ASSIGNMENT 2 - DATA ANALYTICS APPLICATION

UK SALARY PREDICTION MODEL BASED ON RANDOM FOREST REGRESSOR

GROUP NUMBER 10

MEMBER DETAILS:

- TEENU PRATHYUSH (GROUP COORDINATOR)

  STUDENT ID:21262966

  EMAIL ID:teenu.prathyush2@mail.dcu.ie
- NIRANJAN NIMBARTE

  STUDENT ID:21264812

  EMAIL ID:niranjan.nimbarte2@mail.dcu.ie
- TEENA SHARMA

  STUDENT ID:21261593

  EMAIL ID:teena.sharma5@mail.dcu.ie
- FIAZ ASHRAF

  STUDENT ID:21263700

  EMAIL ID:fiaz.ashraf2@mail.dcu.ie
- DIVYASHREE PARAMESHWARA

  STUDENT ID:20211141

  EMAIL ID:divyashree.parameshwara2@mail.dcu.ie
- ANISH PATIL

  STUDENT ID:21260453

  EMAIL ID:anish.patil4@mail.dcu.ie

LINK FOR THE DATASET

Dataset Link: https://www.kaggle.com/c/job-salary-prediction/data

1. INTRODUCTION AND MOTIVATION:

Salary plays a vital role in determining career progression as well as in determining occupational success. Additionally, the pay rate is also considered an influential motivator. With the increasing number of jobs available, it becomes even more crucial to determine how much a particular job can offer in monetary terms. Returns from investments in higher education have a strong correlation with salary outcomes and, hence, influence a student's decision in choosing a degree program or field of study.

Recent trends show that there has been a significant rise in the number of jobs available in the market in a wide range of different domains. It is possible to find a huge number of Job roles on e-Recruitment websites while applying for a job. Still, the underlying problem is that salary insight is often hidden or not readily available to job seekers. To find a solution to this problem, we propose an application that can be used to determine the salaries of the various roles in the dataset. This could be useful for gaining valuable insights. In our application, we analyze the United Kingdom job salary dataset from Kaggle and utilize the **Random Forest Regressor** to predict salaries. These predictions would be helpful for people living in the United Kingdom and those who live overseas who want to land a job in the U.K. Our web application is developed using the Flask framework and integrates machine learning as a key component for making predictions. Out of many possible models in machine learning, we have chosen the **Random Forest Regressor** model since it provided superior results.

2. RELATED WORK:

We can see different types of applications on the internet considering salary prediction. One of them is Morgan Mckinley for salary prediction applications. This application predicts the salary for employees from lots of different disciplines which also can influence the prediction based on different cities in Ireland. There is also an application we found similar to our project which focuses on the challenge of predicting the salary offered by companies through job posts on the web. This application analyses job advertisements collected from Tecnoempleo, an e-Recruitment website specialized in information technology jobs in Spain. The classifiers implied on this application based on ensembles of decision trees (AB and RF) and based on voting ensembles (Vote and Vote3) achieve the best accuracy. The application's average accuracy is ≈ 0.84, with the voting classifiers that lead to a slightly better median accuracy (≈ 0.841 for Vote and ≈ 0.85 for Vote3).

3. DESCRIPTION OF THE DATASET:

Link to the dataset: https://www.kaggle.com/c/job-salary-prediction/data

The dataset we used in our project was downloaded directly from Kaggle. The dataset consists of two parts - The training set and Test Set which is used for the Random Forest Regressor Model. We use the Training set for building our machine learning model.

In the training set, we have approximately 240,000 rows and 12 columns, but for our model we only require 7 columns, therefore, we removed the 5 unusable ones that are Id, FullDescription, LocationRaw, SalaryRaw, SourceName. The 7 columns that are used in our model are described below.

- **Title** column provides the designation of the particular employee as per the data.
- **LocationNormalized** is a normalization of the LocationRaw column that provides information about the city in which the company is established.
- **ContractType** column gives us information on whether the employee is working on a contract basis or working in a permanent job.
- **ContractTime** column gives us information on whether the employee is working full-time or part-time.
- **Category** column provides the field of the job for eg. Engineering, IT, etc
- **SalaryNormalized** is a normalization of the SalaryRaw column that provides the salary of the employee. This is also the target variable of our machine learning model where we will predict the salary for an employee looking for a job.
- **Company** column provides information about all the companies based in the UK.

4. DESCRIPTION OF DATA PROCESSING:

Link to the Pig Script: https://gitlab.computing.dcu.ie/prathyt2/ca675-assignment-2_group-10/-/blob/master/Code/Pigscript.pig

The dataset contains a large number of null values. It is necessary to clean the data before training it with the machine learning model. We have chosen Pig Technology on a Dataproc Cluster (in Google Cloud Platform) for cleaning the dataset. Using Pig, we have selected only the necessary columns and removed all the entries containing null values in the ContractTime and Company Columns. The ContractType column also contains many null values but based on the information present in the dataset we have normalized the missing values in this column.

The majority of the data processing/cleaning has been done on Pig. After cleaning the data on Pig we have used R programming to remove some of the messy characters in the title column. The rest of the columns are clean and do not require any processing.

5. TECHNOLOGIES USED:

Front-end: HTML , CSS , Bootstrap , jQuery, JavaScript.
Back-end: Python
Machine Learning Model: Random Forest Regressor
Web Framework: Flask
Libraries: scikit-learn, NumPy, pandas, flask-cors, pickle
Other applications used: Microsoft Excel, R-Studio, Jupyter Notebook in GCP.

The web application is hosted on the Google AppEngine platform present in the GCP. Google Cloud SDK is used to manage the resources and deploy the application.

6. DEVELOPMENT OF THE APPLICATION PLATFORM:

The following steps have been taken in developing the application:

- The dataset is obtained from the Kaggle website.
- The dataset is then imported into the Google Cloud Staging Bucket present in GCP.
- We load the data into Pig and then process/clean the data.
- The processed data is then loaded into the Machine Learning model (Random Forest Regressor) for training.
- The trained machine learning model produces an accuracy score of 78% and is then saved as a pickle file (serializing into byte stream) for later use.
- The front-end of the application is developed using HTML, CSS, JavaScript, jQuery, and Bootstrap.
- A **main.py** file is then created to build the application using the front-end html file and integrate the trained machine learning model, which was previously saved as a pickle file.
- Flask framework is used for developing the application.
- The application is then hosted on the Google AppEngine platform in GCP using the Google Cloud SDK shell.
- The estimated salary is obtained by entering the values in the given form.

7. RELEVANT SCREENSHOTS:

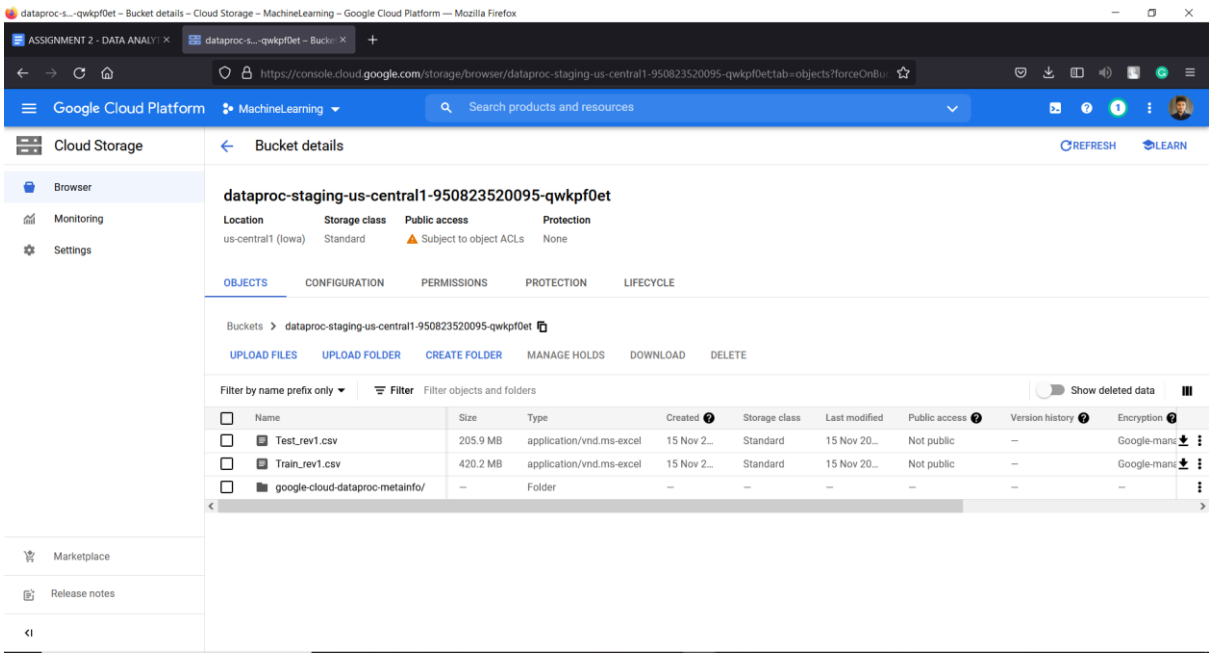


Figure 1: Cloud Storage - Google Cloud Staging Bucket

Figure 1 displays the datasets imported into the Google Cloud Staging Bucket for further processing.

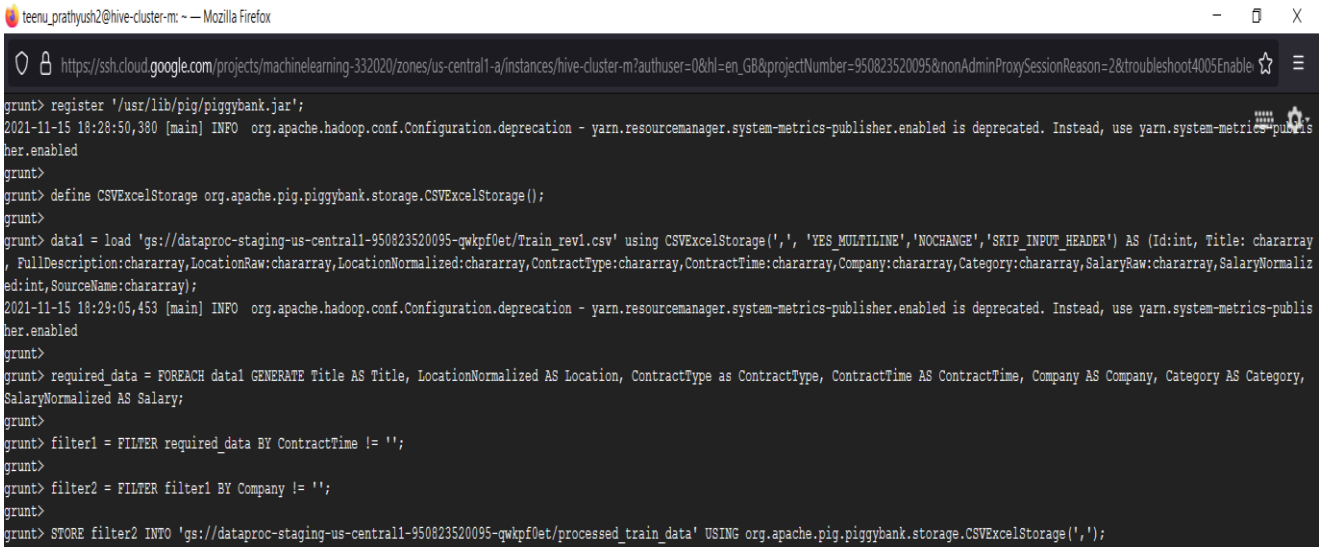


Figure 2: Pig Commands

Figure 2 displays the commands run in the Pig shell for processing/cleaning the data

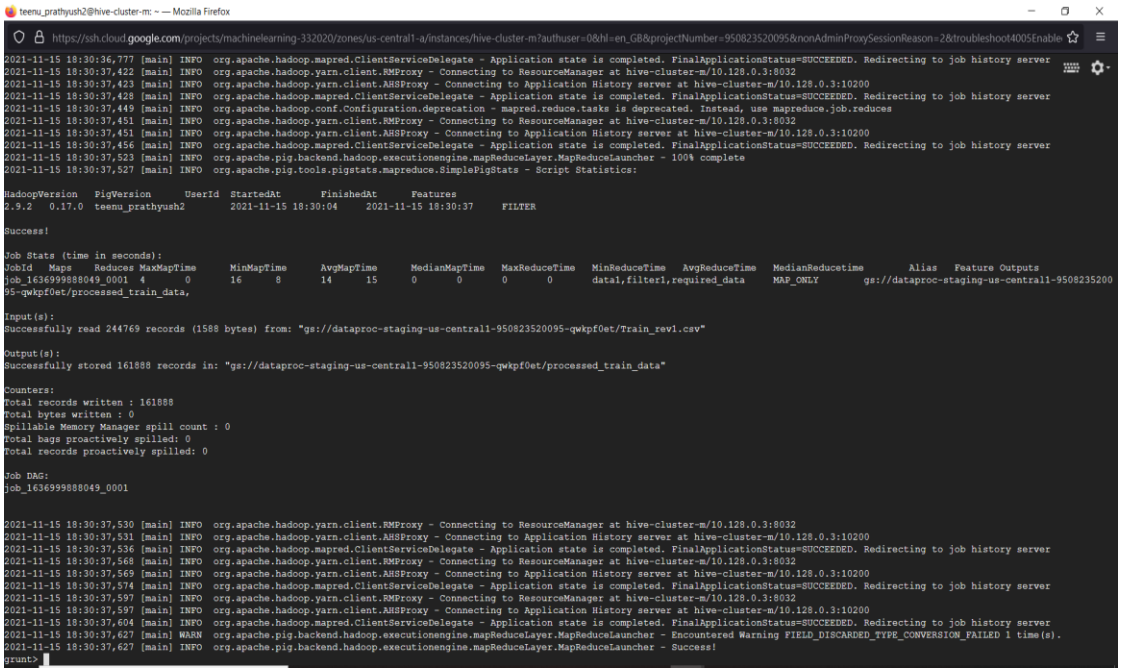


Figure 3: Pig-processed data output

Figure 3 - The processed data is stored back into the Google Cloud Staging bucket using the STORE command. After removing the null values we are left with 160,000 rows (approx.) out of 240,000 rows(approx.) in total. The same procedure is repeated for the test dataset.

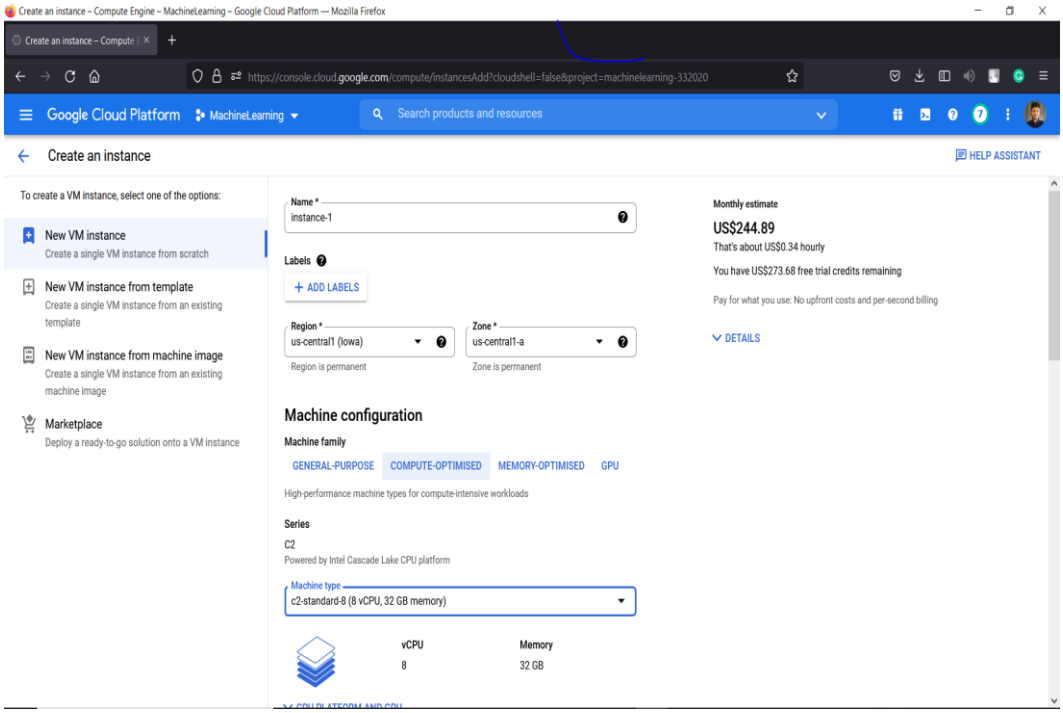


Figure 4: Creating an instance on Google Cloud Platform

Figure 4 displays the process of creating an instance for using Jupyter Notebook on the Cloud.

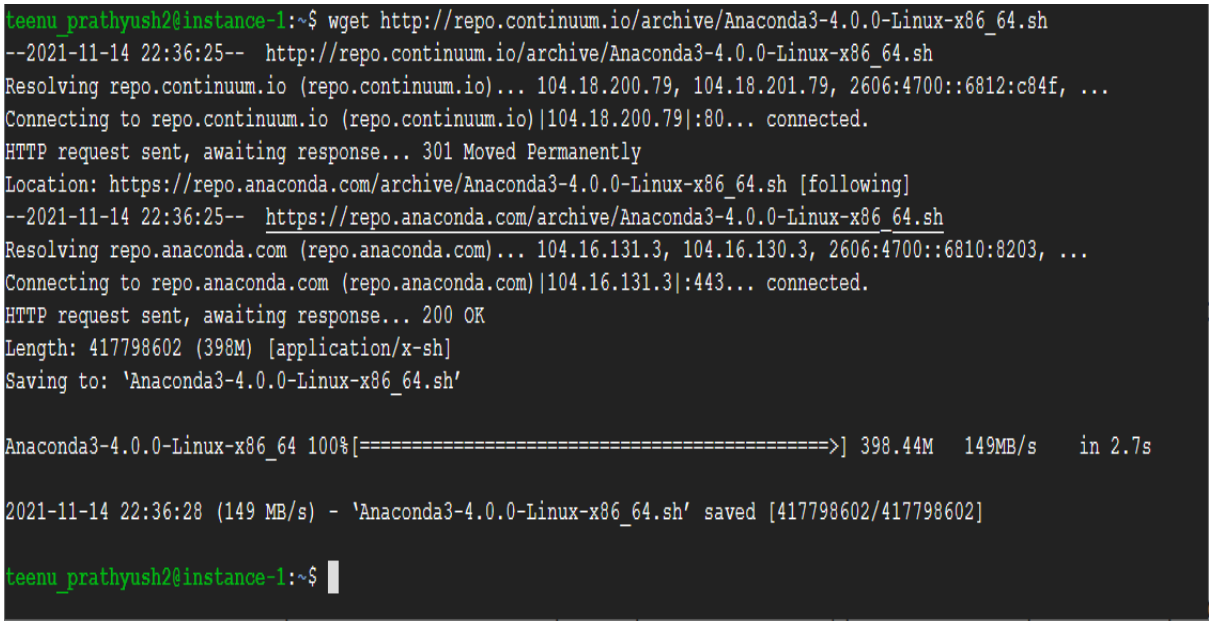


Figure 5: Installing Anaconda on the created instance

Figure 5 displays the installation process of Anaconda, which is required for using Jupyter Notebook.

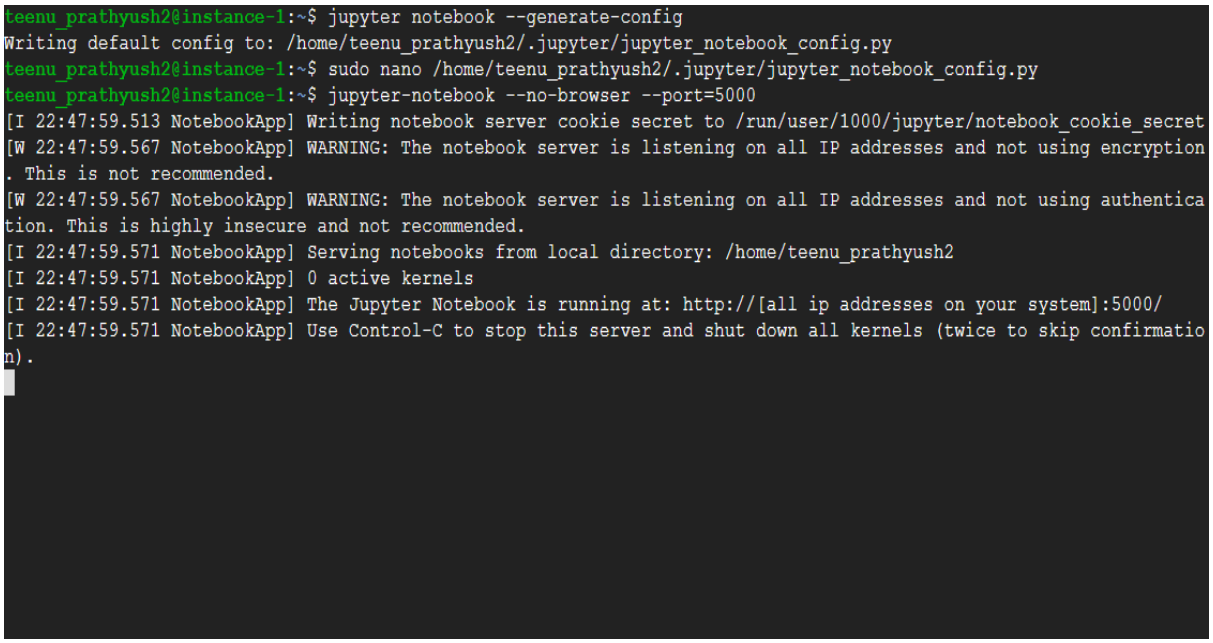
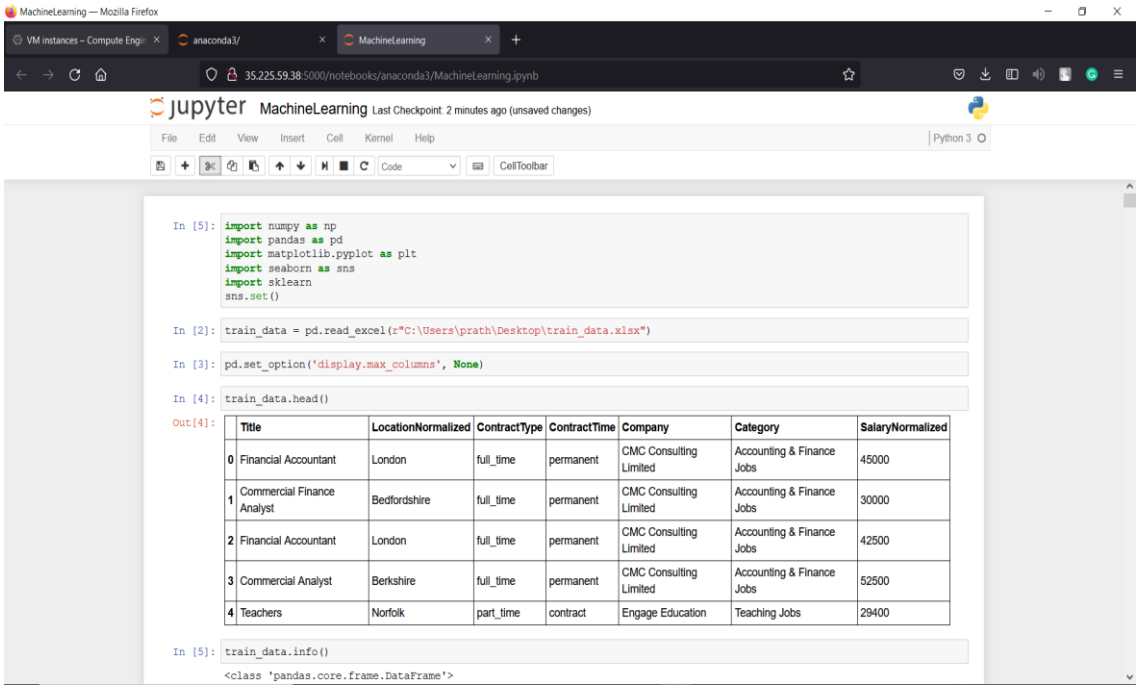


Figure 6: Launching Jupyter Notebook

Figure 6 displays the process of launching Jupyter Notebook on Google Cloud Platform which will be used to create our machine learning model.



	Title	LocationNormalized	ContractType	ContractTime	Company	Category	SalaryNormalized
0	Financial Accountant	London	full_time	permanent	CMC Consulting Limited	Accounting & Finance Jobs	45000
1	Commercial Finance Analyst	Bedfordshire	full_time	permanent	CMC Consulting Limited	Accounting & Finance Jobs	30000
2	Financial Accountant	London	full_time	permanent	CMC Consulting Limited	Accounting & Finance Jobs	42500
3	Commercial Analyst	Berkshire	full_time	permanent	CMC Consulting Limited	Accounting & Finance Jobs	52500
4	Teachers	Norfolk	part_time	contract	Engage Education	Teaching Jobs	29400

Figure 7: Creating the Machine Learning model

Figure 7 displays the creation of the machine learning model on Jupyter Notebook in Google Cloud Platform.

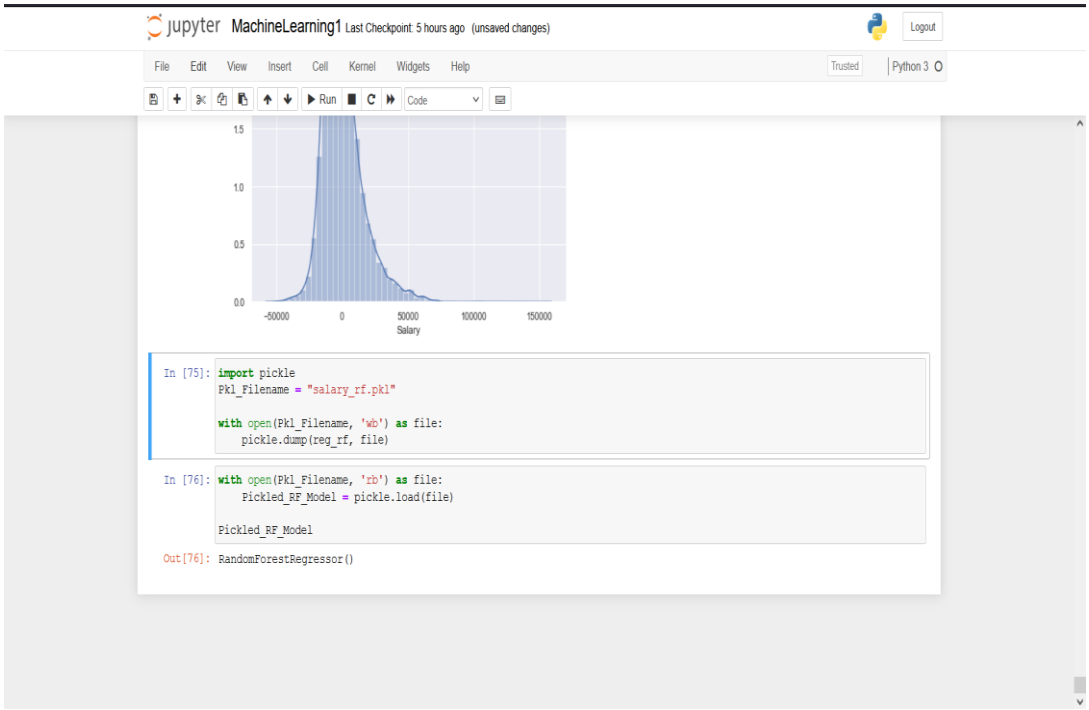


Figure 8: Serializing the machine learning model into a Pickle file

Figure 8 - The trained random forest regression model is saved as a pickle file for later use

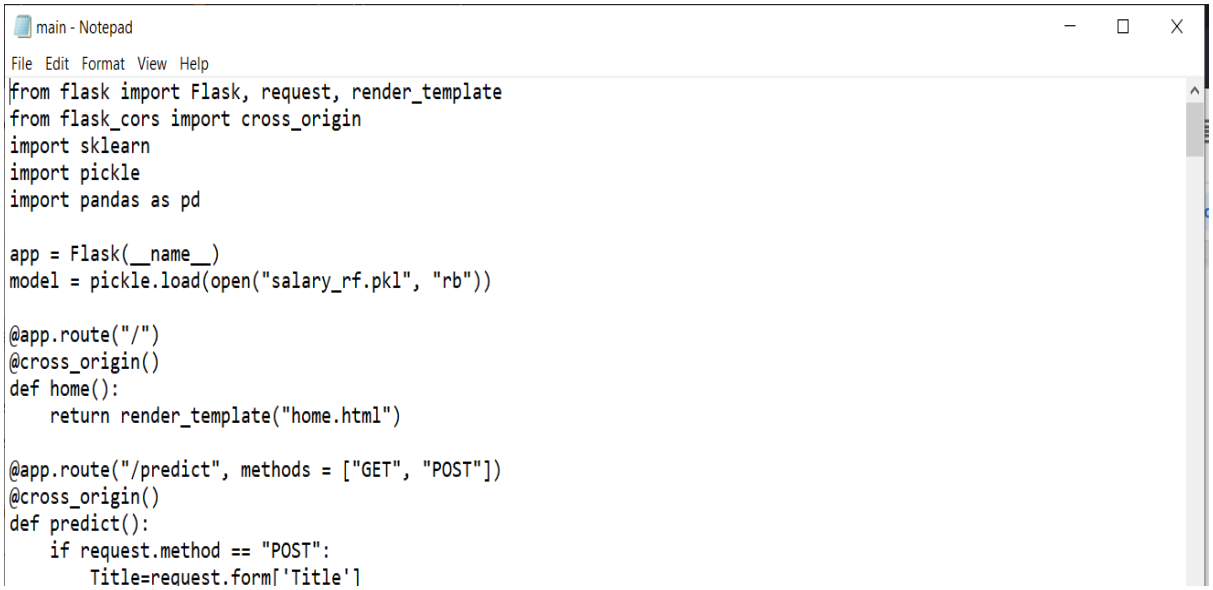


Figure 9: Loading the pickle file into main.py

Figure 9 displays the code snippet of **main.py** which is used to load the machine learning model (pickle file) and also implements the flask framework for developing the web application.

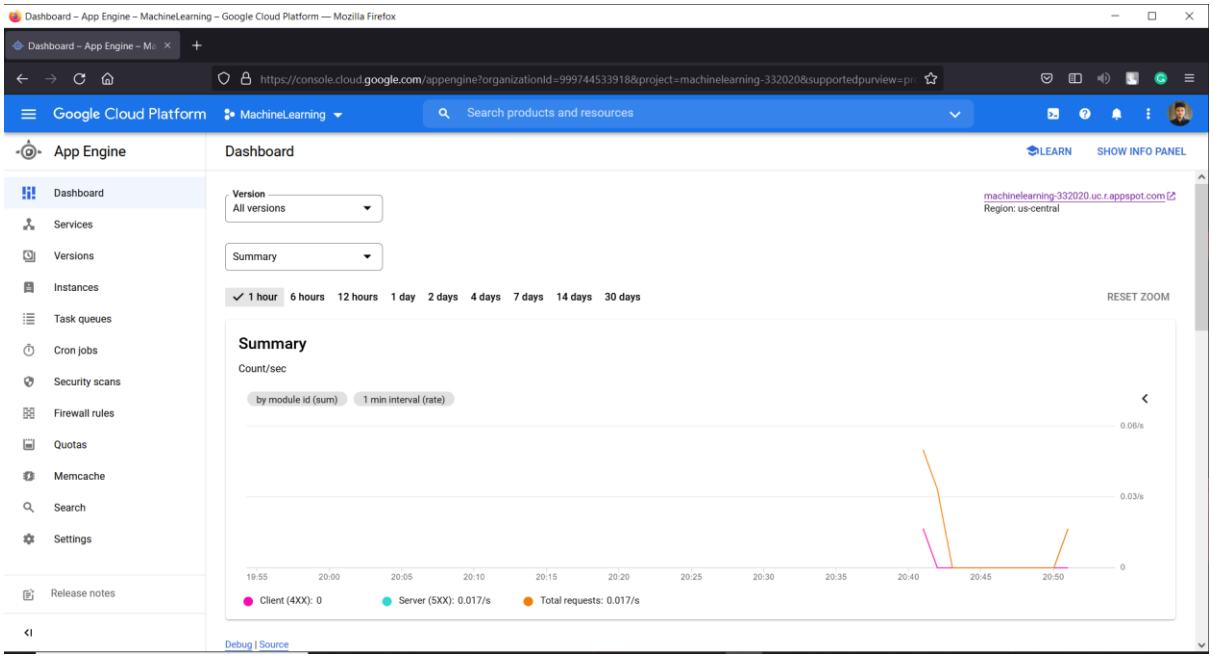


Figure 10: AppEngine Dashboard

Figure 10 displays the dashboard of the AppEngine platform present in GCP.

Figure 11(a): Commands to host the web application

Figure 11(a) displays the commands run on Google SDK to host the web application using the AppEngine service of GCP.

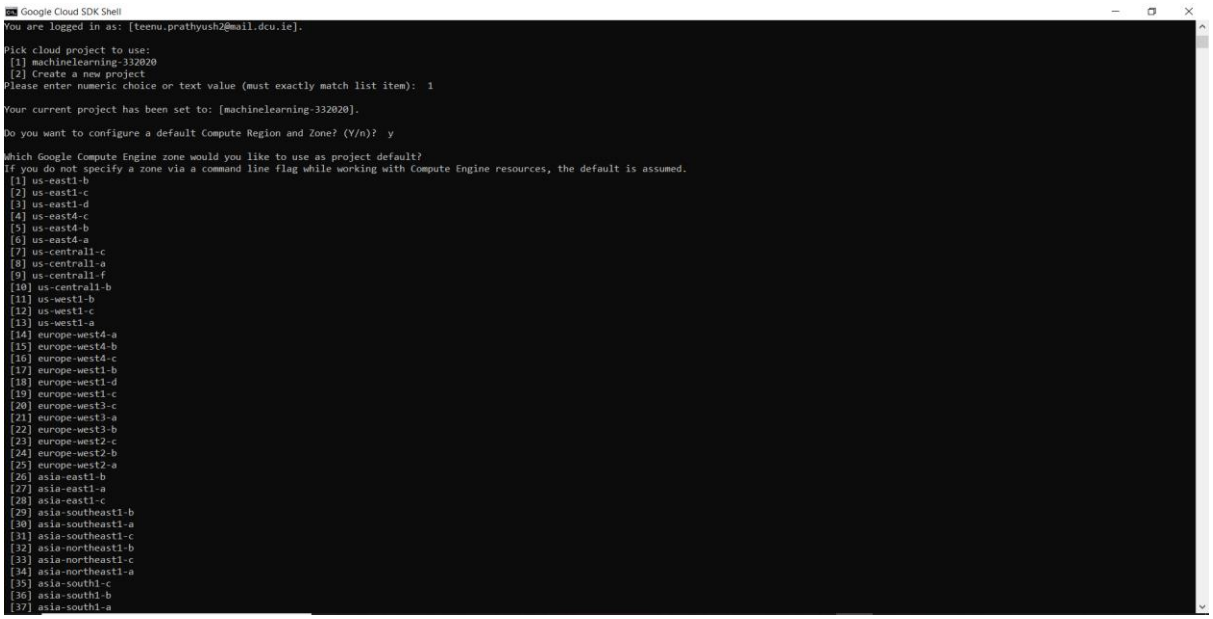


Figure 11(b): Commands to the host the web application

Figure 11(b) is a continuation of the previous step. Here, we configure the zone in which the project will be used.

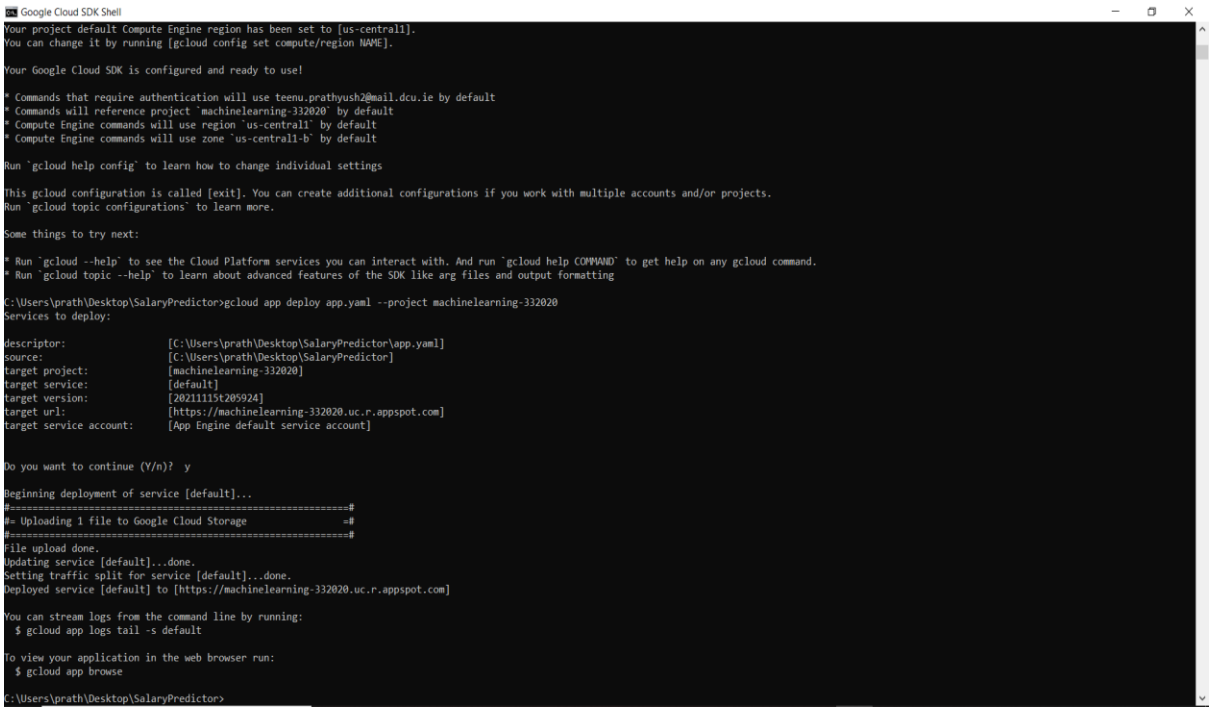


Figure 11(c): Commands to the host the web application

Figure 11(c) displays the progress of deployment of the web application using the AppEngine service of GCP.

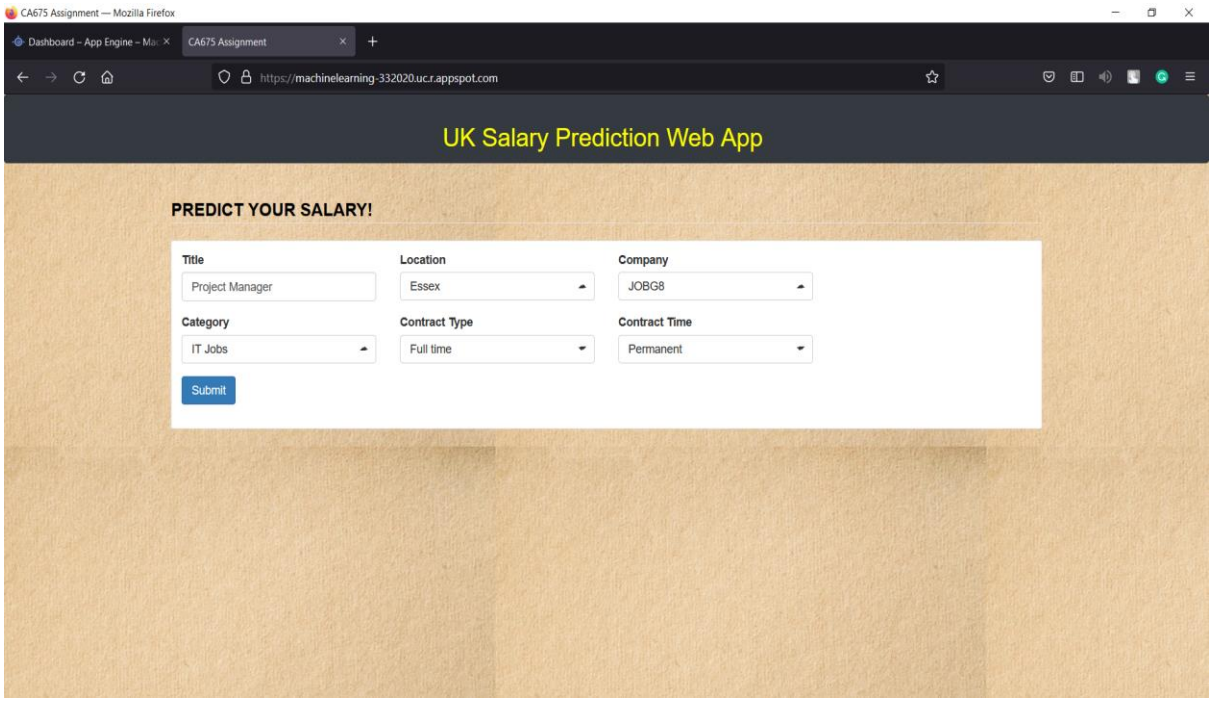


Figure 12: Deployment of the web application

Figure 12 displays the deployment of the web application using AppEngine Service of the Google Cloud Platform, where a user can enter values in the form to get the estimated salary.

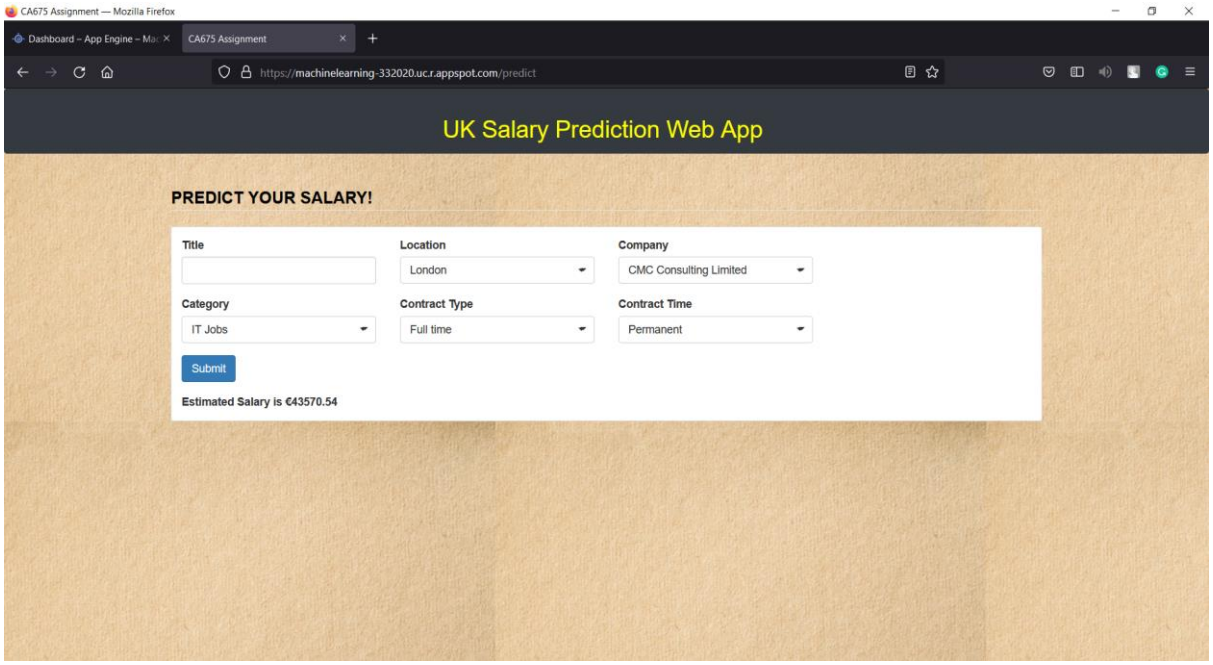


Figure 13: Predicting the estimated salary

Figure 13 displays the estimated salary for the values entered in the form (refer figure 12).

8. CHALLENGES AND LESSONS LEARNED:

The challenges and lessons learned in building the web application are described below:

- **Acquiring the dataset**: Not many datasets are available containing big data for this particular project. We have taken the UK job salary prediction dataset from Kaggle as we found it to be the most suitable dataset for implementing our project.
- **Training the machine learning model with a suitable algorithm**: Having trained the model with different machine learning algorithms like Support Vector Machines, Decision trees, etc., we found Random Forest Regressor to be the best performing model.
- **Designing the web application**: The design of the web application is simple, elegant, and responsive. However, with more time we could have improved the design. For example, the **Title** field in the web application form takes a user input instead of a list of values. Another design feature we could implement in the future is that when a user enters a value in one of the fields, the values in related fields should be auto-populated.

In the future, we could enhance this model to include more features and an improved front-end design. We could also develop a mobile application for predicting salaries.

9. LINK TO THE WALKTHROUGH VIDEO:

https://www.youtube.com/watch?v=6KYlb2Kp4Jc&t=1s

10. TEAM ROLES AND TASKS:

- **Data Collection and Processing:** Fiaz Ashraf and Divyashree Parameshwara - SATISFACTORY
- **Front-end Setup:** Teena Sharma, Anish Patil - SATISFACTORY
- **Back-end Setup:** Teenu Prathyush, Niranjan Nimbarte - SATISFACTORY
- **WebApp Deployment**: Teenu Prathyush - SATISFACTORY
- **Report:** All the members of the group contributed in writing the report.

LIST OF FIGURES