Team R2D2

MODEL APPROACH

1.Data Preparation

Since the complete Train data has ~85L entries, we took a stratified sample of 14 L entries for model training. But, bivariate analysis of existing features and new feature creation has been done using entire data (~85L)

Since the data looked healthy will no NULL values, we straight away started Feature engineering.

2.Feature Engineering

Before creating any new features, we used s subset of variables given to build basic models using different techniques – Random forest, logistic regression, gradient boosting & deep Learning etc and found that gradient boosting model trained better with this data using rudimentary features.

Since gradient boosting algorithms are less prone to multi collinearity, our idea was to create as many features as possible and even overload some features which are most important in the model and train the model with very low learning rate to negate this side effects like overfitting and correlation amongst features.

As one can see in the code, we created new numerical features using large factor variables (example: numeric variables like SOURCE_DEST_COUNT, perc_delay_source_dest as defined below are derived from large factor variables like ROUTE (~4800 categories (distinct routes))

Derived Features:

1. ***Speed of the flight***: Distance between the airports/elapsed time
2. ***ROUTE***: Unique combination of Origin and Destination Airports (Note that we have considered forward (JFK – LAS) and return (LAS – JFK) are two different routes)
3. ***SOURCE_DEST_COUNT***: Historically the number of flights flown in given particular route. More the count of flight ⇔ famous/busier route ⇔ More passenger demand
4. ***perc_delay_source_dest***: Historically Some routes may be more prone to delay than others (this we can directly obtain from bivariate analysis using target variable). It would be logic to assume that the routes which have higher delay rate are more prone to delay in the future also.

   Sample Routes and respective %past delays & Counts

Route	%Delay	Dest_Source_Counts
ABEATL	0.1618677	1285
ABEDTW	0.1608187	1026
ABEORD	**0.2611765**	425
ABIDFW	0.2102590	3938
ABQATL	**0.1102236**	1252
ABQBWI	0.2339109	808

As one can see the route ABQ-ATL has an average delay of 11% where as that of ABI-DFW is 21%

5. ***perc_delay_tail***: With the same logic as above some tail numbers are more prone to delay historically as compared to others.
6. ***TAIL_COUNT***: Number of journeys a particular TAIL_NUM has made in entire 1.5 year span. Similarly for other categorical variables, historical percentage delay variables are created
7. ***DEP_TIME_BLK_perc_delay***: Historically, Some departure times are more busier than others
8. ***ARR_TIME_BLK_perc_delay***: Historically ,Some arrival times are more busier than others
9. ***DAY_OF_MONTH_perc_delay***: Some DAYS_OF_MONTH are more prone to delay than other
10. ***DAY_OF_WEEK_perc_delay***: some DAYS_OF_WEEK are more prone to delay than other

Day of week	***%Delay***
1	0.226612
2	0.2079532
3	0.2085981
4	**0.2331634**
5	0.2270863
6	**0.1784214**
7	0.2061169

11. ***Carrier_perc_delay*** : Historically, Some carriers are more busier than others

Day of week	***%Delay***
AA	0.222278
AS	0.129919
B6	0.226998
DL	0.152878
EV	0.243189
F9	0.274653
FL	0.173713
HA	0.090501
MQ	0.27178
NK	0.324582
OO	0.205231
UA	0.229749
US	0.181483
VX	0.183356
WN	0.238797

12. ***Origin/Destination Traffic (Variables named N.x,N.y in the model) :*** Number of flights dep/arriving in the same time block in the same airport gives a rough estimate of airport traffic (runway availability? may be! :D)

13. ***Repair Time:*** Variable for repair / maintenance time for a flight which just landed in an airport and has to depart to another airport in few hours/minutes. It intuitively makes sense as the more time the crew has for maintenance before next journey the less chance of delay even in the case where previous journey got delayed.
    # Logic used:
    Repair Time or Time available for a flight to get ready before its next journey = (Departure time of the flight - Arrival Time of the same flight (same tail No) before this in the same day)

    Hence we created a lag variable to subtract Arrival time of previous flight and dep time of this flight

14. ***Tail_rank***: Variable for 'n'th journey of a flight (with a particluar tail number) on the same day.
    There is clearly a difference between a flight which has 10 journeys a day and a flight which has only one journey a day.
    For the flight with 10 journeys, the delay in the first few journeys will get compounded and will result in a larger delay in the last journeys of the day
    # Logic used:
    In order to find which journey of the day the flight is currently in, we used a simple rank function partitioned by tail_num, fl_date and ordered by dep time of each journey

Logically combining Some of the derived features to get new features

15. ***Multiplied Delays***

Consider the following scenario,

a) **Tail No**: N0EGMQ - **34.7%** past average delay (perc_delay_source_dest)
b) **Tail No**: N028AA - **16.3%** past average delay
c) **Arrival Time block**: '0700-0759' - **9.6%** past average delay (ARR_TIME_BLK_perc_delay)
d) **Arrival Time block**: '2000-2059' - **29%** past average delay

So, if the flight with tail number N0EGMQ (34.7% past average delay) has an arrival time in between 20:00 - 20:59 (29% past average delay), ***the probability of flight getting delayed seems logically more*** as compared to a flight with tail number N028AA (16.3% past average delay) has an arrival time in between 07:00-07:59 (only 9.6% past average delay)

34.7 * 29 >> 16.3 * 9.6 <==> more delay in first case

In order to capture this pattern, we created a series of multiplied delay variables as shown below,

i) ***Multiplied_delay*** = 100000* perc_delay_source_dest* perc_delay_tail* DEP_TIME_BLK_perc_delay* ARR_TIME_BLK_ perc_delay* DAY_OF_MONTH_perc_delay* DAY_OF_WEEK_perc_delay* Carrier_perc_delay
ii) ***Multiplied_delay_1*** = 1000* perc_delay_source_dest* perc_delay_tail* Carrier_perc_delay
iii) ***Multiplied_delay_2*** = 100* ARR_TIME_BLK_perc_delay* DEP_TIME_BLK_perc_delay
iv) ***Multiplied_delay_3*** = 100*DAY_OF_MONTH_perc_delay* DAY_OF_WEEK_perc_delay

we believe trying out different other logical combinations of multiplied delay can improve model further.

3.Modelling Assumptions:

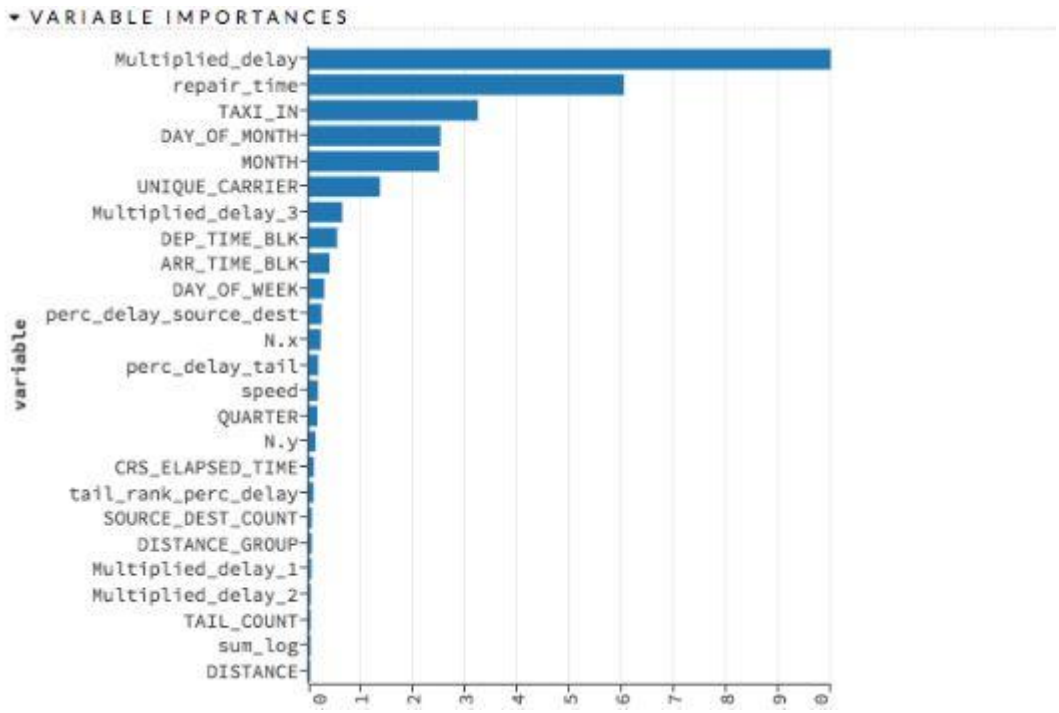
1.The 14 Lakh stratified sample which we took from the entire base to train the model represents entire base

4.MODEL / Variable Importance

As mentioned earlier gradient boosting algorithm performed best on this particular data. Ensemble of different models might improve the performance but it comes at a cost of more computation and eventually more run time

Model is trained with a lower learning rate of 0.01 for around 2400 rounds.

The variable importance is shown below,



Since the availability of weather data is limited to just one month, we tried building a model including weather related variables like temperature, visibility, event and scored July month test data (~3.4 lakh out of 14.2 lakh). We tried to ensemble this score with the non-weather model score which improved the leader board score, but this would require building two different models which would increase the run time.

5.Applicability of model for future flights:

- Since all the new variables used in the model are derived from the historical patterns in the data, they are available way before the actual flight date.
- Moreover, the model evolves itself whenever every time it gets new data.
  For example: A Particular carrier which had high delay rate in the past has changed its management to counteract it. As a result the recent flight delays have decreased drastically. The model captures this change when calculating carrier_perc_delay variable.
- The model will further improve by adding useful weather variables are included.

6. Insights and recommendation for application of the model

- From the variable importance, we can obtain certain variables and their respective effect of the model. We can even obtain a combination of routes, departure times etc where delay is high/low for respective optimal resource allocation.
- Model gives high importance to ***repair-time*** *(flight maintenance time).* It is observed that more the time for maintenance, the less is the chance of delay. Hence either maintenance resources can be optimised according to the time available or the flight schedule itself can be optimised to give enough time for flight maintenance.