**Fake Job Prediction**

**Report**

**Project By:**

Nikunj Ashok Mistry

**Introduction:**

There are many websites where job postings are fake. Many people apply to such jobs and give their information to scammers, thus such scammers are able to gain personal confidential information very easily. Thus, using a publicly available dataset, we are building a model which can predict if a job posting is fake or real. We are building a classifier that can classify the jobs posted to be real or fake based on the training done on the model.

The dataset used in this project is from Kaggle. The details of the dataset are as follows:

The dataset has 18,000 job postings and 800 of these are fake. The dataset contains 18 columns: Job ID, Title, Location, Department, Salary\_Range, Company\_Profile, Description, Requirements, Benefits, and Telecommuting. The website link for the dataset is here: <https://www.kaggle.com/shivamb/real-or-fake-fake-jobposting-prediction>

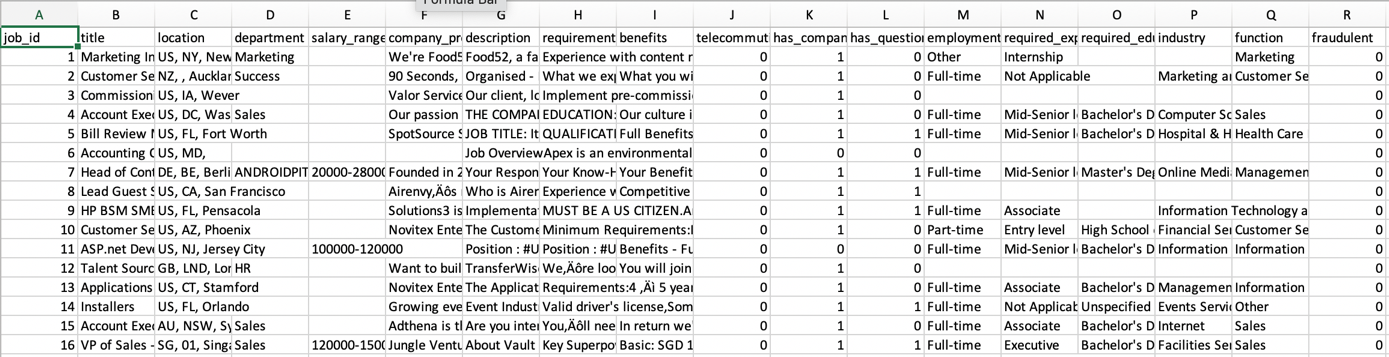


Figure 1- First 16 rows of the dataset

**Data Pre-processing-**

Since a publicly available dataset is used, the first step towards our project is data pre-processing. We are using Jupyter Notebook for implementation of our project.

The main python modules used are pandas, numpy, spacy, sklearn and seaborn.

Pandas is used for data handling. The dataset is in in a form of a data frame, and then operations can be performed on the data frame.

The dataset is uploaded and stored as a Data Frame for processing.

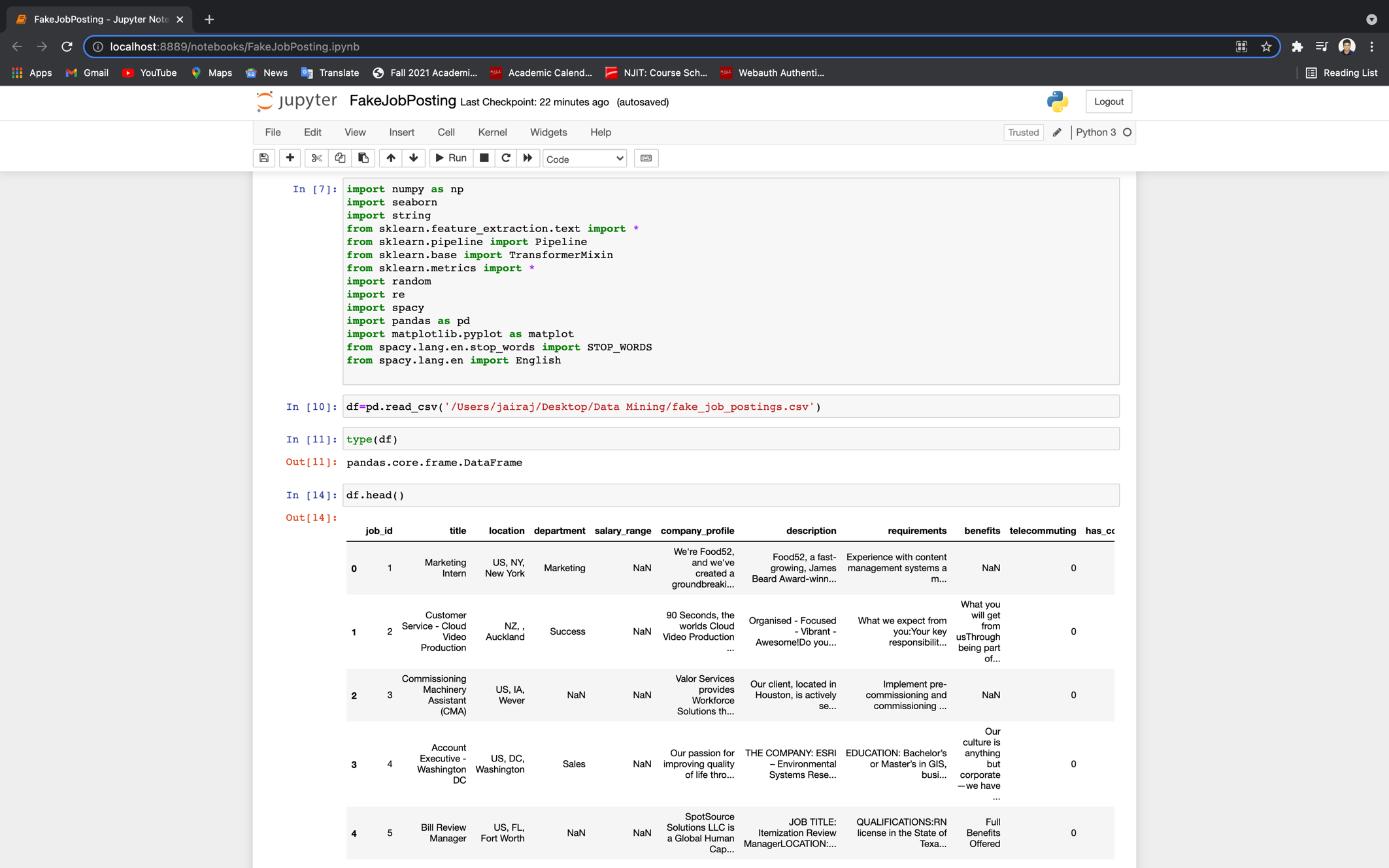


Figure 3-The dataset is uploaded on the Jupyter Notebook.

The dataset pre-processing step is required so that the data used to train a model is a standardized data.

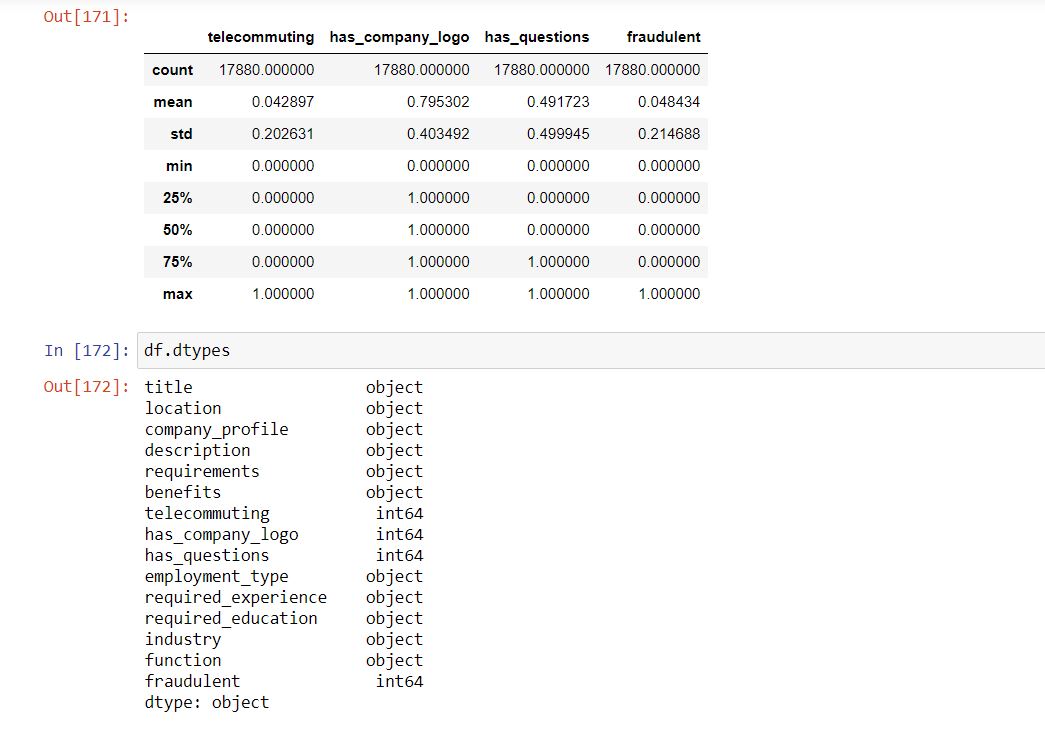


Figure 4-Overall description and data types of the attributes.

Isnull(),sum(): this command shows the number of NULL values in the dataset. Axis is set to 0, so the count will be for each column.

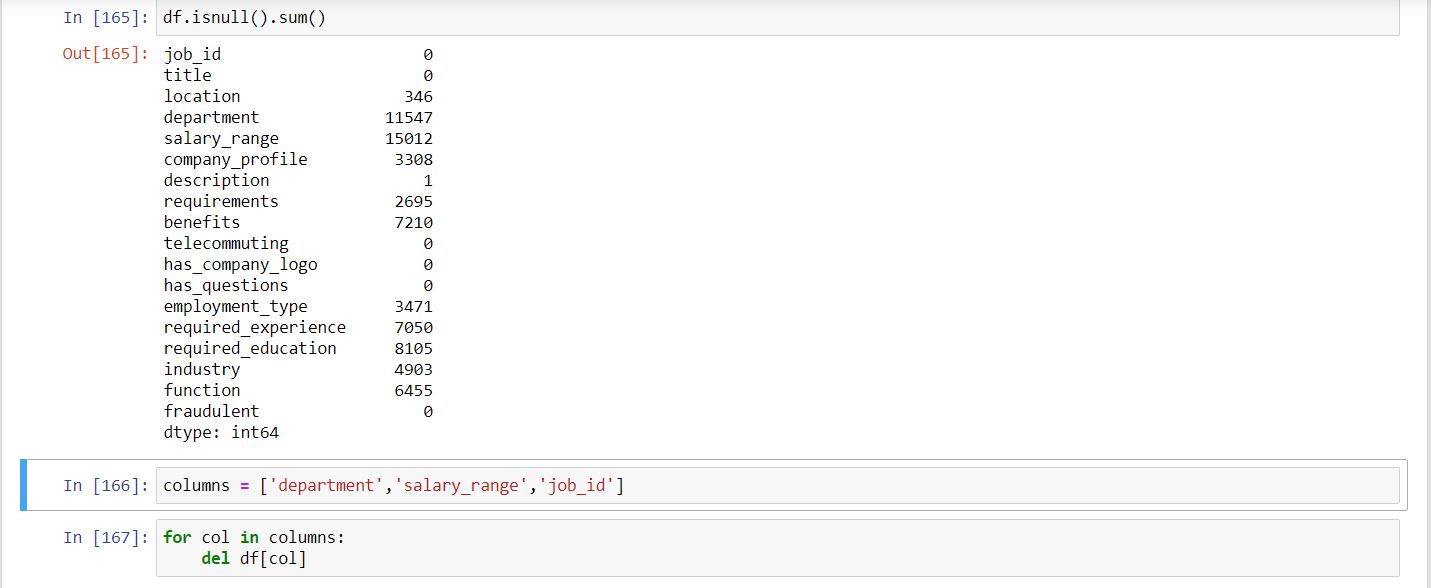


Figure 5- Null attribute count

M9\*columns along with job\_id which does not play a part in analysis.

The cells from other columns where there null values are present are filled with a space by using fillna(‘’, inplace=True).

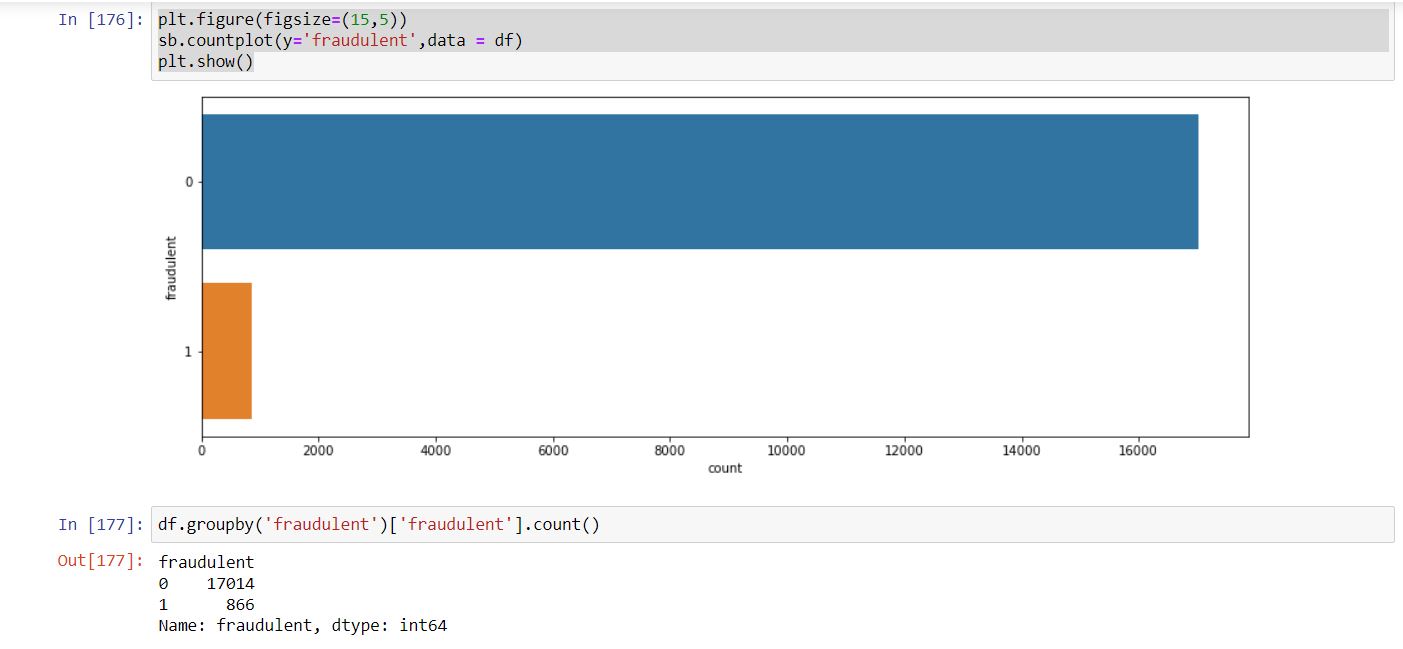
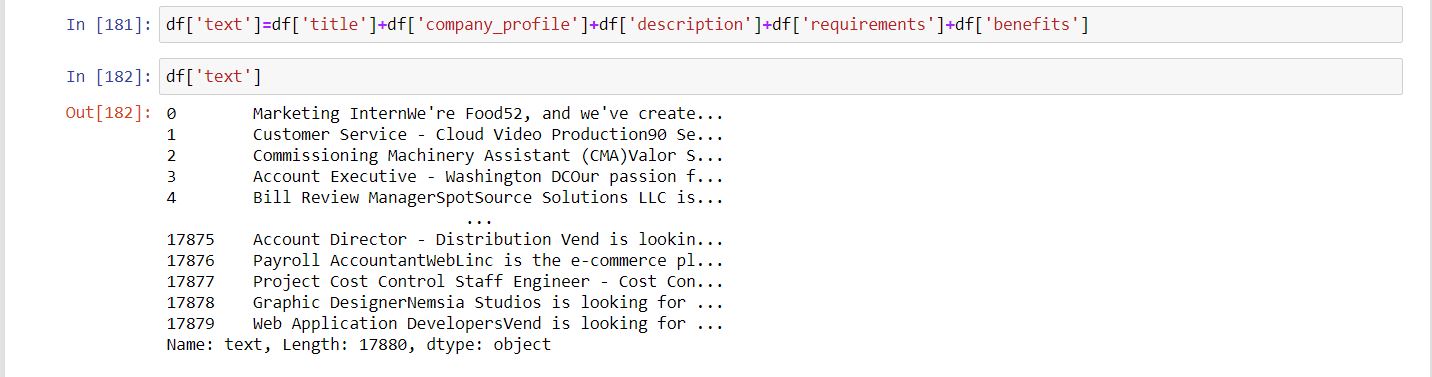


Figure 6- Bar Graph describing the nature of job\_posting.

Of all the data present we can see that majority of the data present is for real jobs. Data present for real job is 17014 while only 866 data items are present for fraudulent jobs.

For pre-processing of the text data we will be using Natural Language Processing. For that we need to create a new attribute which contains prominent textual data features like title, company\_profile, description, requirements and benefits. Then creating a temporary dataframe for NLP processing we only keep 2 attribute- The textual data and whether the posting is fraudulent or not.



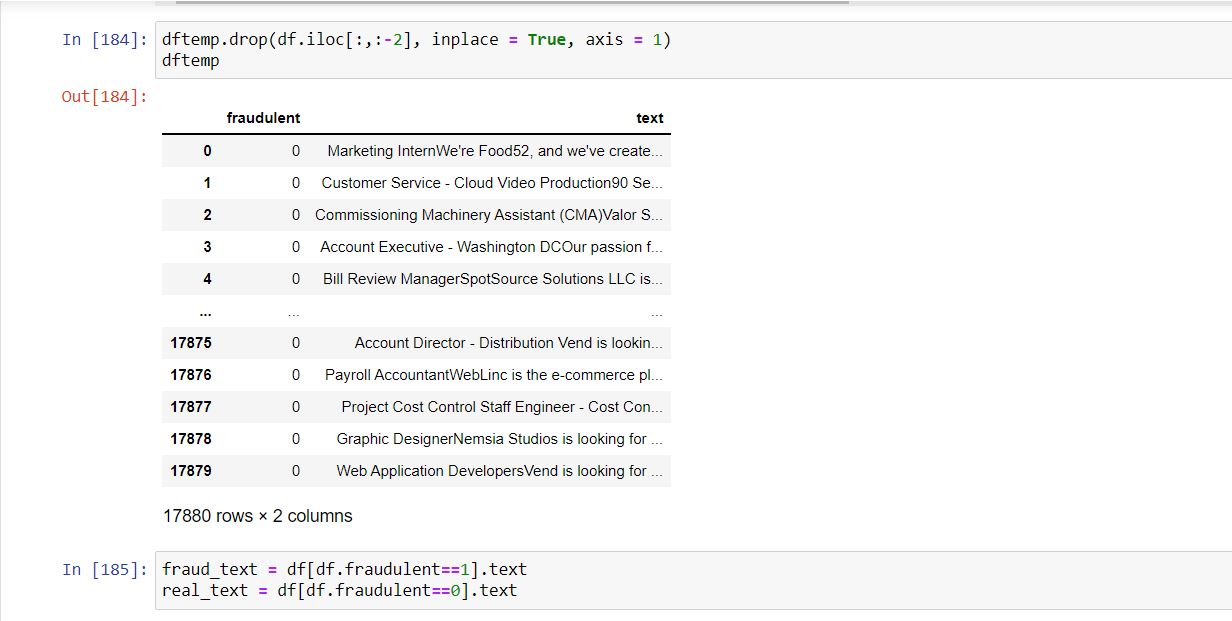


Figure 7- Creating temporary DataFrame for text pre-processing using NLP

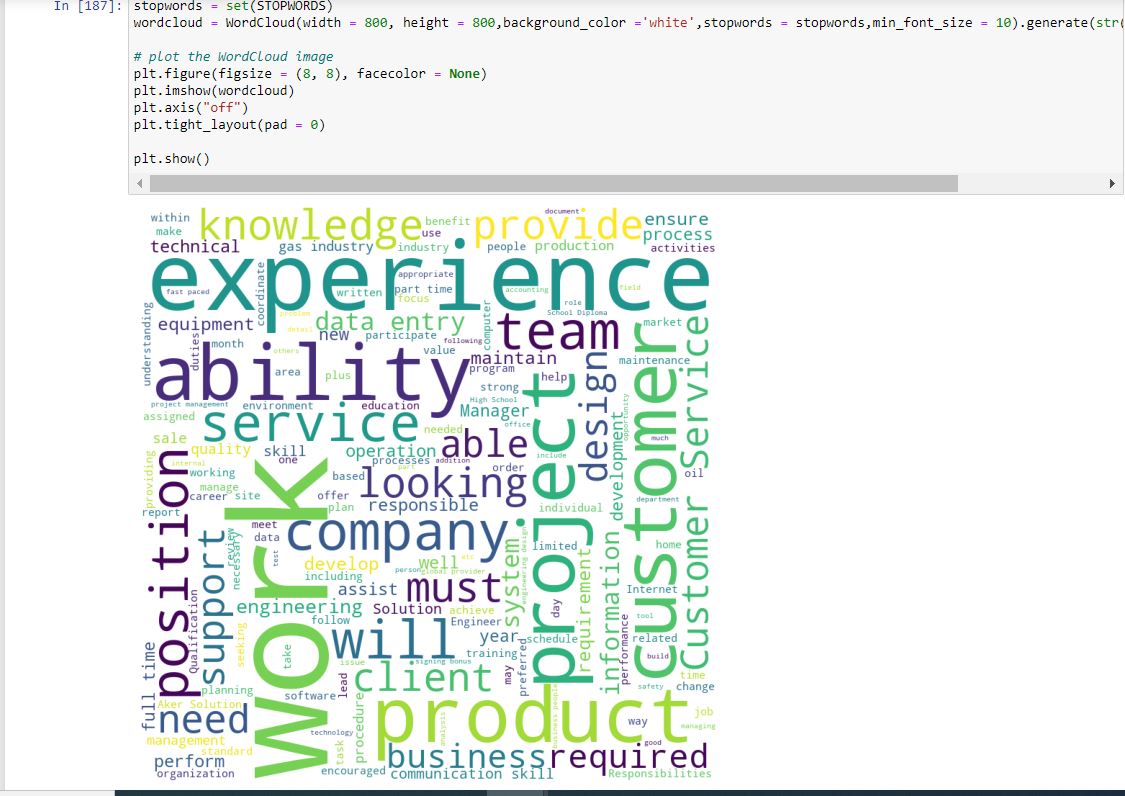


Figure 8- Word Cloud for fraudulent job\_postings.

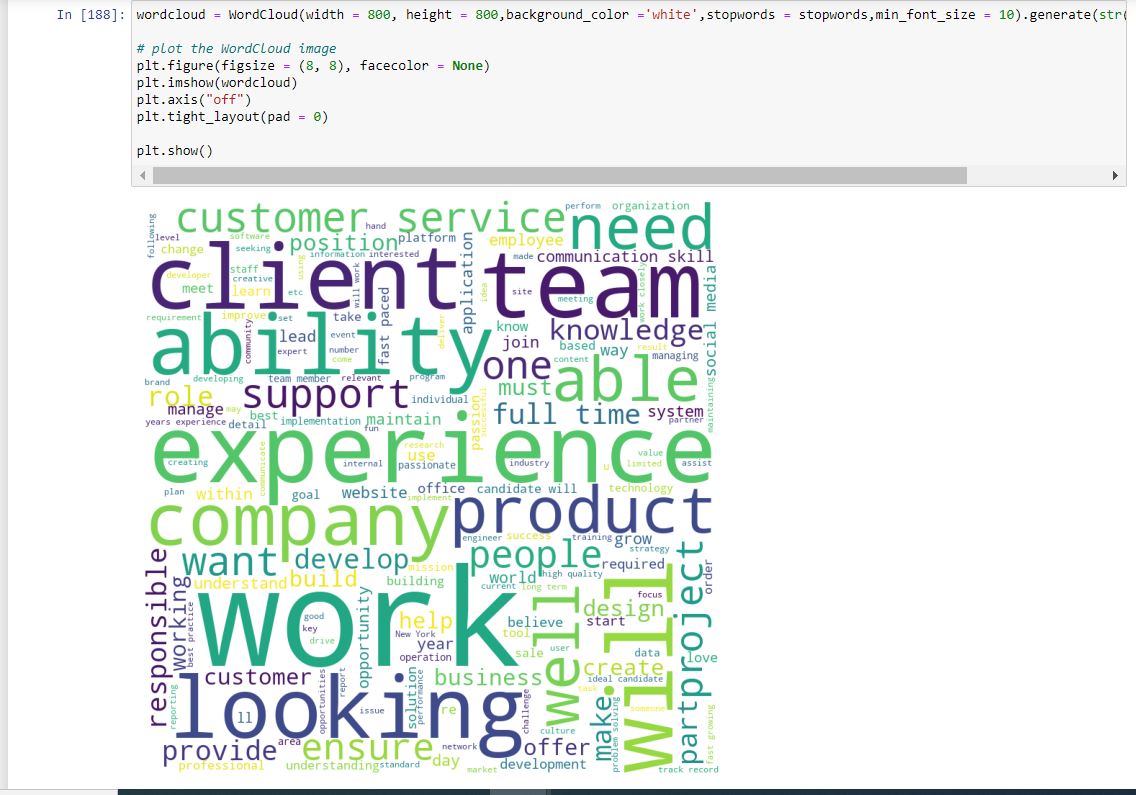


Figure 9- Word Cloud for real Job postings.

**Pre-Processing of Textual Data.**

To signify the prominent words used in both real job posting and fraudulent job postings we use Word Cloud that gives text data in which the text-size of each word indicates its frequency or importance.

The Word Cloud for fake posts and real posts is displayed in Figures 8 and 9 respectively.

The pre-processing stage includes removing Stopwords, text cleaning and lemmatization.



Figure 10- NLP preprocessing of the text data

The text first gets converted into lowercase. All the special characters are then removed from the sentence. The sentence then gets tokenized into words and Stop Words are also removed from the sentence. The remaining words are then lemmatized and these words form a new sentence with clean text.

The final Pre-processed text can be seen in Figure 11, where in the first part we see the DataFrame as a whole and in the second part we can see the list generated for 1st sentence after the pre-processing is done.



Figure 11- Final pre-processed text.

After the pre-processing stage we will be performing Feature Extraction process along with the implementation of a classification algorithm that will classify the testing data as real or fraudulent.

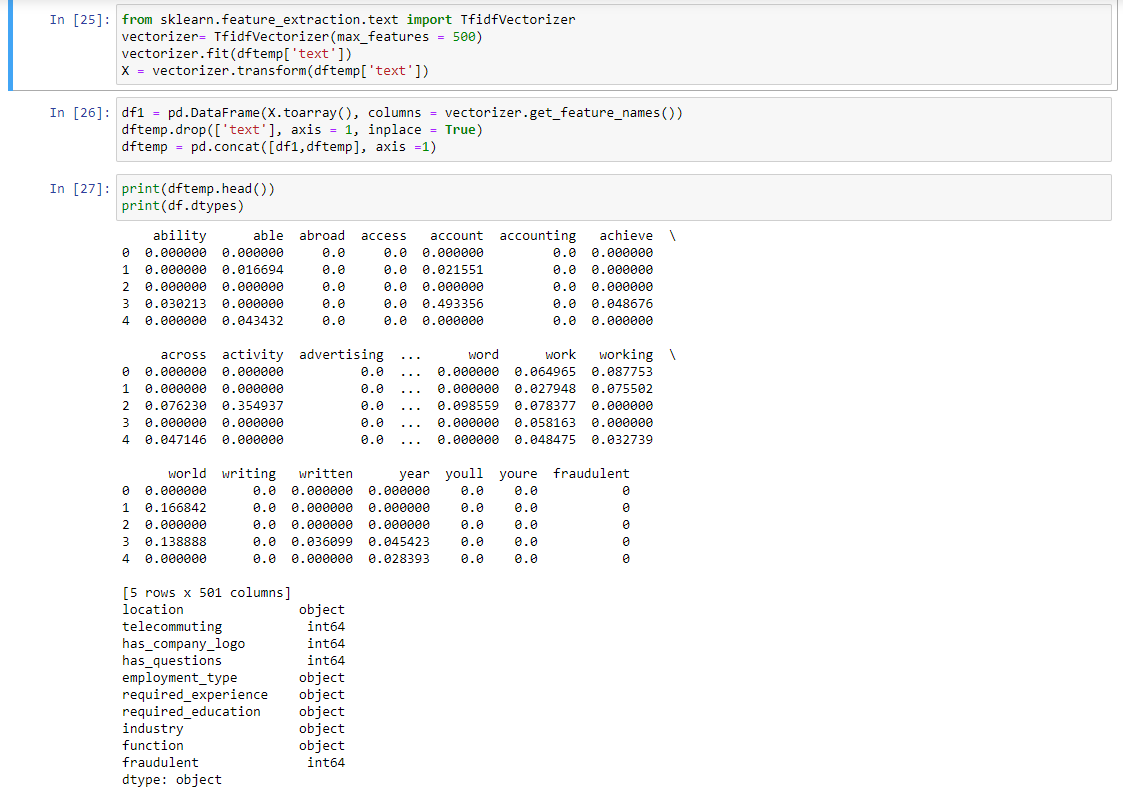


Figure 12- Feature Extraction of the data.

After we have successfully Pre-Professed the data we now move to feature extraction. Here we are using TF-IDF Vectorizer to give equal weightage to each word in the data. Using TF-IDF Vectorizer will convert each word in the data into a specific numeric value it will also show us the importance of a word in the data.

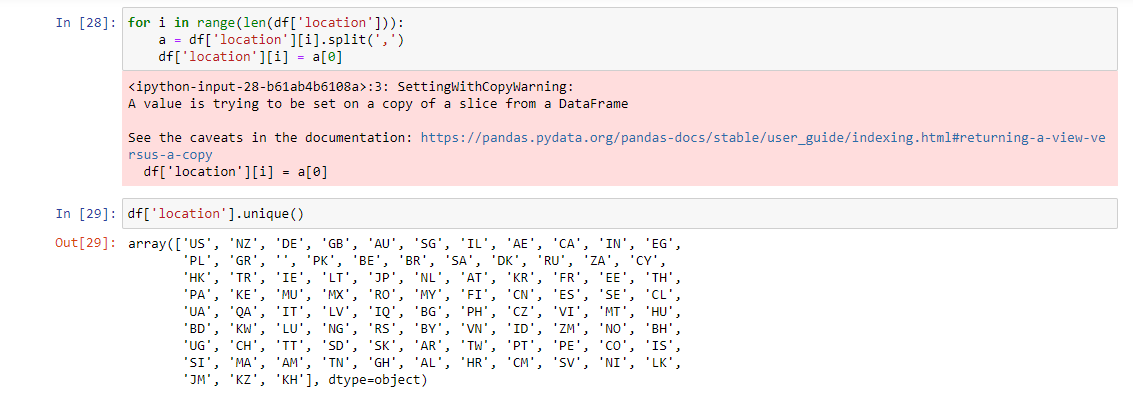
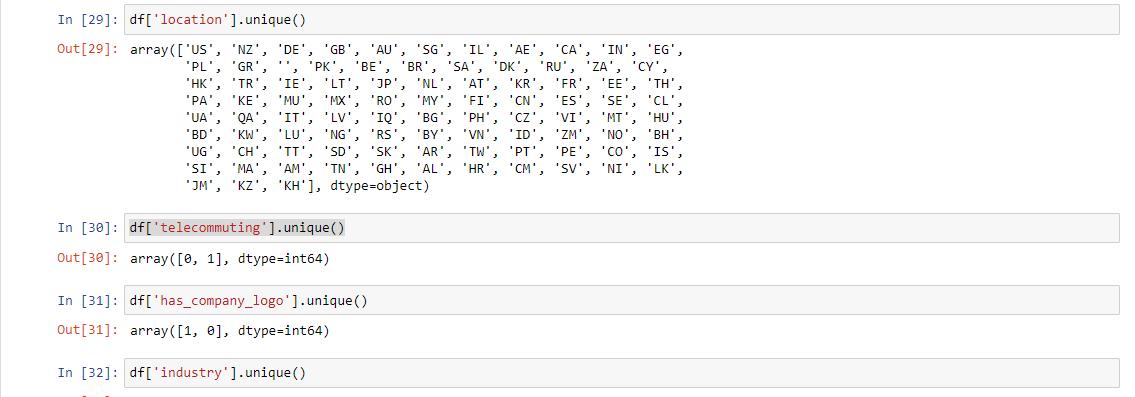
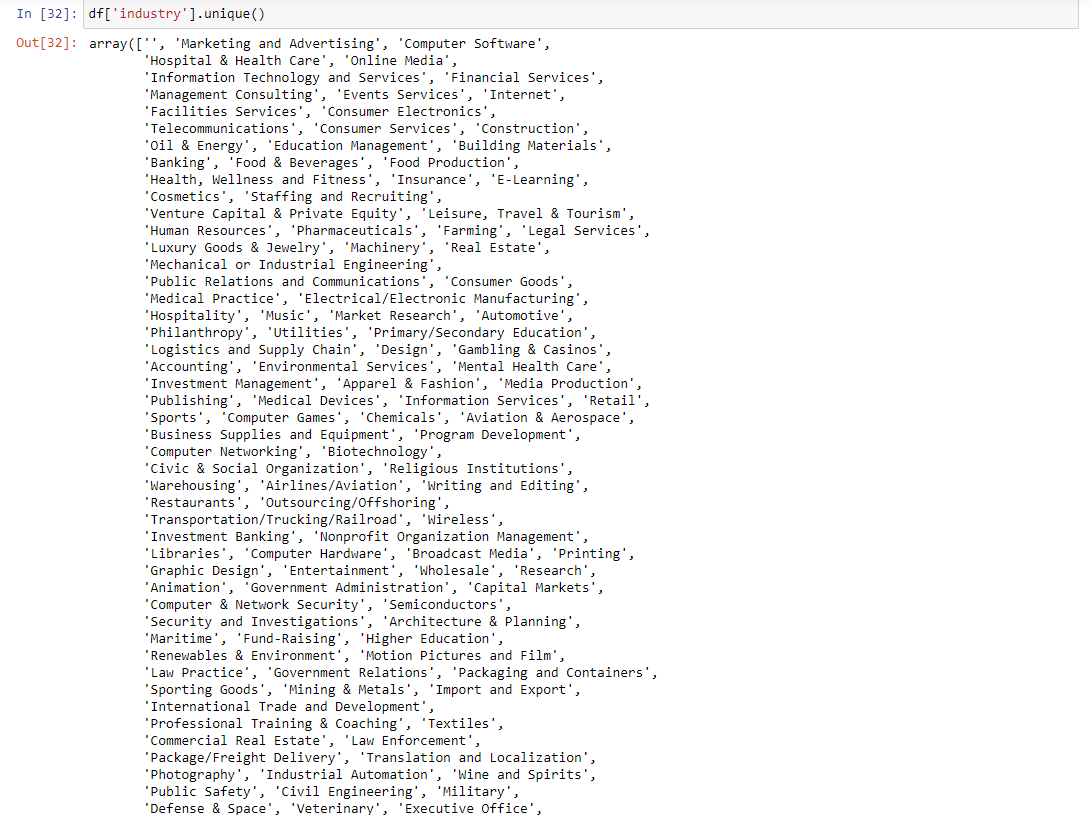
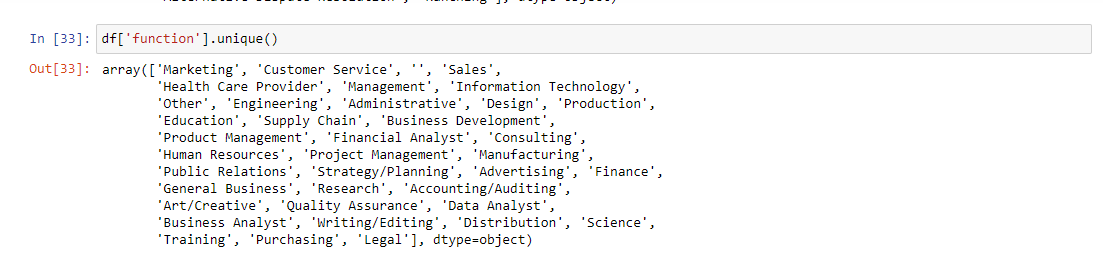


Figure 13- Splitting the location present in the data.

Here we are splitting the location of the job posting present in the data set. We are removing the cities present and just keeping the countries for which the job has been posted for.

Checking all the unique values of the attributes. For example if the company has logo or not, what are the fields of the job posting and what are the functions required in the job postings.



Figure 14-Performing Mean Encoding on each attribute in the dataset.

Here we are preforming mean encoding of each attribute in the data set.

Mean encoding converts the categorical variables in the data into numerical data which helps in calculating the data by taking the mean of all the combined data with respect to the output class attribute. Here we have mean encoded each attribute by taking the specific attribute and combing it with each related attribute and then taking the mean of the whole data.

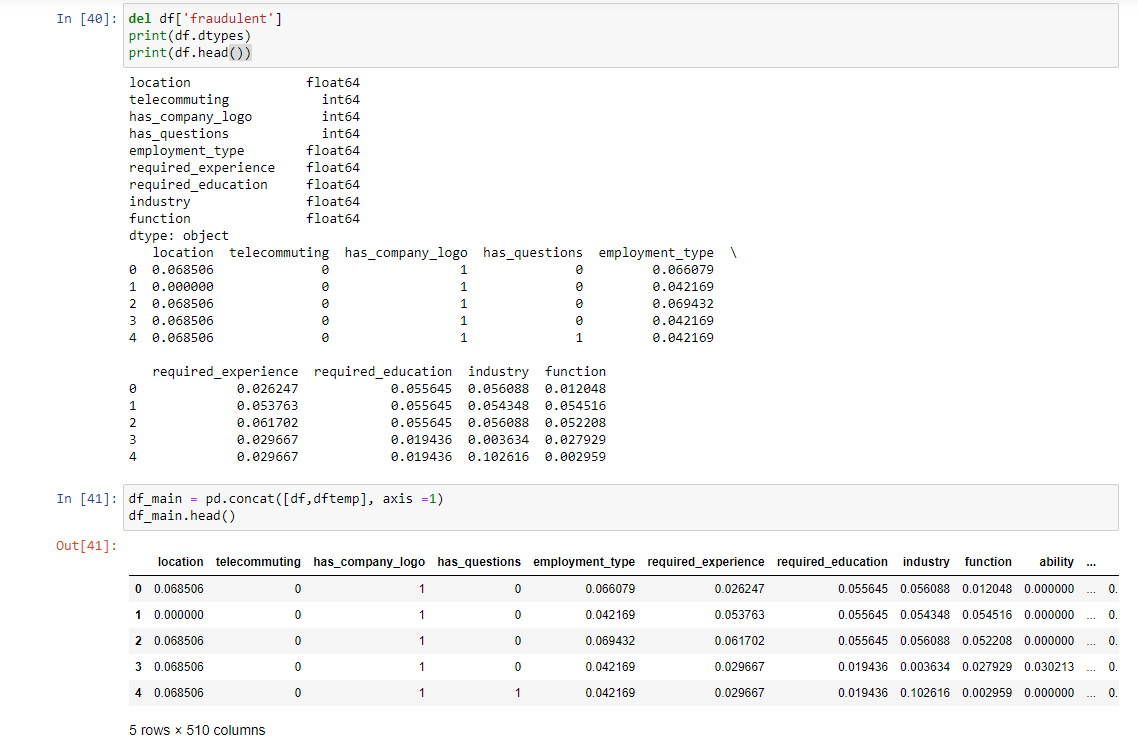


Figure 15- Combining the datasets.

Getting a final dataframe (df\_main) consisting of all the feature attributes by getting dftemp and df dataframe which consist of text features and other features respectively.

**Implementation:**





Setting the X and Y as input and output attributes then splitting them into training and testing datasets.

**Classification Algorithms:**

In classification, a model is created to predict class labels. This involves two steps- learning step and a classification step. The model is built using a predetermined set of data called as the training data. Every data tuple has a class label, thus, classification algorithms are supervised learning algorithms.

**Random Forest Algorithm:**

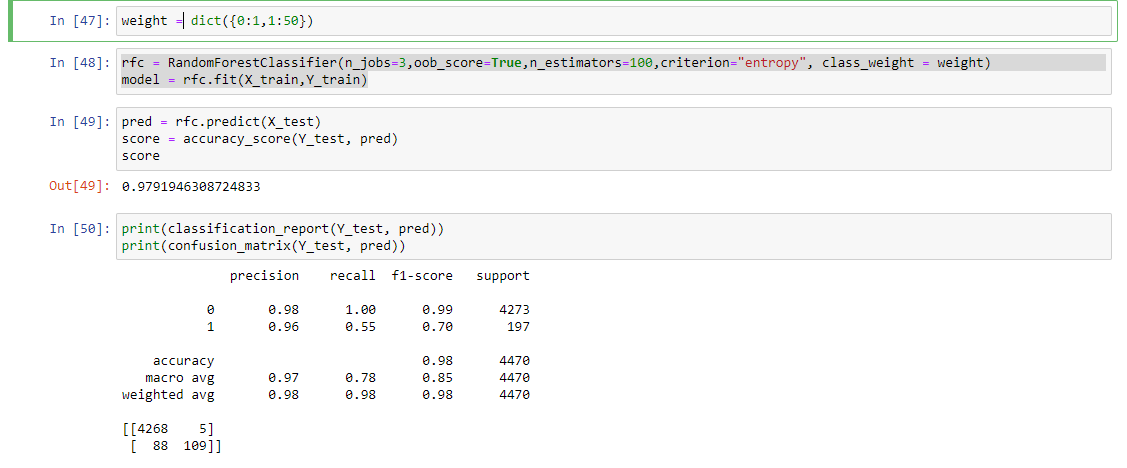
A random forest consists of collection of decision trees. It is a ensemble learning algorithm where better performance is obtained using many decision trees. The decision trees are constructed using random attributes at each node to determine the split. 

Figure 16- Getting the accuracy by using Random Forest Classifier.

By using the Random Forest Classifier we got the value of the precision as 0.96 and the value of recall as 0.55. As Precision is inversely proportional to recall, the value of precision is higher so the value of recall is a little low. We have assigned the value of real data as 1 and for the fraudulent data as 50 because there is less fraudulent data in the dataset.

**Decision Tree:**

Decision Trees is one of the other models used for classification.

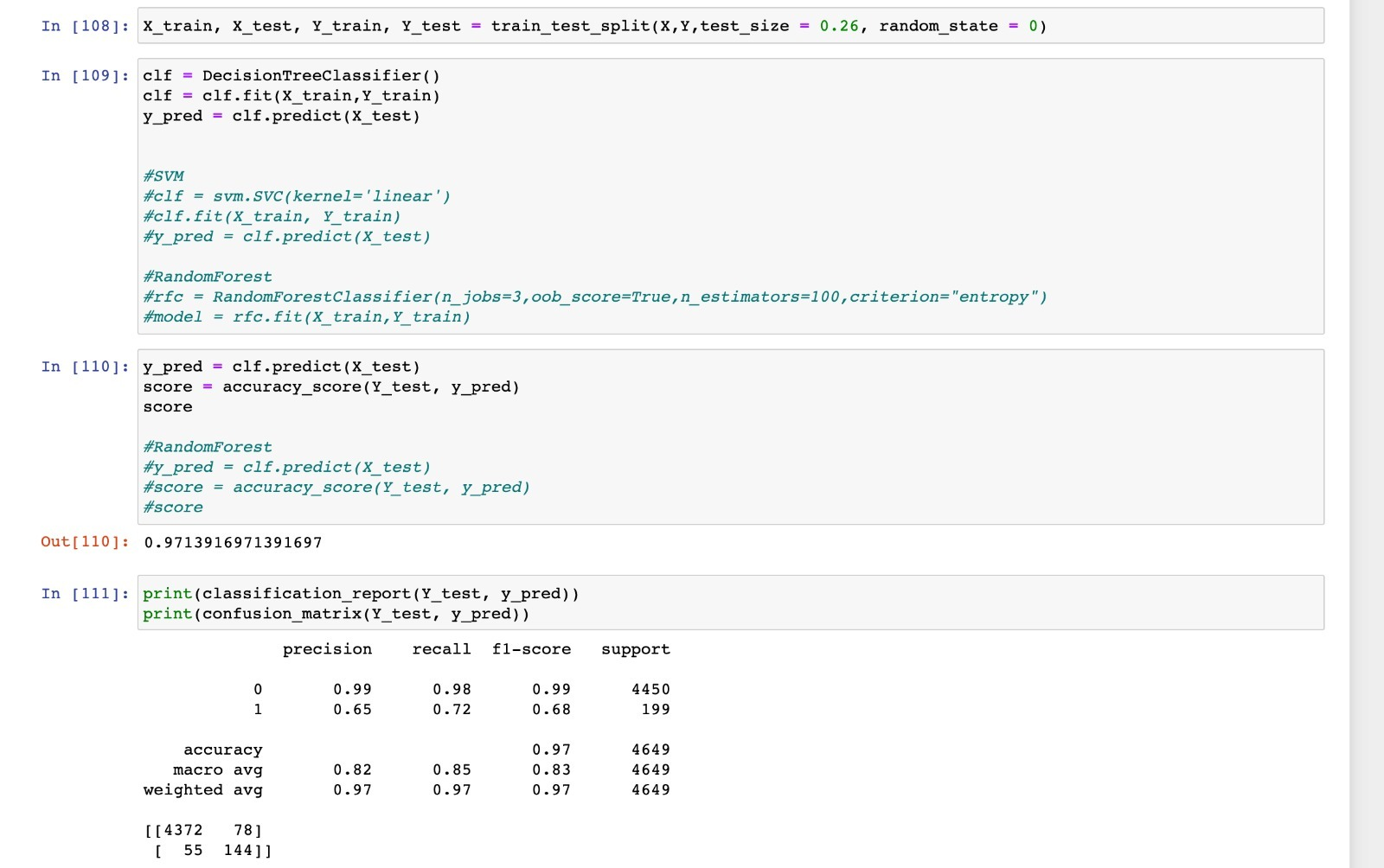


Figure 17- Getting the accuracy by using Decision Tree.

By using Decision Tree as the classifier we the value of the precision as 0.65 and the value of recall as 0.72.

**Conclusion:**

There’s not much to choose between both these models. Both perform very similar and have similar accuracy and other performance metric. Random Forest gives higher precision but lower recall and decision tree gives low precision with better recall. We prefer to use Random Forest as it gives slightly better ‘F1’- score for fraudulent label.