Task 3

As a final task, your supervisor assigned to you to investigate if it is possible to train a linear regression model that could predict the departure delay a flight may have by using, as input, its origin (column "ORIGIN"), its airways (column "CARRIER"), and its departure time (column "DEP_TIME"). Again you should use Python and DataFrames, this time with MLlib. You should pay attention to transform the string-based input features using the proper representation format, and you should explain your choices. Special attention should be given to the "DEP_TIME" column: your supervisor told you that you should only consider the corresponding hour of the day (which, of course, should be transformed in a one-hot representation). For your training and testing workflow you should first remove outliers (see Task 2) and then split your dataset into two parts, one that will be used for training reasons and it will contain 70% of the entries, and a second one containing the remaining entries and which will be used for the assessment of the model. No need to implement a more sophisticated evaluation process (e.g., based on k-fold) is required in this phase. Your code should (a) prepare the feature vectors, (b) prepare the training and testing datasets, (c) train the model, (d) print on the screen the first 10 predictions, i.e., pairs of feature vectors (in compact format) and predicted outputs, on the screen, and (e) evaluate the accuracy of the model and display the corresponding metric on the screen.

The report will be split in five sections according to the tasks we had to solve:

a. **Preparing the feature vectors**

To begin with, we first imported the wanted "pyspark" libraries and then we read the dataset from the csv file. We selected only the airports and the airways belonging to the 1% percentile of the whole dataset, we replaced the null values with 0 and then we created a final table with the wanted columns: airports, airways, departure time (as dep_time) and the positive values of the departure delays. We also used the negative values of the last-named column, because they represent the flights which begun prior to their schedule.

After that, we selected from the "departure time" column only the two first digits, which are referring to the hour. The records of this column were even in 3-digit format (when the hour was less than 10) or in 4-digit format (when it was above 10). So, we

had to make all the records 4-digits, select the first two numbers, and convert them to integers.

Next, we used the “One-Hot-Encoder” in order to map each different element of the columns: 'airports', 'airways' and 'dep_time' to a number. We created indexes, we did transformations and then through pipelines we got for each different element a certain number. We inserted the results in our data-frame and we now have a new column referring to index of each of the above columns and the “features” column that contains the one-hot-encoding. An example of this encoding is:

If we have **(70,[19,31,54],[1...** value on features it means that this line contains 70 0s, the 20^{th} – 32^{nd} and 55^{th} columns (because they start from 0) consist of non-zero values. The **[1...** shows in abbreviation which column there are non-zero elements.

b. Preparing the training and test datasets

For the training and test sets we dropped the columns referred to indexes from the above encoding, keeping only the “features” column which will be used later on as independent variable. Then we split the data into training set, which contained the 70% of the records after the cleaning and the encoding and the rest as the test set, which we are going to use for prediction later.

c. Train the model

We created a linear regression model with “pyspark mlib” library and then we fitted the training set into it. The independent variable was the “features” column and the dependent was the “delay” column.

d. Select the 10 predictions

We fitted the test set in the above model in order to predict departure delays. We can see 10 predictions in the below table:

prediction	delay	airports	airways	dep_time
-0.8605078631363856	0.0	ATL	UA	0
96.906478688287 27	403.0	ATL	UA	2
77.469262996485 45	242.0	ATL	UA	3
77.469262996485 45	243.0	ATL	UA	3
77.469262996485 45	273.0	ATL	UA	3
-0.15742665380191845	-8.0	ATL	UA	5
-0.15742665380191845	-6.0	ATL	UA	5
-0.15742665380191845	-4.0	ATL	UA	5
-0.15742665380191845	-3.0	ATL	UA	5
-0.15742665380191845	-2.0	ATL	UA	5

only showing top 10 rows

e. Evaluation of the model's accuracy

Finally, we evaluated the model's accuracy and goodness of fit. We can observe from our code that as far as the model with the training set is concerned the R^2 is nearly 5.2% and the Root Mean Squared Error (RMSE) is 45.3. It is worth to mention that RMSE shows the differences between predicted values from a model and observed values and the R2 shows how the proportion of the independent variables' variance explained by the independent variable. In this case, one measure indicates somehow good fitting (RMSE: 45.3 and SD: 46.53, are very near to each other) and the other very bad fitting (R2 has very low value). This is also confirmed when we fit the test set to assess the above model. The goodness of fit in this case are R2: 5.3% and RMSE: 45.33 with SD (standard deviation): 46.59. So, taking into account the very bad predictions from the above picture we can conclude that this model is not a good predictor for future observations.

This model is very bad for many reasons. First of all, the two datasets (training and test) are probably wrong, because they are selected from one iteration only. In order to be more accurate, we should select a k-fold cross validation, from which we will select after many tests the most proper datasets. Moreover, it would be a good idea and we will get better results if we standardize (scaled) the dataset in order all of the variables to be measured in the same scale. Finally, we are not sure if the linear regression model is the most appropriate for this dataset and more investigation should be done to decide whether it is the best method.