



iNeuron.ai

**Hackathon**

Restaurant Rating

Prediction

**Application URL**

<https://restaurant-rating-prediction.el.r.appspot.com>

Rohan Waje

Email-Id:- rohann2189@gmail.com

## Contents

1. [The Problem statement 3](#_bookmark0)
2. [Application Design 3](#_bookmark1)
3. [Pre-requisites 5](#_bookmark2)
4. [Python Implementation 5](#_bookmark3)
5. [Flask App 10](#_bookmark4)
6. [Deployment to G-cloud 14](#_bookmark5)

**Preface**

## This document is intended to explain the solution for **Machine Learning** problem challenge which is launched by **iNeuron** on<https://challenge-ineuron.in/> .

## We have to choose between 3 bundles which contain 3 problems and deploy the model to the cloud environment like **Amazon AWS/ Microsoft Azure/ Google Cloud Platform**.

## In this document we are providing detail solution for the **“Restaurant Rating Prediction”** from the **“ML Challenge 1”** and deploying on **“Google Cloud Platform”.**

**Machine Learning with Deployment to Google Cloud Platform**

# The Problem statement:

To build an application which predicts restaurant ratings based on given attributes.

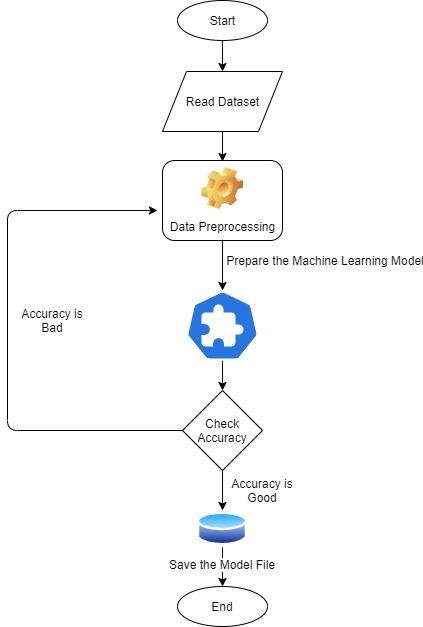
# Application Design:

#### Once we have the data source fixed, the machine learning approach majorly consists of two pipelines:

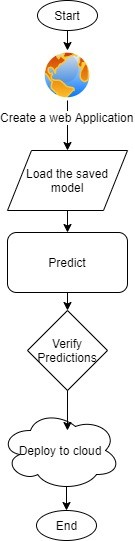
### The Training Pipeline

#### The training pipeline includes data pre-processing, selecting the right algorithm for creating the machine learning model, checking the accuracy of the created model and then saving the model file.

### The Testing Pipeline



#### Once the training is completed, we need to expose the trained model as an API for the user to consume it. For prediction, the saved model is loaded first and then the predictions are done using it. If the web app works fine, the same app is deployed to the cloud platform.



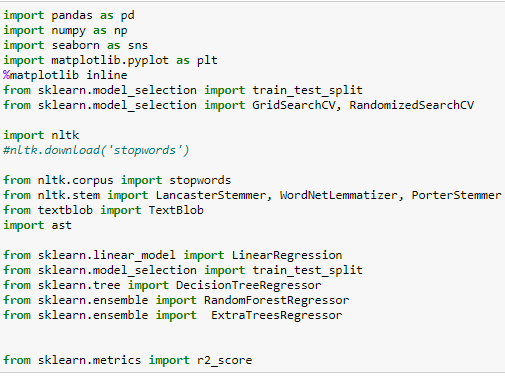
1. **Pre-requisites:**
   * Jupyter Notebook for performing EDA
   * Any Python IDE installed(we are using PyCharm).
   * A Google Cloud Platform account.

.

# Python Implementation:

### Importing the necessary Files

#### We’ll first import all the required libraries to proceed with our machine learning model.

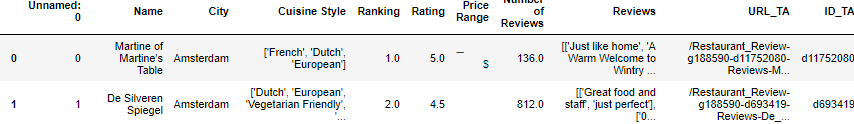


### Reading the Data File

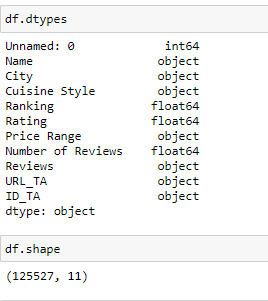
df=pd.read\_csv('TA\_restaurants\_curated.csv')

### Data Pre-processing and Exploratory Data Analysis

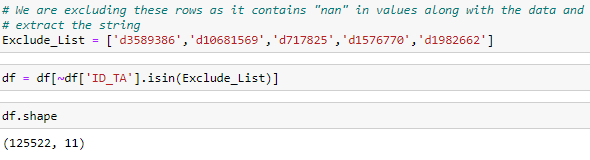
**Checking the sample data**



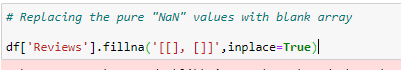
**Analyzing the data**



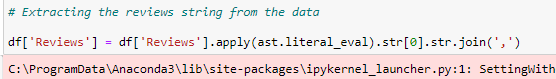
**Excluding the below mentioned records as it contains junk values**

****

**Replacing missing values with blank array**

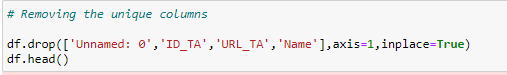
****

**Extracting the reviews string**

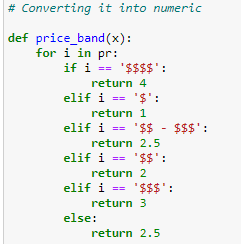
****

****

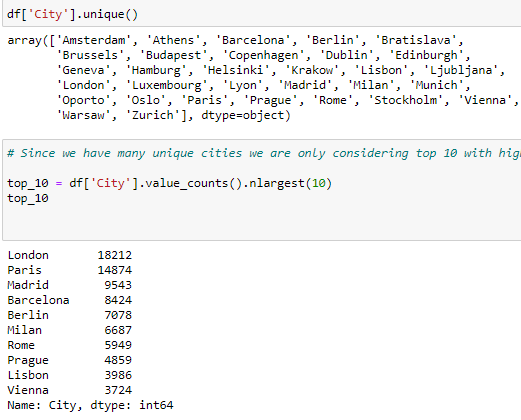
**Removing unique columns**

****

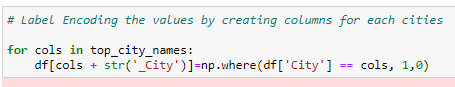
**Converting special characters to numeric for price range**

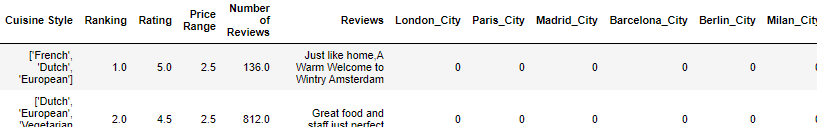
****

**Getting unique City names and getting list of top 10 highest frequencies**

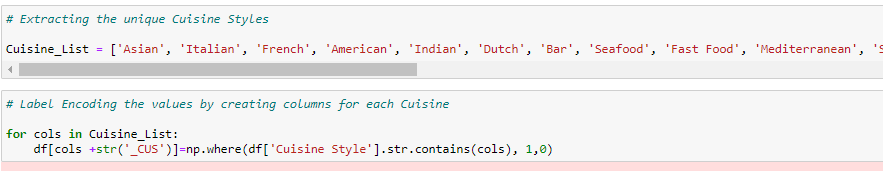
****

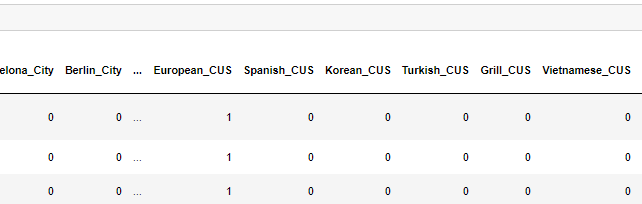
**Label Encoding for each Cities**

****

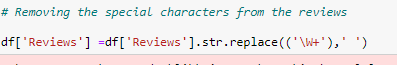
****

**Label Encoding for unique Cuisine Styles**

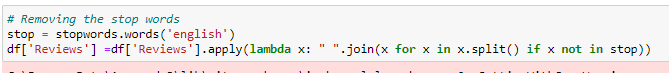
****

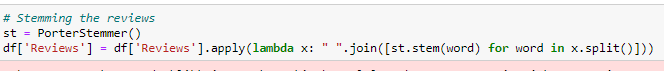
****

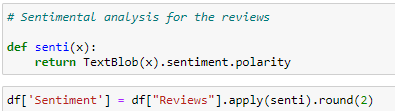
**Performing Sentimental Analysis for the Reviews**

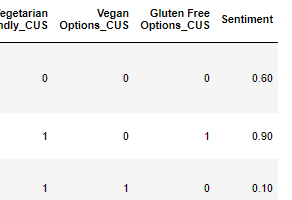
****

****

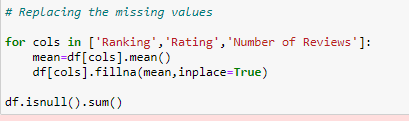
****

****

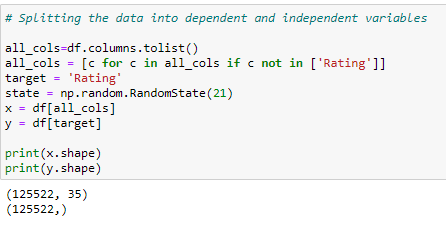
****

****

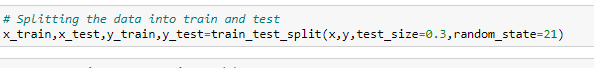
**Handling missing values for other remaining numeric variables**

****

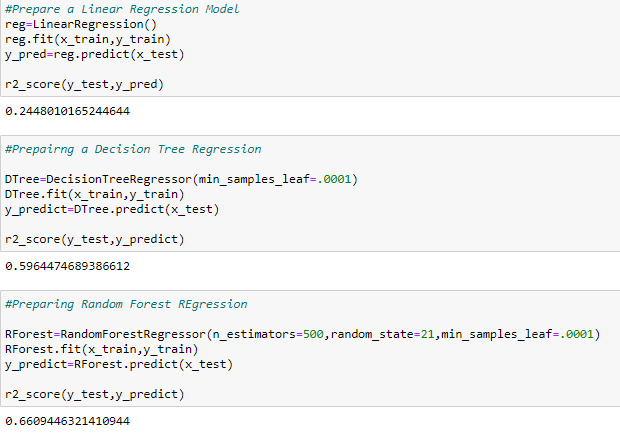
**Splitting dependent and independent**

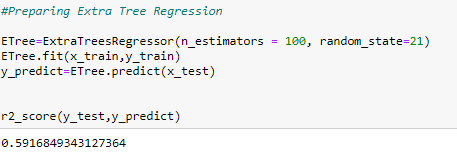
****

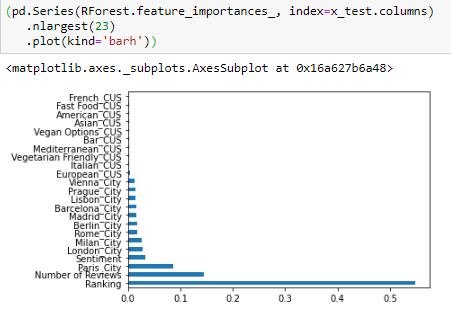
**Splitting data for train and test**

****

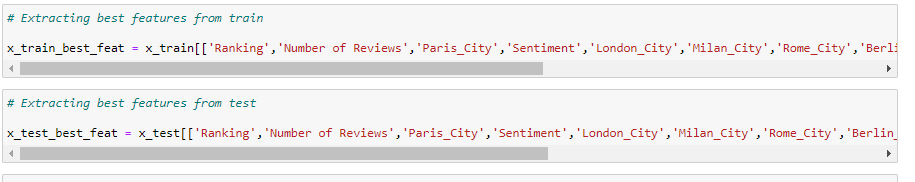
**Checking the accuracy against each model**

****

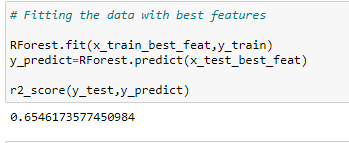
****

****

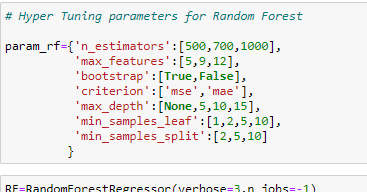
**Selecting best features**

****

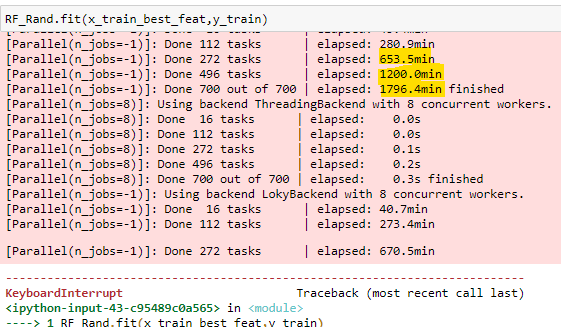
**Checking accuracy with the reduced features**

****

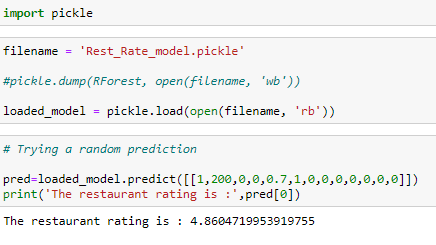
**Hyper Tuning the model**

****

**Note:- We have tried to reduce multiple parameters as the execution was taking more than 4 days but still the execution is pending. So, we have created model with the default parameters.**

****

**Creating the model and predicting with sample**

****

# Flask App:

###### As we’ll expose the created model as a web API to be consumed by the client/client APIs, we’d do it using the flask framework.

###### The flow of our flask app will be:

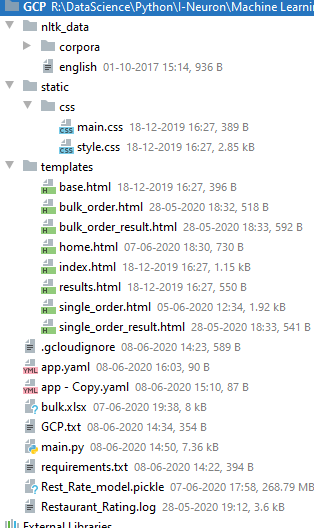
###### 

##### Landing Page (index.jsp)

##### The Python App (main.py)

##### The result page (results.html)

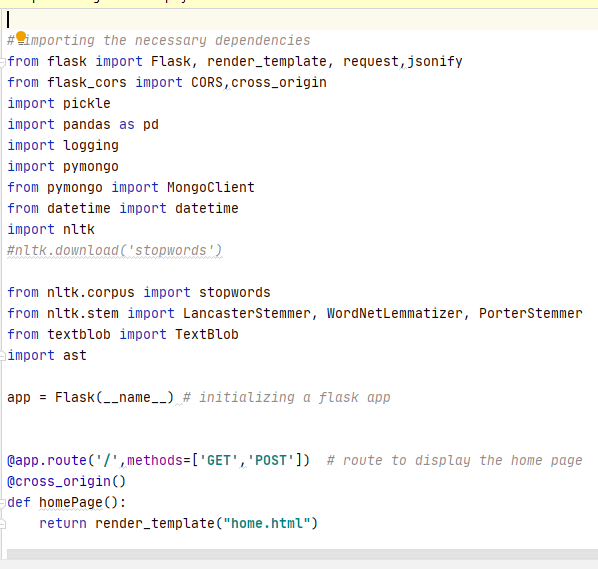
###### **Create the project structure, as shown below:**



**Important files :-**

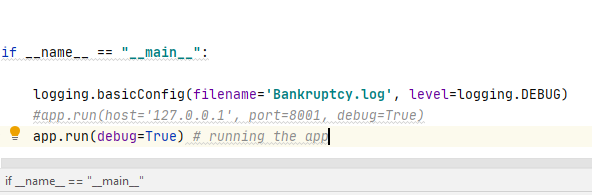
**Static, Templates, app.yaml, main.py, Pickle file, requirements.txt**

**Main.py :**

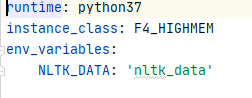
****

****

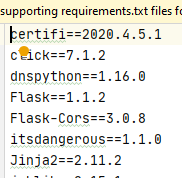
****

****

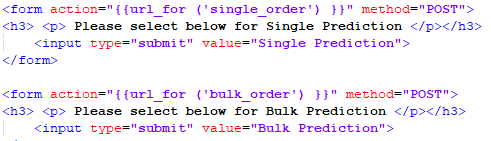
**App.yaml**

****

**Requirements.txt**

****

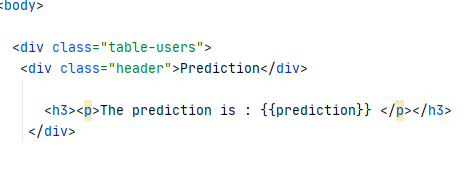
**Home.html**

****

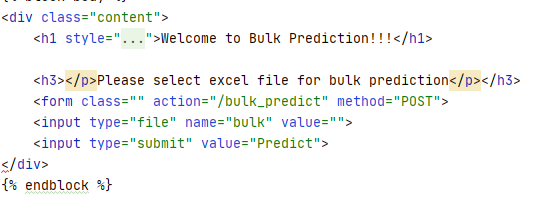
**Single Order.html**

****

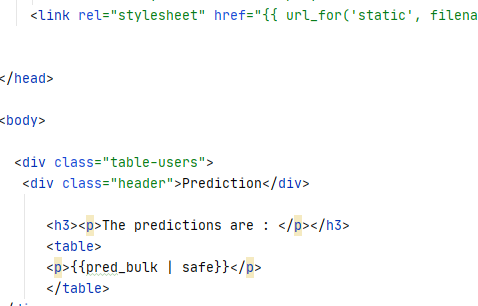
**Single Order Result.html**

****

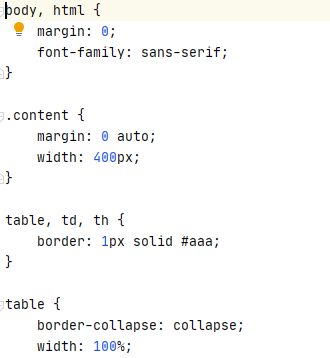
**Bulk Order.html**

****

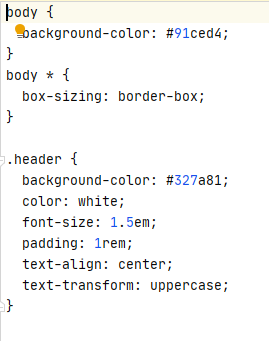
**Bulk Order Result.html**

****

**Main.css**

****

**Style.css**

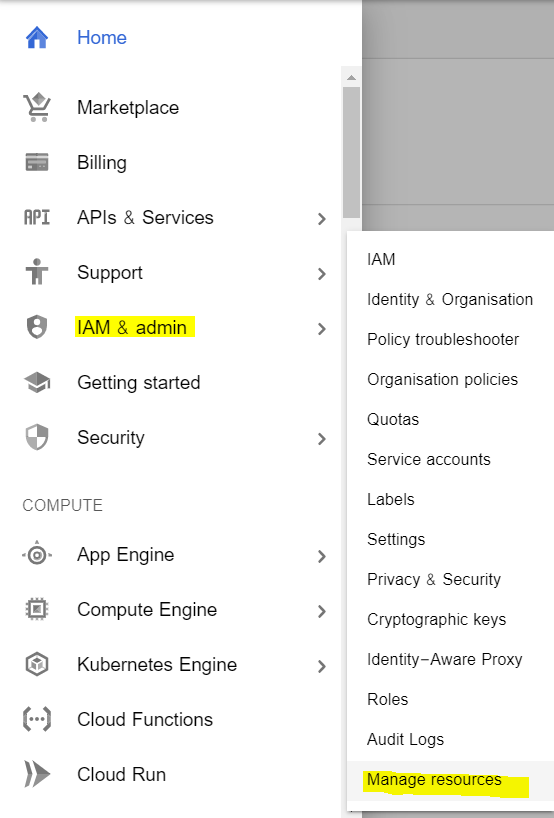
****

# Deployment to G-cloud:

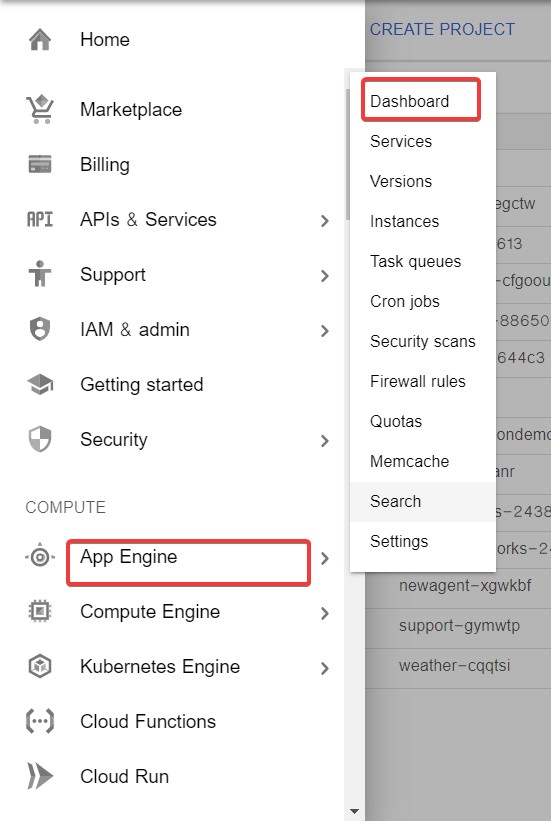
###### Go to <https://cloud.google.com/>and create an account if already haven’t created one.

###### Then go to the console of your account.

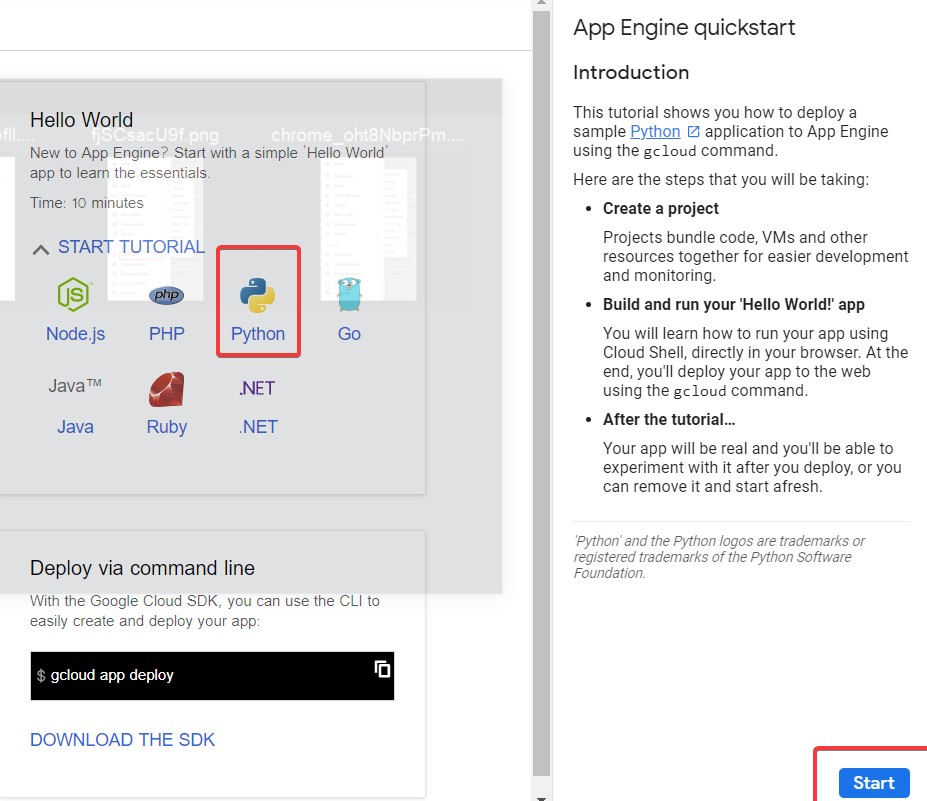
* Go to *IAM and admin*(highlighted) and click *manage resources*.



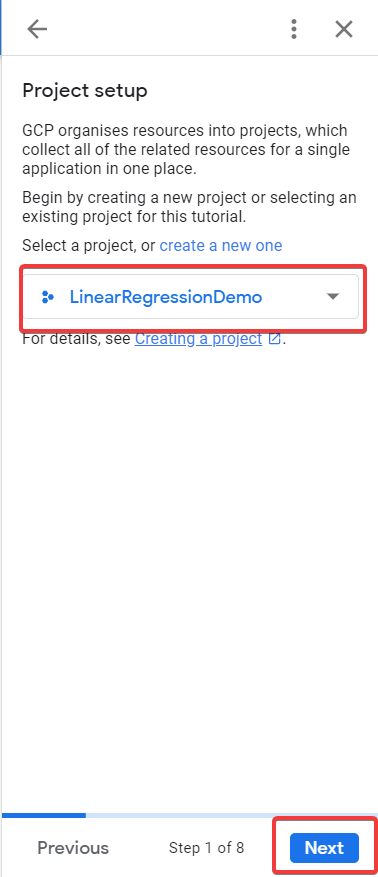
* Click CREATE PROJECT to create a new project for deployment.
* Once the project gets created, select App Engine and select Dashboard.



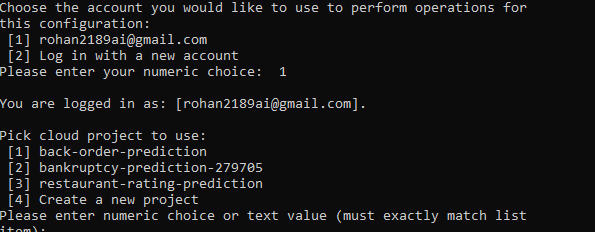
* Go to <https://dl.google.com/dl/cloudsdk/channels/rapid/GoogleCloudSDKInstaller.exe> to download the google cloud SDK to your machine.
* Click Start Tutorial on the screen and select Python app and click start.



###### Check whether the correct project name is displayed and then click next.

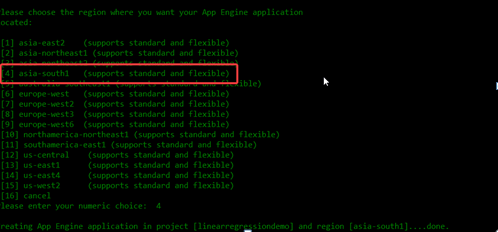


* Create a file ‘app.yaml’ and put ‘runtime: python37’ in that file.
* Create a ‘requirements.txt’ file by opening the command prompt/anaconda prompt, navigate to the project folder and enter the command ‘pip freeze > requirements.txt’. It is recommended to use separate environments for different projects.
* Your python application file should be called ‘main.py’. It is a GCP specific requirement.
* Open command prompt window, navigate to the project folder and enter the command
* gcloud init to initialise the gcloud context.
* It asks you to select from the list of available projects.



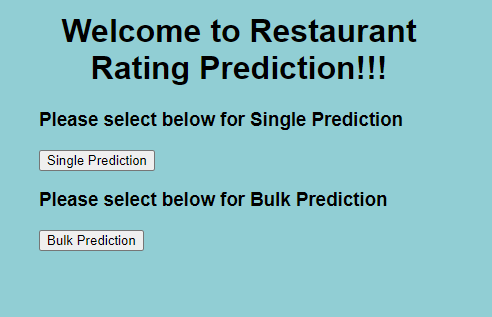
###### Once the project name is selected, enter the command gcloud app deploy app.yaml -- project <project name>.

###### After executing the above command, GCP will ask you to enter the region for your application. Choose the appropriate one.

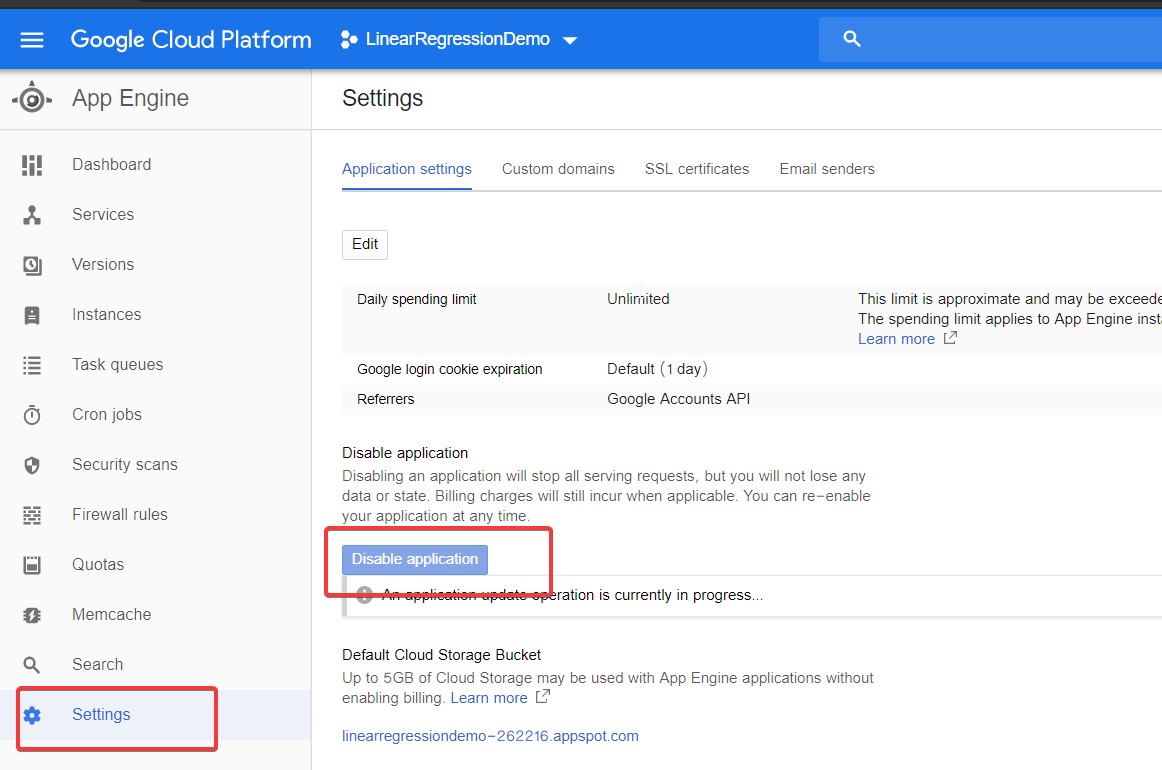


###### GCP will ask for the services to be deployed. Enter ‘y’ to deploy the services.

###### And then it will give you the link for your app, and the deployed app looks like:

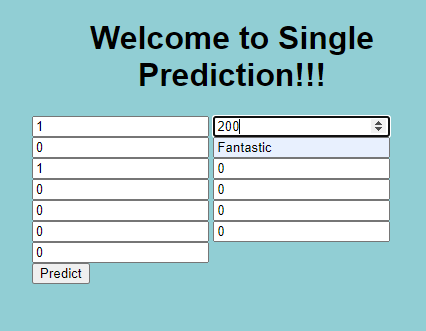


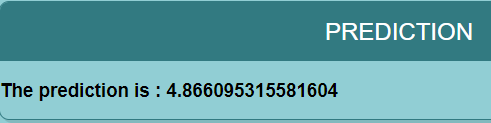
###### To save money, go to settings and disable your app.

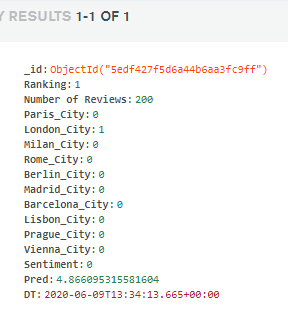


**Validation :-**

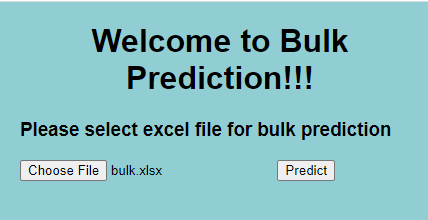
**Single Prediction:-**

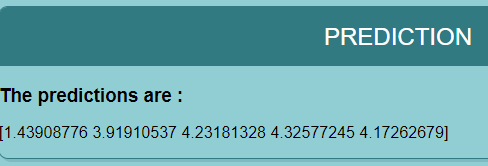






**Bulk Prediction:-**







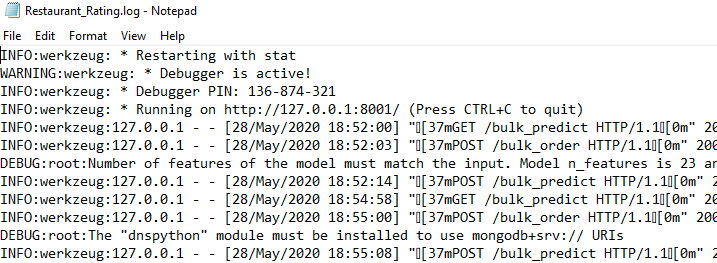
**Retrain Approach :-**

We are storing the data and result in the database so we can randomly

fetch the data and try to check the accuracy. If it is below the

expectation, we will perform the EDA and tuning again.

**Logs:-**

****

Thank You!