COM6012 Scalable Machine Learning **Lida Petropoulou**
Assignment 2 Registration No: 180128240

Introduction

Alongside with this report, the following files have been attached:

- q1_1.py, q1_1.output, q1_1.sh: code, output and bash file for question 1 part 1.
- q1_23.py : contains the code for question 1 part 2 and 3.
- q2.py : contains the code for question 2
- q1_23_10cores.output, q1_23_10.sh: contain the output and bash file of q1_23.py ran with 10 cores.
- q1_23_20cores.output, q1_23_20.sh: contain the output and bash file of q1_23.py ran with 20 cores.
- q2.output, q2.sh: contain the output and bash file of q_2.py ran with 20 cores.

Question 1. Searching for exotic particles in high-energy physics using classic supervised learning algorithms.

1. Using pipelines and cross-validation the best configuration of parameters and the metrics for the three algorithms were found: Decision Trees for Classification, Decision Trees for Regression and Logistic Regression. The cross validation was used in a subset of the full dataset (25%) with 5 k-folds, and the metrics used were the accuracy and the area under the curve. The configurations that were tested, are the following (all the possible combinations):

- Decision Tree Classification: maxDepth: 5, 10 and impurity: gini, entropy
- Decision Tree Regression: maxDepth: 5, 10 and maxBins: 16, 32
- Logistic Regression: maxIter: 10, 20 and regParam: 0.0, 0.5

The best configurations are shown in the following Table 1.

Model	Parameters	Area under curve	Accuracy
Decision Tree Classification	maxDepth: 10, impurity: gini	0.7012	0.7028
Decision Tree Regression	maxDepth: 10, maxBins: 32	0.7014	0.7029
Logistic Regression	maxIter: 20, regParam: 0.0	0.6346	0.6410

Table 1: Best parameters and metric scores.

2. After finding the best parameters for the different algorithms the full dataset was trained. The dataset was split in training and test dataset with the ratio of 7:3. Moreover, the models were trained using 10 and 20 cores and the required times were 13mins 36secs and 12mins 27secs respectively (for all the three algorithms and the time needed for printing, and initiliasing the algorithms. The time does not contain the preprocessing of the data). The results are presented in the Table 2.

Model	Area under curve	Accuracy	Time 10c	Time 20c
Decision Tree Classification	0.7023	0.7039	10mins 18secs	10mins 16secs
Decision Tree Regression	0.7024	0.7042	0mins 47secs	0mins 46secs
Logistic Regression	0.6351	0.6413	1min 27secs	1min 22secs

Table 2: Metric scores for the full dataset.

3. After training the models, the three most relevant features were found for either classification or regression (depending on the method). We notice that the top 3 features were always the same, but they were in a different order, where c26 is m_bb, c27 is m_wbb, and c28 is m_wwbb. The results are shown in Table 3

Model	Top1	Top2	Top3
Decision Tree Classification	c26	c28	c27
Decision Tree Regression	c26	c28	c27
Logistic Regression	c28	c27	c26

Table 3: Most relevant features

Question 2. Senior Data Analyst at Intelligent Insurances Co.

1. For the preprocessing of the data, the following steps were followed. Firstly, as noticed the data contained a lot missing values. In order to handle this issue, the rows that contained at least one missing value were removed. After that the values were changed to doubles or integers depending on what they represented. Furthermore, a stringIndexer was used to convert the categorical values (i.e. strings) to numerical values, thus the predictive model could be designed. Also, as the data was unbalanced (i.e. there were too many zeros in the Claim_Amount column), the amount of rows that had zero to the Claim_Amount column was reduced to the same amount of nonzero Claim_Amount. This was done by sampling the zero_values.

2. For the predictions, two different predictive models were implemented. The dataset was split in two datasets, one for training and one for testing. For the first predictive, linear regression was used. The RMSE for the testing dataset was computed and it was the following:

Root Mean Squared Error (RMSE) on test data = 280.442

RMSE alone is meaningless until we compare with the actual Claim_Amount value, such as mean, min and max, which were 87.28, 0 and 11440.75 respectively. After such comparison, our RMSE looks pretty good.

The second predictive model was built with two separate models. Firstly a Decision Tree Classifier was implemented, were the data was binarized to only predict if the claim is zero or not. Afterwards a Generalized Linear Regression with gamma distribution was used. The RMSE value that was acquired with this predictive model was better since it was the following:

Root Mean Squared Error (RMSE) on test data = 261.97

Both implementations were pretty slow, i.e. almost 60 minutes. If the model was configured in an other way to fit the dataset better, probably the time would reduce a lot. However, due to lack of time I wasn't able to run cross-validation to find the best-parameters, thus the model was trying to find the best parameters on each own, and the required time to fit and train the data was much more than the expected. The results for both 10 and 20 cores were the same, and for this reason only one output and bash file is attached.