Assignment #5

Antonio Mallia
CS6923 Machine Learning, Spring 2018

May 11, 2018

1 Problem definition

The goal of this project is to predict the time delay (in minutes) of a flight based on some features recorded about the flight. This is a regression problem and so we need to predict a quantity which in this case is going to be an integer representing the delay in minutes. Some of the possible algorithm we can use to solve a regression problem are the following: linear regression, polynomial regression, neural network, decision trees, random forests, k-NN.

We decided to focus on three algorithms: **linear regression**, **neural network** and **random forest**. Linear regression will be our simple baseline, while the others two represent the most promising ones to build on top of it.

We are provided with a training set with a feature list of dimension equal to 20, a unique *UID* attribute for each row and a label column containing the delays recorded for the given sample.

2 Feature engineering

2.1 Dropped features

As we might expect some feature are **irrelevant** and do not contain any useful information for discriminating outcome.

Obviously, *UID* is just an additional attributed used for convenience to identify a sample of the dataset, so it will definitely not be included.

Some other features may result to be **redundant**, so highly correlated features end up containing duplicate information. For example the tuples (*ORIGIN*, *ORIGIN_CITY_NAME*, *ORIGIN_STATE_ABR*) and (*DEST*, *DEST_CITY_NAME*, *DEST_STATE_ABR*) are extremely redundant, i.e. *ORIGIN* and *ORIGIN_CITY_NAME* the former implies the latter since from the origin airport con the city can be inferred. The same things applies to *ORIGIN_CITY_NAME* and *ORIGIN_STATE_ABR* since the two have a direct dependency.

Also these categorical features have **high cardinality**, for example *ORIG* and *DEST* have 50 distinct values which would make encoding one-hot impractical.

Attribute *FIRST_DEP_TIME* is missing a value for most of the examples, so we decided to remove the entire column considering that.

Another reason to remove a feature is when two or more are **combined** into a new one. This as been done for *ACTUAL_ELAPSED_TIME* and *DISTANCE*. Finally

Feature	Reason for dropping
UID	Unique value, no information provided
CRS_DEP_TIME	Extract hour into *FL_HH*
ACTUAL_ELAPSED_TIME	Combined with *DISTANCE* into a new feature
DISTANCE	Combined with *ACTUAL_ELAPSED_TIME* into a new feature
FL_DATE	Split into *FL_DD* and *FL_MM*
ORIGIN_CITY_MARKET_ID	Highly correlated to *ORIGIN*
DEST_CITY_MARKET_ID	Highly correlated to *DEST*
FL_NUM	High cardinality
UNIQUE_CARRIER	Highly correlated to *AIRLINE_ID*
ORIGIN	Highly correlated to *ORIGIN_CITY_NAME*
ORIGIN_CITY_NAME	Highly correlated to *ORIGIN_STATE_ABR*
ORIGIN_STATE_ABR	High cardinality
DEST	Highly correlated to *DEST_CITY_NAME*
DEST_CITY_NAME	Highly correlated to *DEST_STATE_ABR*
DEST_STATE_ABR	High cardinality
FIRST_DEP_TIME	Missing a value for most of the examples

2.2 Created features

Creating new features can sometimes capture the important information in a dataset much more effectively than the original features. There are several techniques that can be applied as combine existing features or intersect existing features with external data.

For example, we are provided with the distance between the origin and destination airport and the actual elapsed time of flight. Intuitively, we can calculate the average speed of the flight and so **combine** those two feature into a new one.

We decided to intersect the date column with a dataset of US holidays to produce a new **binary attribute** named *IS_HOLIDAY* which tells if the day of the flight was a holiday or not. We believe this information might influence the prediction quality of the system.

The attribute *FL_DATE* which represents the departure time of the flight is **split** into *FL_DD* and *FL_MM*.

Feature	Description
SPEED	Ratio of *ACTUAL_ELAPSED_TIME* to *DISTANCE*
IS_HOLIDAY	If day was a holiday
IS_WEEKEND	If day was weekend
PART_OF_THE_DAY	Part of the day based on the hour of the flight
FL_MM	The month of the date
FL_DD	The day of the date
FL_HH	The hour of the time

2.3 Feature encoding

The training and test sets contains several categorical features, so they must be converted to a numerical form. To do so we have adopted two different options: **integer** encoding and **one-hot encoding**.

Integer encoding is good enough for features whose a natural ordered relationship between each other is present. As an example, *AIRLINE_ID* identifies a unique carrier and so there is not a specific ordering. In this case, also in consideration of the limited number of distinct values, it makes sense to adopt a one-hot encoding.

3 Cross-Validation

We decided to perform k-fold cross-validation on the dataset. We run the models varying k to find the best value for each of them.

k	5	10	15	20
Linear regression	1928	**1924**	1926	1925
Neural network	1934	**1928**	1930	1928
Random forest	2319	**2261**	2288	2316

According to our experiments the k the minimizes the validation error is always equal to **10**.

4 Grid Search

We performed a grid search over specified parameter values for each of the models. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid. For space reasons we have included detailed results of the grid search in the appendix.

Linear regression

Parameter	Description	Values
fit_intercept	Whether to calculate the intercept for this model	[True, False]
normalize	X will be normalized before regression	[True, False]

Best parameters found on dev test are the following:

```
{
 'fit_intercept': True,
  'normalize': True
}
```

The error with those parameters is equal to **1923**.

Neural networks

Parameter	Description	Values
hidden_layer_sizes	The number of neurons in the hidden layer	[5, 10, 20, 50]
learning_rate	Learning rate schedule for weight updates	['constant', 'adaptive']
learning_rate_init	The initial learning rate used	[0.001, 0.01, 0.1]
alpha	L2 penalty (regularization term) parameter	[0.001, 0.01, 0.1]

Best parameters found on dev test are the following:

```
{
 'alpha': 0.001,
 'hidden_layer_sizes': 50,
 'learning_rate': 'constant',
 'learning_rate_init': 0.001
}
```

The error with those parameters is equal to **1922**.

Random forest

Parameter	Description	Values
n_estimators	The number of trees in the forest	[10, 15, 20]
max_features	The number of features to consider when splitting	['auto', 'sqrt']
max_depth	The maximum depth of the tree	[10, 20, 30, 40, 50, None]

Best parameters found on dev test are the following:

```
{
 'max_depth': 10,
 'max_features': 'sqrt',
 'n_estimators': 20
}
```

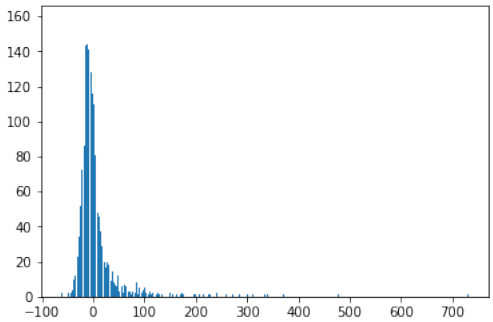
The error with those parameters is equal to **1976**.

In conclusion, **linear regression** seams good enough for this task and our predictions will be based on this model.

5 Data cleaning

The following is the distribution of the ARR_DELAY label. As we can see, most of the delays are within $\pm 2hrs$, with very few examples having longer delays. This unbalance can produce a regressor which tries to minimize the error for all the samples in training, which might end up in having a model that does slightly worst on the common samples, so the short-delayed flights.

```
count    4911.000000
mean        4.316229
std        45.386657
min       -61.000000
25%       -14.000000
50%        -6.000000
75%         7.000000
max       731.000000
Name: ARR_DELAY, dtype: float64
```



We remove those samples from the training set with the intention to maximize the accuracy on the data that is most relevant. This decision was taken also because after running a prediction for the test set we noticed that the predicted values where all within the 2 hours delay and so it makes definitely sense to try to optimize further for this codomain. We have created two different prediction files *prediction.csv* and *prediction_opt.csv* [1]. The former refers to the one produced using the model that has been trained to the entire training set, the latter, on the other hand, has been generated by performing prediction using a model trained on filtered data.

[1] Please consider it as an experimental attempt to improve the final predictions, but do not consider for assignment evaluation as limited analysis of this technique has been performed.