PREDICT FUEL FLOW RATES OF AIRPLANES

A PROJECT REPORT

Submitted in partial fulfilment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY (B.Tech)

In

Computer & Communication Engineering

by

MEHREET SINGH BAJAJ

159103038

MANIPAL UNIVERSITY JAIPUR

INSPIRED BY LIFE

Department of Computer & Communication Engineering,
School of Computing and IT,
Manipal University Jaipur,
MAY, 2019

School of Computing and IT,

DECLARATION

I hereby declare that the project entitled " **PREDICT FUEL FLOW RATES OF AIRPLANES** " submitted as part of the partial course requirements for the course ***Major Project (CC1881)***, for the award of the degree of Bachelor of Technology in Computer & Communication Engineering at Manipal University Jaipur during the ***VIIIth*** semester, has been carried out by me. I declare that the project has not formed the basis for the award of any degree, associate ship, fellowship or any other similar titles elsewhere.

Further, I declare that I will not share, re-submit or publish the code, idea, framework and/or any publication that may arise out of this work for academic or profit purposes without obtaining the prior written consent of the Course Faculty Mentor and Course Instructor.

Signature of the Student:

Place: Manipal University, Jaipur

Date: 14.05.2019

DEPARTMENT OF COMPUTER AND COMMUNICATION

Manipal University Jaipur, 303007(Rajasthan), India

May, 2019

14.05.2019

CERTIFICATE

This is to certify that the project entitled " **PREDICT FUEL FLOW RATES OF AIRPLANES**" is a record of bonafide work carried out as part of the course ***major project (CC1881)*** , under my guidance by ***Mehreet Singh Bajaj (159103038)*** , during the academic semester ***VIII*** th , in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer & Communication Engineering, at Manipal University Jaipur during academic year 2018-2019.

DR. VIJAYPAL SINGH DHAKA
Project Guide, Computer and Communication
Manipal University Jaipur

DR. VIJAYPAL SINGH DHAKA
Head Of Department , Computer and Communication
Manipal University Jaipur

ACKNOWLEDGEMENT

I wish to express my deep sense of gratitude to my guide Dr. Vijaypal Singh Dhaka, Head Of Department, Computer & Communication Engineering, School of Computing and IT, Manipal University Jaipur for his guidance and useful suggestions, for keep on reminding us about the deadlines and encouraging us to complete our work on time, without which it would've been impossible to make this project, that too in time. I'm really thankful to him for believing in me and my project idea and helping me in shaping it up even better than I thought I could.

I feel blessed to have parents Dr Satinderjit Singh Bajaj and Dr Jagminder Kaur Bajaj . It was due to the intellectual discussion that they had with me all the time that always motivated me and inspired me to do something , and i could come up with the willingness to do this project.

I would also like to thank my cousin Preet Kanwal Singh who would keep me updated with the project requirements and helping me out whenever I was stuck in a problem.

I would also like to mention my friend Astha Gupta and younger brother Prabh Arjun Singh Bajaj's little inputs about suggesting good modifications to the documents to make it look more appealing.

MEHREET SINGH BAJAJ (159103038)

Bachelor of Technology (B-Tech)

Computer and communication (8th semester)

ABSTRACT

Fuel is of utmost importance as it is one of the non renewable resource and using it in a efficient and smart way is highly required in today's date. This is the most common problem faced by the airline industry these days .Fuel constitutes around 30% of the operating cost of airlines due to which we have higher cost of tickets . Developing cost saving strategies especially on fuel is of prime importance to airlines and reducing the emission of staggering amounts of greenhouse gases along with reducing fuel intake can have a significant positive impact on the environment .Given an idea of the utility of the resources used by my work so that no extra resources are used and may go in vain

Now the problem statement is that

The ability to predict the Fuel Flow (FF) rate of airplanes during different phases of a flight (Taxi, Takeoff, Climb, Cruise, Approach, and Rollout) will help understand

- ❖ The significant drivers of FF rate for each of these phases and also help understand the factors that make the airplanes perform at higher levels of fuel efficiency during the different phases of a flight.
- ❖ Insights from the exercise can help derive the best practices, which make flights more fuel efficient under different conditions.

After the project it was observed that climb, cruise and approach are most important phases for optimizing fuel and the consumption. The most important features consisted of rate of change of altitude, longitudinal acceleration and ground speed. The clearest visible trend between predictors and fuel flow rate is in the climb phase. In other phases, some of the predictors have a weakly visible trend, but since the root mean squared error is small, it is assumed that the features have strong nonlinear interactions which are not clearly visible in simple plots.

All of the work was done in Jupyter Notebook using Python and with the concepts of supervised Machine Learning with various data cleaning techniques from tidying up the data to applying different models like Random Forests and eXtreme Gradient Boosting to get the best suitable results and then validation and checking performance using Root Mean Square error (rmse) method.

TABLE OF CONTENTS

Page No.

LIST OF FIGURES

INTRODUCTION

1.1 Basic Introduction

In today’s fast growing world in which we are surrounded by problems which can be solved easily using the knowledge of computing science, we discovered one of the common problem which has been neglected from a long time and it really needs a solution. Fuel Management is one of the basic problems which an individual faces in every day to day life. In the busy hustle bustle of life one does not notices that fuel plays an important role , majorly for travelling and as we know the most expensive means of travelling is via Airplanes. So we will try to understand why flight tickets are so expensive and how fuel is the major factor directly related to the cost . Next, we will also come to know how fuel emission through airplane emits greenhouse gases and how we can use the fuel efficiently to produce less of those harmful gases and in a way providing a good solution to the problem which is environment friendly.

Fuel flow rate of an airplane is the key component in deciding the aircraft’s engine performance. Also, the amount of gas it emits , decides the fuel used and directly relates to the cost . That means of the total cost which airline handles, 30 % of it is just the cost of fuel. we can imagine how much fuel the plane consumes for one round , which can be reduced if used efficiently. So developing cost saving strategies especially on fuel is of prime importance to airlines and reducing the emission of staggering amounts of greenhouse gases along with reducing fuel intake can have a significant positive impact on the environment.
we will solve this problem with the concepts of Data Mining and Machine Learning and train our machine with the flight record data to see how the target value (Fuel Flow (FF)) rate varies during different phases of flight, from where we will come to know about the main drivers which have a significant effect on fuel flow during the flight . This later on can help us derive the best practices which makes flight more fuel efficient during different phases of flight. So our main work is to derive the top features from these phases and later on we can do Exploratory Analysis on it.

1.2 Scope of the Work

There are relatively few studies which have taken into account operational flight data to model fuel emission in planes. The integration of various types of operational data has been shown to improve the estimates of aircraft fuel consumption . Uncertainties in fuel flow rate values can be estimated by the Data driven models of engine fuel flow rate . With all this , in our project Fuel Flow rate of airplane is modelled with the help of operational data from our Flight Data Records (FDR). As all the readings including the different parameters for aircraft and engine are recorded with the help of sensors during the flight, So FDR provides a reliable real flight data . Due to random variations in the parameters and phases of flight our model is **stochastic** and not deterministic as a result of the manufacturing , flow turbulence, component deterioration and other internal errors and other external ambient disturbances . "Predicting fuel Flow of Airplanes" focuses mainly on finding the **top features** in different phases of fight responsible for high consumption of fuel during that phases and predicting the Fuel Flow rate in it . These top features will help us understand the nature of the flight . As most of the flight flies in between the same phases . In this project, machine learning principles are applied to airplane flight recorder data to model the fuel flow rate (FF) as a function of the aircraft altitude, ground speed, vertical speed, and takeoff mass in the airborne phases of flight. So overall if we come to know about these top features we can later do an exploratory analysis over it and derive best practices saving the fuel and benefitting airline companies in many ways .

1.3 Purpose and Motivation

Currently there is not much work done with this field using the concepts of Machine Learning and Data Science . So working on this field will open up broadness to this problem as fuel is very essential resource and there will be a time in future where there will be a scarcity of this resource . So this is the need of an hour to develop some strategy to save and use fuel efficiently , so that no resources are wasted . With our model we will be able to develop cost saving fuel strategies in airplanes . We will come out with the top features in airplane that majorly consume the fuel and how to modify our methods to save fuel at the end.
We will build a statistical model using the machine learning techniques for aircraft engine data . All the variables used in our model are continuous and metric , our machine learning problem is basically a regression problem. We are using two models to solve our issue :

A. **RANDOM FOREST MODEL (using extra Tree Regressor method)**
B. **BOOSTING(XGboost - eXtreme Gradient Boosting)**

These algorithms have many widespread applications and helps to conclude to a better result than others. XGboosting have been used in many competitions as it give better results and very less error values, when compared to other models.

REQUIREMENT ANALYSIS

2.1 Functional Requirements

Providing the details of the project, all requirements and specifications will be made. Some of the functionalities we are going to use in our project are the basic principles of Data science along with Python . Supervised data Modelling techniques from Machine Learning and Visualizations using matplotlib and seaborn. Most of the important libraries of python have been implemented in our code in the best possible way to enhance the quality of our project.

All the work has been done in the ipython environment of Jupyter Notebook using base language as python.This provides a better work environment and faster run time than other IDE available out there .

2.2 Dataset

Our dataset consists of uncompressed data of about 14.4 GB of Flight record data (FDR) from 1005 different flight instances which represents readings of numerous sensors on aircraft and other engineering parameters. Every second measurements were done each of which contained 227 parameters. Flight data consisted of measurements during all the **8 phases** (unknown, preflight, taxi, takeoff, climb, cruise, descent and roll out) .

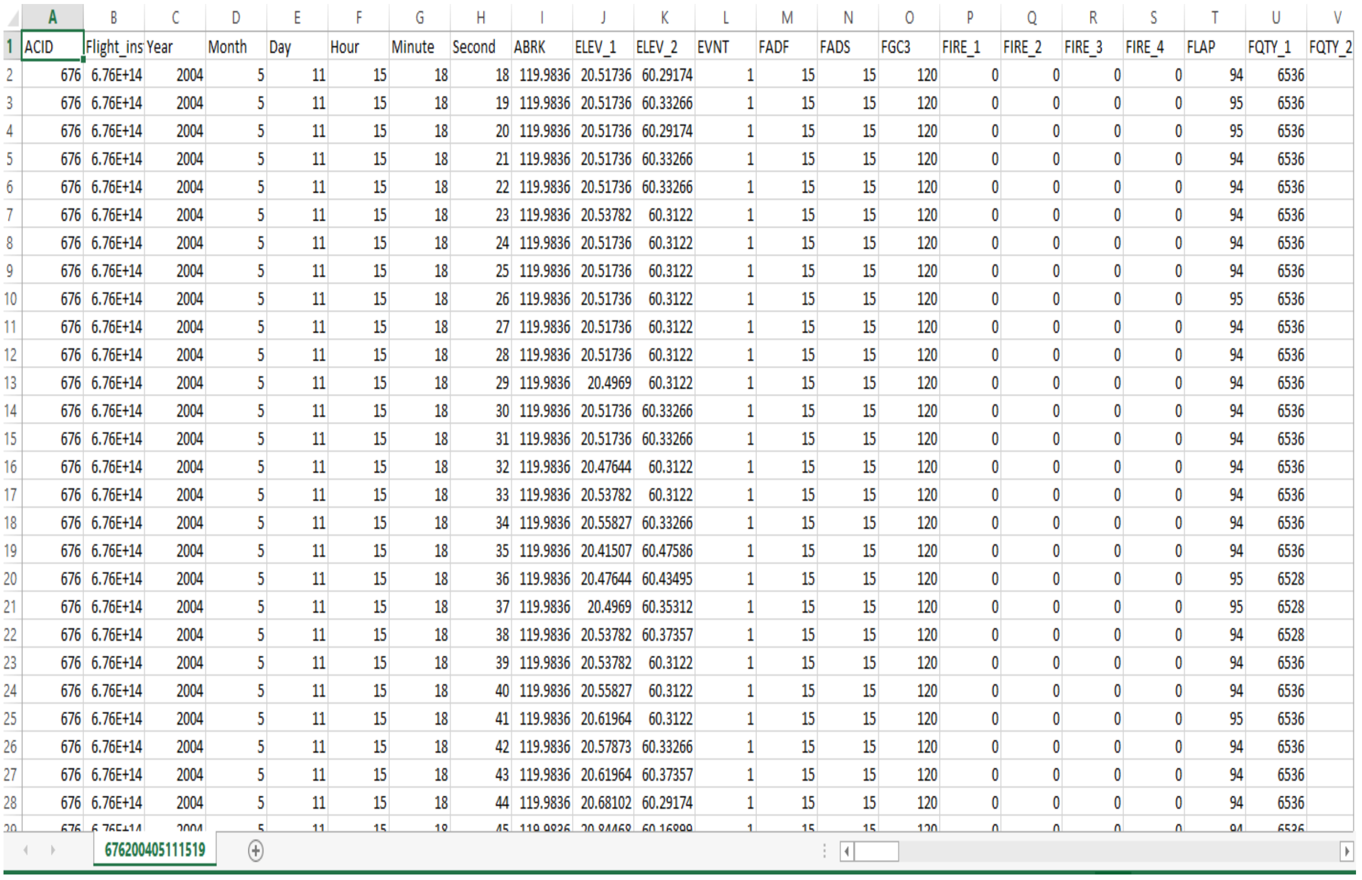
The dataset has been divided into training and testing set .The data distribution across training and test data sets is as follows:

- **Training dataset: 60%**
- **Test set: 40%**

There are 5 training data files each containing 200 csv files with record data for training and 1 testing file for validation of our model.

2.3 A Look At Data

This is what our dataset look like containing **227 variables(columns)** and **4365 rows** . This is just one file. There are **200 CSV** files in one training file and there are **5 training files** each containing 200 files.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	ACID	Flight_ins	Year	Month	Day	Hour	Minute	Second	ABRK	ELEV_1	ELEV_2	EVNT	FADF	FADS	FGC3	FIRE_1	FIRE_2	FIRE_3	FIRE_4	FLAP	FQTY_1	FQTY_2
2	676	6.76E+14	2004	5	11	15	18	18	119.9836	20.51736	60.29174	1	15	15	120	0	0	0	0	94	6536	
3	676	6.76E+14	2004	5	11	15	18	19	119.9836	20.51736	60.33266	1	15	15	120	0	0	0	0	95	6536	
4	676	6.76E+14	2004	5	11	15	18	20	119.9836	20.51736	60.29174	1	15	15	120	0	0	0	0	95	6536	
5	676	6.76E+14	2004	5	11	15	18	21	119.9836	20.51736	60.33266	1	15	15	120	0	0	0	0	94	6536	
6	676	6.76E+14	2004	5	11	15	18	22	119.9836	20.51736	60.33266	1	15	15	120	0	0	0	0	94	6536	
7	676	6.76E+14	2004	5	11	15	18	23	119.9836	20.53782	60.3122	1	15	15	120	0	0	0	0	94	6536	
8	676	6.76E+14	2004	5	11	15	18	24	119.9836	20.51736	60.3122	1	15	15	120	0	0	0	0	94	6536	
9	676	6.76E+14	2004	5	11	15	18	25	119.9836	20.51736	60.3122	1	15	15	120	0	0	0	0	94	6536	
10	676	6.76E+14	2004	5	11	15	18	26	119.9836	20.51736	60.3122	1	15	15	120	0	0	0	0	95	6536	
11	676	6.76E+14	2004	5	11	15	18	27	119.9836	20.51736	60.3122	1	15	15	120	0	0	0	0	94	6536	
12	676	6.76E+14	2004	5	11	15	18	28	119.9836	20.51736	60.3122	1	15	15	120	0	0	0	0	94	6536	
13	676	6.76E+14	2004	5	11	15	18	29	119.9836	20.4969	60.3122	1	15	15	120	0	0	0	0	94	6536	
14	676	6.76E+14	2004	5	11	15	18	30	119.9836	20.51736	60.33266	1	15	15	120	0	0	0	0	94	6536	
15	676	6.76E+14	2004	5	11	15	18	31	119.9836	20.51736	60.33266	1	15	15	120	0	0	0	0	94	6536	
16	676	6.76E+14	2004	5	11	15	18	32	119.9836	20.47644	60.3122	1	15	15	120	0	0	0	0	94	6536	
17	676	6.76E+14	2004	5	11	15	18	33	119.9836	20.53782	60.3122	1	15	15	120	0	0	0	0	94	6536	
18	676	6.76E+14	2004	5	11	15	18	34	119.9836	20.55827	60.33266	1	15	15	120	0	0	0	0	94	6536	
19	676	6.76E+14	2004	5	11	15	18	35	119.9836	20.41507	60.47586	1	15	15	120	0	0	0	0	94	6536	
20	676	6.76E+14	2004	5	11	15	18	36	119.9836	20.47644	60.43495	1	15	15	120	0	0	0	0	95	6528	
21	676	6.76E+14	2004	5	11	15	18	37	119.9836	20.4969	60.35312	1	15	15	120	0	0	0	0	95	6528	
22	676	6.76E+14	2004	5	11	15	18	38	119.9836	20.53782	60.37357	1	15	15	120	0	0	0	0	94	6528	
23	676	6.76E+14	2004	5	11	15	18	39	119.9836	20.53782	60.3122	1	15	15	120	0	0	0	0	94	6536	
24	676	6.76E+14	2004	5	11	15	18	40	119.9836	20.55827	60.3122	1	15	15	120	0	0	0	0	94	6536	
25	676	6.76E+14	2004	5	11	15	18	41	119.9836	20.61964	60.3122	1	15	15	120	0	0	0	0	95	6536	
26	676	6.76E+14	2004	5	11	15	18	42	119.9836	20.57873	60.33266	1	15	15	120	0	0	0	0	94	6536	
27	676	6.76E+14	2004	5	11	15	18	43	119.9836	20.61964	60.37357	1	15	15	120	0	0	0	0	94	6536	
28	676	6.76E+14	2004	5	11	15	18	44	119.9836	20.68102	60.29174	1	15	15	120	0	0	0	0	94	6536	
29	676	6.76E+14	2004	5	11	15	18	45	119.9836	20.84468	60.16899	1	15	15	120	0	0	0	0	94	6536	

676200405111519

2.4 Methodology

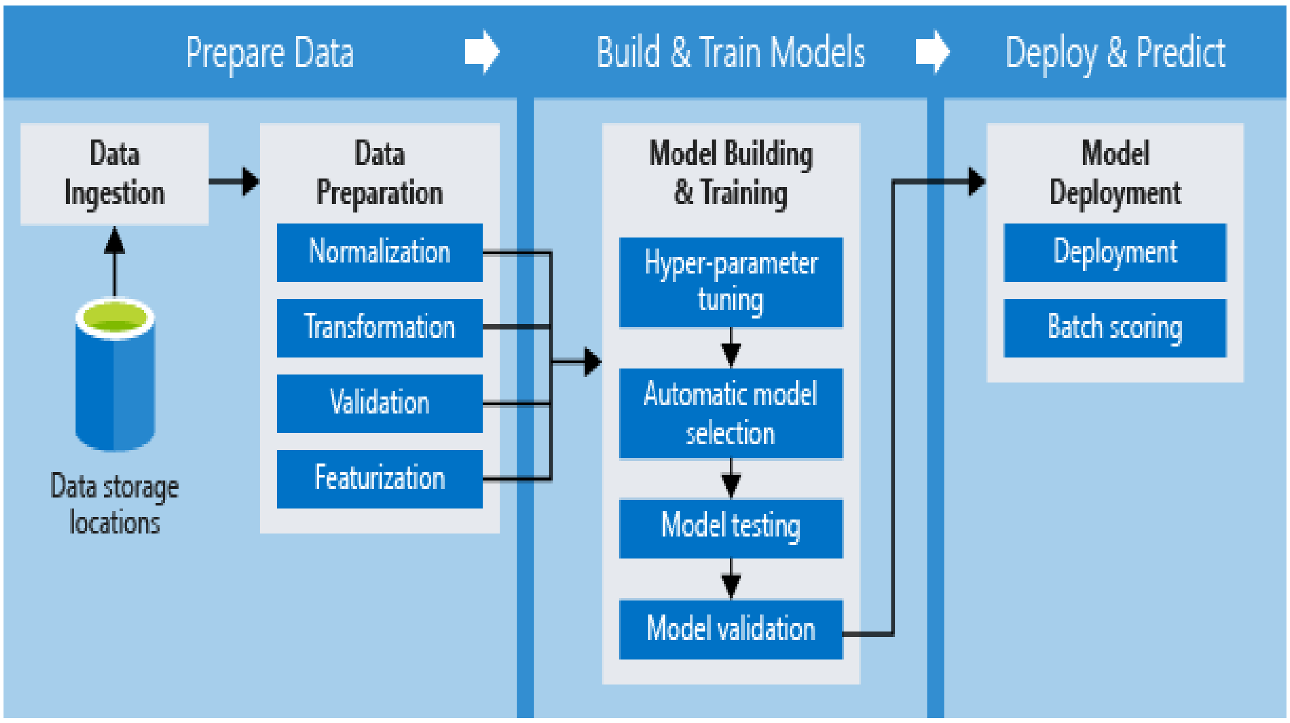


Fig 12 : The methodology diagram

This is the methodology which we will follow:

❖ **Data storage locations** : The recorded data or the raw data that is stored and will be used as the base for our whole project. More heavy the data is , more trained our machine will be and more accurate the results will be.

❖ **Data ingestion** : Importing our set of data from the stored location to our main code which will further be processed.This includes steps like understanding our data, collecting it , describing the data, exploring and verifying the quality of it.

❖ **Data Preparation** : Consists of steps like selecting the data, cleaning it, constructing , integrating and formatting the data using technique like Normalization , Transformation, Validation and Featurization

❖ **Model Building & Training** : In this stage we will apply hyper parameter tuning (choose a set of optimal hyperparameters for a learning algorithm . It's value controls the learning process). Model selection(selecting various supervised algorithmic models which gives a better results to the given model), Model Testing(Testing the models and checking the errors with the rmse methods), scoring(applying those algorithmic models from historical data set to new dataset)

- **Model Deployment:** scoring (applying those algorithmic models from historical data set to new dataset in order to uncover the practical insights that help solving a problem) and finally deploying the model.

BACKGROUND OVERVIEW

3.1 Conceptual overview

So Reducing fuel consumption is extremely important for aviation industry as fuel constitutes ~30% of the operating cost of airlines. Reducing fuel intake can also have a significant positive impact on the environment. Hence, developing cost saving strategies especially on fuel is of prime importance to airlines.

Driving fuel efficiency involves developing strategies that touch upon various aspects of airplanes - broadly some of which are highlighted below:

- **Aspects related to Aircraft's actions on the ground** - e.g. include reducing taxiing times to reduce engine running times which translate in to reduced fuel intake.
- **Aspects related to route planning** – e.g. taking shorter routes when inflight to destination taking in to consideration any altitude restrictions that exist.
- **Aspects related to aircraft design** – e.g. improving aerodynamics, redesigning aircraft components to conserve fuel or reducing the weight on board like installation of lighter seats.

There are different phases in flight during a flight instance. All the readings from the sensors fit into the airplane are taken from these sensors every second and saved in flight record data . There are mainly 7 phases of flight and one unknown phase which holds the data which was not recognised to be from any phase .

Here is the description of the phases of the flight from 0 to 7 . Variable PH in our dataset refers to the phase of flight .

3.2 PH TABLE

PH	*PHASE NAME*	*DESCRIPTION*
0	**Unknown**	Consists of those readings in which phase cannot be determined
1	**Preflight**	Phase before the flight includes some checkups of airplane and ensure everything is working fine
2	**Taxi**	This phase refers to the estimated time an aircraft spends taxing between it's parking stand and runway or vice versa
3	**Takeoff**	Fuel required to takeoff the plane
4	**Climb**	phase pointing to increasing altitude of an airplane
5	**Cruise**	Phase when an aircraft levels after a climb to set altitude before it sets to descend
6	**Approach**	A final approach is the last leg in an aircraft is lined up with the runway and Descending for landing
7	**Rollout**	The phase of an aircraft's landing during which it travels along the runway while losing speed

3.3 HOW TO OVERCOME THE PROBLEM ?

The Flight Record Data (FDR) can be used to

❖ Understand which features of dataset are correlated with Target(FF) we are trying to predict during different phases of a flight (Taxi, Takeoff, Climb, Cruise, Approach, and Rollout)
❖ Derive best practices that make flights fuel efficient under different conditions?
❖ We can later on do Exploratory Analysis on these top features

APPROACH

This is just a short overview of what strategies have been used to solve the problem . The detailed description will be in the code and explanation section.

3.1 Importing libraries

Firstly we imported some of the basic important libraries used in python like numpy, pandas, glob, os, sklearn, matplotlib and seaborn for enhanced visualization of our results using plots.

3.2 Loading into Pandas

Some pre-processing steps included downloading the data and loading it. , pre-processed to make it smaller and faster to load. All the CSV(comma separated) files were loaded into pandas dataframe and concatenated. After this the 64-bit data-types (int64,float64) were converted to 32-bit data-types (int32, float32). Finally, the combined Data Frame was saved as **pickle** files. The total size of train and test data pickle files on disk was about 6.5 GB.

3.3 Tidying the Data

Now , to check the the basic statistics of the data and separate the unwanted data from the important data DataFrame.describe() method was used and some variables showing no variance were observed and dumped using VarianceThreshold method and all N/A if were present were also thrown out.

3.4 Visualization

We compared fuel flow across various flight phases (There are 7 phases) excluding Unknown Phase. As different phases have very different fuel flows.We also segregated why few flight instances are different from others in terms of Fuel Flow.
 We have 600 flight instances . Data visualization was made clear using distplot and boxplot.
Then using FacetGrid function we obtained fuel flow plot for every phase on a separate axis .
Violin Plot and swarmplot showed the spread of fuel flow in the cruise phase was very wide .
Then compared and visualised Flight_instance with FF using visualization through boxplots.
As the instances were very large in number it became difficult to visualise.

3.5 Splitting the data

Now the sklearn train_test_split , cross_validation K Fold strategy was used here to split data into pairs into groups .

3.6 Checking the Errors

Using the Root Mean Square Error method the more validation error that comes up the poor the model is .

3.7 Models

We have used two models here for better results . Algorithms like ExtraTreesRegressor and eXtreme Gradient Boosting have shown amazing results in many competitive coding contests and are known for their less error in rmse and fast computation. in the end one with less validation error was considered as better model .

3.8 Finding the Top Features

Now top features were found using both the models using plot_importance method we created which was using .feature_importances method. Then the feature ranking was done, from where we come to know that PH(phase of flight) is one of the most important variable

3.9 Finding the correlation using Heatmaps

TO find the correlation between top features including the FF variable with each other we get the correlated features and remove highly correlated ones.

From this we will also get other top features which are important for minimizing the Fuel Flow rate.

WORK DONE

5.1 Details as required

- ❖ Understanding the problem statement and thinking of its possibilities
- ❖ Learning the basics of python and its important libraries and implementing them
- ❖ DATACAMP, Coursera and Udacity course for more implementation practice and sharpening the concepts of python in Data Science
- ❖ Quick crash course on aircraft flight parameters, an article explaining flight controls of a Boeing 737 and categorized the 227 parameters in different categories.
- ❖ Understanding the Data Dictionary, its values in terms of aviation and its meaning.
- ❖ Cleaning the large amount of data to make it simple and relatable
- ❖ Coding and visualization of the characteristics showing variance in the given data set
- ❖ Removing unnecessary and noisy variables that could potentially confuse the predictive model
- ❖ Reading through research papers, finding various strategies through which our problem can solved
- ❖ Finding Different ways of Supervised Learning Techniques for solving the problem and at end performing comparison between those methods
- ❖ Learning various Visualization Techniques to better understand the plots and graphical representations.
- ❖ Implementing Some efficient statergies like GroupKFold , Cross-Validation for better output.
- ❖ Studying various concepts like plot_importances , Heatmap so as to find the top features.
- ❖ understanding the code and implementing it properly to get the desired top features for future exploratory analysis.
- ❖ Made the analysis document and original project report for reference in future

ANALYSIS OF CODE

EXPLANATION OF CODE STEP BY STEP

In this section the code is explained line by line and why that method was applied and its description in detail. From 7.1 to are the parts the problem was divided into to solve .

7.1 IMPORTING BASIC LIBRARIES OF PYTHON

In this section some of the important libraries of python like pandas, numpy, os, matplotlib and others, are imported so as to use their functionalities later in code.

- ❖ **Pandas** : It is the library providing high performance ,easy to use data analysis tools and high data structures. Mainly known for importing csv datafiles to our dataframe.

- ❖ **Numpy**: Mainly used to create numpy arrays. its is a powerful N dimensional array object.

- ❖ **glob** : Returns an array of filenames or directories maintaining a specific pattern

- ❖ **OS** : Can use os line commands in Python workspace .

- ❖ **Matplotlib** : Matplotlib tries to make easy things easy and hard things possible. We can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code.

- ❖ **% matplotlib inline** : Sets the back end of matplotlib to the ‘ inline’ back end, It ensures that output of plot is just shown with the code for better visualization .

- ❖ **Seaborn** : It is higher version of matplotlib library . It provides a high-level interface for drawing attractive and informative statistical graphics.
- ❖ **Sklearn** : It is a library which features various algorithm like classification regression and clustering including support vector machines, random forests, gradient boosting, *k*-means and DBSCAN.

7.2 LOADING UP THE DATA

The load_data() function with argument as path is created for loading the desired data csv files in the code.

def load_data (path):

Now we have 5 training folders and 200 csv files in each folder and one testing folder . All the CSV files were loaded into pandas DataFrame and concatenated.

all_files = glob.glob(path + "/*.csv")

To find the files with same pattern like **.csv** we used **glob** function to match the pattern and load only those files with .csv pattern matching and put them in all_files.

list = [ ]
Initialising an empty list so as to store the values .

for i , file in enumerate (all_files[: 200])

For i folder we are going to consider all 200 files. Now in our for loop we used **enumerate -**this is used to loop over something and have an automatic counter.

df= pd.read_csv(file=index , col=none , header=0)

Reading the csv file one by one using the **read_csv** function using **Pandas** as pd and putting it in dataframe df

df ['flight_instance'] = i

From the dataframe df choosing flight_instance variable and putting it in variable i .
From here we will come to know which element of file 1, file2 …… ,file200 was captured in 'flight instance'

list.append(df)
Moves to the next item on list appending it by 1.

return pd.concat(list)
This function returns a concatenated list.

path = r"data"
train = load_data(path)
r refers to raw string treats / backslash as literal character and
calling the load _data function and putting its resulted concatenated list in train dataframe .

This combined DataFrame was saved as pickle file (pickle the object to save it on disc . It is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.)

7.3 CHECKING BASIC STATISTICS ON DATA :

pd.set_option("display.max_columns", 250)

If there are a lot of variables and columns covered are more it does not print all columns as default so to print all the columns lets change the max_columns to 250

train.describe()

.describe() method gives detailed description of the dataset variables. Descriptions about count, mean, std, min, 25%. 75%, 50 % and max are seen using this method

Observation :

*From observing we came to know that few columns had **same values (no variance)** and they are not useful so we will remove them in next steps .*

train.isnull().sum().sum()

This removes the N/A in our columns , as there are no n/A in our dataset we assured if there were we would come to know but we can see 0 as output meaning there are no N/A .

7.4 REMOVING THE VARIANCE

This is one of the step of **tidying up** the data .Here we are throwing the columns whose variance is less than 0

def remove_low_varcols(df, threshold):

A function to remove no variance variables using threshold and it will return a new dataframe with only those factors who are showing variance. No variance means those variables who don't have any direct connection with the target variable

return df_new

train = remove_low_varcols(train, 0)
Calling our function with our original dataframe train and threshold as 0 and the resultant new df will be known as train dataframe.

Output:
Columns with Variance Less than or equal to threshold are:
['ACID' 'FIRE_2' 'FIRE_3' 'FIRE_4' 'FQTY_3' 'POVT' 'SMOK' 'WAI_2'
'APUF_Mean' 'APUF_Min' 'APUF_Max' 'TOCW_Min' 'CALT']
New shape (1217028, 214)

7.5 VISUALIZATION

Now here we will use the visualization techniques to :

- ❖ **compare Fuel Flow** across various **phases** of flights. Different phases have different Fuel Flow
- ❖ Also we will understand why few **flight instances** are different from others in terms of fuel flow.

plt.figure(figsize=(12,10))
sets the size of figure with 12 inches width and 10 inch height

sns.distplot(train['FF'], bins=100, color='red')
Flexibly plot a univariate distribution of observations.This function combines the matplotlib hist function (with automatic calculation of a good default bin size) with the seaborn kdeplot() and rug plot() functions.
Our DataFrame is train with FF , distributed the bins in 100 and color of graph set to red

plt.show()
TO show the plotted graph on output screen

Observations:

Here in the output graph in the code section we can learn that on the x axis is Fuel Flow and y axis The observations of FF for various phases of flight.

train.PH.value_counts( )

To get the observation count value of FF with phase number

Output:

```
 5   361086
2    242135
6    229122
4    216804
0    125960
3    27418
1    10447
7    4056
Name: PH, dtype: int64
```

We can observe there are **maximum** no of observation values for **phase 5 (Cruise phase)** because the flight is actually in this state for a long period of time as compared to other phases

Creating Boxplot for FF across different phases flight

```
plt.figure()

train[["PH", "FF"]].boxplot( by="PH")
```

here we are making boxplot graphs of FF across various phases [PH] of flight with PH on x axis

Output:

From the boxplots we can see the min , max , 25%, 50%, 75% of Fuel FLow observation for different phases of flight.

g = sns.FacetGrid(train, col="PH", col_wrap=3, size=4)
g = g.map(sns.distplot , 'FF', bins=100)
plt.ylim([0, 0.001])
plt.show()

Now to see different individual plot for each phase with seperate axis we will use **sns.FacetGrid**

Output:

From the plots it was observed that **PH 3,4,5,6** were **spread the most** and PH **2** plot had **outliers**(an outlier is a datapoint in a dataset that is different from all other observations)

we know now that the flight spends most of its time in Cruise Phase [PH 5] so by using **Violin Plot** we will try to see its spread

train_ph5 = train.loc[train.PH == 5]

Focusing on phase 5 from train dataframe and putting it into a new DataFrame named train_ph5(Training dataframe for phase 5)

train_ph5_agg = train_ph5.groupby('flight_instance').agg('mean')

.groupby splits the data into groups based on some criteria and as there are so many flight instances so we are considering their mean here.

train_ph5_agg.head()

.head() method shows first 5 rows of our dataset by default.

sns.violinplot(y='FF', data=train_ph5_agg)
sns.swarmplot(y = 'FF', data= train_ph5_agg, color='red')
plt.show()

For Cruise Phase plotting a violin plot with FF observations for different Flight_instances on y- axis and using swarmplot also looking at the data points.

Output : From the violin plot it was seen that the spread was quite large

USING BOXPLOT FOR CRUISE PHASE

sns.boxplot(data=train_ph5, x='flight_instance', y='FF')
plotting boxplot with flight instances against FF

sns.swarmplot(data=train_ph5, x='flight_instance', y='FF')
plt.show()

Output : It was observed that a lot of values for flight_instance was far away from the mean value . Now to understand this we observed plots for various flight_instances with respect to duration of flight

so here is the explain for one of the flight instance :

flight_instance = 4
Picking up flight_instance value 4

plt.plot(train_ph5.loc[train_ph5.flight_instance == flight_instance, 'FF']
Constructing a plot of train _ph5 dataframe for flight instance where it is 4 against FF observations .
plt.show()

Observations:

- There **patches** of values where there are **no readings** in phase 5; probably **phase changed** intermittently (to be checked below)
- There are few values for each instance where **sudden drop in fuel flow** is observed (could be **measurement/data processing error**?)
- **Last** value and sometimes **beginning** values of a given phase are **far off** (could be due aggregation of values??)
- There is mean **shift** of fuel flow in some instances (could be change of cruise altitude etc.)

CONFIRMING FAULTY ALLOCATIONS
To see what actually is happening here we tried confirming the faulty allocation that we allocated before for various flight phases :

plt.plot(train.loc[train.flight_instance == 4, 'PH'])
plt.show()
Here we have assigned flight_instance while plotting,

Observations:

It was observed that **change in phase** was **not monotonous** .This is most likely, a**ssignment error;** Plane would not oscillate so many times between climb and
cruise or cruise and approach.

7.6 SPLITTING DATA

In this section of our code we are using various techniques to efficiently split our data into training and testing set .

from sklearn.model_selection import train_test_split, GroupKFold, cross_val_score
here we are importing various algorithms and techniques from sklearn described below:

- **train_test_split:** to divide our dataset into training and testing datasets
- **GroupKFold :** KFold is used when we want to divide chunks of observations across folds. Normal K Fold randomly assign observations to folds whereas GroupKFold first groups the the observations and then assigns them randomly to the folds
- **cross_val_score:** Measures performances of different algorithms and prepares scores array . Mainly tells us the error rate and helps selecting best fit model.
  It also ensures that model is non -overfit

#5 fold cv strategy
folder = GroupKFold(n_splits=5)
applying GroupKFold method with no of folds =5

cvlist = list(folder.split(train, y=None, groups=train.flight_instance))
splitting part and saving it into list

tr = train.iloc[cvlist[0][0]]
val = train.iloc[cvlist[0][1]]
Use first split as Hold out cv - for quick checking

set(tr.flight_instance.unique()) & set(val.flight_instance.unique())
Checking to ensure we are mixing flight instances between train and validation sets

7.7 ROOT MEAN SQUARE ERROR (rmse)

It can be used to measure the accuracy and performance of different models or algorithms on a dataset . The less the rmse the better the model .That means less the error the better the model will be.

def rmse(y_true, y_pred):

Creating the root mean square error method so as to call it for different models.

return np.sqrt(metrics.mean_squared_error(y_true, y_pred))

returns the output to tell us the error and tells us the accuracy by doing the root mean square.

7.8 IMPLEMENTING THE MODELS

In our project we are going to use two models **Random Forests** and **extreme Gradient Boosting** because of their better performance and efficiency rate .

so now we are using **ExtraTreeRegressor()** method firstly to see which features comes up on the top which actually is our main objective .

etr = ExtraTreesRegressor(max_depth=7, n_estimators= 200, n_jobs=-1, verbose=1)

So here we are dumping everything in etr(ExtraTreeRegressor) and checking which features come on the top .

max_depth : The max_depth of a tree, if none the nodes are expanded until all leaves are pure or until all leaves contains less than min_sample_split samples . Here we have chosen the depth of tree as 7 we can change it again and again and see where we get best output .

n_estimators: The number of trees in the forest

n_jobs= - 1 : this means we are using all processors

verbose = 1 : controls the verbosity while fitting and predicting

feats = [f for f in train.columns if f not in ['FF', 'flight_instance']]
etr.fit(tr[feats], tr['FF'])

We are fitting the ExtraTreeRegressor model with feats and FF samples.

The **preliminary feature selection** was done fitting this model with **200 trees** and **max_depth** of trees as **7** . Later on the importance of each feature was obtained from this and using the plot_importance method **top 25 features** were selected . The original data was then replaced by the subset of data with only these 25 important features.

TO CHECK THE rmse VALUE FOR etr

print("RMSE on train set :", rmse(tr['FF'], etr.predict(tr[feats])))
predicting rmse value on training set

print("RMSE on hold out validation set:", rmse(val['FF'], etr.predict(val
predicting rmse for validation set

Output:

RMSE on train set : 329.0492207519591

RMSE on hold out validation set: 341.46706972668557

NOW trying with xgboost

import xgboost
from xgboost.sklearn import XGBRegressor
importing the XGBRegressor from sklearn library

X_tr = tr[feats]
y_tr = tr['FF']
X_val = val[feats]
y_val = val['FF']
splitting the data in training and validation sets using tr referring to training and vr for validation for x and y

xgb_dump = XGBRegressor(max_depth=6, n_estimators=1000, colsample_bytree=0, subsample=0.8, learning_rate=0.2)
calling the regressor method with max depth =6 ,1000 trees and 0.2 learning rate

xgb_dump.fit(X_tr, y_tr, eval_set=[(X_tr, y_tr), (X_val, y_val)], eval_metric='rmse', verbose=1,early_stopping_rounds=1)

fitting the model with training and val set and getting the rmse to check the error rate
Output : VALIDATION rmse = 200

Observation :

SO NOW WE CAN SEE THAT rmse VALIDATION VALUE FOR EXTRATREEREGRESSOR IS 341.46706972668557 AND FOR XGBOOST IS 200 . THIS MEANS

rmse VALUE OF XGBOOST < EXTRATREEREGRESSOR

SO XGBOOST IS BETTER METHOD.

7.9 FINDING THE TOP FEATURES

In this section we are going to find the top features using **plot_importance** method.
This method is also called **Feature Selection .** We have so many variables and we do not know which of them are important to the model, Random Forest helps to decide the relative importance of the variable by a plot called **VARIABLE VARIANCE PLOT.**
Features refers to independent variables . The process of selecting important features is called Feature Selection .

def plot_importance(model, feats, n_feats):
Creating a plot_importance method with arguments as model which we are taking into use . the feats and n_feats no of features .
This will return the top features responsible for fuel flow consumption

_, top_feats = plot_importance(etr, feats, 25)
Calling our plot_imprtance method with etr (ExtraTreeRegressor) model and feats list and 25 n_feats

Output :

Feature Ranking:

```
1   IVV_Mean (0.084489) (0.148454)
2   CAS_Min (0.084107) (0.167770)
3   CAS_Mean (0.076386) (0.158847)
4   IVV_Min (0.075828) (0.134424)
5   IVV_Max (0.066964) (0.123356)
6   ALTR_Min (0.054371) (0.112998)
7   ALTR_Max (0.050322) (0.108571)
8   ALTR_Mean (0.041237) (0.099513)
9   MACH_Mean (0.037555) (0.119481)
10  CAS_Max (0.036193) (0.115377)
11  GS_Mean (0.025937) (0.100067)
12  MACH_Max (0.025539) (0.098464)
13  GS_Max (0.023136) (0.092056)
14  GS_Min (0.022971) (0.096472)
15  N1T_Min (0.019834) (0.041106)
16  N1T_Max (0.017343) (0.036877)
17  N1T_Mean (0.016924) (0.038210)
18  PH (0.014879) (0.050823)
19  PI_Max (0.013783) (0.072954)
20  MACH_Min (0.013265) (0.073216)
21  FPAC_Max (0.012998) (0.014419)
22  VMODE (0.012004) (0.026403)
23  TAS_Max (0.011043) (0.064733)
24  FPAC_Mean (0.010024) (0.012104)
25  FPAC_Min (0.009425) (0.012013)
```

As expected we get Phase as one of the important variables.

We will look into other variables also :

Scatter plot between LONG_Max and FF (Fuel Flow)

plt.figure()
To plot a figure

sns.jointplot("LONG_Max" , "FF", data=train)
Here we are constructing a joint plot between LONG_MAX on x - axis and FF on y - axis

plt.show()
To show the constructed plot .

Similarly we will plot a graph for IVV_MAX and FF

plt.figure()
sns.jointplot("IVV_Max" , "FF", data=train)
plt.show()

Observations :

Vibrations are related to acceleration, engine health and Phase.

Using **Heat Maps** we stepped up the game
Heatmaps represents the correlation between various variables with each other

corr_feats = list(top_feats) + ['FF']
Firstly combining the features with whom we want see the correlation the top_feats we obtained from the plot_importance method and FF variable and concatenating them into a list named corr_feats meaning correlating features

corr_df = train[corr_feats].corr()
Now in Pandas **dataframe.corr()** is used to find the pairwise correlation of all columns in a dataframe and any N/A values are automatically removed.
Here we will find correlation of items in the list we created corr_feats with each other.

fig, ax = plt.subplots(figsize=(16,16))
Changes the size of the figure

sns.heatmap(corr_df, robust =True, annot=True, ax=ax, annot_kws={'size':9}

Calling the heatmap method from seaborn library ,
corr_df is our **dataset**

robust = true takes a boolean entry . if it is True or Vmin or Vmax values are absent , the color map range is computed with in **robust quantities** instead of extreme values .

annot =True means to write data values in each cell

OBSERVATIONS FROM HEATMAP :

Few other features that show up high are
CAS(Corrected air speed)
Mach, Ground speed
True airspeed and
Altitude related features.
CAS and Mach are correlated. Also, min, max and mean are highly correlated for many features. It might be a good to remove some of the highly corelatedones.

DIFFERENCE IN VARIOUS PHASES OF FLIGHT

There are seven different phases of the flight in this dataset, marked by the values of ‘PH’ 0 to ‘7’ a ‘PH’ value of 0 indicates that the phase is unknown.

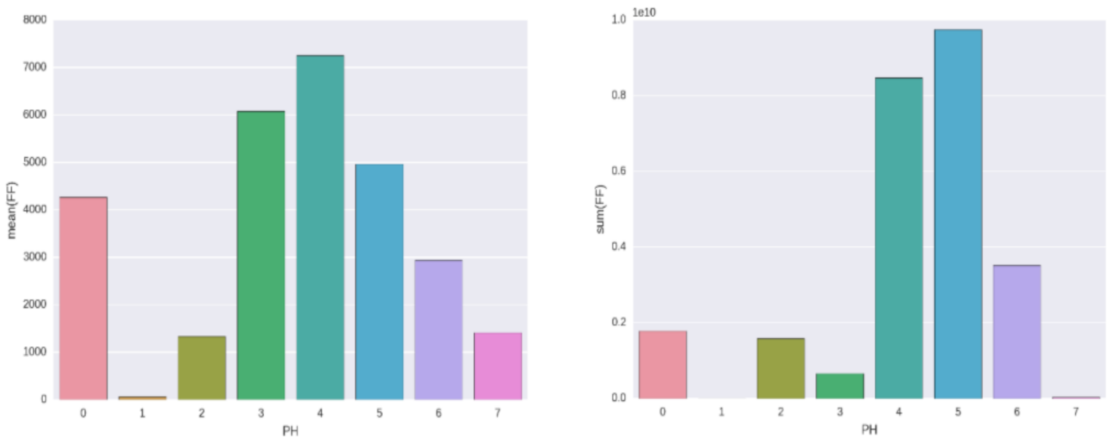


Fig 1 The mean fuel flow rate (left) and total fuel consumption (right) in each phase of flight. From these graphs, phase 4 and phase 5 appear to be the most important phases that drive the fuel consumption in an aircraft.

The plots in fig 1 show mean and total fuel consumption in different phases of flight.
The mean and standard deviation of the subset of data with unknown phase (mean = 4266, std = 2440) is similar to those of the entire dataset(mean = 4219, std = 2356). This observation gives a strong indication that the **subset of data with PH = 0 is representative of entire dataset and thus have a distribution of flight-phases similar to entire dataset**.

The preflight phase is small and in the fuel consumption is zero for most of the time (80% of the instances in this phase have FF = 0). The total fuel consumption is **highest** in the **climb (PH=4) and cruise (PH=5) phases**, which is obvious because these two phases are among the l**ongest phases of flights**.

The **approach (PH=6)** phase is also of a significant duration, but since the aircraft is **not accelerating anymore**, the fuel consumption is **smaller** in this phase.

The **takeoff (PH=3**) has a high rate of fuel consumption, but since this phase **does not last too long,** the total fuel consumption is rather small in this phase.

The difference between total fuel consumed in different flight instances is driven mainly by the **distance between source and destination**. Since the data does not contain the distance information, we should derive it from the **ground speed**. The GS_Mean column has ground speed as function of time, and thus we can obtain distance by simply integrating this column. The following plot shows the total fuel consumption as a function of total distance for each of the flight instances given in the data.

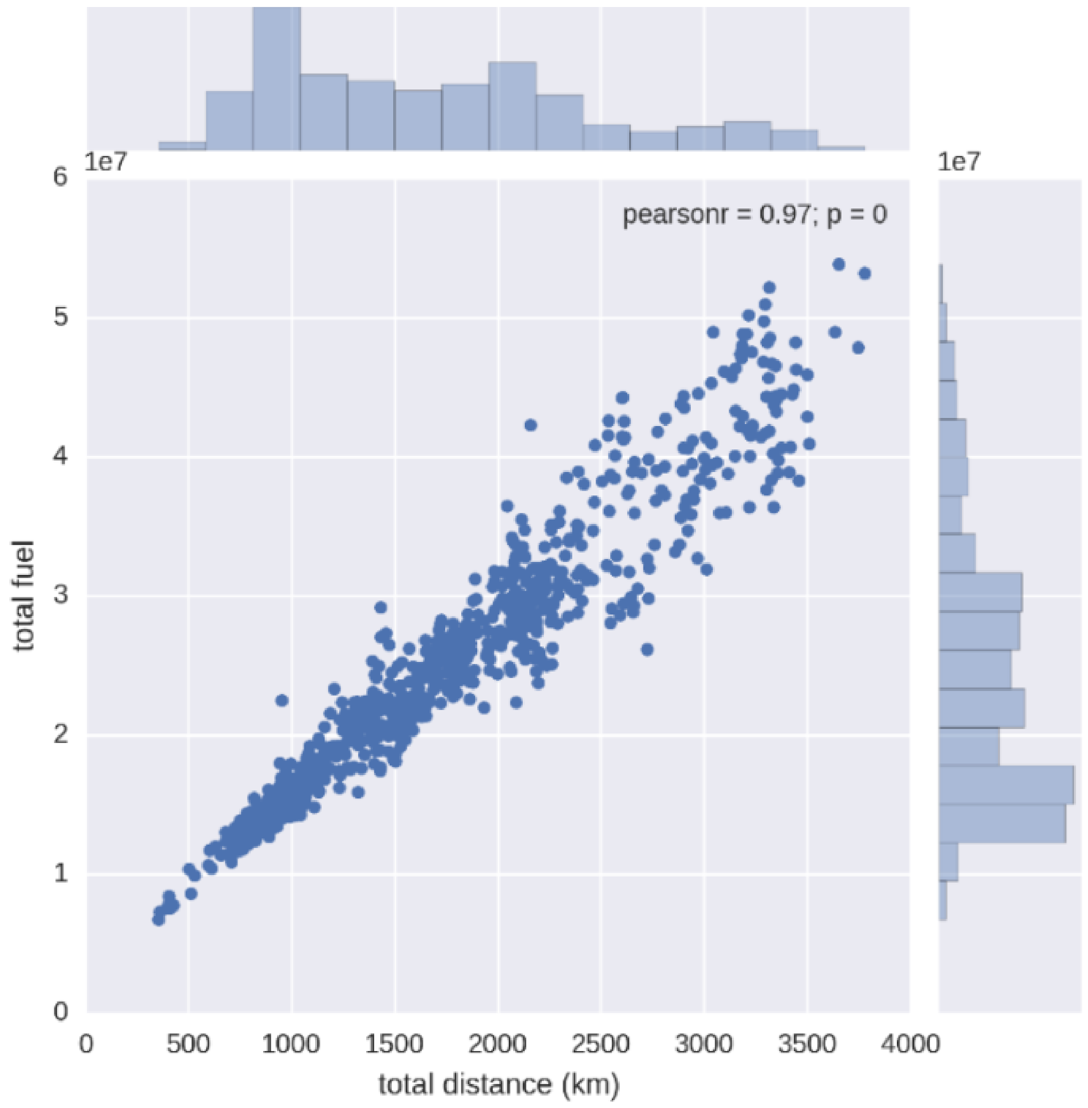
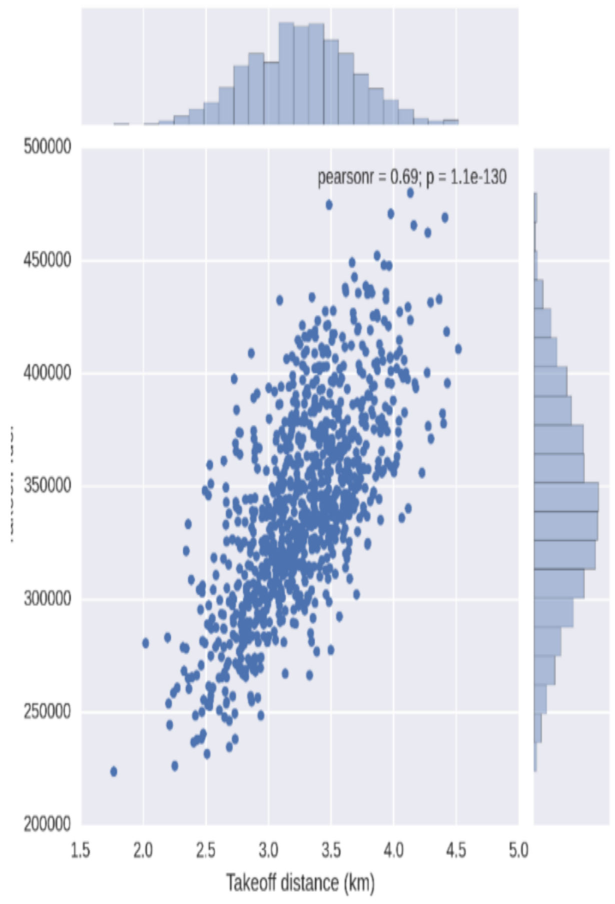
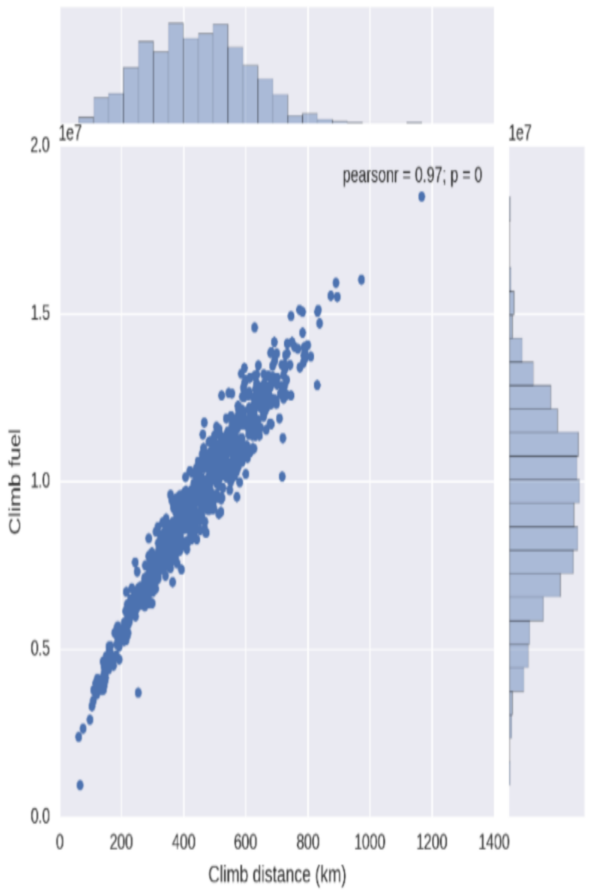


Fig 2 Joint distribution of fuel consumption and distance covered by the aircraft. A very strong linear correlation between distance and fuel consumed (pearson coefficient = 0.97) is found.

Since the taxi phase is mostly determined by the traffic and distance between aircraft bay and runway, it is not really in the direct control of the flight operators. The rollout phase is very small and hence not particularly interesting. The other phases, namely takeoff, climb, cruise and approach are the ones which correspond to travel between source and destination and are deemed to be more important.





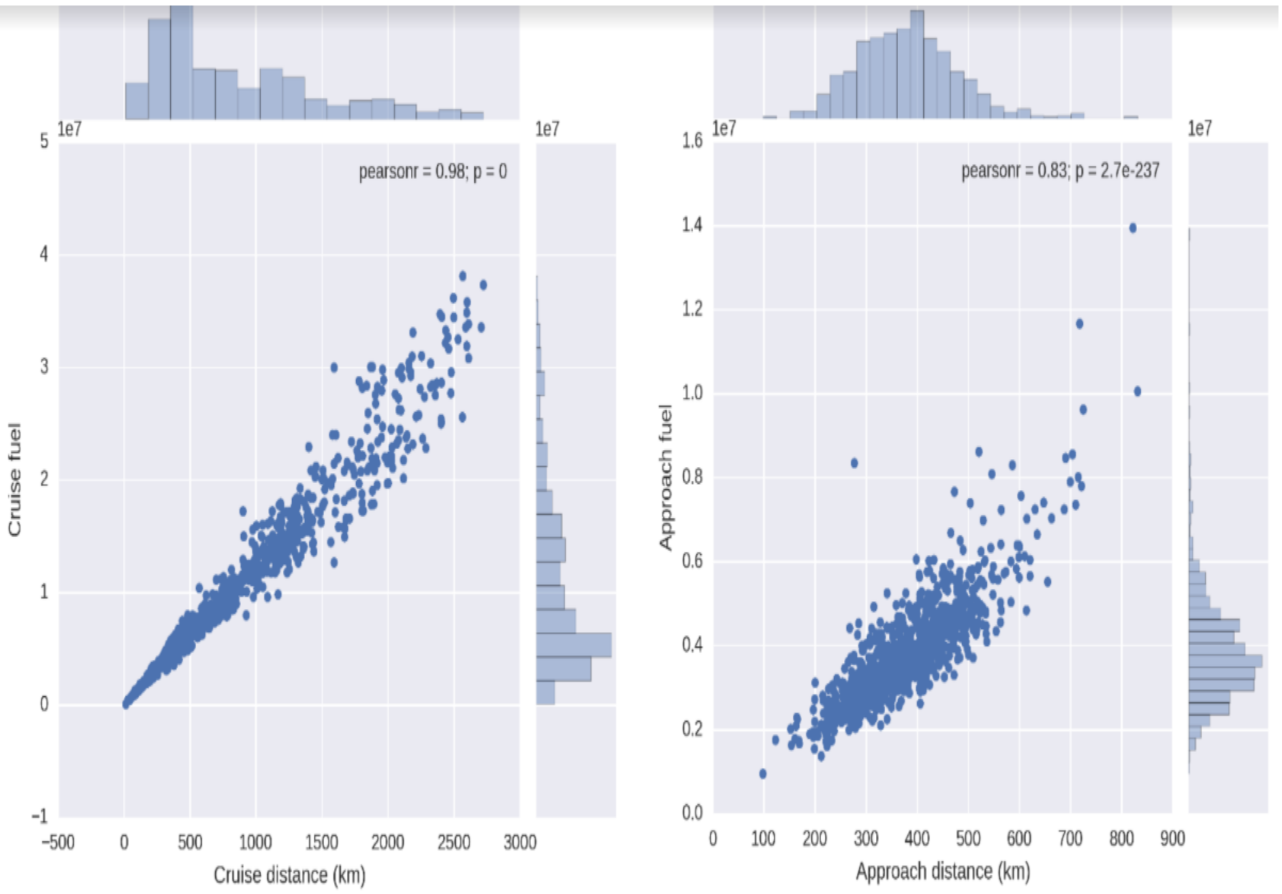


Fig 3 The joint distribution of fuel consumption and distance for takeoff, climb, cruise and approach phases. **The Climb and cruise phases have the largest fuel consumption**, but the relationship of fuel consumed with distance has a very strong correlation. On the other hand, the takeoff and approach phases have significant deviation from a linear relationship which hints at possible improvements.

A breakdown of the analysis of fuel vs distance into different phases gives an idea about **which phases need to be and can be optimized for low fuel consumption**. In fig 3, we show joint distribution of fuel and distance for different phases. This analysis has an implicit assumption that **flight path is straight between source and destination**; which is a reasonable but not completely accurate assumption.
In the following subsections, the report will focus on **key differences in the predictors in different phases of flight**. As mentioned above, the **takeoff, climb, cruise and the approach** phases are the important phases as far as optimization of fuel consumption is concerned.

PH = 0 (Unknown phase)

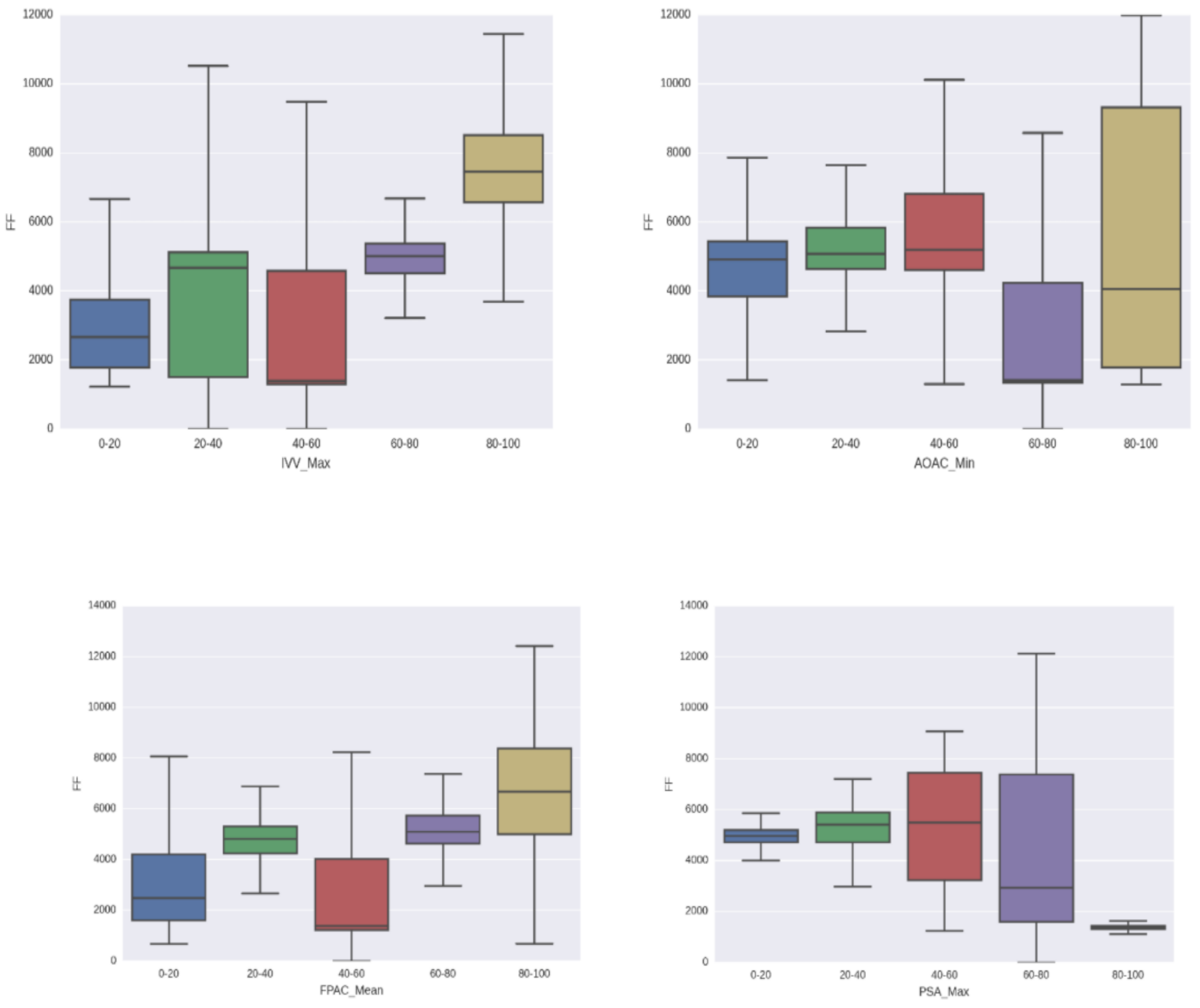


Fig 4 The relationship of fuel flow in the unknown phase with top individual predictors - *IVV_Max*, *AOAC_Min*, *FPAC_Mean* and *PSA_Max*. Here the x-axis is five equally sized groups of samples such that the first box corresponds to values between 0 and 20th percentile of predictor, the second box corresponds to values between 20th and 40th percentile and so on. Clearly, there is a significant relationship between fuel flow rate and these predictors but none of these four relationships are linear.

As mentioned before, PH=0 is actually a mini version of the entire dataset. The top individual predictors for this part of data were IVV_Max, AOAC_Min, FPAC_Mean and PSA_Max. The top combined feature for this phase was a combination of IVV_Mean, PT_Min and CAS_Max. The relationship of the important individual predictors with the fuel consumption is shown in the fig 4.

PH = 1 (Preflight)

The preflight phase is a small phase and **80% of the time in this phase does not consume any fuel.** The average fuel consumption for the total time in this phase is just **61** which is very small and insignificant compared to the average of all flight phases (4219). The top four individual predictors of fuel consumption for this phase are **VIB_1_Mean, OIT_4, OIPL and N1T_Max** and the top combined predictor was a combination of **SAT, PAC_Mean, TAS_Mean and IVV_Min.**

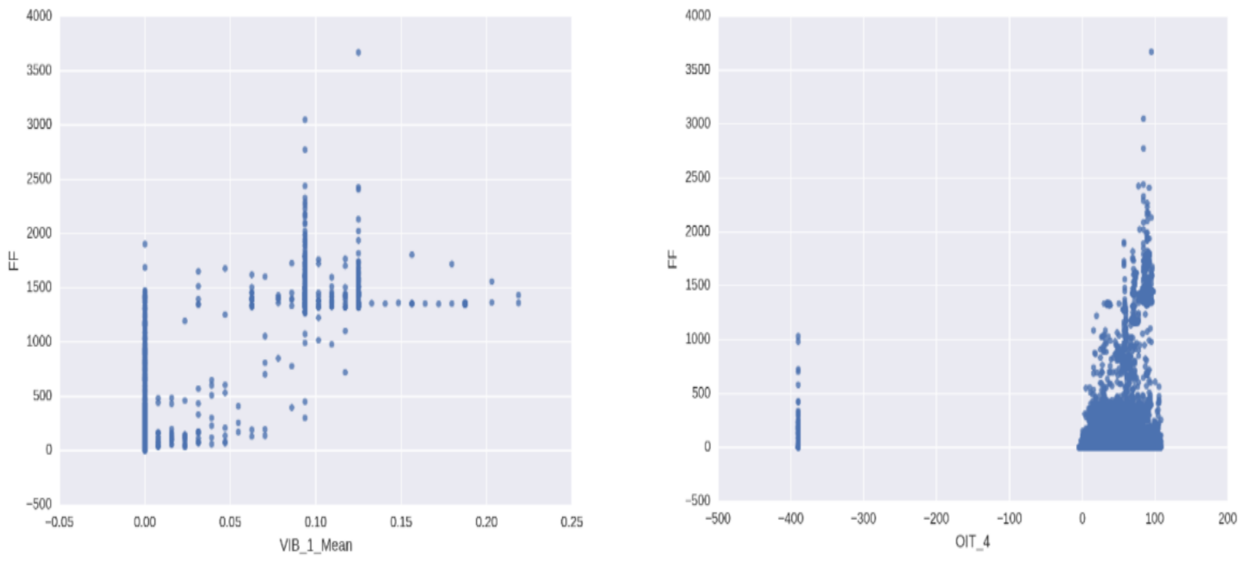


Fig 5 The relationship of fuel flow in the preflight phase with top individual predictors - **VIB_1_Mean, OIT_4, OIPL andN1T_Max**. Because of a skewed distribution of fuel flow (most of the samples having zero fuel flow), the distribution of fuel flow as a function of percentiles of predictors was **not informative.**

PH = 2 (Taxi)

There is a significant amount of time spent in this phase (**20%** of the time in flights data given in this competition). The average fuel consumption (**1334**) is just above half of the overall average,and thus this phase accounts for about **6% of the total fuel consumed**. The top four individual predictors of fuel consumption for this phase are **LONG_Max, OIPL, VIB_1_Mean and FQTY**_4and the top combined predictor is a combination of **PT_Mean, OIPL and FPAC_Min.** The following graphs shows the relationship of fuel flow rate with respect to these four individual predictors.

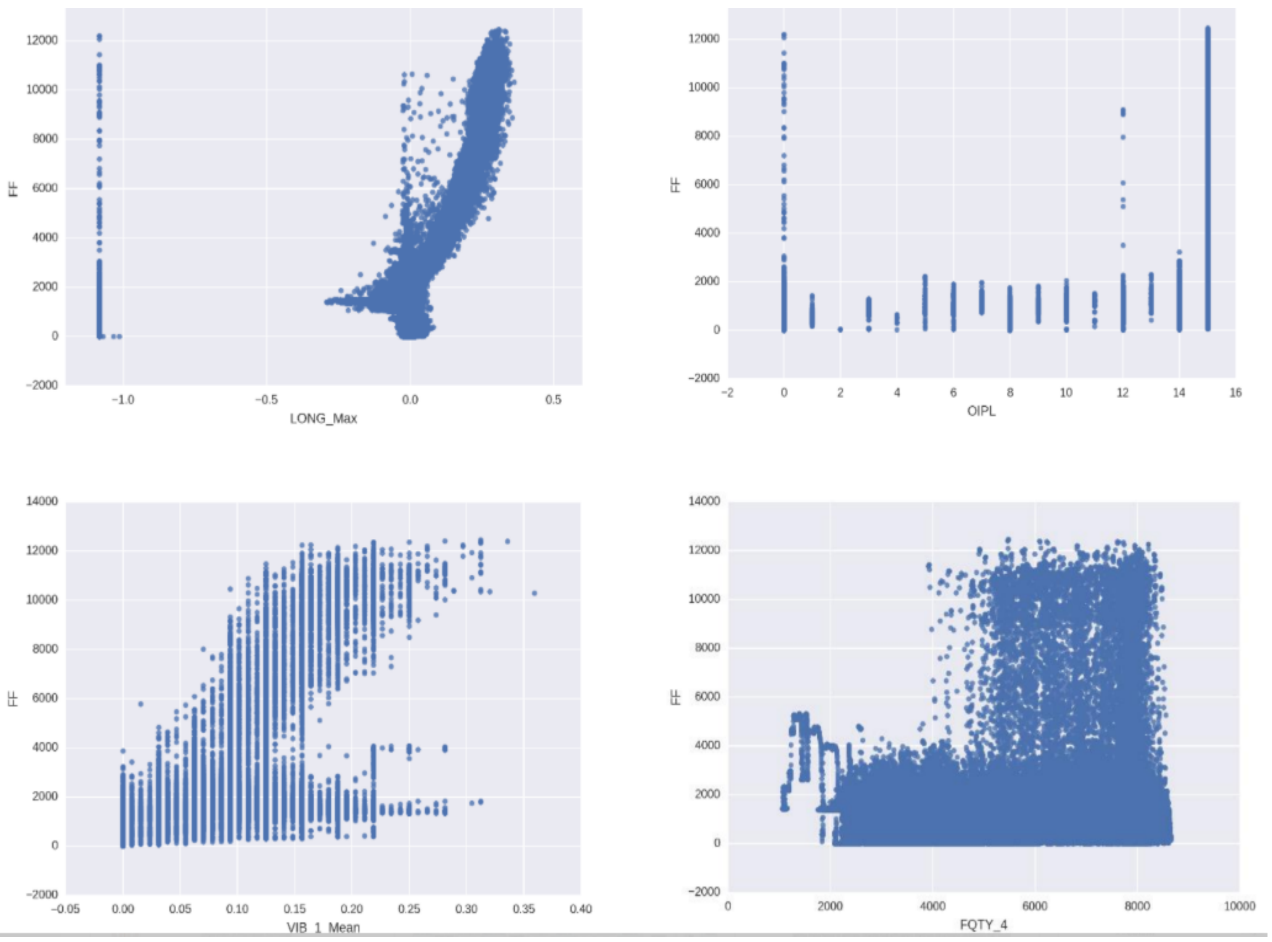


Fig 6 The relationship of fuel flow in the taxi phase with top individual predictors - **LONG_Max, OIPL, VIB_1_Mean and FQTY_4.Only** LONG_Max shows a clear relationship, for which the fuel flow rate increases with increase in the value of LONG_Max. The fuel flow was VIB_1_Mean also has a faint visible pattern, but the other two graphs (OIPL and FQTY_4) are very uninformative. The validation root mean squared error in this phase was about 120, so the absence of a clear visual relationship between fuel flow and top predictors is indicative of strong non-linear interaction between predictors.

PH = 3 (Takeoff)

The average fuel flow rate in this stage is very large **(6077)** but since this phase lasts for a **small time**, it accounts for only about **2.5%** of the total fuel consumption. The top individual predictors are **LONG_Max, FLAP, IVV_Max and AOAC_Min**. The best combination of features for prediction of fuel consumption in this phase was **GS_Min, N1T_Max, IVV_Min and FPAC_Min.**

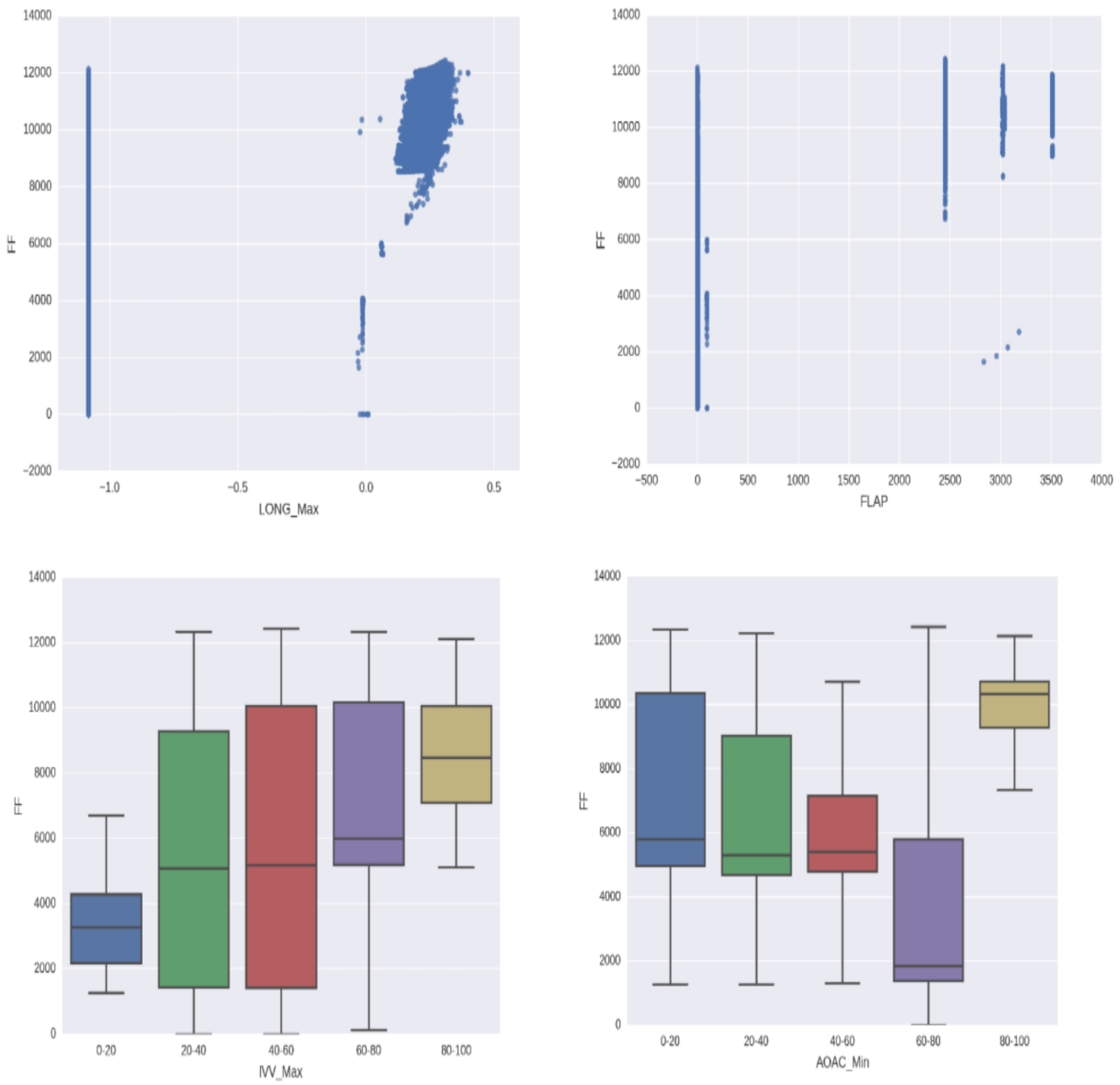


Fig 7 The relationship of fuel flow in the takeoff phase with top individual predictors - **LONG_Max, FLAP, IVV_Max and AOAC_Min**. The first two graphs are shown as simple scatterplots, while the latter two are shown as distribution of fuel flow rate in different percentile ranges of predictors. The graph format is chosen for best possible clarity.

The relationship of fuel flow rate with LONG_Max and IVV_Max (somewhat linearly increasing) and AOAC_Min (nonlinear relationship) are visible from the plots in fig 7.

PH = 4 (Climb)

This is the phase with the **highest average fuel flow rate (7254)** and it accounts for about **33%** of the total fuel consumption during a flight. The most important predictors for this phase are **PT_Mean, LONG_Max, FLAP and ALTR_Mean**. The fuel flow rate increases with increase in PT_Mean, LONG_Max, and ALTR_Mean as is clear from boxplots in fig 8 below. However, the relationship of fuel flow rate with FLAP is not clear from a plot, which implies a nonlinear relation or a strong interaction of FLAP with other features.

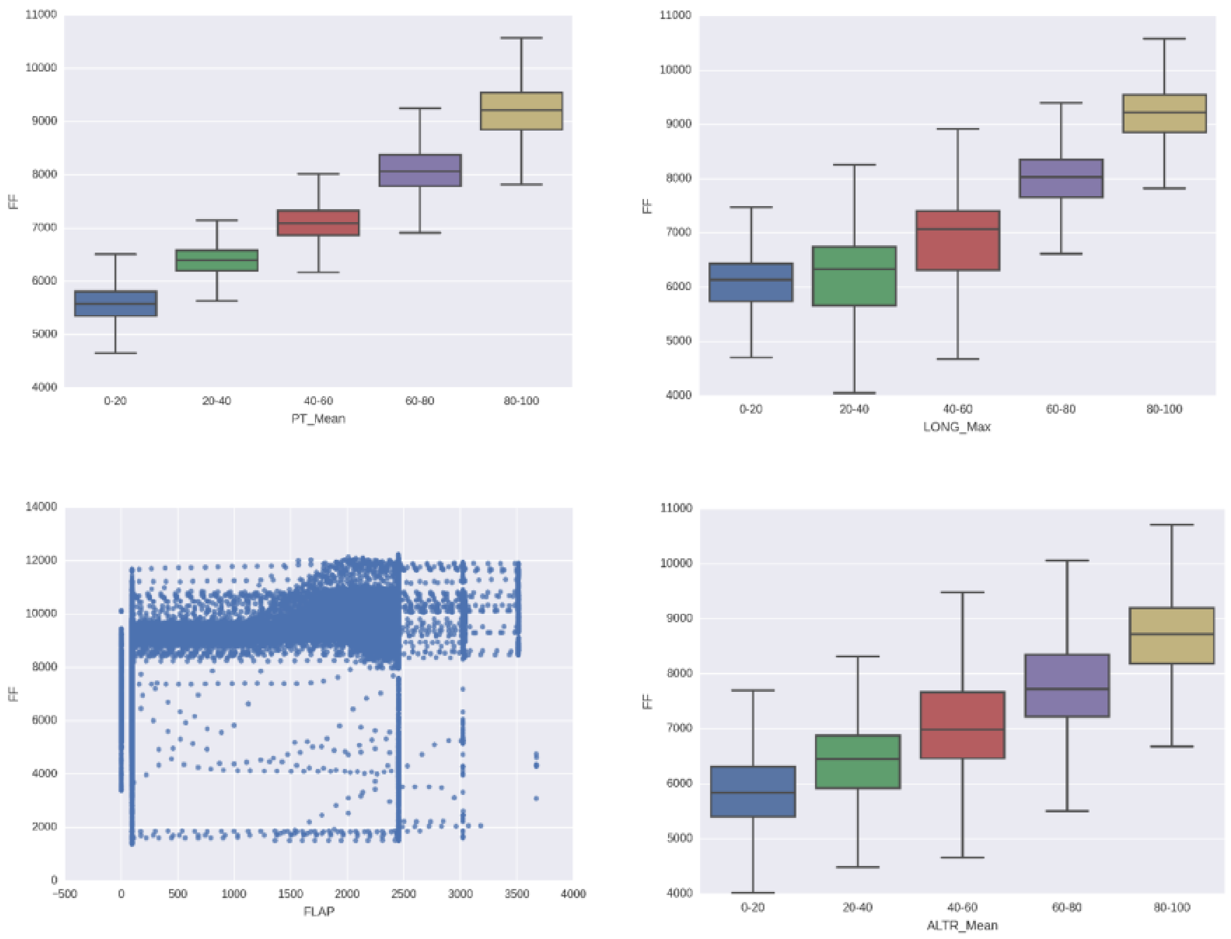


Fig 8 The relationship of fuel flow in the climb phase with top individual predictors - **PT_Mean, LONG_Max, FLAP and ALTR_Mean**. The third graph is shown as simple scatterplots, while the other three are shown as distribution of fuel flow rate in different percentile ranges of predictors. The graph format is chosen for best possible clarity.**The best combination of features as a predictor for climb phase was LONG_Max, OIT_4 and PT_Max.**

PH = 5 (Cruise)

This is the **longest phase of flights**, and constitutes on an average **32%** of total flight time, and so the total fuel consumption in this phase (~ **37.8%**) is more than any other phase. The average fuel consumption **(4966) is** smaller than the **takeoff and climb phase** because the average altitude is very high, and the air is thin. Thus the work done against the drag of atmosphere is small in the cruise phase, even if the ground speed is very high. The important individual predictors of fuel consumption in this phase are **CASS, N1T_Max, LONG_Max and FPAC_Max**, whereas the combined features which could best predict the fuel flow is a group of **PT_Min, CAS_Min, PI_Min and LONG_Min.**

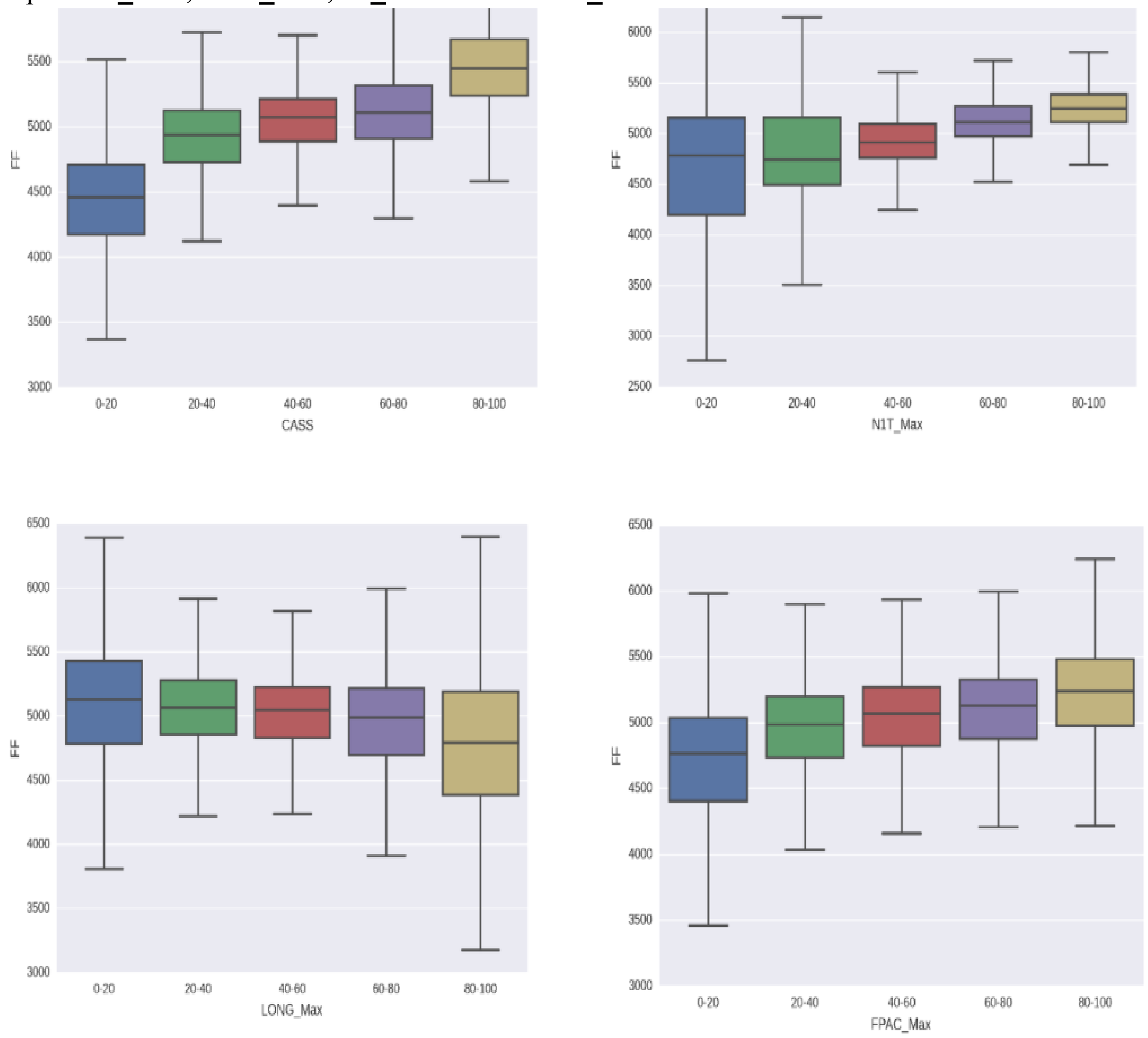


Fig 9 The distribution of fuel flow rate for cruise phase in different percentile ranges of top individual predictors - **CASS, N1T_Max, LONG_Max and FPAC_Max**. The fuel flow rate slightly increases with increase in **CASS, N1T_Max and FPAC_Max**, whereas it decreases slightly with increase in LONG_Max.

PH = 6 (Approach)

At approach, the average fuel consumption **(2940)** is much smaller than **climb, takeoff** or **cruise.**But this phase still accounts for about **13.6%** of total fuel consumed, as the average time duration of this phase is significant (**19.5% of total flight time**). The best individual predictors for this phase are **N1T_Max, MACH_Min, LONG_Max and IVV_Min**. There was no combination of four or less features which could explain even **40%** of the variance in fuel consumption for this phase. Fig 10 below shows a rather weak relationship between fuel flow rate and the top individual predictors.

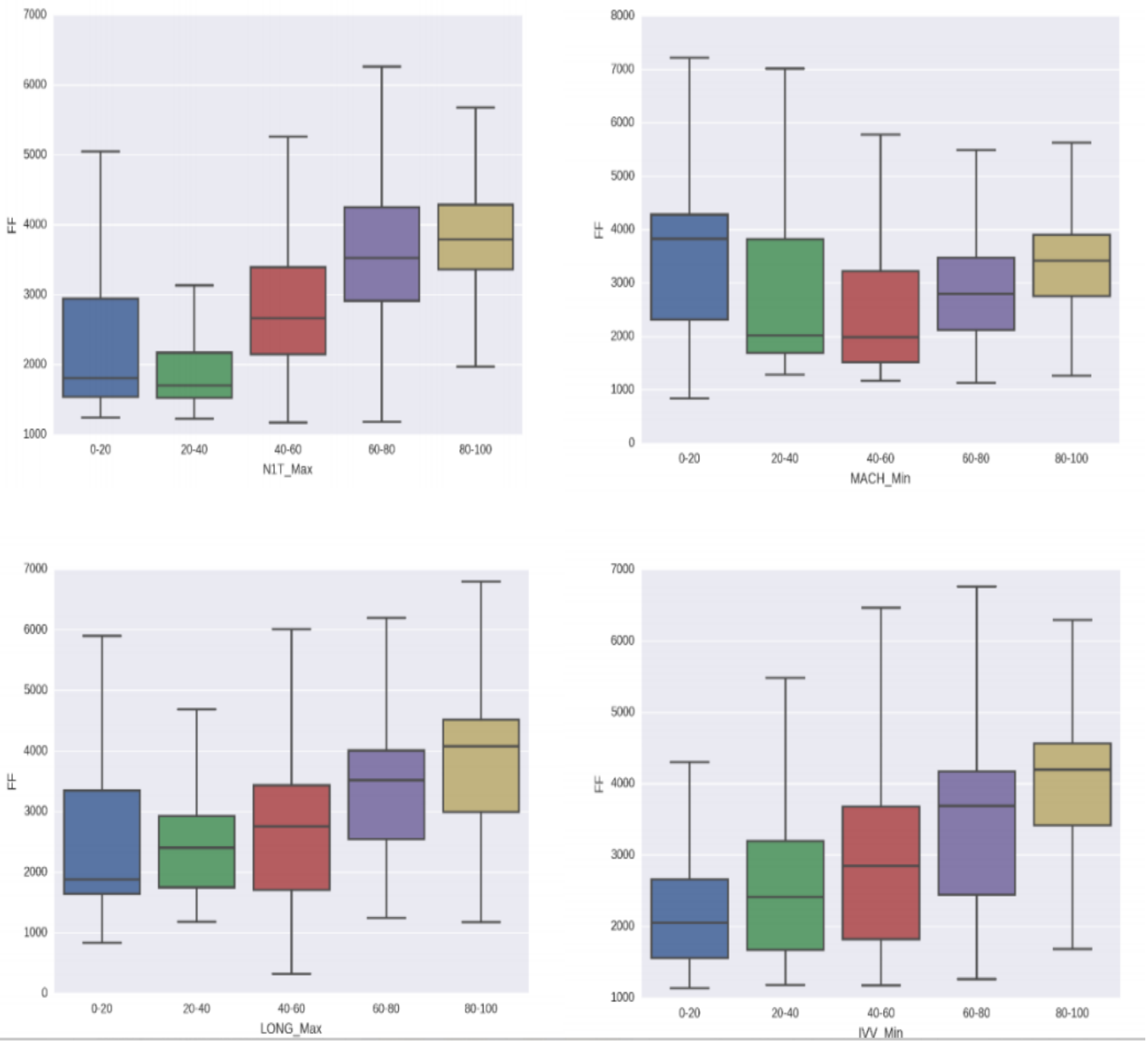


Fig 10 The distribution of fuel flow rate for approach phase in different percentile ranges of top individual predictors -**N1T_Max, MACH_Min, LONG_Max and IVV_Min**. The fuel flow rate slightly increases with increase in N1T_Max,LONG_Max and IVV_Min, whereas it first decreases and then slightly increases with increase in MACH_Min. The visual dependence in each of these graphs is very poor.

PH = 7 (Rollout)

This phase consists of the **smallest amount of time and smallest total fuel consumption**, which is just **0.12%** of the total fuel consumed. Thus this phase does **not** have any **meaningful influence** on the amount of fuel used. Nevertheless, we did modelling to predict fuel flow for this phase and found that the top predictors were **LONG_Max, FADS, FPAC_Mean and CAS_Mean**. The relationship of fuel flow rate with LONG_Max, and FPAC_Mean is shown as scatterplot in fig 11, and those with FADS and CAS_Mean is shown as boxplots with various values and various percentile ranges of predictors respectively.

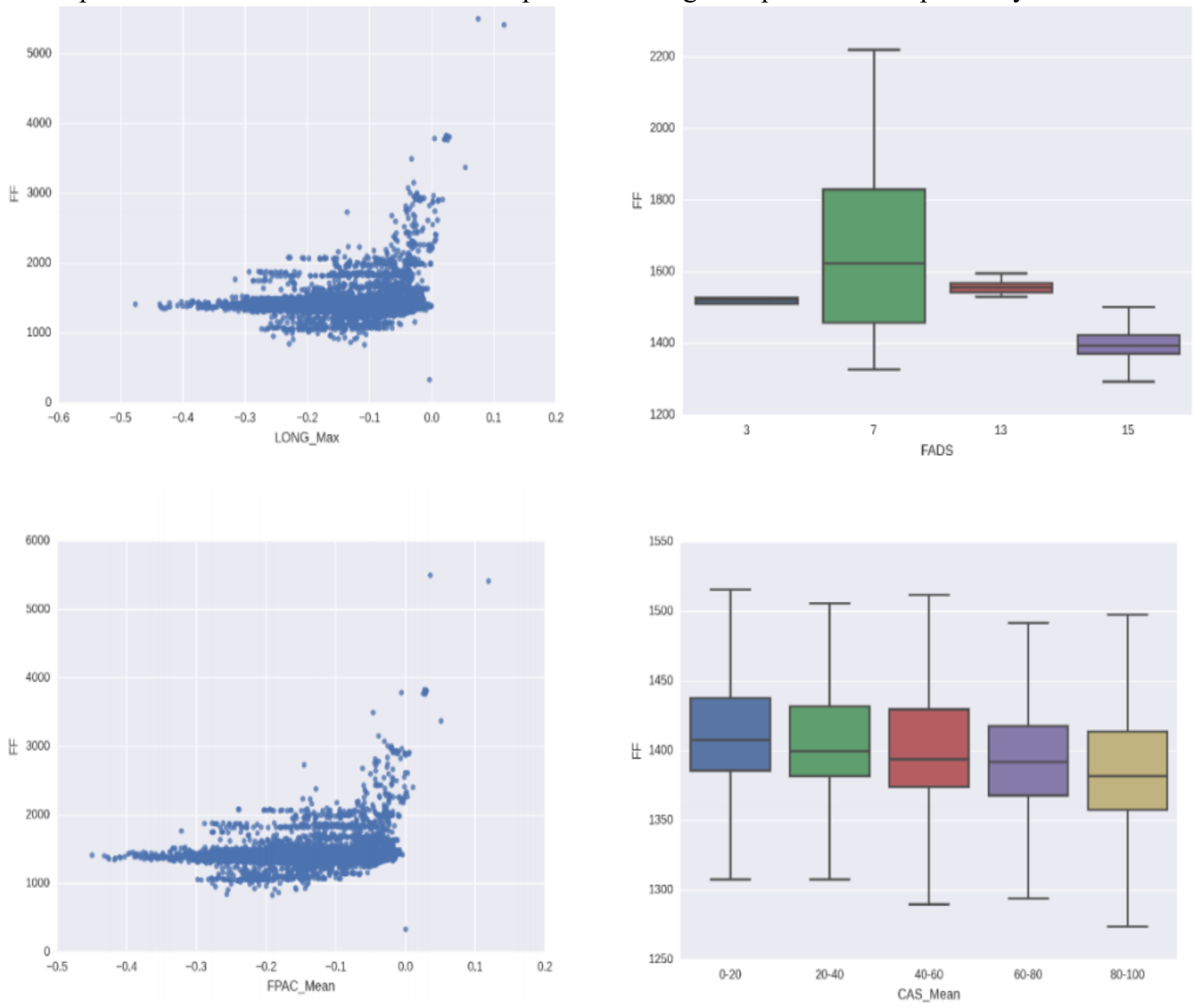


Fig 11 The distribution of fuel flow rate for rollout phase in different percentile ranges of top individual predictors **-LONG_Max, FADS, FPAC_Mean and CAS_Mean**. The dependence of fuel flow on these predictors is not visually clear from the graphs. The format of graph (scatter plot vs boxplot) is chosen for best possible clarity.

CONCLUSION

The most important phases for optimizing fuel consumption are **climb, cruise and approach**. The overall important features are **rate of change of altitude, longitudinal acceleration** and **ground speed**. The clearest visible trend between predictors and fuel flow rate is in the **climb phase**. In other phases, some of the predictors have a weakly visible trend, but since the root mean squared error is small, it is assumed that the features have strong nonlinear interactions which are not clearly visible in simple plots.

REFERENCES

- ❖ Yashovardhan S. Chati*, Hamsa Balakrishnan*, "**STATISTICAL MODELING OF AIRCRAFT ENGINE FUEL FLOW RATE**" , International Council of the Aeronautical Sciences, 2016 , 10 pages
(Last Accessed 18th May 2019)

- ❖ **Predict fuel consumption of airplanes** , Magic Data , (Last Accessed - 18th may 2019)

- ❖ **Predict fuel consumption of airplanes during different phases of flight ,** Crowdanalytix, (Last Accessed - 18th May 2019)

- ❖ **Data Science With Python course** , Datacamp , (Last Accessed - 14th may 2019)

- ❖ **Machine Learning Nanodegree Program** , Udacity (Last Accessed - 28th April 2019)

- ❖ **Data Science for all** channel , Youtube (LAst Accessed 16th May 2019)