**Introduction**

The given code performs a data analysis and model building on a dataset contained in a file called "ahs\_insurance\_sample.xlsx". The dataset seems to be related to insurance policies, and the ultimate goal is to predict whether a customer will buy insurance or not. The code can be divided into several parts, which are explained below.

**Data exploration:**

The code reads the data from the file using pandas library's read\_excel function, and stores it in a dataframe called "df". The describe() method is then used to get a summary of the dataset, which includes count, mean, standard deviation, minimum, and maximum values of each column. The info() method is used to get information about the dataset, such as the number of non-null values in each column and the data type of each column.

**Handling missing values:**

Next, the code checks for missing values using the isnull() method followed by the sum() and sort\_values() methods. The result shows the number of missing values in each column in descending order. There are 5 columns that contain null values, which will be handled later on. The code then checks for duplicate values in the dataset using the duplicated() method followed by the sum() method. The result shows that there are no duplicate values in the dataset.

**Exploratory analysis of the BUYI column:**

The code then performs exploratory analysis of the target column, which is called "BUYI". The value\_counts() method is used to get the count of each unique value in the column. Since the dataset is imbalanced, with much fewer 0 values than 1 values, the code decides to use the f1\_score as a metric to evaluate the performance of the models. The code also uses a bar plot to visualize the distribution of the target variable.

Below is the graph that shows how small the 0 values are compared to the ones :

Chart, bar chart

Description automatically generated

**Exploratory analysis of the features:**

The code then performs further exploratory analysis on the other columns in the dataset using a pair plot. The pair plot shows the relationship between each pair of features in the dataset. The code observes that some features are uniformly distributed, while others have anomalies. The code decides to investigate the correlation between features and the target variable and uses a catplot to plot the ZSMHC and HHAGE columns against BUYI. The result shows that the larger the ZSMHC value, the closer it gets to 1 as a target. For the HHAGE column, there is barely any difference between the 0 and 1 values. The code uses a scatter plot to illustrate these results.

Chart, histogram, scatter chart

Description automatically generated

Pair plot 1 : AMTI feature

Chart, histogram

Description automatically generated

Pair plot 2 : ZSMHC Feature

Chart, histogram

Description automatically generated

Pair plot 3 : HHAGE feature

Chart, box and whisker chart

Description automatically generated

Box Plot 1 : ZSMHC feature vs target variable.

Chart, box and whisker chart

Description automatically generated

Box Plot 1 : ZSMHC feature vs target variable.

Chart, scatter chart

Description automatically generated

Scatter Plot 1 : Illustrating the distribution of the target according to ZSMHC and HHAGE features

**Features correlation**

The code then creates a correlation matrix plot using the heatmap method from seaborn library. The correlation matrix shows the correlation coefficients between each pair of features in the dataset. The code uses the correlation matrix to identify the columns that have a lot of missing values and do not correlate with other features and removes them from the dataframe.

A picture containing text

Description automatically generated

Fig : Correlation matrix of all the dataset features

**Training and comparing machine learning models.**

The code then builds several machine learning models using 5-fold cross-validation. The models are Decision Trees, Random Forest, Light GBM, SVM, KNN, and XG Boost. The code uses the f1\_score as the evaluation metric for the models. The code outputs the f1\_score for each model in each fold of the cross-validation and calculates the average f1\_score for each model over all folds. The code selects the Random Forest model as the best model and decides to tune its hyperparameters for better performance.

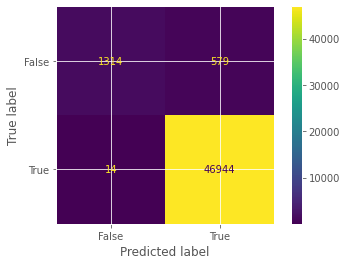


Fig : Confusion matrix of Random Forest Prediction

**Calculating the annual profit**

The goal of the annual profit calculation is to estimate the potential profit or loss that the insurance company can expect to earn or lose based on the model's predictions. The calculation is based on a set of assumptions about the costs and revenues associated with each policy.

In this particular code snippet, the calculation of annual profit is done in the following way:

1. A new data frame dfCopyForProfit has been created that excludes the target column BUYI, as it is not needed for the calculation of annual profit.
2. The missing values in the target column BUYI are replaced with 0, which is the negative class of the target.
3. The randomForestModel is used to predict the target values for each row of the dfCopyForProfit data frame.
4. For each row in the predicted target values, the corresponding annual profit is calculated using the following formula:

* If the predicted value is 1 and matches the actual value in the data frame, then 30% of the corresponding AMTI value is added to the total profit and a fixed fee of 500 USD is subtracted from it.
* If the predicted value is 1 but does not match the actual value in the data frame, then a fixed penalty of 200 USD is subtracted from the total profit.
* If the predicted value is 0, then there is no profit or penalty associated with it.

1. The total profit or loss is calculated by summing the profit/penalty associated with each row.
2. The final result is displayed as the total annual profit or loss that the insurance company can expect to earn or lose based on the model's predictions.

The amount of profit made by the company (according to this dataset) is **36472606.1 $**.

It is important to note that the calculation of annual profit is based on certain assumptions and may not reflect the actual performance of the insurance policy in practice. The assumptions made in this code snippet include the fixed fee and penalty amounts, as well as the assumption that the AMTI column is a reliable estimate of the actual amount of insurance purchased by the customer. Therefore, it is important to carefully review and adjust these assumptions based on real-world data and feedback from customers.

**Steps that were not used in the project.**

Just like every other machine learning project, sometimes it is better to avoid some steps in order to receive the maximum accuracy, which is exactly what we encountered here in this project.

1. Row removal: Removing rows that contain missing values can result in a loss of information, especially if the missing values are randomly distributed throughout the dataset. Additionally, removing a large number of rows may lead to a reduction in the sample size, which can impact the statistical power of the analysis. In our code, you handled missing values by imputing them with the mean value of the column, which is a common strategy for dealing with missing data.
2. Outlier removal: Removing outliers from the dataset can be useful if the outliers are due to measurement errors or other artifacts that do not reflect the true underlying distribution of the data. However, if the outliers are genuine observations that represent extreme values of the variable, then removing them can result in a loss of important information. In our code, we did not remove any outliers, but instead explored the distribution of the features using pair plots and box plots, which can help identify potential outliers and inform the modeling strategy.
3. Encoding: Encoding categorical variables can be useful for machine learning algorithms that require numerical inputs. However, some algorithms, such as tree-based models (Random Forest in our case), can handle categorical variables without the need for encoding. In our code, we did not encode the categorical variables, but instead used them as is in the modeling process. This is a reasonable approach, especially if the categorical variables have a small number of unique values and are already in a format that can be directly used by the model.
4. Numerical transformations were not used as well because the dataset only contains numerical data.
5. Oversampling : for the sampling routines, we used oversampling as a method to add more rows containing 0 values in the target BUYI column. But it was not useful as it only lowered the score of our model. Consequently, in this case sampling should not be included.