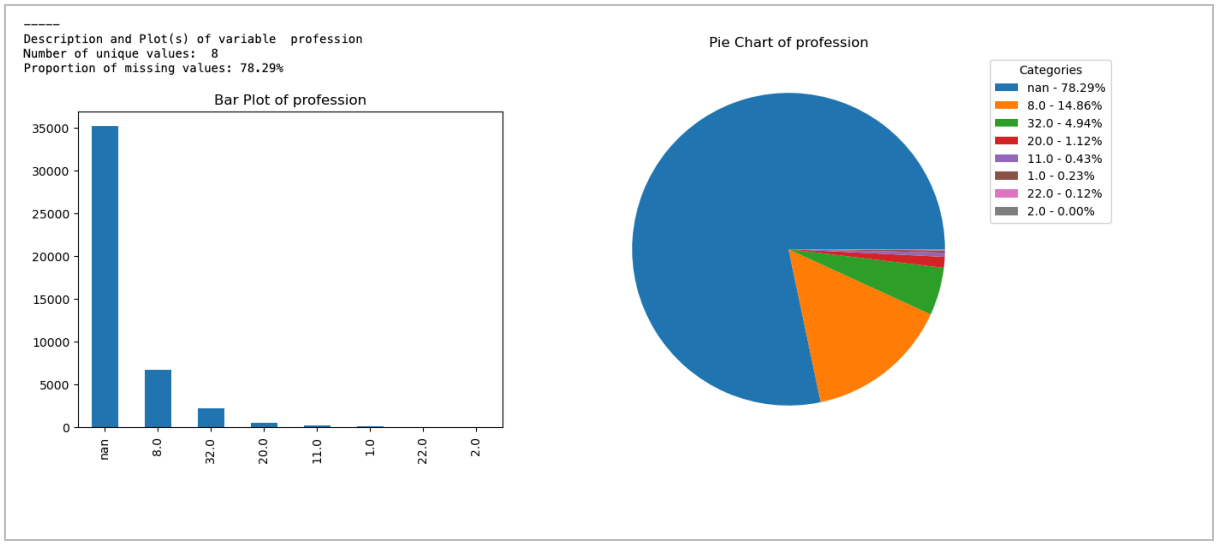
- **Can month be considered as an ordinal variable?**
  No, a month is typically considered a categorical variable rather than an ordinal variable. Categorical variables represent categories or groups with no inherent order or ranking. ...

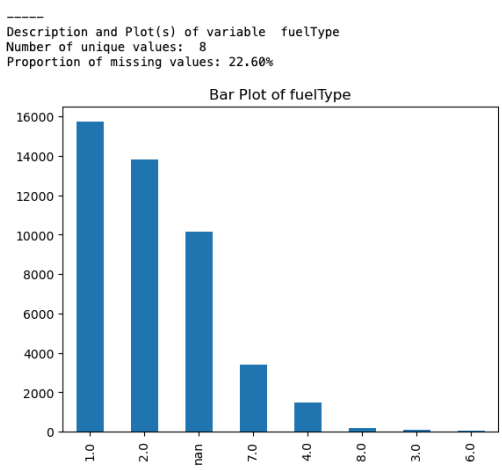
Nevertheless code 1 integrates it as such motivated by the intuition of a commercial ranking related to time, as such assuming, as Nerad from Forbes, that car prices lower towards the end of the year (Nerad, 2020, https://www.forbes.com/sites/jacknerad2/2021/12/29/buying-a-car-at-the-end-of-the-year-can-save-you-thousandsif-you-do-it-right/).
Asked at chatGTP (chat.openai.com, consulted on 07/12/2023)

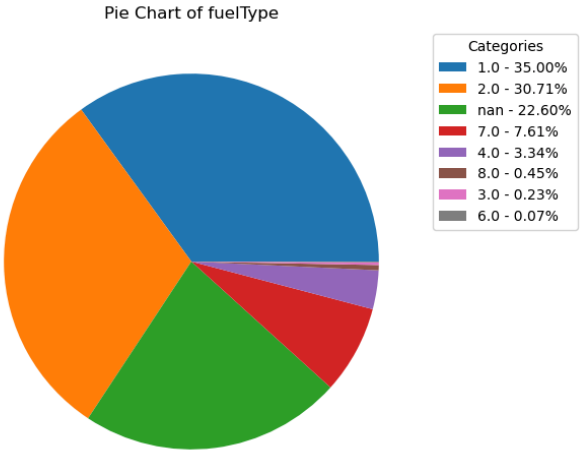
Appendix 2: Data understanding

	Nominal variables	Ordinal Variables	Discrete Variables	Continuous Variables
Pie Chart	X			
Bar Chart	X	X	X	X
Boxplot Scatterplot				X
N° Unique Values	X	X	X	X
% Missing Values	X	X	X	X
Descriptive Stats			X	x

Nominal Variables



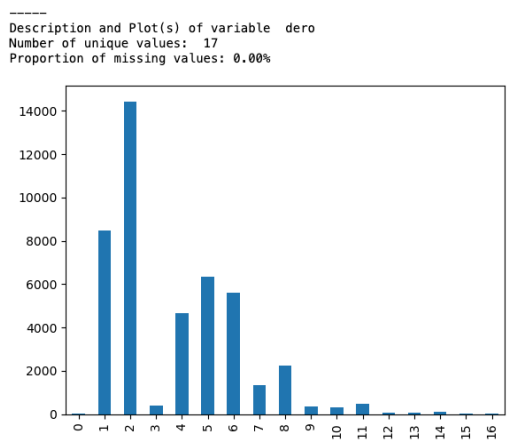


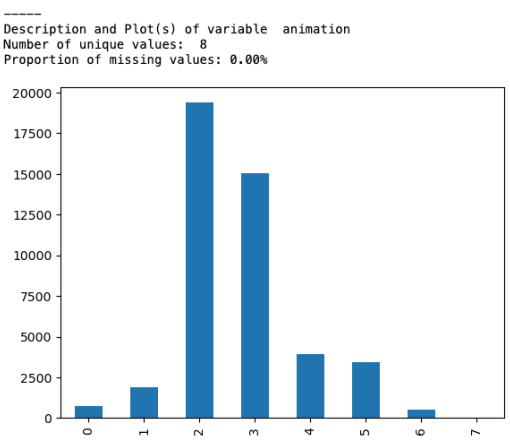


Description and Plot(s) of variable make
Number of unique values: 266
Proportion of missing values: 0.00%

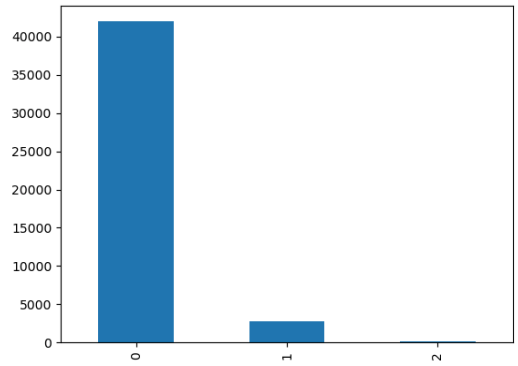
Description and Plot(s) of variable model
Number of unique values: 2072
Proportion of missing values: 0.00%

Ordinal Variables

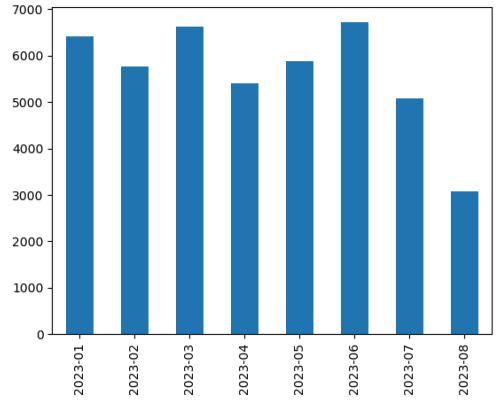




```
_____
Description and Plot(s) of variable  animation_2
Number of unique values:  3
Proportion of missing values: 0.00%
```




```
_____
Description and Plot(s) of variable  Year_Month
Number of unique values:  8
Proportion of missing values: 0.00%
```

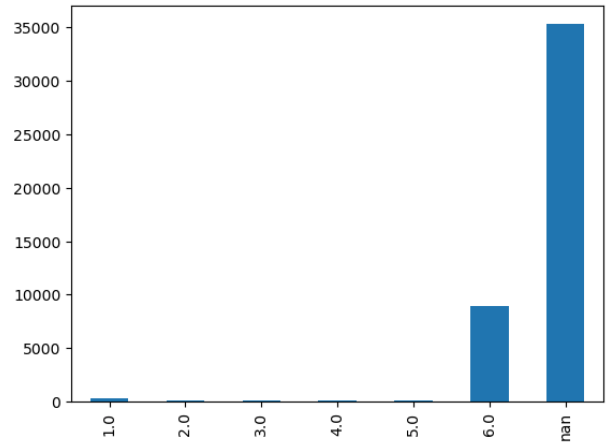



```
_____
Description and Plot(s) of variable  postal_code_XX
Number of unique values:  78
Proportion of missing values: 0.00%
```

```
_____
Description and Plot(s) of variable  customerScore
Number of unique values:  76
Proportion of missing values: 0.00%
```

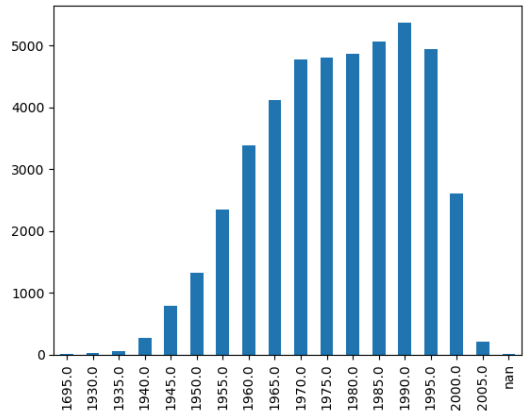
Discrete Variables

```
_____
Description and Plot(s) of variable  nbYearsAttest
Number of unique values:  7
Proportion of missing values: 78.59%
```



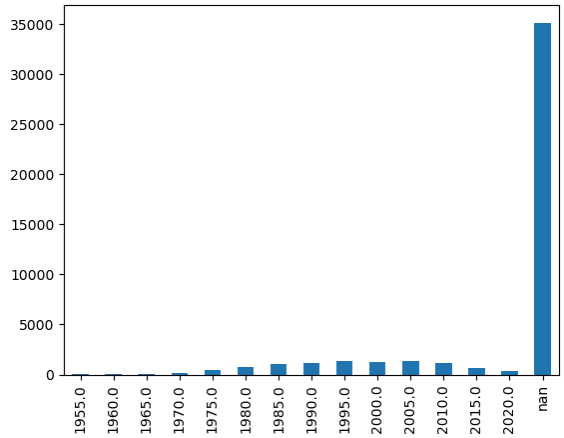

```
Statistics for nbYearsAttest:
Mean: 5.742750233863424
Median: 6.0
Mode: 0    6.0
Name: nbYearsAttest, dtype: float64
Standard Deviation: 1.0126182500086223
Minimum: 1.0
Maximum: 6.0
25th Percentile: 6.0
75th Percentile: 6.0
```

```
-----
Description and Plot(s) of variable  birthday_5y
Number of unique values:  18
Proportion of missing values: 0.01%
```



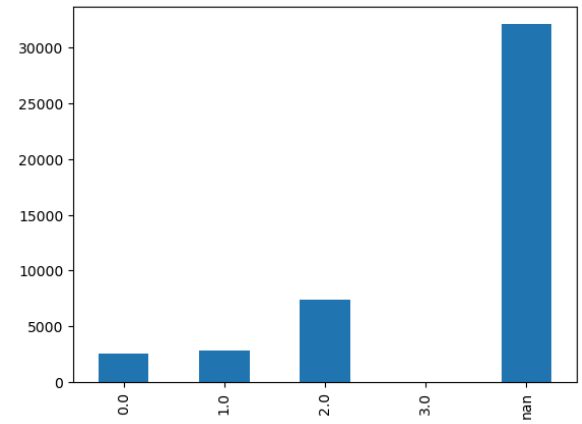

```
Statistics for birthday_5y:
Mean: 1977.092264885921
Median: 1980.0
Mode: 0    1990.0
Name: birthday_5y, dtype: float64
Standard Deviation: 14.718035591671722
Minimum: 1695.0
Maximum: 2005.0
25th Percentile: 1965.0
75th Percentile: 1990.0
```

```
-----
Description and Plot(s) of variable  license_year_5y
Number of unique values:  15
Proportion of missing values: 78.29%
```



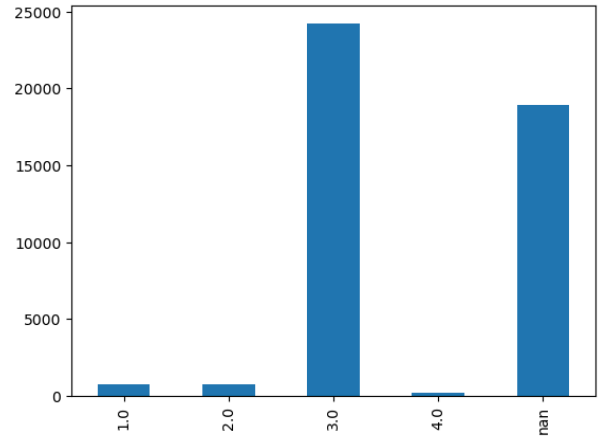

```
Statistics for license_year_5y:
Mean: 1996.9983594791347
Median: 1995.0
Mode: 0    2005.0
Name: license_year_5y, dtype: float64
Standard Deviation: 12.449660630333284
Minimum: 1955.0
Maximum: 2020.0
25th Percentile: 1990.0
75th Percentile: 2005.0
```

```
-----
Description and Plot(s) of variable  availableActions
Number of unique values:  5
Proportion of missing values: 71.41%
```



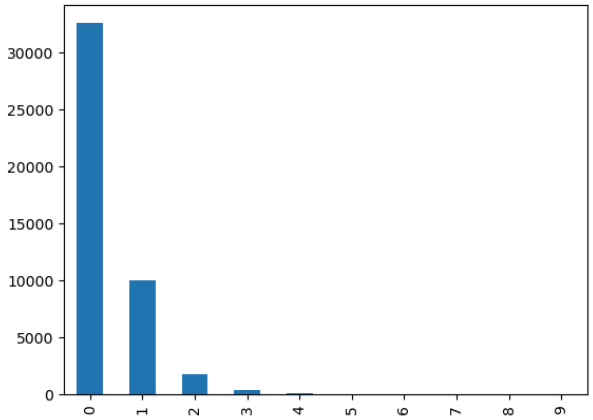

```
Statistics for availableActions:
Mean: 1.3771602055114434
Median: 2.0
Mode: 0    2.0
Name: availableActions, dtype: float64
Standard Deviation: 0.7983858181970417
Minimum: 0.0
Maximum: 3.0
25th Percentile: 1.0
75th Percentile: 2.0
```

```
-----
Description and Plot(s) of variable  purchaseTva
Number of unique values:  5
Proportion of missing values: 42.13%
```




```
Statistics for purchaseTva:
Mean: 2.9190353475133657
Median: 3.0
Mode: 0    3.0
Name: purchaseTva, dtype: float64
Standard Deviation: 0.3902106894053681
Minimum: 1.0
Maximum: 4.0
25th Percentile: 3.0
75th Percentile: 3.0
```

```
-----
Description and Plot(s) of variable  nbbackoffice
Number of unique values:  10
Proportion of missing values: 0.00%
```

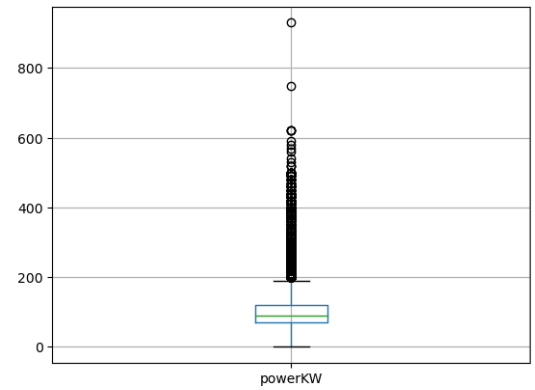


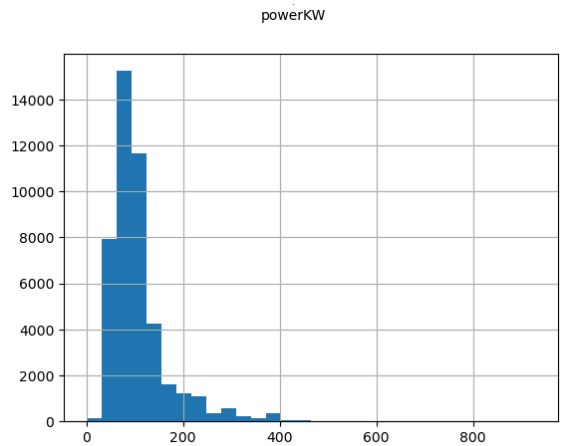

```
Statistics for nbbackoffice:
Mean: 0.34134615384615385
Median: 0.0
Mode: 0    0
Name: nbbackoffice, dtype: int64
Standard Deviation: 0.6343235790462088
Minimum: 0
Maximum: 9
25th Percentile: 0.0
75th Percentile: 1.0
```

```
-----
Description and Plot(s) of variable  vhConstructionYear
Number of unique values:  83
Proportion of missing values: 9.31%
Statistics for vhConstructionYear:
Mean: 2015.0420665112283
Median: 2017.0
Mode: 0    2023.0
Name: vhConstructionYear, dtype: float64
Standard Deviation: 9.206320096394666
Minimum: 1014.0
Maximum: 2023.0
25th Percentile: 2011.0
75th Percentile: 2021.0
```

Continuous Variables

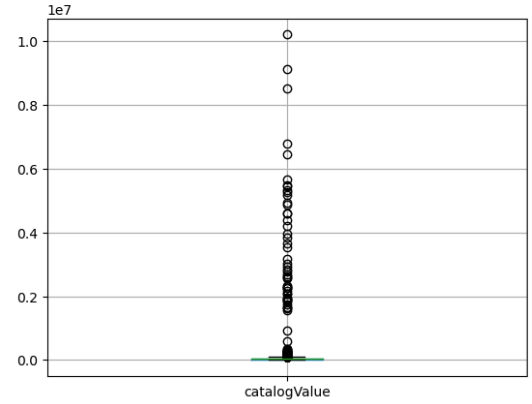
```
_____
Description and Plot(s) of variable  powerKW
Number of unique values:  61
Proportion of missing values: 0.00%
```

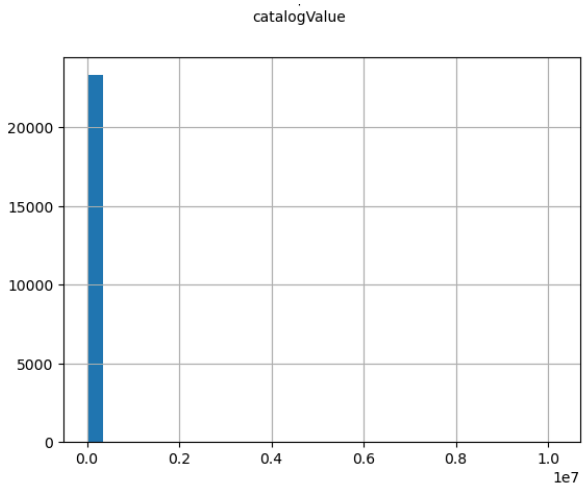




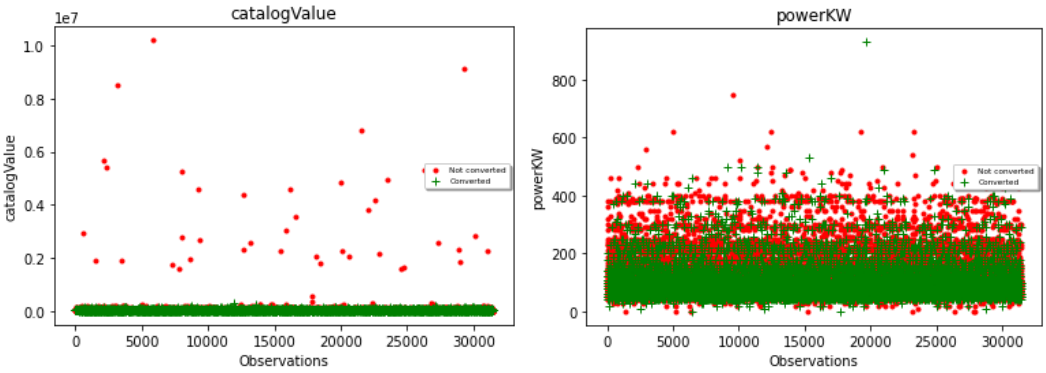

```
Statistics for powerKW:
Mean: 107.205974002849
Median: 90.0
Mode: 0    80
Name: powerKW, dtype: int64
Standard Deviation: 61.32617922686786
Minimum: 0
Maximum: 930
25th Percentile: 70.0
75th Percentile: 120.0
```

```
_____
Description and Plot(s) of variable  catalogValue
Number of unique values:  106
Proportion of missing values: 47.94%
```



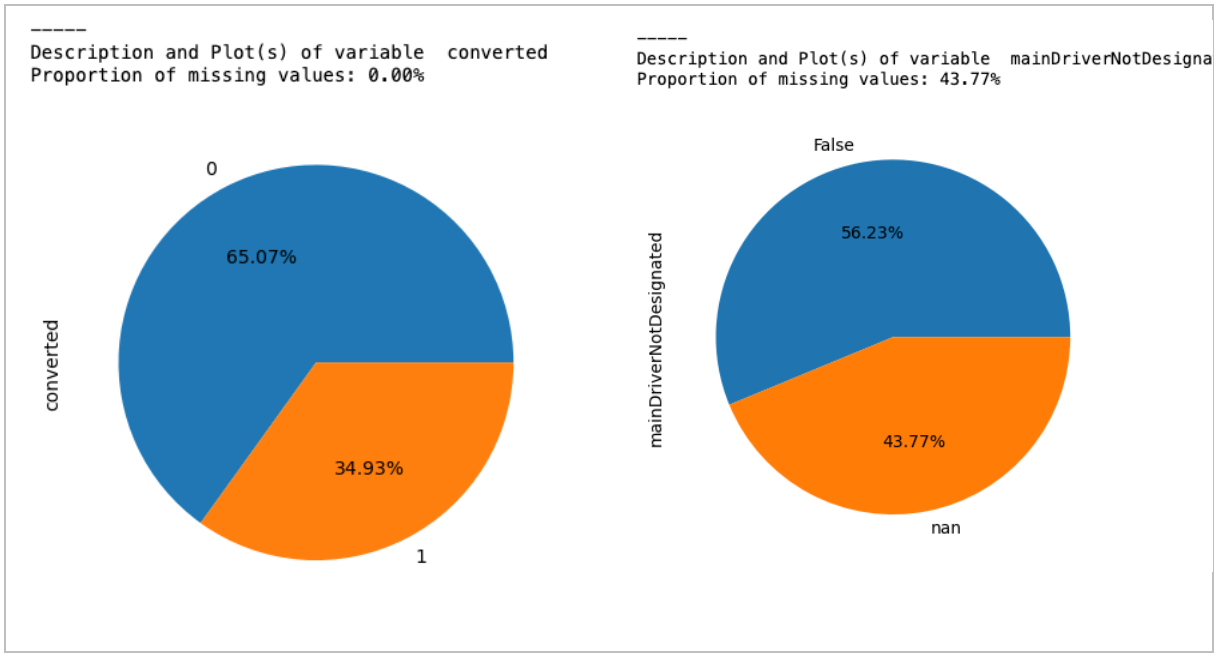


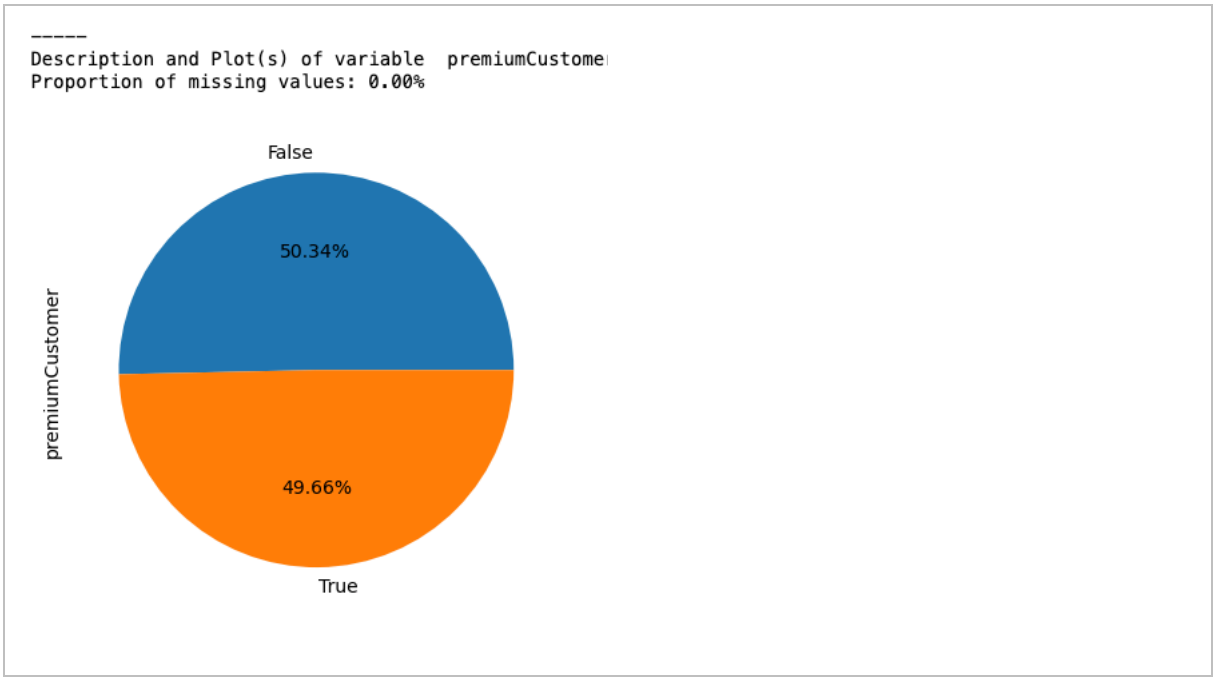

```
Statistics for catalogValue:
Mean: 47759.96237386694
Median: 35000.0
Mode: 0    25000.0
Name: catalogValue, dtype: float64
Standard Deviation: 190733.68530694384
Minimum: 0.0
Maximum: 10190000.0
25th Percentile: 20000.0
75th Percentile: 50000.0
```



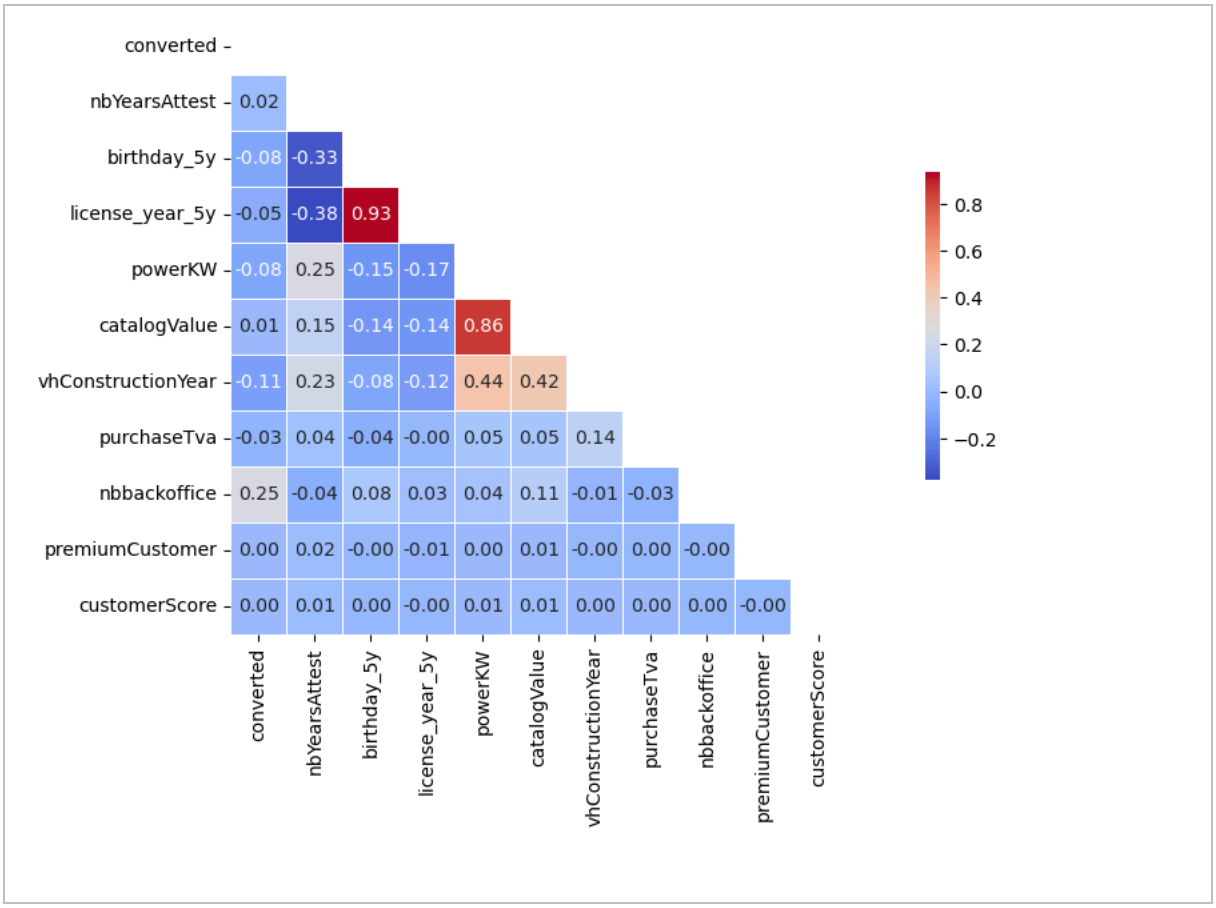
Both graphs strongly suggests that the expensive cars, likely attracted by the upper segment insurance agent Jean Verheyen, is less likely to be converted.

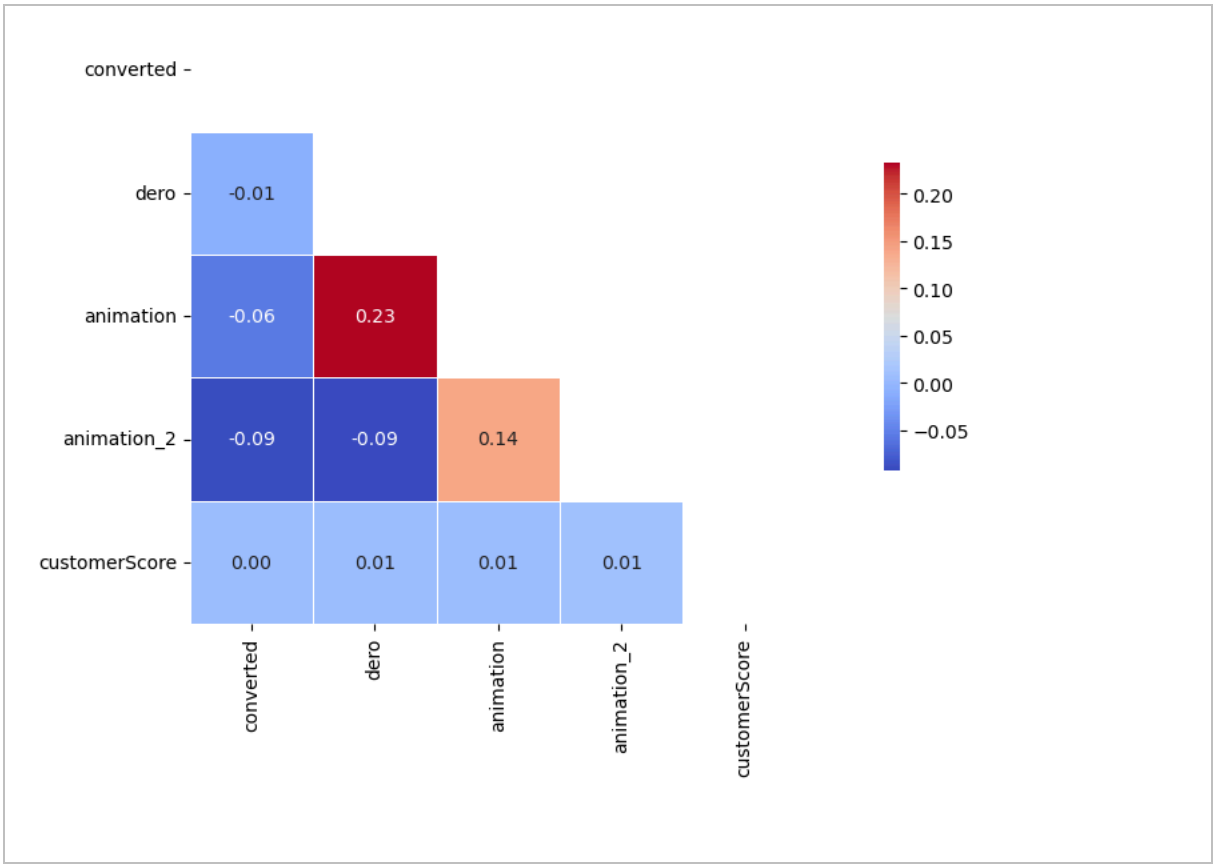
Binary Variables





Correlation Matrices





Appendix 3: Missing values replacements for the best performing model

Variable	Missing Value Replaced By	Additional Interpretation
nbYearsAttest	7	People that did not have an accident for more than 6 years
profession	'Other'	People that do not belong to any of the specified profession (categories) belong to a category 'Other)
birthday_5y	1990 (mode training set)	/
license_year_5y	2000 (mode training set)	/
fuelType	5	We have fuel types ranging between 1 and 8, but 5 was not present

catalogValue	35000 (median training set)	Since catalogValue contains many outliers, we opted for median rather than mean
vhConstructionYear	2023 (mode training set)	/
mainDriverNotDesignated	True	Only False was present in the data
purchaseTva	4	We have categories 0, 1, 2, and 3, so 4 seems like a reasonable choice

Appendix 4: Model Description

Decision Tree

A Decision Tree is a supervised learning algorithm used for both classification and regression. By learning simple decision rules, learned from the data, the model is able to predict the value of the target variable. The algorithm has a hierarchical tree structure, consisting of a root node, branches, internal nodes, and leaf nodes. The hyperparameters that we will consider in hyperparameter tuning are: splitting criterion (criterion), maximum depth of the tree (max_depth), the minimum number of samples required to split an internal node (min_samples_split), the minimum number of sample

required to be at a leaf node (min_samples_leaf), and the number of features to consider when looking for the optimal split (max_features) (ScikitLearn, 2019).

Logistic Regression

A logistic regression is a supervised learning algorithm that estimates the probability of an event occurring, using a logit transformation on the probability of success divided by the probability of failure (log odds). Through maximum likelihood estimation, the best fit of log odds is found. This best fit is the log likelihood function, and the logistic regression tries to maximize this function and find the best parameter estimate. From this parameter estimate, the predicted probabilities can be calculated. The hyperparameters that we will consider are the penalty form (penalty), the inverse of the regularization strength (C), and the algorithm to use in the optimization problem (solver).

Random Forest

A Random Forest is an example of a supervised learning ensemble method, in essence combining the predictions of multiple (simple or weak) decision trees to become a strong learner. In classification tasks, every decision tree in the random forest makes a prediction for an individual observation. Subsequently, the mode of all predictions is taken, and the value is assigned as the final prediction for a specific observation. The hyperparameters to be optimized are: the number of trees in the forest (n_estimators), the splitting criterion (criterion), the maximum depth of the tree (max_depth), the

minimum number of samples required to split an internal node (min_samples_split), the minimum number of samples required to be at a leaf node (min_samples_leaf), and the number of features to consider when looking for the optimal split (max_features). Note that four out of five hyperparameters are the same as for the decision tree.

Support Vector Machine

A Support Vector Machine is a supervised learning algorithm that in essence tries to find an hyperplane in a high-dimensional space, that is able to separate observations in terms of class, hence allowing to classify the observations. The best possible plane is the plane that has the maximum margin, which is the maximum distance between observations from both classes. In our modelling approach, we made a distinction between a linear SVM (kernel type = linear) and a non-linear SVM (kernel type = rbf). For the linear SVM, the hyperparameters that we considered is the regularization parameter (C). For the non-linear SVM, the hyperparameters that we considered are again the regularization parameter (C) and the kernel coefficient for kernel type rbf (gamma).

KNeighborsClassifier (kNN)
No model is constructed. Classification relies on a majority vote of the nearest neighbors of each point.

HistGradientBoostingClassifier (HGB)
An ensemble booster known for his strength in handling missing values, yet selected because its high calculation speed (as it is histogram-based).

GradientBoostingClassifier (GBC)
Another booster of an ensemble, however aiming at improving the loss function.

MLPClassifier (MLP)
Like the Support Vector Machines, a neural network that uses backpropagation, yet optimizing the log-loss function.

Stochastic Gradient Descent (SGD)
In essence, an optimization technique for loss function of the linear models Support Vector Machines and Logistic Regression classifiers.

Appendix 5: Optimal Hyperparameter Values

Decision Tree	
Criterion	Gini
Max_depth	6
Min_samples_leaf	4
Min_samples_split	20

Max_features	None

Logistic Regression	
C	0.01
Penalty	L2
Solver	Newton-cg

Random Forest	
Criterion	Entropy
Max_depth	50
Min_samples_leaf	2
Min_samples_split	20
Max_features	sqrt
N_estimators	1000

Linear SVM	
C	0.1
Kernel	linear

Non-linear SVM	
C	100
Kernel	Rbf
Gamma	scale

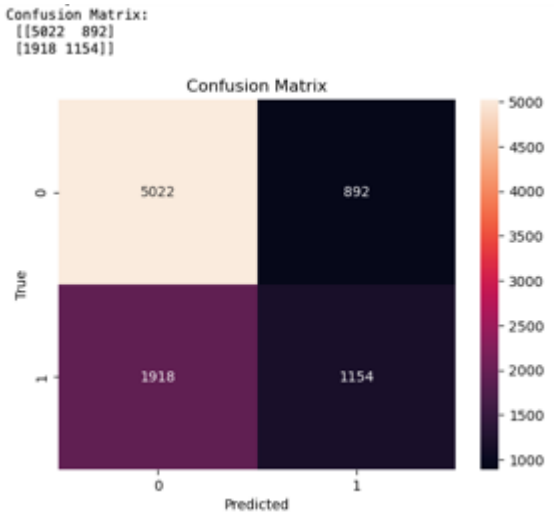
Appendix 6: Model Performance & Performance Metrics

	Base Models				
	Decision Tree	Logistic Regression	Random Forest	Linear SVM	Non-Linear SVM
AUC	0,5917	0,7113	0,7311	0,7086	0,6684
Accuracy	0,6316	0,6877	0,6931	0,6867	0,6581

	After Hyperparameter Tuning				
	Decision Tree	Logistic Regression	Random Forest	Linear SVM	Non-Linear SVM
AUC	0,7201	0,7168	0,7444	0,7029	0,7197
Accuracy	0,6873	0,6877	0,7003	0,6848	0,6581
Cost	6368	5306	5684	5788	3072
Lift (10%)	1,2791	1,2272	1.2602	1.2510	1.1530

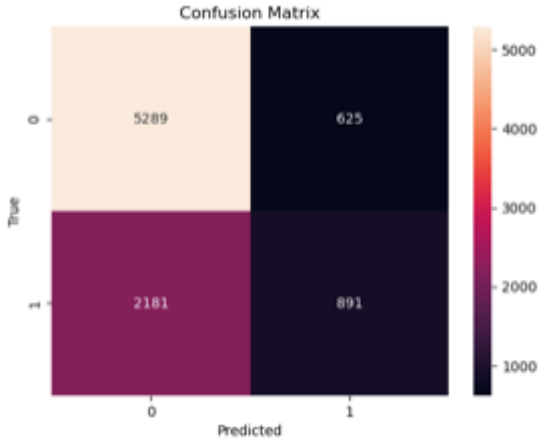
Confusion Matrices

Decision Tree



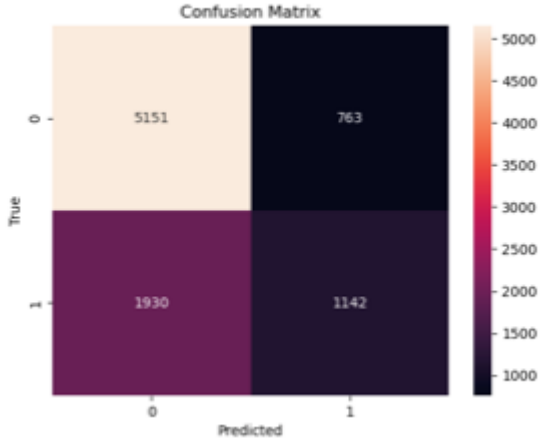
Logistic Regression

```
Confusion Matrix:
 [[5289  625]
 [2181  891]]
```



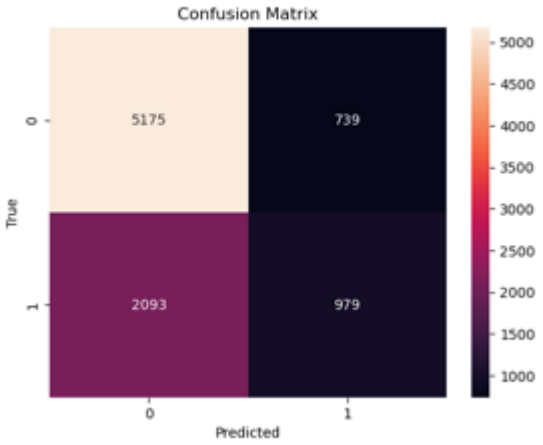
Random Forest

```
Confusion Matrix:
 [[5151  763]
 [1930 1142]]
```



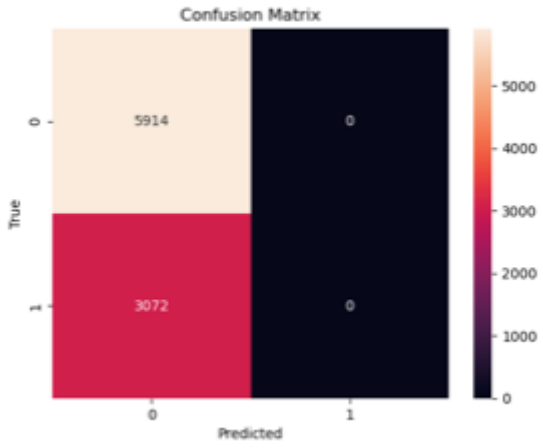
Linear SVM

```
Confusion Matrix:
 [[5175  739]
 [2093  979]]
```



Non-linear SVM

```
Confusion Matrix:
 [[5914    0]
 [3072    0]]
```



Appendix 7: total overview models of the different coding approaches

code	model	acc	bal. acc.	AUC	TP	FN	FP	TN	total cost	precision -	precision +	recall	F1 -	F1 +
1	RFC	0,7013	0,6218	0,7444	5164	750	1934	1138	5684	0,6028	0,7275	0,8732	0,7132	0,7937
2	RFC			0,7370										
2	DTC			0,7367										
1	DTC	0,6875	0,6128	0,7201	5042	890	1918	1154	6368	0,5646	0,7244	0,8500	0,6785	0,7822
1	SVM non-lin.	0,6581	0,5000	0,7197	5914	0	3072	0	3072		0,6581	1,0000		0,7938
1	LR	0,6877	0,5922	0,7168	5289	625	2181	891	5306	0,5877	0,7080	0,8943	0,7093	0,7903
2	LR			0,7104										
1	SVM linear	0,6848	0,5969	0,7029	5175	739	2093	979	5788	0,5698	0,7120	0,8750	0,6902	0,7852
3	HGB	0,5599	0,6204	0,6828	1834	440	2526	1939	4726	0,8150	0,4206	0,8065	0,8108	0,5529
3*	RFC	0,6599	0,6658	0,6662	4324	2232	60	123	11220	0,0522	0,9863	0,6595	0,0968	0,7905
3	SVM linear	0,4612	0,5876	0,6447	994	265	3366	2114	4691	0,8886	0,2280	0,7895	0,8361	0,3538
3	LR	0,4751	0,5830	0,6435	1156	333	3204	2046	4869	0,8600	0,2651	0,7764	0,8161	0,3953
3	SGD	0,4928	0,5799	0,6300	1361	419	2999	1960	5094	0,8239	0,3122	0,7646	0,7931	0,4433
3	GBC	0,6354	0,5912	0,6285	3311	1408	1049	971	8089	0,4082	0,7594	0,7016	0,5161	0,7294
2	kNN			0,6257										
3	RFC	0,6091	0,5654	0,5917	3142	1416	1218	963	8298	0,4048	0,7206	0,6893	0,5101	0,7046
3	DTC	0,5749	0,5493	0,5521	2745	1250	1615	1129	7865	0,4746	0,6296	0,6871	0,5614	0,6571
3	MLP	0,5885	0,5365	0,5519	3139	1552	1221	827	8981	0,3476	0,7200	0,6692	0,4576	0,6936
	All higher values in green color scales are more favorable. The lower values are prefered in red color scales.													

Appendix 8: Performance Metrics Explanation

AUC

The AUC (Area Under the ROC Curve) as the name suggest, is the area under the ROC curve. ROC (Receiver Operating Characteristics curve) is basically a graph that shows the performance of a classifier at different classification thresholds. The graph plots the TPR (True Positive Rate) on the y-axis and the FPR (False Positive Rate) on the x-axis. The ROC is a concave function, that ideally lies as close as possible to the upper left corner. A straight diagonal represent the performance of a random model.

Accuracy

Accuracy is calculated as the fraction of correct predictions over the total number of observations.

Balanced accuracy

The balanced accuracy score takes into account possible imbalances in the datasets used. By theory, this score should equal the accuracy when datasets are perfectly balanced.

Confusion Matrix

The confusion matrix compares the predicted values to the actual values, in essence a 2x2 matrix. It contains the positive observations correctly classified as positive (True Positives), the positive observations wrongly classified as negative (False Positives, FP), the negative observations correctly classified as negative (True Negatives), and the negative observations wrongly classified as positive (False Negatives, FN). Note that positive and negative refer to the (predicted) target class (i.e. respectively 1 or 0).

Cost

Another performance metric is total cost, which represents the total cost of the misclassified predictions. The total cost is calculated as Cost_FN * FN + Cost_FP * FP. The costs were specified as follows:

- Cost_FN = 5
- Cost_FP = 1

Lift

The lift score is a performance metric that measures how good a model is at predicting positive outcomes compared to a random model (i.e. randomly guessing if an instance is positive or negative). The higher the lift score, the better the model is at predicting positive outcomes.

F1 score

The F1 score can be interpreted as a harmonic mean of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. It is used to judge the quality of the model.

Appendix 9: overview data and codes

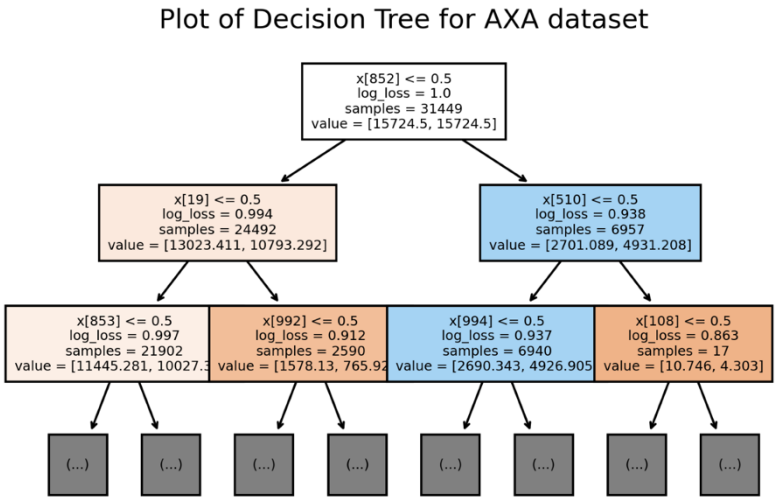
deleted (from)
unchanged
replaced by mean
replaced by mode
replaced by median
replaced with
clipped in range
replacements based on training set solely
min: replacement with min training set used in that run after excluding the outliers

** reduced data set for testing 'heavy' models

approaches						code 1		code 2		code 3	
				[Train]/[Valid & Test] split		80/20		80/20		70/30	
				[Valid]/[Test] split		75/25		80/20		50/50 and 50/20**	
				Tuning		GridSearch		GridSearchCV		Grid- & RandomSearchCV	
				Replacement values derived from		training		training		training	
				Set used for fitting		train		train		train	
				Set used for tuning		valid		train		valid	
				Set used for prediction		test		test		test	
variable initial name	code 1	code 2	code 2	unique values	NA	missing	outliers	missing	outliers	missing	outliers
Month	nominal			8	0,00%						
converted	binary			2	0,00%						
dero	ordinal			17	0,00%						
animation	ordinal			8	0,00%						
animation_2	ordinal			3	0,00%						
nbYearsAttest	discrete			7	78,59%	7		7		0	
profession	nominal			8	78,29%	other				0	
birthday_5y	discrete			18	0,01%						<1923: min
license_year_5y	discrete			15	78,29%						
fuelType	nominal			8	22,60%	5		5		5	
powerKW	continuous			61	0,00%		"-3:3"		"-3:3"		<2: min
make	nominal			266	0,00%						
model	nominal			2072	0,00%						
catalogValue	continuous			106	47,94%		"-3:3"		"-3:3"		<1: min
isSecondHandVehicle	binary			2	21,53%						
vhConstructionYear	discrete			83	9,31%						<1970: min
mainDriverNotDesignated	binary			2	43,77%	TRUE		TRUE		TRUE	
postal_code_XX	nominal	ordinal	ordinal	78	0,00%						
availableActions	discrete			5	71,41%						
purchaseTva	discrete			5	42,13%	4		4		4	
nbbackoffice	discrete			10	0,00%						
premiumCustomer	binary			2	0,00%						
customerScore	ordinal			76	0,00%						

Appendix 10: Explainable AI

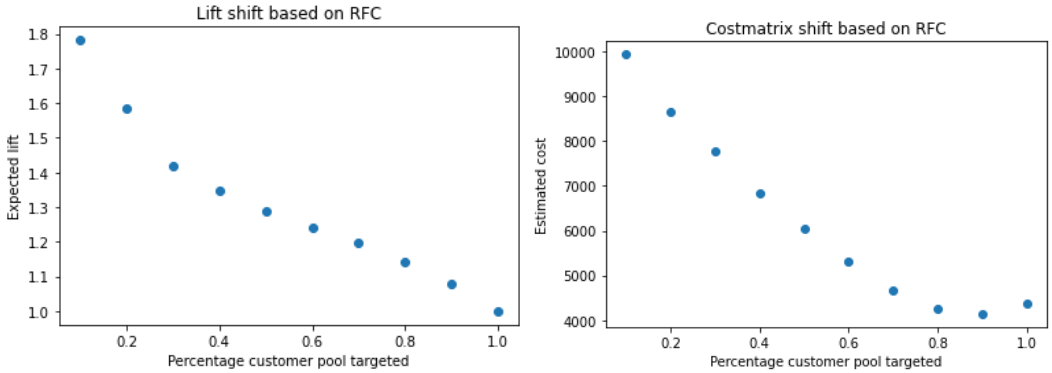
In order to provide insights in the dynamics behind this report, we elaborate on explainable AI by exploring a decision tree as well as a cost and lift curve.



Above an illustrative print of a DecisionTree based on the AXA dataset. Variable x[852] in the top node is a dummy variable for the sixth category of nbYearsAttest and enables splitting the training set in 15,724.5 observations (with dummy value 0 (i.e. <= 0.5)) and target class converted, and 15,724.5

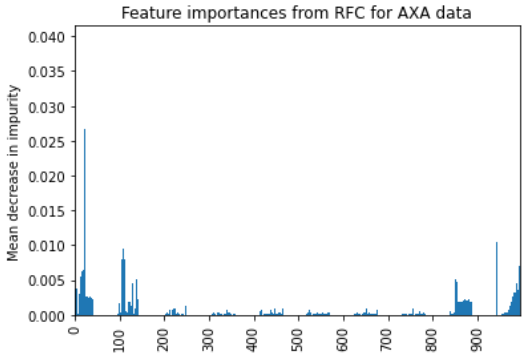
observations (with dummy value 1 (i.e. > 0.5)) and target class not-converted. Similarly the sample is further dividable by the rule <=0.5 across dummy variable of a category of e.g., birthday (x[19]), car model (x[510]) or nbbackoffice (x[853]). We note that this specific tree has a rather low AUC score (below 0.6). Yet it illustrates the working principle of classification nicely.

Illustrated example of lift and cost matrix



If the almost 50,000 observations selected by AXA are representative for the much larger insurance market, the lift in graph 1 reveals that targeting 10% of this market with this specific RFC model results in predictions 1.8 times better than random. As more customers are targeted the quality of the model's output will be reduced. Contradictory to the lift curve, the cost matrix suggest targeting larger parts of the market to reduce the cost of wrong predictions. The cost of falsely predicting a customer to be converted weighs heaviest in this matrix. As such this essential trade-off has to made to optimize the working model.

Illustrative feature importances based on RFC (the number of the variable on X-axis)



Besides business indicators, weighing feature importances of used variables is a continuous process. Graph 3 illustrates how birthday (close to nr 0) and dero (close to nr 100) are among others more clearly more important variables than e.g. model (between nr 300 and 800). Thus dropping the variable 'model' as done in code 2 is justifiable. However, bundling its current categories likely has potential to extract the predictive power of the information in it.