INFO 7390 Advances in Data Sciences & Architecture

# Air Pressure Failure Prediction System



**Introduction:**

|  |  |
| --- | --- |
| **Part**  Part 1: Exploratory Data Analysis | **Description**  Conduct EDA using Principal Component Analysis(PCA) |
|  |  |
| Part 2: Feature Engineering | Conduct thorough feature analysis and use pre-processing techniques to make the data usable. |
| Part 3: Predicting Algorithms | Try Logistic Regression, Random forest, Support Vector Machine and XGBoost to build prediction models in using sklearn in Python.  Compute AUC\_ROC\_score for evaluation. |
| Part 4: Serialization | Used pickle operation to serialize our machine learning algorithms and save the serialized format to a file. |
| Part 5: Web Application | Developed a web application using Flask for client -server communication and to take input from client and make predictions according the models developed. |
| Part 6: Dockerization | Docker is used to containerize the application to address it across the cloud. |
| Part 7:  Deployment | Deployed model using Amazon S3 and AWS Lambda |



**Project Tools:**

* Language: Python
* Process: Data Ingestion, Exploratory Data Analysis, Feature Engineering, Prediction Algorithms, Serialization, Pipelining, Dockerizing, Flask Web Application, Scaling with Dask

● Tools used: Jupyter Notebook, Docker, Flask, Dask, Sublime text

**Team:**

Ankur Jain - 001206900

Amandeep Singh - 001271649

Eklavya Saxena - 001850025

**GitHub Link:**

[**https://github.com/eklavyasaxena/Advances-in-Data-Sciences-and-Architecture**](https://github.com/eklavyasaxena/Advances-in-Data-Sciences-and-Architecture)

**PART -1: Exploratory Data Analysis**

Conducted an exploratory data analysis using Python packages (plotly, seaborn, matplotlib etc.) to understand the energy dataset.

**1.a General Tips followed**

Before plotting/joining/doing something, have a question or hypothesis that you want to investigate

* Draw a plot of what you want to see on paper to sketch the idea



• Write it down, then make the plan on how to get there

* How do you know you aren't fooling yourself?
* What else can I check if this is actually true?
* What evidence could there be that it's wrong?

### 

### **1.b Libraries Used**

**Plotting** – Matplotlib, Seaborn

**Wrangling** – Numpy, Pandas

**Modelling** – LogisticRegression, KFold, cross\_val\_score, svm

**Evaluation** - accuracy\_score, roc\_curve, confusion\_matrix, precision\_recall\_curve

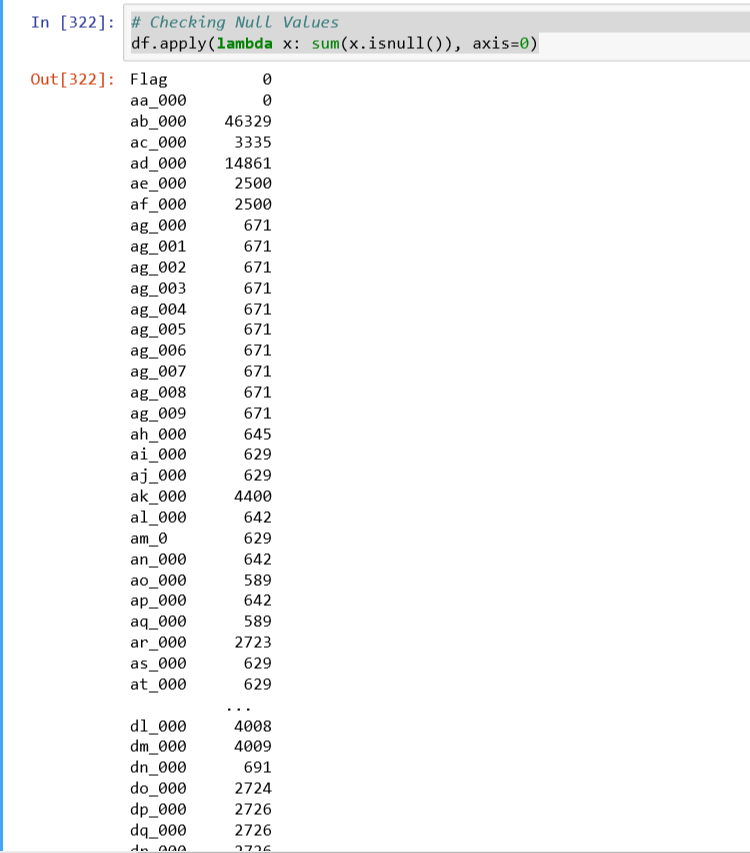
**1.c Description of the Data columns**

The dataset consists of data collected from heavy Scania trucks in everyday usage. The system in focus is the Air Pressure system (APS) which generates pressurized air that are utilized in various functions in a truck, such as braking and gear changes. The dataset’s positive class consists of component failures for a specific component of the APS system. The negative class consists of trucks with failures for components not related to the APS. The data consists of a subset of all available data, selected by experts.

The attribute names of the data have been anonymized for proprietary reasons. It consists of both single numerical counters and histograms consisting of bins with different conditions. Typically, the histograms have open-ended conditions at each end.



**1.d Handling Missing Data**



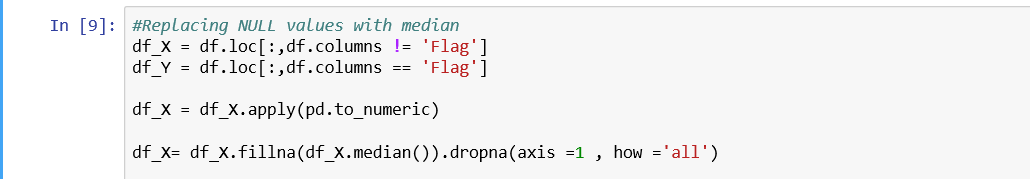


From the above, we observed that every column in this dataset has missing or Null values.

This is highly imbalanced dataset. We will have to do some sampling technique before modelling.

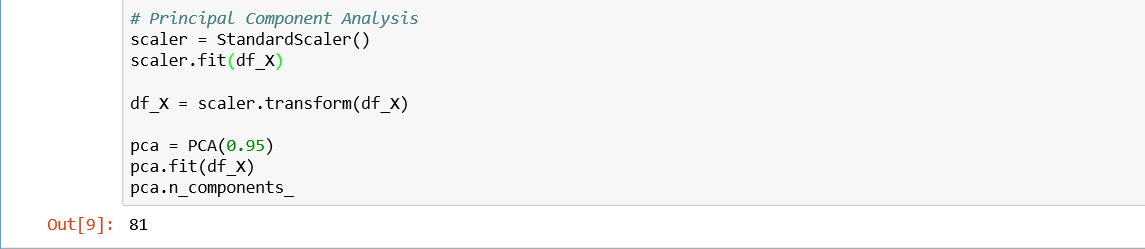
We are imputing NA values in our data with the median value since all the fields are numeric.

Now we are using a median Normalization to remove NULL values.



**1.e Principal Component Analysis**

Principal component analysis (PCA) is a statistical procedure that uses an [orthogonal transformation](https://en.wikipedia.org/wiki/Orthogonal_transformation) to convert a set of observations of possibly correlated variables into a set of values of [linearly uncorrelated](https://en.wikipedia.org/wiki/Correlation_and_dependence) variables called principal components.





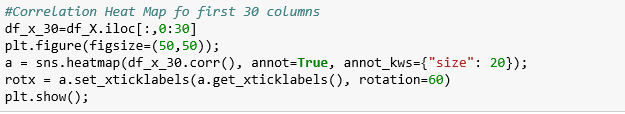
We have reduced number of columns from 171 to 81 here which explains **95 %** of the data.

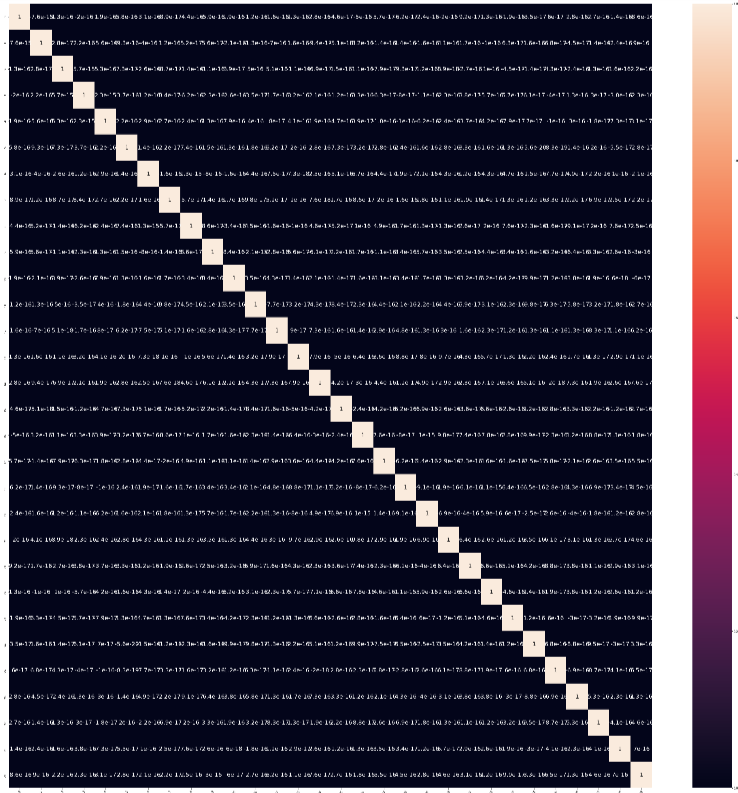
**1.f Correlation Analysis**

The term correlation most often refers to the linear association between two quantities or variables. Correlation analysis is well-known statistical approach used in the study of relationships among multiple physical properties.

Here we have computed the relationship between variables by splitting the dataset.

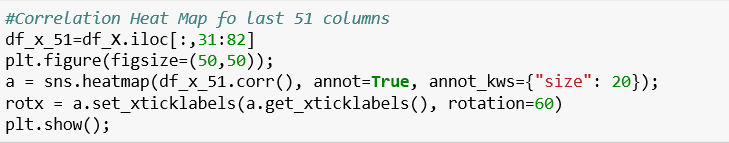
Initially considering first 30 columns of the dataset

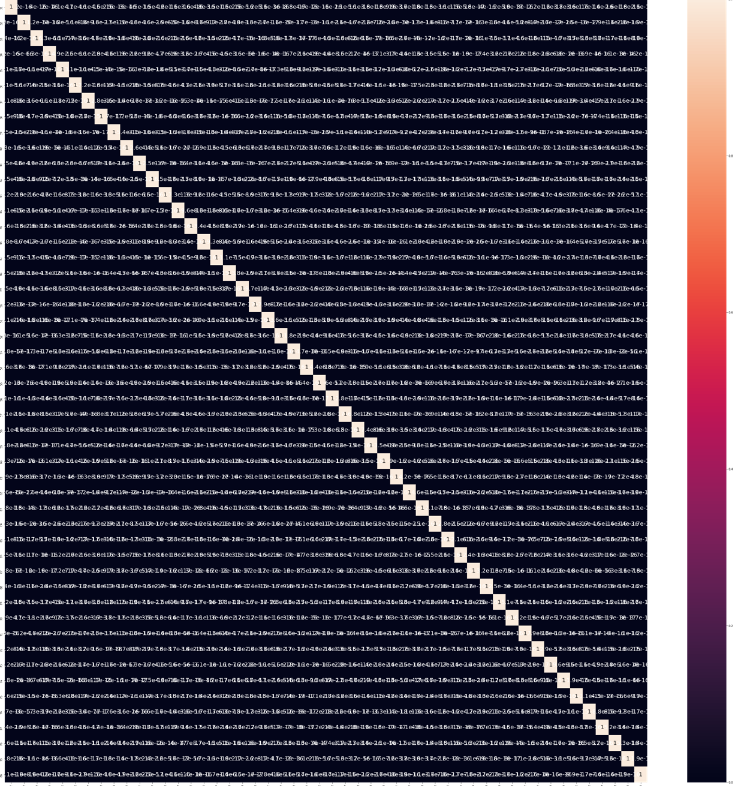






Now considering the other 51 columns of the dataset





From the above correlation matrix, we can say that there is **no co-relation** between the columns. Which is good for our model.

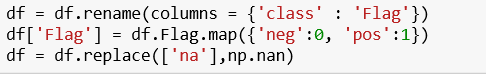


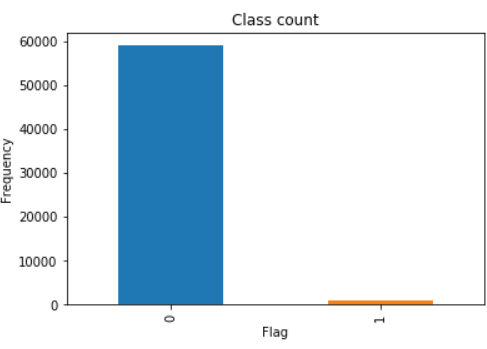
## **PART -2: Feature Engineering**

In the dataset:

* There are columns with a scope of transformation, like class
* Values of target variables can be modified to numeric values

**2.a Renaming the target variable and mapping values**

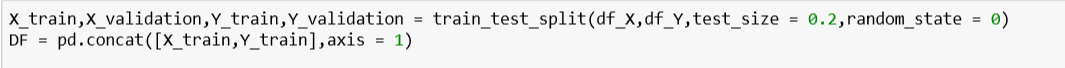




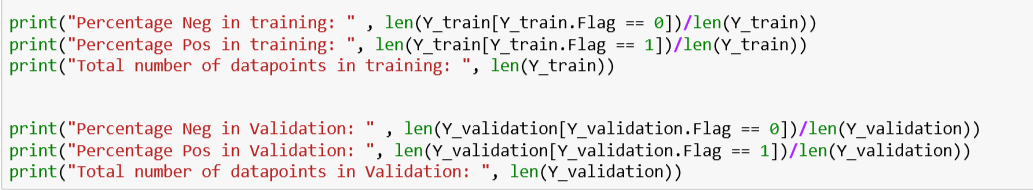
## 

## **PART -3: Predicting Algorithms**

## **3.a Dividing dataset into Training and Testing data**



**3.b Calculating Percentage of Negative and Positive values in target variable**



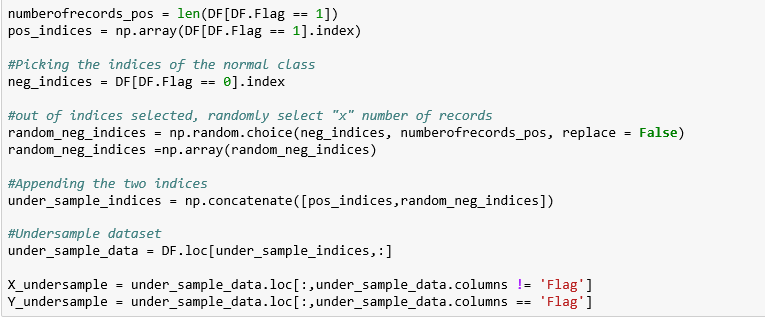


**3.c Sampling**

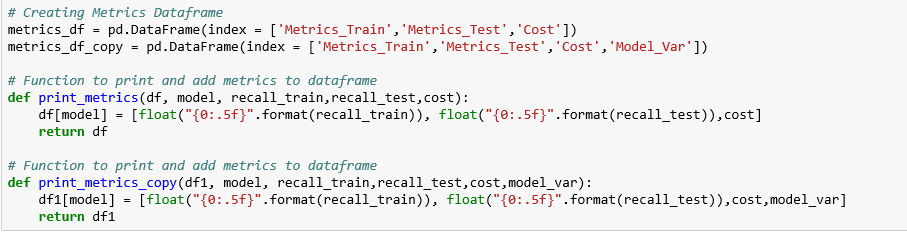
Sampling is the selection of a subset (a [statistical sample](https://en.wikipedia.org/wiki/Statistical_sample)) of individuals from within a [statistical population](https://en.wikipedia.org/wiki/Population_(statistics)) to estimate characteristics of the whole population.

Which is

For this imbalanced dataset we have implemented Under sampling which is a technique used to adjust the class distribution of a [data set](https://en.wikipedia.org/wiki/Data_set) (i.e. the ratio between the different classes/categories represented).



**3.d Creating Metrics Dataframe**







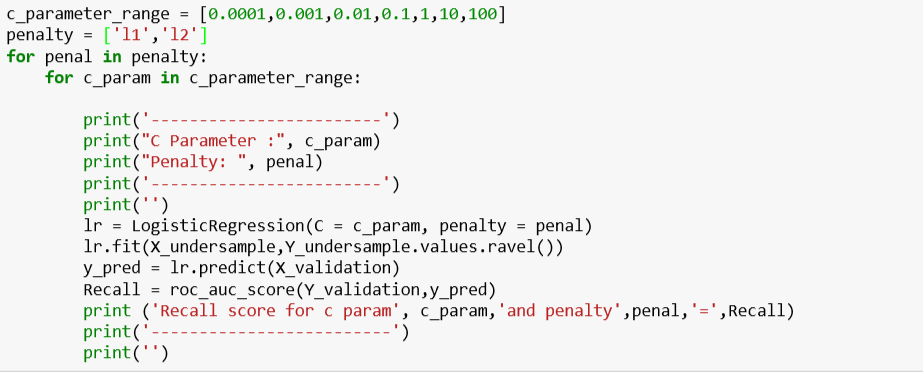
The above functions will print

* **AUC\_ROC\_Score**

It is Area under ROC curve which is most common evaluation metric for binary classification problems to assess how good a machine learning model is.

* **Cost Function**
* **Receiver Operating Characteristic**

**3.e** **Logistic Regression Model**



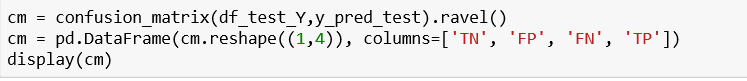


**Parameter range = [0.0001,0.001,0.01,0.1,1,10,100]**

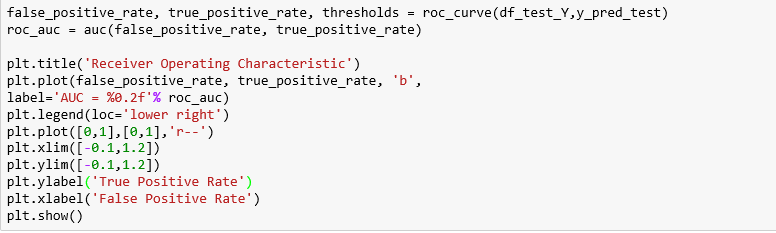
**Penalty = [l1, l2]**

* **Output**

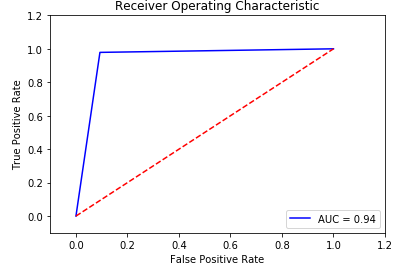
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Parameter | 0.0001 | 0.001 | 0.01 | 0.1 | 1 | 10 | 100 |
| L1 | 0.883299 | 0.883299 | 0.933629 | 0.92795 | 0.928810 | 0.924527 | 0.922148 |
| L2 | 0.899294 | 0.938587 | 0.941856 | 0.92405 | 0.931104 | 0.926737 | 0.925383 |







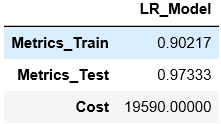




* **Cost Function**

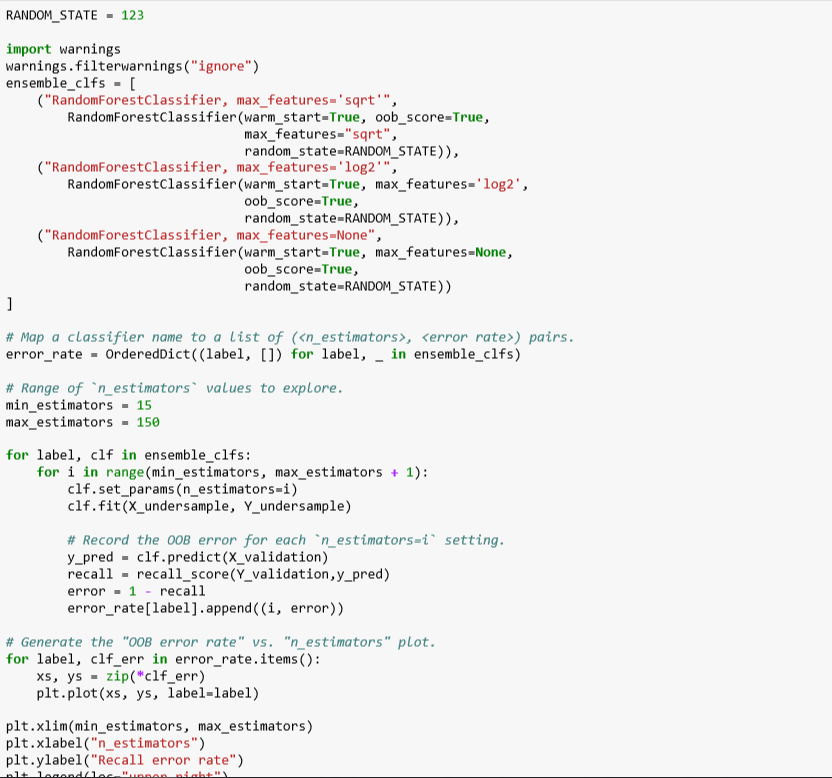


* **Output of Logistic Regression Model:**



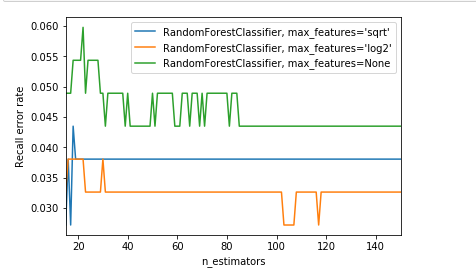


**3.f Random Forest**

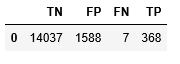


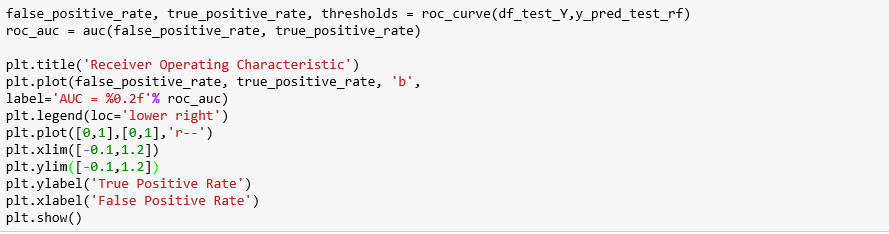


Above will results in below plot

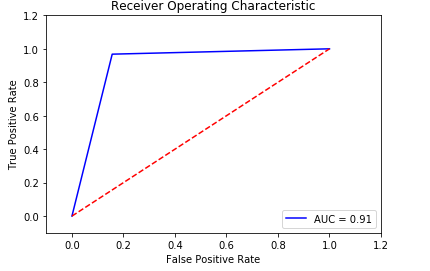








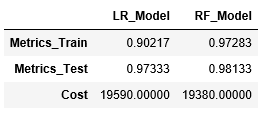




* **Cost Function**

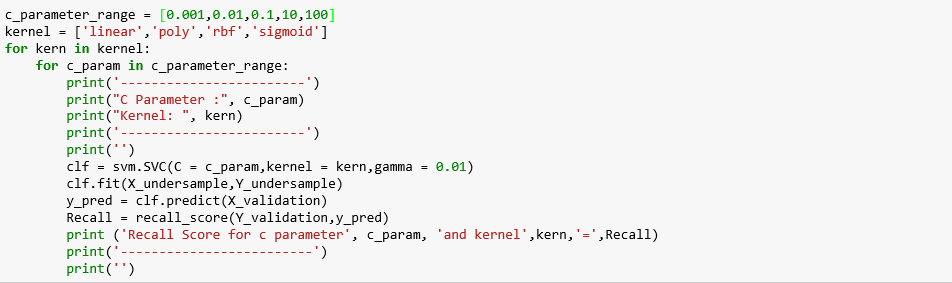


* **Output of Random Forest Model:**





**3.g Support Vector Machine**



**Parameter range = [0.001,0.01,0.1,1,10,100]**

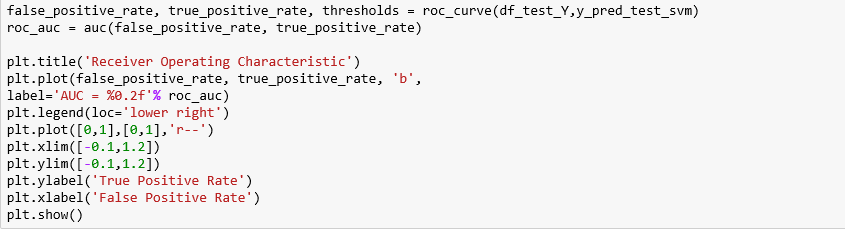
**Penalty = ['linear','poly','rbf','sigmoid']**

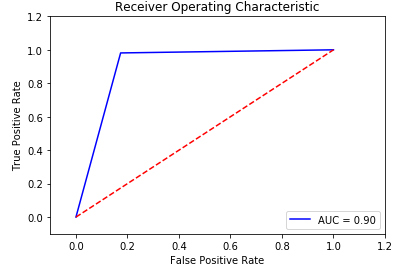
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Parameter | 0.001 | 0.01 | 0.1 | 10 | 100 |
| Linear | 0.932882 | 0.93581 | 0.93530 | 0.92962 | 0.92138 |
| poly | 0.848276 | 0.887641 | 0.92586 | 0.93041 | 0.92741 |
| rbf | 0.909328 | 0.921811 | 0.93760 | 0.94212 | 0.93887 |
| sigmoid | 0.929362 | 0.897541 | 0.82966 | 0.80355 | 0.803508 |









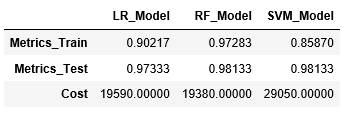


* **Cost Function**

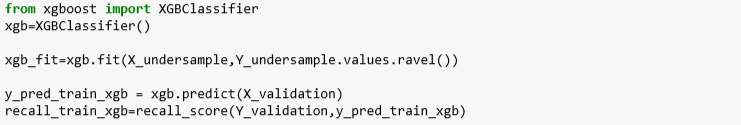




* **Output of Random Forest Model:**



**3.h XGBoost**



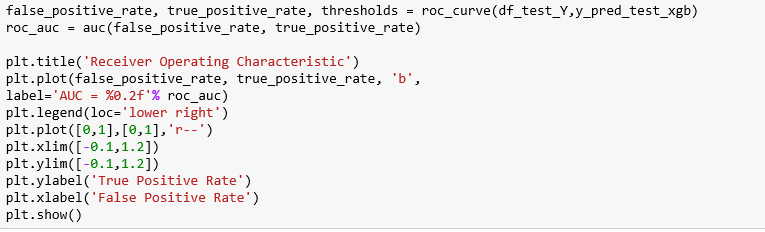


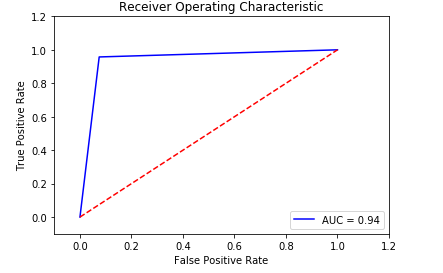




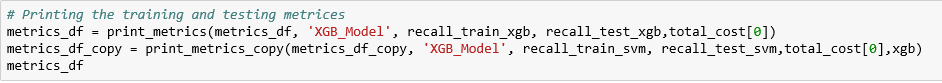
* **Cost Function**



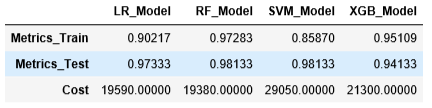




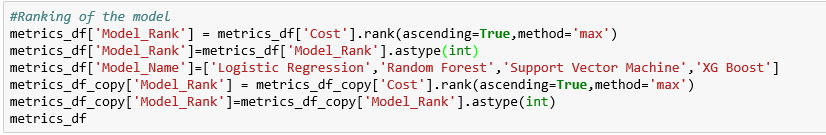




* **Output of XGBoost Model:**

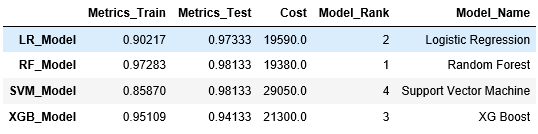


**3.i Ranking the model based on metrics value**





**Output:**



**The above XGBoost model fits best.**

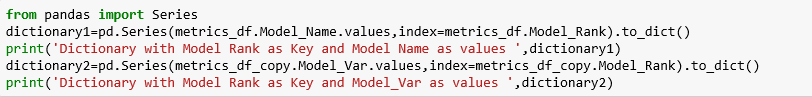
* From the above models, we can say that the SVM is out figured and has accuracy of 98.133 but it has the highest cost. Since, our aim is to reduce the cost, therefore Logistic Regression is the best model for this problem.

**PART-4: SERIALIZATION**

In Data Science, in the context of data storage, **serialization** is the process of translating [data structures](https://en.wikipedia.org/wiki/Data_structure) or [object](https://en.wikipedia.org/wiki/Object_(computer_science)) state into a format that can be stored (for example, in a [file](https://en.wikipedia.org/wiki/Computer_file) or memory [buffer](https://en.wikipedia.org/wiki/Data_buffer)) or transmitted (for example, across a [network](https://en.wikipedia.org/wiki/Computer_network) connection link) and reconstructed later (possibly in a different computer environment).

**Pickle** is the standard way of serializing objects in Python. We can use the pickle operation to serialize our machine learning algorithms and save the serialized format to a file. Later we can load this file to deserialize our model and use it to make new predictions.





* The above will store the Models rank as keys and Models names as values in dictionary 1.
* And it will also save Models ranks as keys and Model\_var as values in dictionary 2.



* Above will save the pickle format in a file.

**PART-5: Web Application**

Website Link - <http://167.99.147.122>

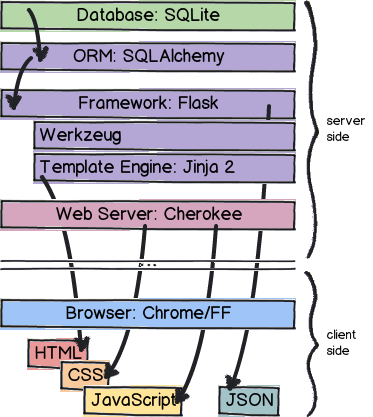
In computing, a web application or web app is a [client–server](https://en.wikipedia.org/wiki/Client%E2%80%93server_model) [computer program](https://en.wikipedia.org/wiki/Computer_program) which the client (including the user interface and client-side logic) runs in a [web browser](https://en.wikipedia.org/wiki/Web_browser).

We have used **Flask** platform to create a web application.



**Flask** is a micro [web framework](https://en.wikipedia.org/wiki/Web_framework) written in [Python](https://en.wikipedia.org/wiki/Python_(programming_language)) and based on the Werkzeug toolkit and [Jinja2](https://en.wikipedia.org/wiki/Jinja_(template_engine)) template engine. Flask is called a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself.

This web app has been setup by primarily following https://pythonprogramming.net/practical-flask-introduction/, and other model deployment examples available online.





**Steps are explained below:**

* **Application Detailing:**

1. Setup DigitalOcean Server (Use referral **https://m.do.co/c/48bcc6762c20** to earn $10 instant credit) with Ubuntu 16.04 with 1 vCPUs and 1GiB Memory and 25GiB Storage

2. Download puTTy and WinSCP

3. Once in the server, start with an update and upgrade:

**sudo apt-get update && sudo apt-get dist-upgrade**

4.Install mysql as primary database:

**sudo apt-get install apache2 mysql-client mysql-server**

5. Install Apache webserver for Python 3.6:

**sudo apt-get install apache2 apache2-dev**

6. Enable wsgi:

**sudo a2enmod wsgi**

7. Create required folders (not necessary to be followed):

**cd /var/www**

**sudo mkdir FlaskApp**

**cd FlaskApp**

**sudo mkdir FlaskApp**

**cd FlaskApp**

**sudo mkdir static**

**sudo mkdir templates**

8. Create & edit a python file in Nano Editor (optional):

**sudo nano \_\_init\_\_.py**

9. Install & update python pip module:

**sudo apt-get install python-pip**

**pip install --upgrade pip**



10. Install Virtual Env package:

**sudo pip install virtualenv**

11. Create Virtual Env to segregate it from webserver and for Flask to run Python and application in it:

**sudo virtualenv venv**

12. Activate the virtual environment:

**source venv/bin/activate**

13. Install Flask in the VirtualEnv:

**sudo pip install Flask**

14. Deactivate the virtual environment:

**Deactivate**

15. Setup the Flask Conf File

**sudo nano /etc/apache2/sites-available/FlaskApp.conf**

**<VirtualHost \*:80>**

**ServerName yourservername**

**ServerAdmin youremailid@gmail.com**

**WSGIScriptAlias / /var/www/FlaskApp/FlaskApp.wsgi**

**<Directory /var/www/FlaskApp/FlaskApp/>**

**Order allow,deny**

**Allow from all**

**</Directory>**

**Alias /static /var/www/FlaskApp/FlaskApp/static**

**<Directory /var/www/FlaskApp/FlaskApp/static/>**

**Order allow,deny**

**Allow from all**

**</Directory>**

**ErrorLog ${APACHE\_LOG\_DIR}/error.log**

**LogLevel warn**

**CustomLog ${APACHE\_LOG\_DIR}/access.log combined**

**</VirtualHost>**

16. Enable the Server

**sudo a2ensite FlaskApp**



**service apache2 reload**

17. Configure the WSGI File:

**cd /var/www/FlaskApp**

**sudo nano Flaskapp.wsgi**

**import sys**

**import logging**

**logging.basicConfig(stream=sys.stderr)**

**sys.path.insert(0,"/var/www/FlaskApp/")**

**from FlaskApp import app as application**

**application.secret\_key = '<any\_random\_gibberish'**

18. To interact with a MySQL database:

**sudo apt-get install python-mysqldb**

19. To install Flask built in forms module called WTForms:

**pip install flask-wtf**

20. Install passlib for password encryption:

**pip install passlib**

21. To email from within the app:

**sudo pip install Flask-Mail**

22. To create SVG (Scalable Vector Graphics) graphs/charts in a variety of styles:

**sudo pip install pygal**

23. To run machine\_learning model, following libraries were installed:

**pip install pandas**

**pip install sklearn**

**pip install scipy --no-cache-dir**

24. For pickle installing dill package:

**pip install dill**

25. For url requests:

**sudo pip install requests**

26. For dask:

**sudo pip install "dask[complete]"**



**PART 6: Dockerization**

Docker is a Container platform to address every application across the hybrid cloud. Today’s businesses are under pressure to digitally transform but are constrained by existing applications and infrastructure while rationalizing an increasingly diverse portfolio of clouds, datacenters and application architectures. Docker enables true independence between applications and infrastructure and developers and IT ops to unlock their potential and creates a model for better collaboration and innovation.

**PART 7: Deployment**

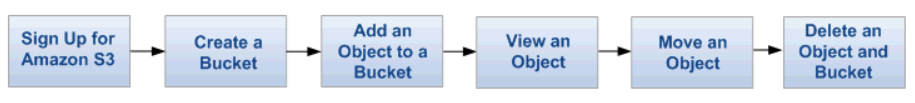
**7.a Amazon S3**

Deployment is the activities that make a [software system](https://en.wikipedia.org/wiki/Software_system) available for use in various environments.

We have used Amazon S3 to containerize and store the model for the internet.

We can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere on the web.

**Steps:**



**7.b Lambda**

**References**

* Luigi:<https://github.com/Atreya22/luigi_rosmann_sales>
* Pickle: <https://pythontips.com/2013/08/02/what-is-pickle-in-python/>
* Amazon S3 :<https://docs.aws.amazon.com/AmazonS3/latest/gsg/GetStartedWithS3.html>