Student n: 2929300

Word count: 2376

Data Mining Report

Assignment Resit, June 2023

# Business Understanding

The task at hand consists in building a classifier for a car insurance company that predicts if a client will claim insurance or not, with the end goal being to develop an ML model that uses customer details as input and outputs binary predictions of whether customers will make insurance claims or not.

The data provided in the *resit\_dataset\_prepared.csv* file consists of 1000 records of customer information from the car insurance company. The data set includes the following relevant attributes: gender, age, race, driving experience, education, income, credit score, vehicle details, marital status, number of children, country, post code, annual mileage, vehicle type, speeding violations, driving under influence, and past accidents. The target variable is *HAS\_CLAIMED\_INSURANCE* which indicates if a customer has claimed insurance or not.

The finished system will require a classifier model capable of predicting whether or not a customer will claim insurance based on the attributes listed above. The system will take into consideration customer details (input) and will produce a binary prediction (output) of “claim” (1) or “no claim” (0).

This task suits the requirements of a Data Mining Approach due to: its large dataset which provides a substantial amount of data to extract patterns and insights; its complex relationships in insurance claiming decisions which tend to depend of factors such as customer demographics, driving history, and policy details, which are analysed through data mining techniques, thus finding patterns that are hard to find and discern manually; its classification problem since predicting whether or not a customer will claim insure is a classification problem, and a common application of data mining techniques; and its decision support to develop an accurate classifier which will help the car insurance company to assess risks, optimize premiums, and make informed decisions based on customer data.

The following terminology can be found throughout the report:

* Attribute: also known as feature or variable, it is a specific characteristic or property of data. Within this report, attribute will refer to the various attributes of the csv file dataset which help describe and categorize the customers.
* Binary prediction: prediction or classification with two and only two outcomes or categories. Within this report, binary prediction will refer to the model’s prediction as to whether or not the customer will claim insurance. It will be represented as “1” for “claim” and “0” for “not claim”.
* Input: independent variables/features. Within this report, input will refer to the customer attributes provided in the csv file to be used in the model for prediction.
* ML Model (or simply Model): a mathematical representation of the relationships between input attributes and the target variable.
* Output: prediction or result generated by a ML model. Within this report, output will refer to the binary prediction of whether or not a customer will claim insurance as represented by the variable “HAS\_CLAIMED\_INSURANCE”.
* Predictive Model: similar to Model. It is a mathematical or statistical model that uses historical data and patterns to make inferences about future outcomes or events by identifying relationships and patterns in the input data and applying them to make predictions upon the output variable.
* Target variable: also known as dependent variable or response variable, it is the variable that will be predicted. Within this report, target variable will refer to the “HAS\_CLAIMED\_INSURANCE” column within the csv file which indicates whether or not a customer has claimed insurance.
* Task: the task at hand, to build a classifier ML model that predicts whether or not a customer will claim insurance based on their attributes as listed in a csv file.
* Variable: an attribute or feature of a dataset. E.g.: age, gender, country, …
  + Categorical: see Nominal.
  + Continuous: it refers to a numeric variable that can be any value within a given range. It can be fractional and/or decimal and it has an infinite number of possible values. E.g.: income, annual mileage …
  + Discrete: it refers to a numeric variable that can only take certain values or integers. E.g.: past accidents, speeding violations, …
  + Nominal: also known as Categorical, it represents categorised/labelled data without any inherent order or numeric value and is treated as labels that are later used to represent different groups/categories. It can have a finite number of distinct values. E.g.: gender, race, education, …
  + Numeric: it refers to data that is represented by numeric values, categorized into continuous and discrete variables. E.g.: credit score, speed violations, …

With regards to project methodology, I will be using the Cross-Industry Standard Process for Data Mining (CRISP-DM) which will ensure a structured and systematic approach to the project, allowing me to conduct reliable analysis and to make informed decision-making based on the output of the predictive model. CRIPS-DM consists of the following 6 stages:

1. **Business Understanding:** acknowledging business goals and defining the problem.
2. **Data Understanding:** exploring and understanding the data provided. Assess data quality and identify relevant attributes.
3. **Data Preparation:** readying data for modelling by addressing missing values, data cleaning, normalisation, feature selection, and data splitting into the training, validating, and testing datasets.
4. **Modelling:** choosing the appropriate ML model algorithms, training them using the training dataset, and tuning hyperparameters for optimal performance.
5. **Evaluation:** evaluating the model’s performance through appropriate evaluation metrics and selecting the best-performing model.
6. **Deployment:** documenting findings, preparing a summary of the project, and presenting said report to stakeholders.

# Data Understanding

To aid in building the solution, I have listed the variables found in the dataset alongside their data type and suitability for the project:

|  |  |  |
| --- | --- | --- |
| **Variable** | **Data Type** | **Suitability** |
| ID | Nominal - unique identifier for each customer | Not considered, it has no predictive value |
| AGE | Nominal – categorical age range | Considered, should be treated as discrete nominal variable |
| GENDER | Nominal – categorical gender | Considered, should be treated as discrete nominal variable |
| RACE | Nominal – categorical race | Considered, should be treated as discrete nominal variable |
| DRIVING\_EXPERIENCE | Nominal – categorical driving experience range | Considered, should be treated as discrete nominal variable |
| EDUCATION | Nominal – categorical education level | Considered, should be treated as discrete nominal variable |
| INCOME | Nominal – categorical income level | Considered, should be treated as discrete nominal variable |
| CREDIT\_SCORE | Numeric – continuous credit score | Considered, should be treated as continuous variable |
| HAS\_OWN\_VEHICLE | Nominal – categorical ownership of a vehicle | Considered, should be treated as discrete nominal variable |
| VEHICLE\_YEAR | Nominal – categorical vehicle year range | Considered, should be treated as discrete nominal variable |
| IS\_MARRIED | Nominal – categorical marital status | Considered, should be treated as discrete nominal variable |
| HAS\_CHILDREN | Nominal – categorical presence of children | Considered, should be treated as discrete nominal variable |
| COUNTRY | Nominal – categorical country of residence | Considered, should be treated as discrete nominal variable |
| POSTAL\_CODE | Nominal – categorical postal code | Not considered, specified to customer location, has no predictive value |
| ANNUAL\_MILEAGE | Numeric – continuous estimated annual mileage | Considered, should be treated as continuous variable |
| VEHICLE\_TYPE | Nominal – categorical type of vehicle | Considered, should be treated as discrete nominal variable |
| SPEEDING\_VIOLATIONS | Numeric – discrete count of speeding violations | Considered, should be treated as discrete variable |
| DRIVING\_UNDER\_INFLUENCE | Nominal – categorical driving under the influence | Considered, should be treated as discrete nominal variable |
| PAST\_ACCIDENTS | Numeric – discrete count of past accidents | Considered, should be treated as discrete variable |
| HAS\_CLAIMED\_INSURANCE | Nominal – categorical insurance claim status | Target variable, to be predicted |

Note that “AGE” could also be classified as an ordinal variable rather than nominal as it represents data with a meaningful order but without a consistent numeric difference between categories. However, since some ML algorithms may treat this ordinal variable as nominal during modelling, it made more sense to consider “AGE” as being nominal since in practice the treatment of ordinal variables varies accordingly to the goals of the project at hand.

In sum, all variables are suitable for the project except for “ID” and “POSTAL\_CODE”.

With regards to inputs and outputs, I selected input variables based on the assumption that they may have predictive value in determining whether or not a customer will claim insurance. The output variable will be determined based on the nature of the task – to build a classifier to predict the status of insurance claims – and the model will be training to predict its binary status based on input attributes.

|  |  |
| --- | --- |
| Input | AGE, GENDER, RACE, DRIVING\_EXPERIENCE, EDUCATION, INCOME, CREDIT\_SCORE, HAS\_OWN\_VEHICLE, VECHICLE\_YEAR, IS\_MARRIED, HAS\_CHILDREN, COUNTRY, ANNUAL\_MILEAGE, VEHICLE\_TYPE, SPEEDING\_VIOLATIONS, DRIVING\_UNDER\_INFLUENCE, PAST\_ACCIDENTS |
| Output | HAS\_CLAIMED\_INSURANCE |

These decisions were made with the task goal in mind and will contribute to the building of a predictive model that can effectively classify whether or not a customer will claim car insurance.

# Data Preparation

After loading the dataset into a pandas DataFrame using pandas’ ‘pd.read\_csv()’, I performed the following data pre-processing steps:

1. Dropped rows and columns with missing values using ‘dropna()’.
2. Filled missing values with their respective median using ‘fillna(df.median())’.
3. Checked for duplicate rows and removed them using ‘drop\_duplicates()’.
4. Corrected mis-typed entries for the GENDER and EDUCATION variables using ‘replace()’. In GENDER, ‘m’ and ‘f’ were replaced with ‘male’ and ‘female’, and in EDUCATION, ‘hs’ and ‘na’ were replaced with ‘high school’ and ‘none’.
5. For Data Transformation and Scaling, I one-hot encoded discrete nominal variables with ‘pd.get\_dummies()’ to represent them as binary indicators, normalized continuous variables with min-max scaling using scikit-learn’s ‘MinMaxScaler()’, and label encoded discrete variables using ‘LabelEnconder()’. [[1]](#footnote-1)

The AGE variable is the one whose histogram shows more obvious changes before and after pre-processing. Both histograms were scaled from 0 to 3000 in jumps of 500.

A picture containing text, screenshot, diagram, line

Description automatically generated

From the histograms, we can gain the following insights into the effect of data transformation on the variable:

* Shape of the Distribution: both before and after histograms have a similar shape to one another, with two peaks at 40-64 and 26-39 which suggests a bimodal distribution. If we were to order the bars from younger age groups to older age groups, we would notice a somewhat bell-shaped histogram, suggesting normal distribution in the data.
* Central Tendency: all three values of Mode, Median and Mean are higher before preprocessing and lower after preprocessing.
* Spread: in both histograms, there are higher number of customers in the age groups 40-64 and 26-39, suggesting that customers within the age range 26-64 are significantly more likely to own a car than those outside of this range.
* Skewness and Kurtosis: for both histograms the condition of Mode < Median < Mean is met, suggesting that the graphs are positively skewed.

Data was split 50/50 using scikit-learn’s ‘train\_test\_split()’, dividing the data into equally sized test and training datasets. The test dataset was then separated to evaluate the model’s performance on unseen data.

# Modelling

I have chosen to build models using Decision Trees, Random Forests and Support Vector Machines (SVM). The following table lists the hyperparameters per model and the effects of each parameter:

|  |  |  |  |
| --- | --- | --- | --- |
| **Model** | **Hyperparameters** | **Effects** | **Validation** |
| Decision Trees | max\_depth | Maximum depth of tree | k-fold cross-validation |
| min\_samples\_split | Minimum number of samples needed to split and internal node |
| min\_samples\_leaf | Minimum number of samples needed to be a leaf node |
| Random Forests | n\_estimators | Number of trees in the forest | Randomized search with cross-validation |
| max\_depth | Maximum depth of each tree |
| min\_samples\_split | Minimum number of samples needed to split and internal node |
| min\_samples\_leaf | Minimum number of samples needed to be a leaf node |
| Support Vector Machines (SVM) | C | Regularization parameter | Coarse-to-Fine Search[[2]](#footnote-2) |
| kernel | Kernel type used in the algorithm |
| gamma | Kernel coefficient |

Summary of Decision Tree results:

A picture containing text, screenshot, font, number

Description automatically generated

Summary of Random Forest results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| n\_estimators | min\_samples\_split | min\_samples\_leaf | max\_depth | mean\_test\_score | std\_test\_score |
| 100 | 10 | 1 | 0 | 0.844607654 | 0.006080036 |
| 50 | 10 | 2 | 0 | 0.841540287 | 0.00696728 |
| 100 | 5 | 4 | 0 | 0.840925848 | 0.008252704 |
| 100 | 5 | 1 | 10 | 0.840158742 | 0.006032502 |
| 200 | 2 | 2 | 0 | 0.839546069 | 0.007032148 |
| 200 | 10 | 4 | 5 | 0.824359195 | 0.00974315 |

Based on these results, the Random Forest model seemed to work better. Running it on all data, we obtain the following metrics:

* Accuracy: 0.8386503067484663
* Precision: 0.7525562372188139
* Recall: 0.7215686274509804
* F1 score: 0.7367367367367368

These metrics further provide information on the model’s accuracy, ability to correctly identify positive cases (precision), ability to capture positive cases (recall), and the overall balance between precision and recall (F1 score).

# Results and Errors

To evaluate how the final model performs, we test it on the test dataset.

The Random Forest model achieved an accuracy of approximately 0.844 meaning that it correctly predicted the target variable 84.4% of the time during the testing period.

The confusion matrix obtained (left) shows that the model wrongly predicts false positives (120) and false negatives (135) more often than it returns true positives (375) and true negatives (1000). Depending on the problem at hand, these errors can have different implications. On the current task of predicting whether a customer will claim car insurance, we need to have in consideration that false positives can occur due to unnecessary costs (which can lead to additional expenses being allocated to customers who wouldn’t claim insurance) and customer experience (where it might cause frustration or inconvenience for customers who are wrongly mislabelled), whereas false negatives can occur due to missed opportunities (where customers could’ve claimed insurance but did not) and customer satisfaction (where a customer’s potential to be identified as a possible claimant was overlooked). Based on this and my own experience, I would deem false positives as being more hurtful for the insurance company.

The metrics of accuracy, precision, recall and F1 score, as identified in the previous section, also hold valuable insights into the model. Accuracy is ratio of correctly predicted samples to the total number of samples; in this case, the model resulted in an accuracy of approximately 83.87%. Precision is ration of true positive predictions to the total number of positive predictions; in this case, the model resulted in a precision of approximately 75.26%. Recall is ratio of true positive predictions to the total number of actual positive samples; in this case, the model resulted in a recall of approximately 72.16%. F1 score is a balanced measure of the model’s performance by considering both precision and recall, ranging from 0 to 1, where 1 stands for the best performance; in this case, the model resulted in a F1 score of approximately 0.7367, indicating a reasonably balanced performance between precision and recall. A picture containing text, screenshot, diagram, rectangle

Description automatically generated

1. While building the Decision Trees model, I was getting multiple errors due to just one-hot encoding certain variables. To fix this, I later label encoded all variables. [↑](#footnote-ref-1)
2. Initially, I tried validating this model with Grid Search with Cross-Validation. However, since it kept taking too long for my machine to effectively run it, I changed to Coarse-to-Fine Search. [↑](#footnote-ref-2)