Flight Delay Prediction For Aviation Industry Using Machine Learning

Performance Testing :

Performance Testing For the ML the workflow was pretty straight forward by starting defining the target, which was the FLIGHT_STATUS, and then dropping it alongside the DEP_DELAY (for the first set of models only) from the dataframe to define X (features). With this done, I split the data with a 25 and 75% for the test and training set respectively and used a typical random_State of 42. Again, all the code is available separated by models tested here. Then this was followed by these steps:

- Building a Regular Tree as Baseline
- Created Bagged Trees
- Ran a Random Forest with no Class Weighting (ran a feature importance plot as a QC tool)
- Random Forest with Bootstrat Class Weighting
- Ran a AdaBoost with and without the DEP_DELAY
- RAN Gradient Boosted Trees with and without the DEP_DELAY
- RAN XGBoost with and without the DEP_DELAY

And every model went through a performance evaluation, but as you will see the highest accuracy achieved wasn’t great at 70% when the DEP_DELAY (departure delay) was dropped, without dropping the DEP_DELAY the highest accuracy obtained was 86%, so a hard 16% higher with only one feature that suggest a late arrival. Now you can get the picture of how much the accuracy can improve if I add the rest of these predictive features, certainly it will increase above the 90%.

Figure 4 is a summary of the models evaluated with and without balancing the data, and with and without the DEP_DELAY feature. It is quite obvious the increase in performance when you add a real predictive feature such as the DEP_DELAY (departure delay). I have highlighted in green the XGBoost which outperform the other models, which also as you can see, didn't use any feature that could biased the model towards a predicted delay.

		Inbalanced data			balanced data		
	Algorithm	Precision	Recall	Accuracy	Precision	Recall	Accuracy
1	Baseline Tree (without DEP_DELAY)	64.00	51.00	63.25	57.00	57.00	57.76
2	Bagged Tress (without DEP_DEPAY(	66.00	51.00	63.33	61.00	52.00	63.68
3	Random Forest (without DEP_DELAY)	74.00	50.00	62.89	57.00	57.00	56.76
4	Random with Bootstrat Weighting	57.00	57.00	56.67	57.00	57.00	57.25
5	AdaBoost_V1 (with DEP_DELAY)	84.00	82.00	81.72			
6	AdaBoost_V2 (without DEP_DELAY)	65.00	54.00	64.64			
7	Gradient Boosted Trees (with DEP_DELAY)	85.00	79.00	83.09			
8	Gradient Boosted Trees (without DEP_DELAY)	70.00	57.00	66.84			
9	XGBoost (with DEP_DELAY)	88.00	82.00	86.00	87.00	83.00	85.65
10	XGBoost (without DEP_DELAY)	71.00	61.00	69.37	69.00	63.00	69.68

Figure 4. ML model Evaluation Summary

The Classification Report for the XGBoost model selected as the best performant can be seen on Figure 5. This model ended up with an accuracy of 70% but very low Recall and Precision for the category 1. Let's keep on doing tests and see if that can be improved with the Deep Neural Networks.

	precision	recall	f1-score	support
0	0.70	0.90	0.79	1889408
1	0.67	0.36	0.47	1116784
accuracy			0.70	3006192
macro avg	0.69	0.63	0.63	3006192
weighted avg	0.69	0.70	0.67	3006192

Figure 5. XGBoost Classification report from the model with the best performance

It can be seen that during the prediction, the one with the best performance among the seven algorithms is the Decision Tree model. For example, the accuracy value for the Decision Tree is 0.9778. This value is significantly higher than that of Gradient Boosted Tree, with the second-greatest value of 0.9334 accuracy. Similar patterns of noticeable differences for performance scores of Decision Tree can also be seen in the other three measures. Besides the Decision Tree, the two tree-based ensemble classifiers Random Forest and Gradient Boosted Tree, are also better performed than others. The values of measures of these two algorithms are relatively similar. The difference between their performance scores and the other four algorithms is also significant.

The lowest scores for the four measures occur in two of the seven models, and they are both base classifiers. KNN model has the lowest accuracy and recall, while Gaussian Naïve Bayes has the lowest precision and f1-score. Meanwhile, the precision value of KNN is particularly low, which is only 0.7501. KNN also has the second-smallest value of accuracy and recall, while the second-smallest F1-Score belongs to the Logistic Regression model instead of the Gaussian Naïve Bayes model. Therefore, it is possible to conclude that the worst-performing algorithm among the seven selected models when predicting the given data set is the KNN model.

$H(s) = -\sum p(c) log p(c) c \in C$

$H(s) = 1 - \sum p(c) 2 c \in C$

Logistic Regression: Logistic regression an algorithm that performs classification using,

$h\theta x) = g(\theta Tx) = 1 + 1e{-}\theta Tx$W