CE802-7-AU-CO Machine Learning
Dr. Luca Citi

Assignment 1: Report on the Investigation

Student ID: 2003838 – PEREZ62206
January 2021

(Note: This document has 1213 words in total. However, 18 of them are for the cover page, 130 of them are for the references, and 44 of them are for this note.
The rest **1021 words** are **designated** to the **narrative** of this report)

## 1. Introduction

In this assignment, a Decision Tree Classifier,[1] Random Forest Classifier[2] and Logistic Regression[3] have been built, trained and evaluated, as a Machine Learning algorithms capable of predicting whether the insured will file a claim or not (y), based on given features (x).
Similarly, a Linear Regression[4], Random Forest Regressor[5] and Lasso regression[6] have been created, fitted and measured their performance, as a ML models capable of predicting not only if the customer claims its travel-insurance, but also the value of the claim.

## 2. Procedures used.

The Machine Learning procedures proposed in these comparative studies have been organized in a logical way according to the seven stages of ML proposed by Yufeng (2017):

- Gathering data
- Preparing that data
- Choosing a model
- Training
- Evaluation
- Hyperparameter tuning
- Prediction.

Each of these steps are summarized in the table 1.0, but for a more detailed explanation, please go to the Jupyter files: "*CE802_P2_Notebook.ipynb*" and "*CE802_P3_Notebook.ipynb*" and read all the comments and markdowns located in all the stages.

---

[1] A **Decision Tree Classifier** tries to solve a classification problem, by using tree representation. Each internal node of the tree corresponds to an attribute, and each leaf node corresponds to a class label. (Alpaydin 2014)

[2] "A **Random Forest Classifier** creates a "forest" of decision trees, each of which votes on the predicted class of an observation". (Albon 2018)

[3] A **Logistic Regression** "allow us to predict the probability that an observation is of a certain class using a straightforward and well-understood approach" (Albon 2018)

[4] A **Linear Regression** is a useful method of making predictions when the target vector is a quantitative value. "It assumes that the relationship between the features and the target vector is approximately" (Albon 2018)

[5] A **Random Forest Regressor** "operates by constructing a multitude of decision trees at training time and outputting the class that is the mean prediction (regression) of the individual trees". (Chakure 2018)

[6] A **Lasso Regression** is a type of linear regression that uses shrinkage. In this case, "Lasso uses regularization to control overfitting issues by applying a penalty on the absolute values of the coefficients" (Dangeti 2017)

Table 1.0

| Step / ML Model | Classification problem | | | Regression problem | | |
|---|---|---|---|---|---|---|
| | **Decision Tree Classifier** | **Random Forest Classifier** | **Logistic Regression** | **Linear Regression** | **Random Forest Regressor** | **Lasso Regression** |
| **Data collection** | • Dataset: CE802_P2_Data.csv<br>• Pandas library to observe the data | • Dataset: CE802_P2_Data.csv<br>• Pandas library to observe the data | • Dataset: CE802_P2_Data.csv<br>• Pandas library to observe the data | • Dataset: CE802_P3_Data.csv<br>• Pandas library to observe the data | • Dataset: CE802_P3_Data.csv<br>• Pandas library to observe the data | • Dataset: CE802_P3_Data.csv<br>• Pandas library to observe the data |
| **Preprocessing data** | • Missing value matrix [1] to analyse the randomness of the data<br>• Dropping columns with missing values [2]<br>• Splitting the dataset in feature variables (X) and target variables (Y)<br>• Randomise the dataset<br>• Converting the dataset to Numpy arrays | • Mean imputaton of missing values [3]<br>• Splitting the new dataset in feature variables (X) and target variables (Y)<br>• Randomise the new dataset<br>• Converting the new dataset to Numpy arrays | • Mean imputaton of missing values [3]<br>• Splitting the new dataset in feature variables (X) and target variables (Y)<br>• Randomise the new dataset<br>• Converting the new dataset to Numpy arrays | • One-hot encoding for categorical data<br>• Splitting the dataset in feature variables (X) and target variables (Y)<br>• Randomise the dataset<br>• Converting the dataset to Numpy arrays | • One-hot encoding for categorical data<br>• Splitting the dataset in feature variables (X) and target variables (Y)<br>• Randomise the dataset<br>• Converting the dataset to Numpy arrays | • One-hot encoding for categorical data<br>• Splitting the dataset in feature variables (X) and target variables (Y)<br>• Randomise the dataset<br>• Converting the dataset to Numpy arrays |
| **Feature selection** | N/A | N/A | N/A | • Performing an Exploratory Data Analysis (EDA):<br>• Scatterplots<br>• Correlation matrix<br>• Dropping features with cero correlation with the target. | • Performing an Exploratory Data Analysis (EDA):<br>• Scatterplots<br>• Correlation matrix<br>• Dropping features with cero correlation with the target. | • Performing an Exploratory Data Analysis (EDA):<br>• Scatterplots<br>• Correlation matrix<br>• Dropping features with cero correlation with the target. |
| **Building the model** | • sklearn library: DecisionTreeClassifier<br>• Splitting the dataset into training set (70%) and testing set (30%) [4] | • sklearn library: RandomForestClassifier<br>• Splitting the new dataset into training set (70%) and testing set (30%) [4] | • sklearn library: LogisticRegression<br>• Splitting the new dataset into training set (70%) and testing set (30%) [4] | • sklearn library: LinearRegression<br>• Splitting the dataset into training set (70%) and testing set (30%) [4] | • sklearn library: RandomForestRegressor<br>• Splitting the dataset into training set (70%) and testing set (30%) [4] | • sklearn library: Lasso<br>• Splitting the dataset into training set (70%) and testing set (30%) [4]<br>• Calculation of the best alpha parameter (GridSearchCV) |
| **Training the model** | • sklearn library: (model.fit) | * sklearn library: (model.fit) | • sklearn library: (model.fit) | • sklearn library: (model.fit) | • sklearn library: (model.fit) | • sklearn library: (model.fit) |
| **Evaluating the model** | • Calculation of the following metrics :<br>• Accuracy Score<br>• Precision Score<br>• Recall Score<br>• F1-Score<br>• Support Score<br>• Confusion Matrix<br>• Visualizing the predictions of the model given the testing set:<br>• Comparison table between "Real values" vs "Predicted values"<br>• Visualizing the Decision Tree: | • Calculation of the following metrics :<br>• Accuracy Score<br>• Precision Score<br>• Recall Score<br>• F1-Score<br>• Support Score<br>• Confusion Matrix<br>• Visualizing the predictions of the model given the testing set:<br>• Comparison table between "Real values" vs "Predicted values" | • Calculation of the following metrics :<br>• Accuracy Score<br>• Precision Score<br>• Recall Score<br>• F1-Score<br>• Support Score<br>• Confusion Matrix<br>• Visualizing the predictions of the model given the testing set:<br>• Comparison table between "Real values" vs "Predicted values" | • Calculation of the following metrics :<br>• Mean Absolute Error<br>• Mean Squared Error<br>• Root Mean Squared Error<br>• $R^2$ score<br>• Model accuracy<br>• Linear regression equation<br>• Visualizing the predictions of the model given the testing set:<br>• Bar chart ("Real Values vs "Predicted values"<br>• Comparison table between "Real values" vs "Predicted values"<br>• Visualizing the linear regression: | • Calculation of the following metrics :<br>• Mean Absolute Error<br>• Mean Squared Error<br>• Root Mean Squared Error<br>• $R^2$ score<br>• Model accuracy<br>• Visualizing the predictions of the model given the testing set:<br>• Bar chart ("Real Values vs "Predicted values"<br>• Comparison table between "Real values" vs "Predicted values" | • Calculation of the following metrics :<br>• Mean Absolute Error<br>• Mean Squared Error<br>• Root Mean Squared Error<br>• $R^2$ score<br>• Model accuracy<br>• Visualizing the predictions of the model given the testing set:<br>• Bar chart ("Real Values vs "Predicted values"<br>• Comparison table between "Real values" vs "Predicted values" |
| **Make predictions (Production set)** | • sklearn library: (model.fit) | • sklearn library: (model.fit) | • sklearn library: (model.fit)<br>• Preprocessing the Production set: CE802_P2_Test.csv<br>• Exporting predictions to the CSV file: CE802_P2_Test | • sklearn library: (model.fit)<br>• Preprocessing the Production set: CE802_P3_Test.csv<br>• Exporting predictions to the CSV file: CE802_P3_Test | • sklearn library: (model.fit) | • sklearn library: (model.fit) |

**Source:** Own elaboration in Excel to illustrate the methods used in each Machine Learning model for both comparative studies part 2 and part 3.

# 3. Results obtained

Table 2.0 shows the results of the performance measures of 3 ML algorithms capable of predicting whether the insured will complain or not. Similarly, table 2.1 displays the scores of various evaluation metrics[7] for 3 machine learning regression techniques .

[7] **Evaluation metrics** are "used to measure the performance of a machine learning model" (Kumar 2020).

- For a classification machine learning algorithm, the output of the model can be a target class label or probability score. The different evaluation metrics used for these two approaches can be found on table 2.0
- For a regression machine learning algorithm, the output of ML models is real-valued. Various metrics to compute the performance of regression models can be found on table 2.1.

Table 2.0

**Evaluation metrics for Classification models**

| Model | Accuracy Score | Precision Score (False) | Precision Score (True) | Recall Score (False) | Recall Score (True) | F1-Score (False) | F1-Score (True) | Support Score (False) | Support Score (True) |
|---|---|---|---|---|---|---|---|---|---|
| Decision Tree Classifier | 70.88% | 73.00% | 69.00% | 75.00% | 66.00% | 74.00% | 67.00% | 247 | 203 |
| Random Forest Classifier | 86.22% | 87.00% | 86.00% | 88.00% | 84.00% | 87.00% | 85.00% | 241 | 209 |
| Logistic Regression | 88.00% | 89.00% | 87.00% | 89.00% | 87.00% | 89.00% | 87.00% | 241 | 209 |

**Source:** Own elaboration in Excel to illustrate different performance measures obtained by 3 different machine learning classification algorithms

**Note**: It is important to mention that one dataset (after mean imputation of missing values) has been used to create the Random Forest Classifier and Logistic Regression, while another dataset (after dropping F15 feature) has been utilized to build the Decision Tree Classifier.

This is a significant remark, since It may be true that imputation techniques can have a positive impact in this model accuracy, while dropping features can cause a lower performance.

According to the results of table 2.0, the Logistic Regression model is considered the optimal ML classifier, since the algorithm got the highest scores in all the metrics.

To sum up, the Logistic Regression has classified **88%** of the clients correctly as customers who file or not file a claim, based on the historical data provided by the manager.

In other words, if the manager takes 100 random customers from the whole database, this model will be misclassifying only **12 clients**.

This not too bad if we consider that anything over 50% means the model is better than random. However, we should consider applying different methods in preprocessing phase, or even performing feature selection methods or hyperparameter tuning in order to improve the accuracy of the model.

Table 2.1

**Evaluation metrics for Regression models**

| Model | Mean Absolute Error (MAE) | Mean Squared Error (MSE) | Root Mean Squared Error (RMSE) | $R^2$ Score | R Score (Accuracy) |
|---|---|---|---|---|---|
| Linear Regression | 390.2992 | 246057.5619 | 496.0418 | 0.8022 | 0.7834 |
| Random Forest Regressor | 426.8305 | 357207.7960 | 597.6686 | 0.7128 | 0.9612 |
| Lasso Regression | 390.3037 | 246059.6093 | 496.0439 | 0.8022 | 0.7834 |

**Source:** Own elaboration in Excel to illustrate different performance measures obtained by 3 different machine learning regression algorithms.

Similarly in table 2.1, it can be observed that Linear Regression represents the best ML regressor, since the model obtained the lowest RMSE Score and the highest $R^2$ score.

To sum up, it can be referred that **80%** of the changeability of the dependent output attribute can be explained by the model, while the remaining **20%** of the variability is still unaccounted for. Additionally, the MAE score informed that on average, the predictions made by the model were 390.299 away from the true prediction, while the RMSE value indicated that the algorithm was not very accurate, since 496.02 is more than 10% of the mean value of column "Target" (92.208).

Table 2.1.1

| | Mean value of the column Target | RSME | 10% of Target mean |
|---|---|---|---|
| 0 | 922.082533 | 496.041895 | 92.208253 |

**Source:** Own elaboration in Python to illustrate the importance of RSME score.

All in all, as mentioned above, this model is not too bad if we consider that anything over 50% means the model is better than random. However, we should consider analyzing the factors that may have contributed to the inaccuracy of a linear regression, such as the data size, bad assumptions of linear relationship or poor features. (not high correlation to the values we are trying to predict).

Although the choice of an evaluation metric should be well aligned with the business objectives, everyone can come up with his/her own evaluation metric as well. As Agarwal (2019) mentions: "a correct choice of an evaluation metric is very essential for a model, but it is a bit subjective".

## 5. Conclusion

In this assignment, a comparative study of the mentioned Machine Learning procedures for classification and regression have been performed to predict if the insured will file a claim or not, and if he/she will complain, to forecast also the value of the claim.

## References


Agarwal, Rahul. 2019. "The 5 Classification Evaluation Metrics Every Data Scientist Must Know." *Towards Data Science*. Retrieved January 19, 2021 (https://towardsdatascience.com/the-5-classification-evaluation-metrics-you-must-know-aa97784ff226).

Albon, Chris. 2018. *Machine Learning with Python Cookbook: Practical Solutions from Preprocessing to Deep Learning*. Sebastopol, CA: O'Reilly Media.

Alpaydin, Ethem. 2014. *Introduction to Machine Learning*. Third edit. Cambridge, Massachusetts: The MIT Press.

Chakure, Afroz. 2018. "Random Forest Regression. Along with Its Implementation in Python." *Medium*. Retrieved January 19, 2021 (https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f).

Dangeti, P. 2017. *Statistics for Machine Learning: Build Supervised, Unsupervised and Reinforcement Learning Models Using Both Python and R*. Birmingham: Packt Publishing.

Kumar, Satyam. 2020. "14 Popular Evaluation Metrics in Machine Learning." *Towards Data Science*. Retrieved January 19, 2021 (https://towardsdatascience.com/14-popular-evaluation-metrics-in-machine-learning-33d9826434e4).

Yufeng, G. 2017. "The 7 Steps of Machine Learning." *Towards Data Science*. Retrieved January 19, 2021 (https://towardsdatascience.com/the-7-steps-of-machine-learning-2877d7e5548e).