Predict Fraudulent Job posts

TEAM 2

Pooja Malage | Saranya Pandiaraj | Mercy Bose

Prof. Shih Yu Chang | DATA 245 Machine Learning | May 11 2021 | SJSU

https://www.youtube.com/watch?v=_uKrpXFlhSA

Contents

Introduction

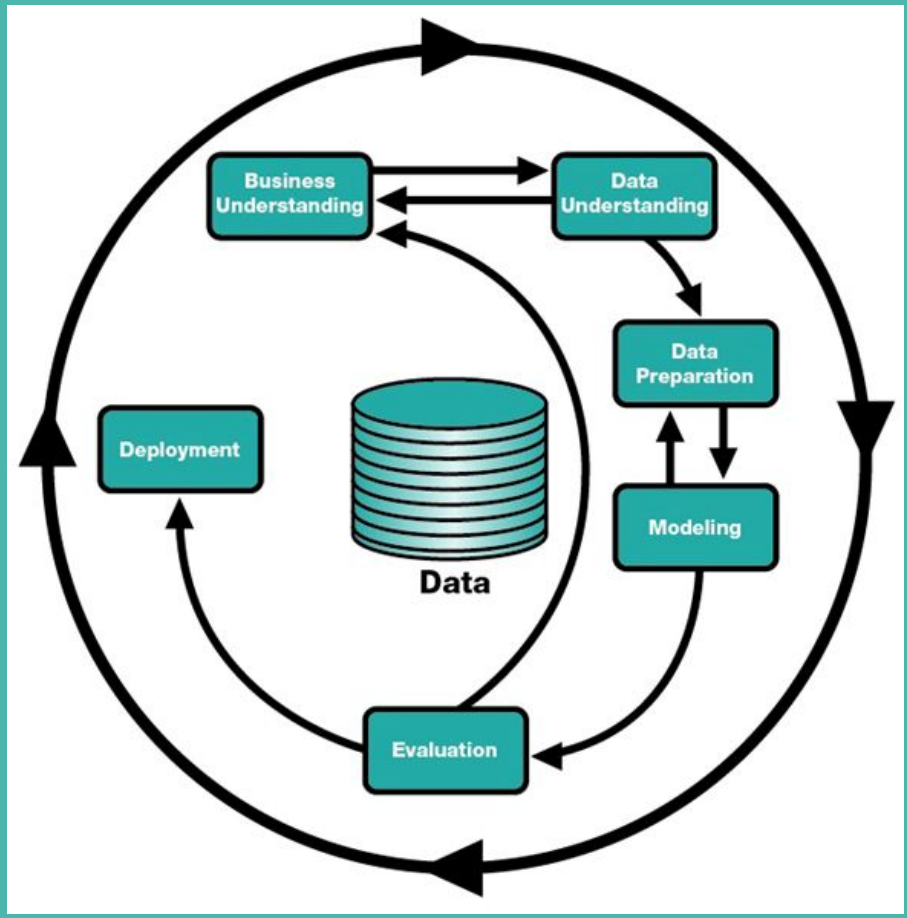
- Social Media and Advertisements in electronic media created newer opportunity to share job details.
- Rapid growth of this has increased the percentage of fraud job postings.

Introduction

- Fake recruiters post job posting on these media with a motive to get money
- Creates an unsecured environment and huge waste of time in professional life.
- It’s important to detect real and fraudulent job postings
- For this purpose,machine learning approach is applied for recognizing fake jobs.

METHODOLOGY

CRISP-DM



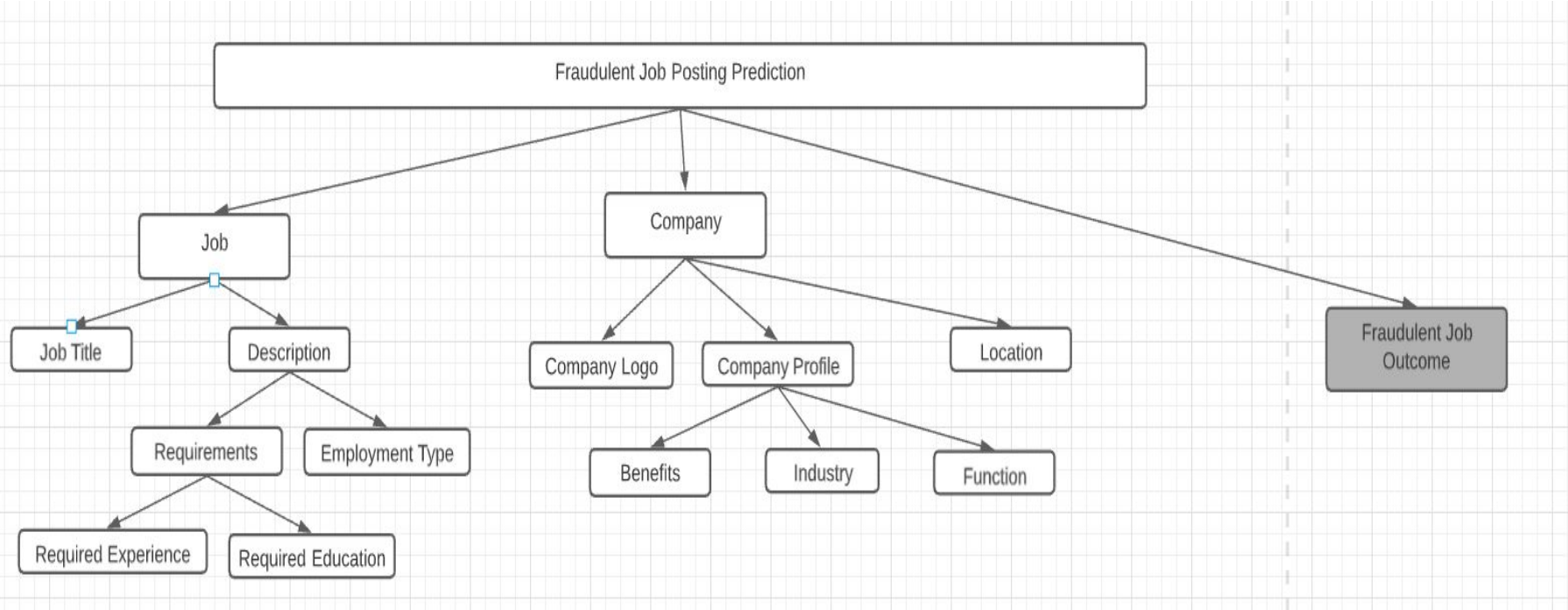
BUSINESS UNDERSTANDING

- Project Goals
- Domain Concept diagram

Project Goal

- To make an prediction about fraudulent job advertisement and detect the job advertisement is real or fake
- Find the best attribute for descriptive and predictive model
- To select appropriate predictive model for fraudulent job prediction

Domain Concept Diagram for Fraudulent Job Prediction



DATA UNDERSTANDING

- About the dataset
- Raw Data exploration
- Data Quality Report

About the Dataset

- Retrieved from the data collected by *The University of the Aegean | Laboratory of Information & Communication Systems Security*

- The Fraudulent Job Posting Prediction dataset contains job description. About 800 of the 18K job descriptions are fake.

The dataset has the following variables :

title: The title of the job ad entry

location: Geographical location of the job ad.

department: Corporate department (e.g. sales).

salary_range: Indicative salary range (eg. 50000 to 60000)

company_profile:A brief company description.

description: The details description of the job ad.

requirements:Enlisted requirements for the job opening

benefits: Enlisted offered benefits by the employer.

telecommuting:True for telecommuting positions.

Has_company_logo: True if company logo is present.

Has_questions: True if screening questions are present.

Employment_type: Full-type, Part-time, Contract, etc.

Required_experience: Executive, Entry level, Intern, etc.

Required_education: Doctorate, Master's Degree, Bachelor, etc.

Industry: Automotive, IT, Health care, Real estate, etc.

Function: Consulting, Engineering, Research, Sales etc.

Fraudulent: target - Classification attribute.

Sample of Dataset

job_id	title	location	department	salary_range	company_profile	description	requirements	benefits	telecommuting	has_company_logo	has_questions	employment_type	required_experience	required_education	industry	function	fraudulent
1	Marketing Intern	US, NY, New York	Marketing	NaN	We're Food52, and we've created a groundbreaki...	Food52, a fast-growing, James Beard Award-winn...	Experience with content management systems a m...	NaN	0	1	0	Other	Internship	NaN	NaN	Marketing	0
2	Customer Service - Cloud Video Production	NZ, , Auckland	Success	NaN	90 Seconds, the worlds Cloud Video Production ...	Organised - Focused - Vibrant - Awesome!Do you...	What we expect from you:Your key responsibilit...	What you will get from usThrough being part of...	0	1	0	Full-time	Not Applicable	NaN	Marketing and Advertising	Customer Service	0
3	Commissioning Machinery Assistant (CMA)	US, IA, Wever	NaN	NaN	Valor Services provides Workforce Solutions th...	Our client, located in Houston, is actively se...	Implement pre-commissioning and commissioning ...	NaN	0	1	0	NaN	NaN	NaN	NaN	NaN	0
4	Account Executive - Washington DC	US, DC, Washington	Sales	NaN	Our passion for improving quality of life thro...	THE COMPANY: ESRI – Environmental Systems Rese...	EDUCATION: Bachelor's or Master's in GIS, busi...	Our culture is anything but corporate—we have ...	0	1	0	Full-time	Mid-Senior level	Bachelor's Degree	Computer Software	Sales	0
5	Bill Review Manager	US, FL, Fort Worth	NaN	NaN	SpotSource Solutions LLC is a Global Human Cap...	JOB TITLE: Itemization Review ManagerLOCATION:...	QUALIFICATIONS:RN license in the State of Texa...	Full Benefits Offered	0	1	1	Full-time	Mid-Senior level	Bachelor's Degree	Hospital & Health Care	Health Care Provider	0

Data Quality Report - Continuous Variables

- It can be noted that for all the features, data cleaning is required
- ***Job Id*** - High cardinality = number of instances -> so its an identity column
- ***Telecommuting, has_company_logo, has_questions, fraudulent columns*** - Low cardinality, min value = 0 and max values =1 -> so it needs to be converted to boolean column

Feature	Count	Miss %	Card.	Min.	1st Qrt	Mean	Median	3rd Qrt	Max	Std. Dev	Problem	Solution
Job_id	17880	0	17880	1	4470.75	8940.5	8940.5	13410.25	17880	5161.65	Identity column	Convert it to index column
telecommuting	17880	0	2	0	0	0.042	0	0	1	0.202	Actually a boolean column but it is numeric here	Convert it to boolean column
has_company_logo	17880	0	2	0	1	0.795	1	1	1	0.403	Actually a boolean column but is numeric here	Convert it to boolean column
has_questions	17880	0	2	0	0	0.4917	0	1	1	0.499	Actually a boolean column but is numeric here	Convert it to boolean column
fraudulent	17880	0	2	0	0	0.0484	0	0	1	0.2146	Actually a boolean column but is numeric here	Convert it to boolean column

Data Quality Report- Categorical Features - 1

All these columns have to be converted to string, remove NaN values, handle missing values appropriately for each column

Location - Split into 3 new columns -> Country, State and City
Department & Salary_range - have high missing% > 60% -> 85% and 64% -> Remove features
Company_profile & Description - text preprocessing must be done to remove unwanted stop words, characters, html code

Feature	Count	Miss %	Card.	Mode	Mode Freq.	Mod %	2nd Mode	2nd Mode Freq	2nd Mode %	Problem	Solution
title	17880	0.00%	11231	English Teacher Abroad	311	1.74%	Customer Service Associate	146	0.82%	Not a string column	Convert to string
location	17534	1.94%	3105	GB, LND, London	718	4.09%	US, NY, New York	658	3.75%	1) Format is concatenated string -> Country, State, City 2) Handle missing values	1) Split the column and create 3 new columns - Country, State, City 2) Missing values - replace with 'Unspecified'
department	6333	64.58%	1337	Sales	551	8.70%	Engineering	487	7.69%	High percentage of missing values > 60%	Remove feature column
salary_range	2868	83.96%	874	0-0	142	4.95%	40000-50000	66	2.30%	High percentage of missing values > 60%	Remove feature column
company_profile	14572	18.50%	1709	We help teachers get safe & secure jobs abroad :)	726	4.98%	We Provide Full Time Permanent Posit	674	4.63%	1) &, URLs, special characters values are present 2) Missing values -> 18.5% 3) Not a string column	1) Remove all unwanted characters 2) Replace missing values with 'Unspecified' 3) Convert to string column
description	17879	0.01%	14801	Play with kids, get paid for it Love travel? Jobs in Asia$1,500+ USD monthly ($200 Cost of living)Housing provided (Private/Furnished)Airfare ReimbursedExcellent for student loans/credit cardsGabriel Adkins : #URL_ed9094c60184b8a4975333957f05be37e69d3cdb68decc9dd9a4242733cfd7f7##URL_75db76d58f7994c7db24e8998c2fc953ab9a20ea9ac948b217693963f78d2e6b#12 month contract : Apply today	379	2.12%	Play with kids, get paid for it :-)Love travel? Jobs in Asia$1500 USD + monthly ($200 Cost of living)Housing providedAirfare ReimbursedExcellent for student loans/credit cardsGabriel Adkins : #URL_ed9094c60184b8a4975333957f05be37e69d3cdb68decc9dd9a4242733cfd7f7##URL_75db76d58f7994c7db24e8998c2fc953ab9a20ea9ac948b217693963f78d2e6b#	66	0.37%	1) &, URLs, special characters values are present 2) Missing values -> 0.01% 3) Not a string column	1) Remove all unwanted characters 2) Replace missing values with 'Unspecified' 3) Convert to string column

Data Quality Report- Categorical Features - 2

Requirements, Benefits, Employment Type- Convert to string, Remove NaN values, Perform text preprocessing to remove unwanted stop words, characters, html code

Feature	Count	Miss %	Card.	Mode	Mode Freq.	Mod %	2nd Mode	2nd Mode Freq	2nd Mode %	Problem	Solution
requirements	15185	15.07%	11968	University degree required. TEFL / TESOL / CELTA or teaching experience preferred but not necessaryCanada/US passport holders only	410	2.70%	University degree required. TEFL / TESOL / CELTA or teaching experience preferred but not necessaryPositive attitude required. Canada/US passport holders only	163	1.07%	1) &, URLs, special characters, -- values are present 2) Missing values -> 15% 3) Not a string column	1) Remove all unwanted characters 2) Replace missing values with 'Unspecified' 3) Convert to string column
benefits	10670	40.32%	6205	See job description	726	6.80%	Career prospects.	158	1.48%	1) "See job description" - implies "description" value- can be replaced 2) Missing values -> 40% -> may be it implies description is enough? 3) Not a string column 4) benefits and description columns are interchangeably used for some instances	If "See job description" is replaced to be same as description, then during concatenation of strings, there will be a duplicate So - how to handle this? Replace with empty? Replace with some other values? What are the implications of this? If missing values are there so that description needed to be considered, then the problem remains the same
employment_type	14409	19.41%	5	Full-time	11620	80.64%	Contract	1524	10.58%	Missing values	1) Since cardinality is low, check for values in *requirements and description column* to see if something matches full time, part time, temporary (contract-not making sense) Example - 'full time' is matching in requirement 2) For further missing values, can be replaced with 'other' 3) Internship can be searched based on description values and a new field called Internship can be added or Temporary can be used

Data Quality Report - Categorical Features- 3

Required Requirements, Required Education, Industry, Function-

Convert to string, Remove NaN values, Perform text preprocessing to remove unwanted stop words, characters, html code

Feature	Count	Miss %	Card.	Mode	Mode Freq.	Mod %	2nd Mode	2nd Mode Freq	2nd Mode %	Problem	Solution
required_experience	10830	39.43%	7	Mid-Senior level	3809	35.17%	Entry level	2697	24.90%	Missing values	1) If blank and 'no minimum experience" -> change to 'Not applicable 2) If blank and 'entry level' is there, then change to Entry level 3) If blank and 'experience' is there, then change to 'Mid-senior level' or Entry level
required_education	9775	45.33%	13	Bachelor's Degree	5145	52.63%	High School or equivalent	2080	21.28%	Missing values	1) Look for values like 'high school', 'diploma, certification, degree, bachelors, masters in that order and keep replacing with proper values
industry	12977	27.42%	131	Information Technology and Service	1734	13.36%	Computer Software	1376	10.60%	Missing values	Replace with Unspecified
function	11425	36.10%	37	Information Technology	1749	15.31%	Sales	1468	12.85%	Missing values	Replace with other

DATA PREPARATION

- Data Cleaning & Preprocessing
- Data Visualization
- Oversampling of data using SMOTE

Percentage of missing values

```
#Determining the Null Value Data

round(( Fraud_Job_df.isna().sum()/len(Fraud_Job_df) ) * 100).sort_values(ascending=False)
```

```
salary_range           84.0
department             65.0
required_education     45.0
benefits               40.0
required_experience    39.0
function               36.0
industry               27.0
company_profile        19.0
employment_type        19.0
requirements           15.0
location                2.0
title                   0.0
fraudulent              0.0
description             0.0
telecommuting           0.0
has_company_logo        0.0
has_questions           0.0
job_id                  0.0
dtype: float64
```

Data Cleaning

- Replacing the Null Value Data with ‘Unspecified’
- Replacing #NAME? with ‘Unspecified’ in the whole Dataset
- Converting Numerical values to bool type
- Removing special characters from text using nltk
- Replacing Non-Alphanumeric values

Data Cleaning - Continuous Columns

- Converted to Index Column for the **Job_id** feature since it is an identity column

```
#Converting to index column

Fraud_Job_df = Fraud_Job_df.set_index('job_id')
Fraud_Job_df.head(5)
```

- Converted the numerical columns to boolean type for the telecommuting, has_company_logo, has_questions, fraudulent features

```
#Converting the below Numerical Columns to bool type
Fraud_Job_df['telecommuting'] = Fraud_Job_df['telecommuting'].astype('bool')
Fraud_Job_df['has_company_logo'] = Fraud_Job_df['has_company_logo'].astype('bool')
Fraud_Job_df['has_questions'] = Fraud_Job_df['has_questions'].astype('bool')
Fraud_Job_df['fraudulent'] = Fraud_Job_df['fraudulent'].astype('bool')
```

Data Cleaning - Categorical Column [1]

- Converted the **title** feature to string since it was not a string column

- The **Location** feature is split to Country, State and City columns

- Replaced the Missing Values with 'Unspecified' for the Location, Country, State & City Columns.

Title

```
[272] #Converting to String since it was not a string column

      Fraud_Job_df["title"] = Fraud_Job_df["title"].astype("str")
```

Location

```
[273] # Splitting the Location column and creating 3 new columns - Country, State, City

      Fraud_Job_df['Country'] = Fraud_Job_df['location'].str.split(',').str[0]
      Fraud_Job_df['State'] = Fraud_Job_df['location'].str.split(',').str[1]
      Fraud_Job_df['City'] = Fraud_Job_df['location'].str.split(',').str[2]
      Fraud_Job_df[['location','Country','State','City']]
```

```
# Missing/ Empty values - replacing with 'Unspecified'

col_1 = ['location','Country','State','City']
Fraud_Job_df[col_1] = Fraud_Job_df[col_1].fillna('Unspecified')
```

Data Cleaning - Categorical Column [2]

- Removed the **Department & Salary** Range feature since it has high percentage of missing values > 60%

```
[276] #Checking for any missing values

      round(( Fraud_Job_df[['department', 'salary_range']].isna().sum()/len(Fraud_Job_df) ) * 100).sort_values(ascending=False)

      salary_range    84.0
      department      65.0
      dtype: float64
```

```
[277] #Removing the Feature Column since it has High percentage of missing values > 60%

      Fraud_Job_df.drop(['department', 'salary_range'], axis=1, inplace=True)
```

- Replaced the Missing Values with 'Unspecified' for the **Industry** feature.

```
#Checking for any missing values
col_3 = ['industry','function']
round(( Fraud_Job_df[col_3].isnull().sum()/len(Fraud_Job_df) ) * 100).sort_values(ascending=False)
```

```
function    36.0
industry    27.0
```

- Replaced the Missing Values with Other for the **Function** feature.

```
#Replacing missing values with 'Unspecified' for Industry

Fraud_Job_df['industry'] = Fraud_Job_df['industry'].fillna('Unspecified')

#Replacing missing values with 'Other' for Function

Fraud_Job_df['function'] = Fraud_Job_df['function'].fillna('Other')
```

Data Cleaning - Categorical Column [3]

Title, Company Profile, Description, Benefits and Requirements

- Removed all the unwanted characters for the Title, Company Profile, Description, Benefits and Requirements features.
- Replaced the Missing Values with ‘Unspecified’ and Converting to String.

Textual Preprocessing : Title,Company Profile , Description, Benefits & Requirements

```
# function to remove non-ASCII
def remove_non_ascii(text):
    ret = []
    for i in text:
        if ord(i) < 128:
            ret.append(i)
        else:
            ret.append("")
    return ''.join(ret)


def tokenise_text(txt):

  tokenizer = RegexpTokenizer(r'\w+')
  tokens = tokenizer.tokenize(txt)
  filtered_words = [w for w in tokens if len(w) > 2 and w not in STOP_WORDS]
  non_ascii_removed = [remove_non_ascii(w) for w in filtered_words]

  return " ".join(non_ascii_removed)

def preprocess_column(df,col_name):

  df[col_name] = df[col_name].apply(lambda x: x.replace("nan",""))
  df[col_name] = df[col_name].apply(lambda x: x.strip().lower())
  df[col_name] = df[col_name].apply(lambda x: x.replace('{html}',""))

  reg_obj = re.compile('&[a-zA-Z0-9]*;|[0-9]+|http\S+|<.*?>')

  df[col_name] = df[col_name].apply(lambda x: re.sub(reg_obj, '', x))
  df[col_name] = df[col_name].apply(tokenise_text)

  return df[col_name]
```

Data Cleaning - Categorical Column [4]

Employment Type, Required Experience and Required Education

- Missing Values has been replaced based on the feature values matching in the description and requirement column for the Employment Type, Required Experience and Required Education feature.
- Replaced the Missing Values with ‘Unspecified’ and Converting to String.

```
def transform_row(single_row):

  concat_req_des = single_row["requirements"] + " " + single_row["description"]
  emp_type = single_row["employment_type"].strip()
  req_exp = single_row["required_experience"].strip()
  req_edu = single_row["required_education"].strip()

  # If employment type is missing or is empty
  if len(emp_type) == 0:
    if (re.search(r'[fF]ull.[tT]ime',concat_req_des)):
      single_row["employment_type"] = "Full-time"
    elif (re.search(r'[pP]art.[tT]ime',concat_req_des )):
      single_row["employment_type"] = "Part-time"
    elif (re.search(r'[cC]ontract', concat_req_des)):
      single_row["employment_type"] = "Contract"
    elif (re.search(r'[tT]emporary', concat_req_des)):
      single_row["employment_type"] = "Temporary"
    elif (re.search(r'[iI]nternship',concat_req_des)):
      single_row["employment_type"] = "Internship"
    else:
      single_row["employment_type"] = "Unspecified"


  # if Required experience is empty

  if len(req_exp) == 0:
    if (re.search(r'[nN]o.[mM]inimum.[eE]xperience',concat_req_des)):
      single_row['required_experience'] = 'Not Applicable'
    elif (re.search(r'[eE]ntry.[lL]evel',concat_req_des)):
      single_row['required_experience'] = 'Entry level'
    elif (re.search(r'[eE]xperienced',concat_req_des)):
      single_row['required_experience'] = 'Mid-Senior level'
    elif (re.search(r'[dD]irector',concat_req_des)):
      single_row['required_experience'] = 'Director'
    elif (re.search(r'[iI]nternship',concat_req_des)):
      single_row['required_experience'] = 'Internship'
    elif (re.search(r'[eE]xecutive',concat_req_des)):
      single_row['required_experience'] = 'Executive'
    elif (re.search(r'[aA]ssociate',concat_req_des)):
      single_row['required_experience'] = 'Associate'
    elif (re.search(r'[Mm]id.[Ss]enior.[lL]evel',concat_req_des)):
      single_row['required_experience'] = 'Mid-Senior level'
    else:
      single_row["required_experience"] = "Unspecified"

    # If Required Education is missing/empty

  if len(req_edu) == 0:
    if (re.search(r'[hH]igh*school',concat_req_des)):
      single_row['required_education'] = 'High School or equivalent'
    elif (re.search(r'[hH][sS].[dD]iploma',concat_req_des)):
      single_row['required_education'] = 'Vocational - HS Diploma'
    elif (re.search(r'[cC]ertification',concat_req_des)):
      single_row['required_education'] = 'Certification'
    elif (re.search(r'[bB]achelor',concat_req_des)):
      single_row['required_education'] = 'Bachelor\'s Degree'
```

Exploratory Data Visualization - after Data cleaning

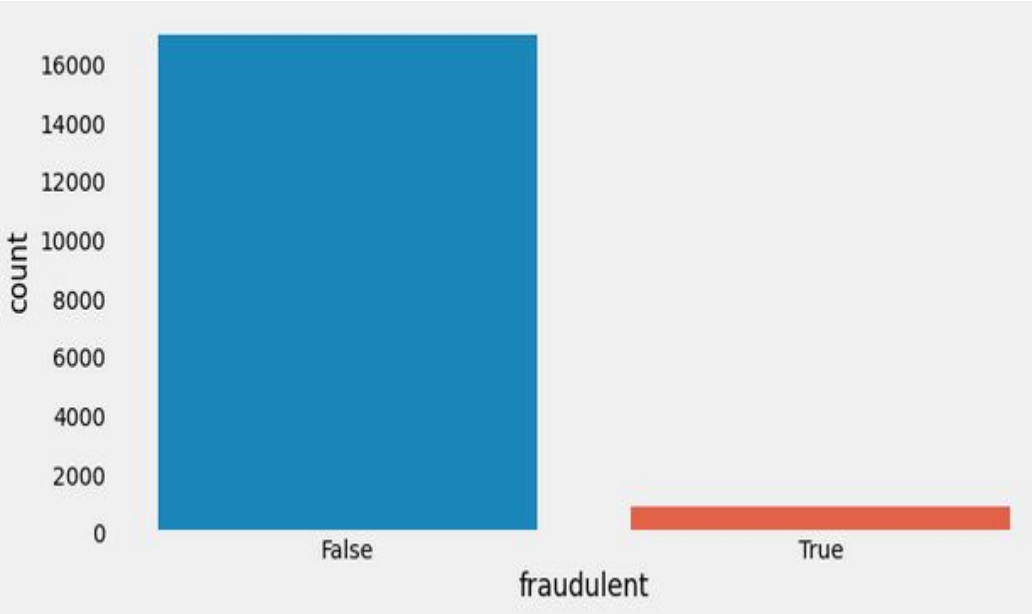
- Once data cleaning is completed, we started exploring the data
- The data was visualized using pandas, matplotlib, seaborn and plotly
- Word clouds, Grouped bar plots, Bar plots, geo maps were created to explore the data
- Based on this exploration, next step of Data preparation can be done for modeling

Target feature

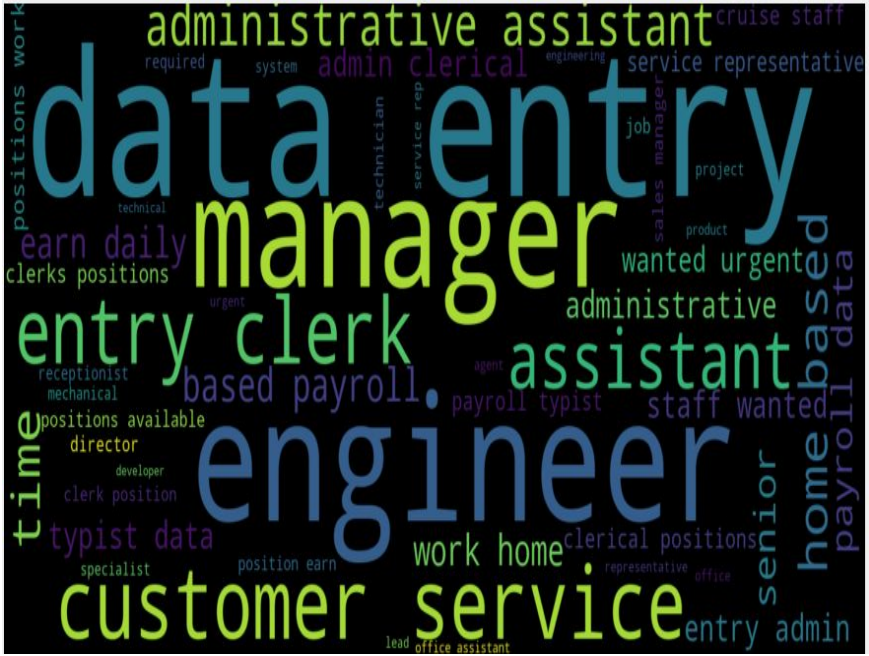
This indicates an imbalance in dataset wrt target feature.

- ***The data is very imbalanced between the actual and fake jobs.***
- ***Fraudulent job promotion postings are pervasive but are overshadowed by the real ads.***

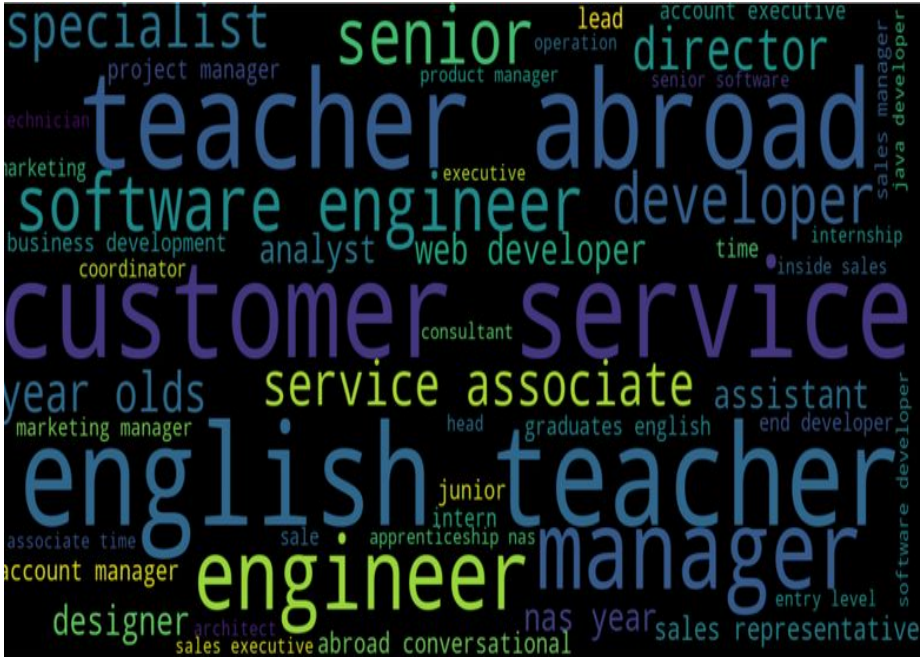
Sampling must be done before data modeling to avoid underfitting or overfitting



Word cloud - *title* for fake & actual job posts



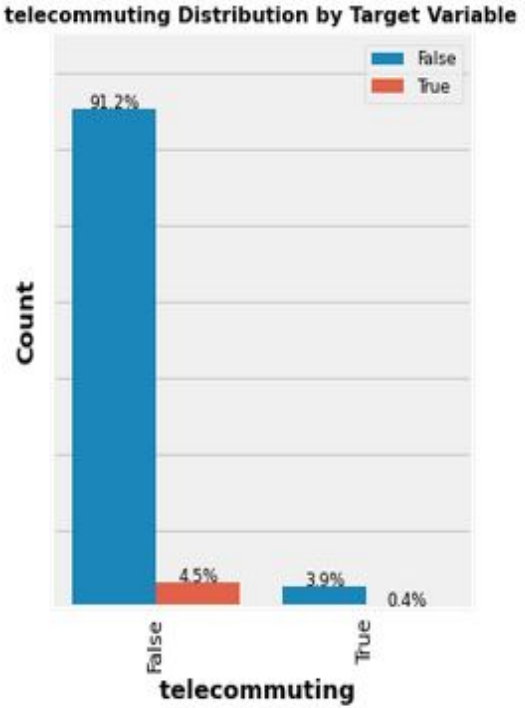
Fraudulent Job Posts

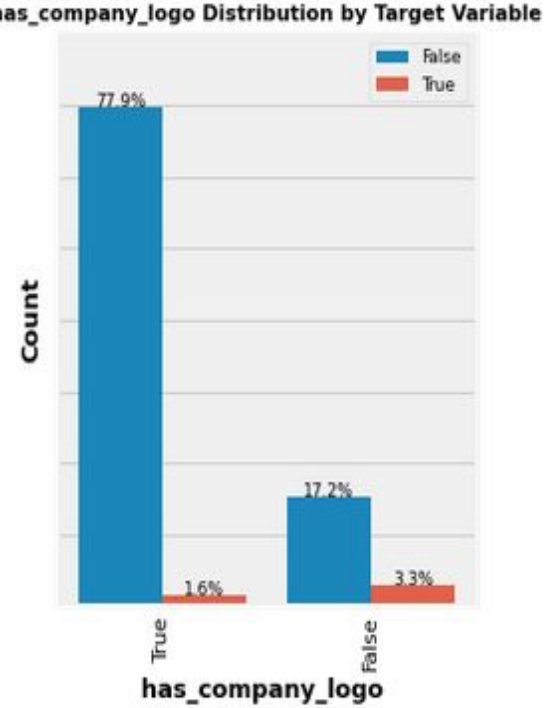


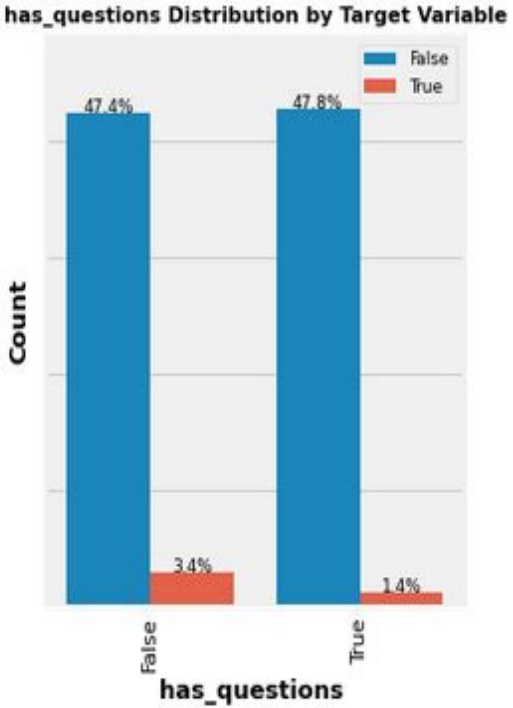
Non- Fraudulent Job Posts

Majority of ***Fraudulent*** jobs - *telecommuting* is False, No company logo, Has no questions

Majority of ***Non Fraudulent*** jobs - *telecommuting* is False, Has company logo, almost equally Split between having and not having questions



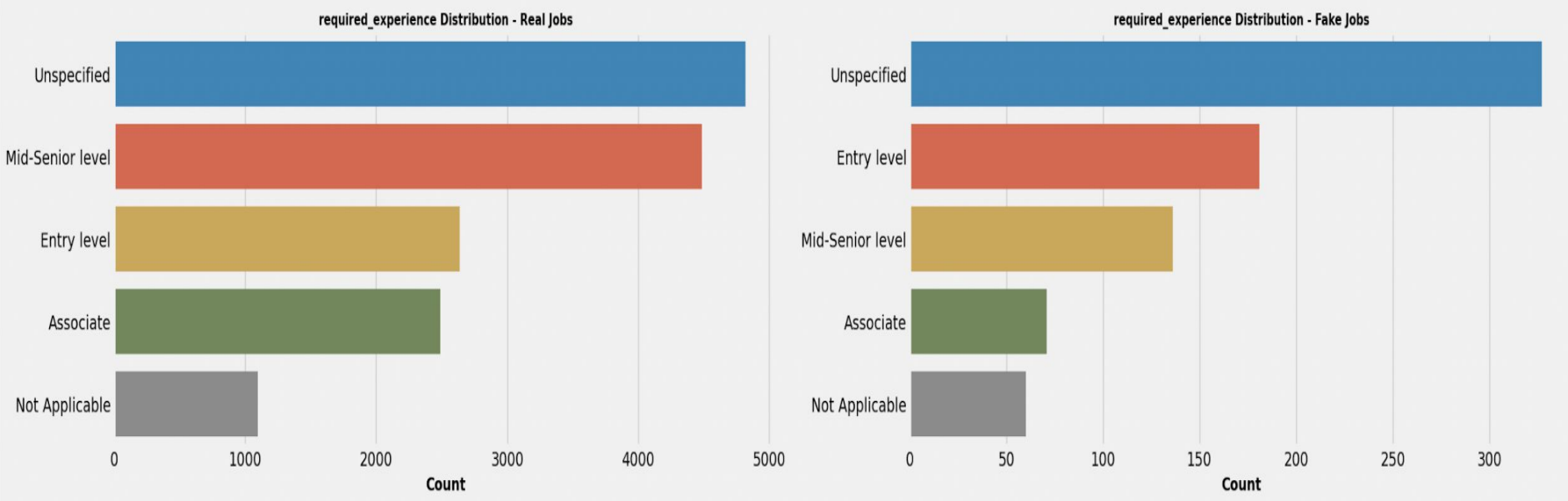




required_experience - Top 5 values

Real - Unspecified, Mid-senior level, Entry level
Fake - Unspecified, Entry level, Mid-senior level

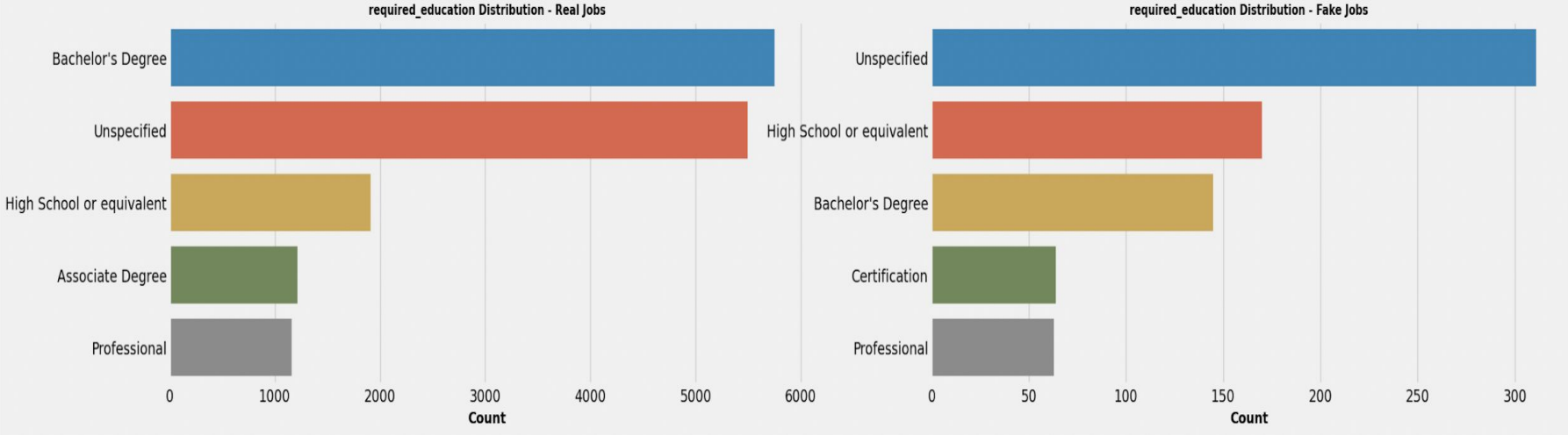
We can see that most of ***real*** jobs require **mid-senior level** experience whereas most of the ***fake*** jobs require **entry level** experience



required_education - Top 5 values

Real- Bachelor's Degree, Unspecified, High School
Fake - High School or equivalent , Bachelor's degree

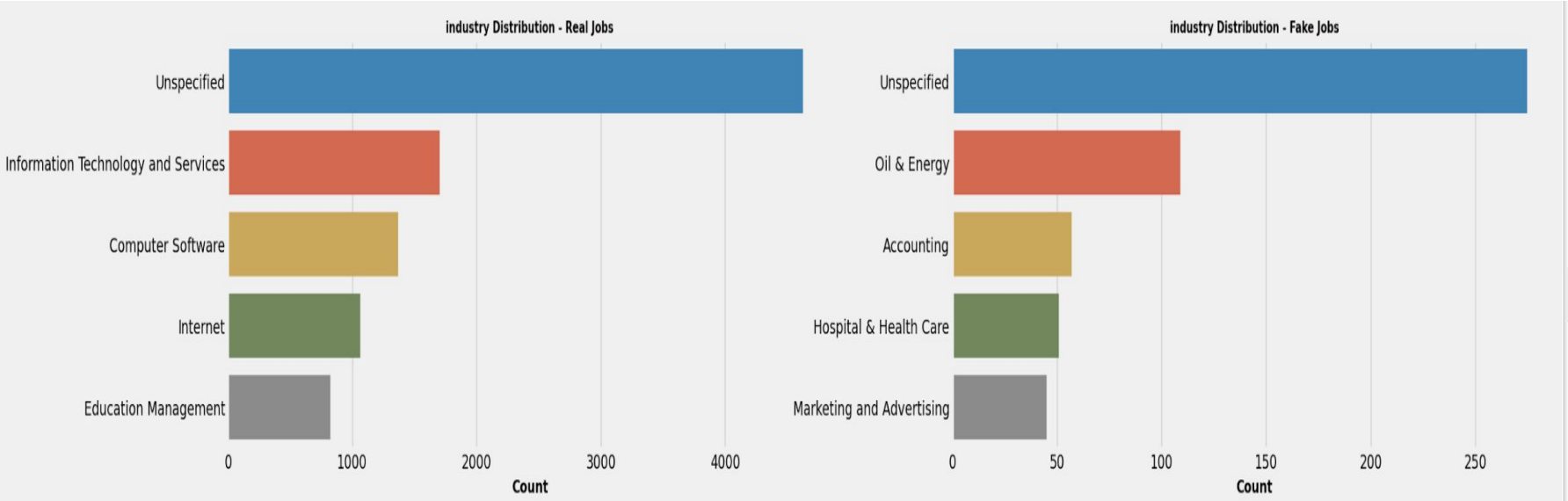
Most of the ***real*** jobs require **bachelor's degree** whereas most of ***fake*** jobs require **high school or equivalent** education apart from unspecified



Industry - Top 5 values

Real - Unspecified, Information Technology & Services, Computer Software
Fake - Unspecified, Oil & Energy, Accounting

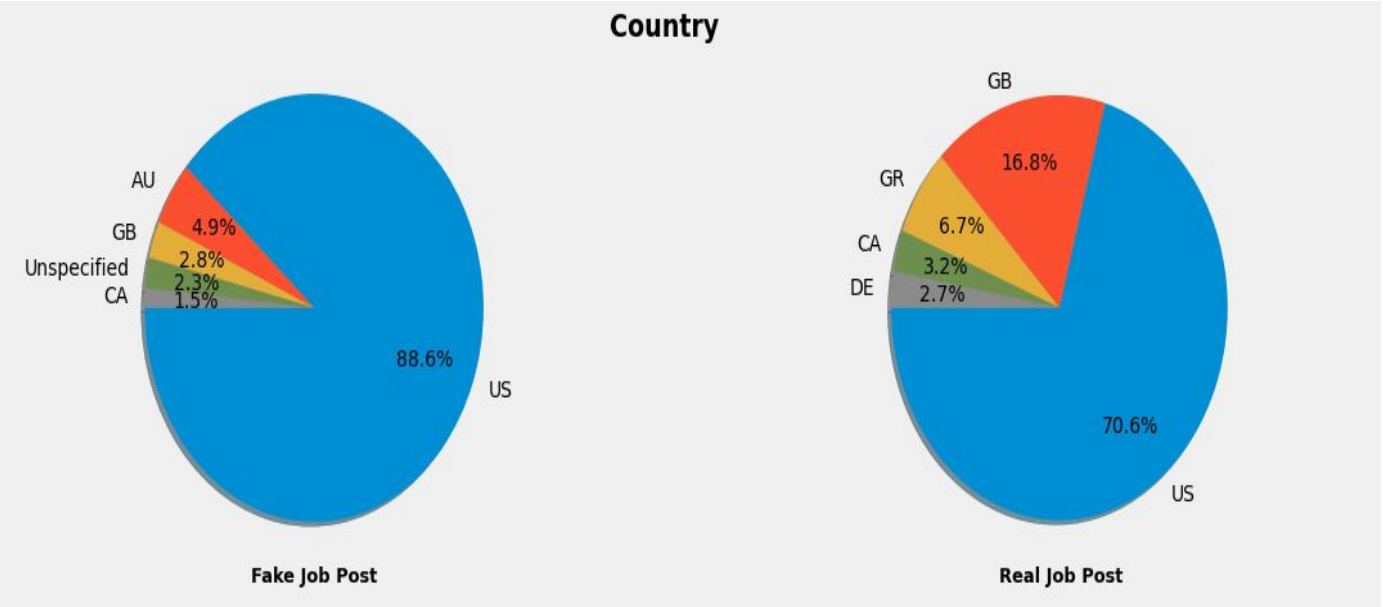
Information and technology services industry has lot of **real** jobs whereas most of **fake** jobs has **Oil & Energy**



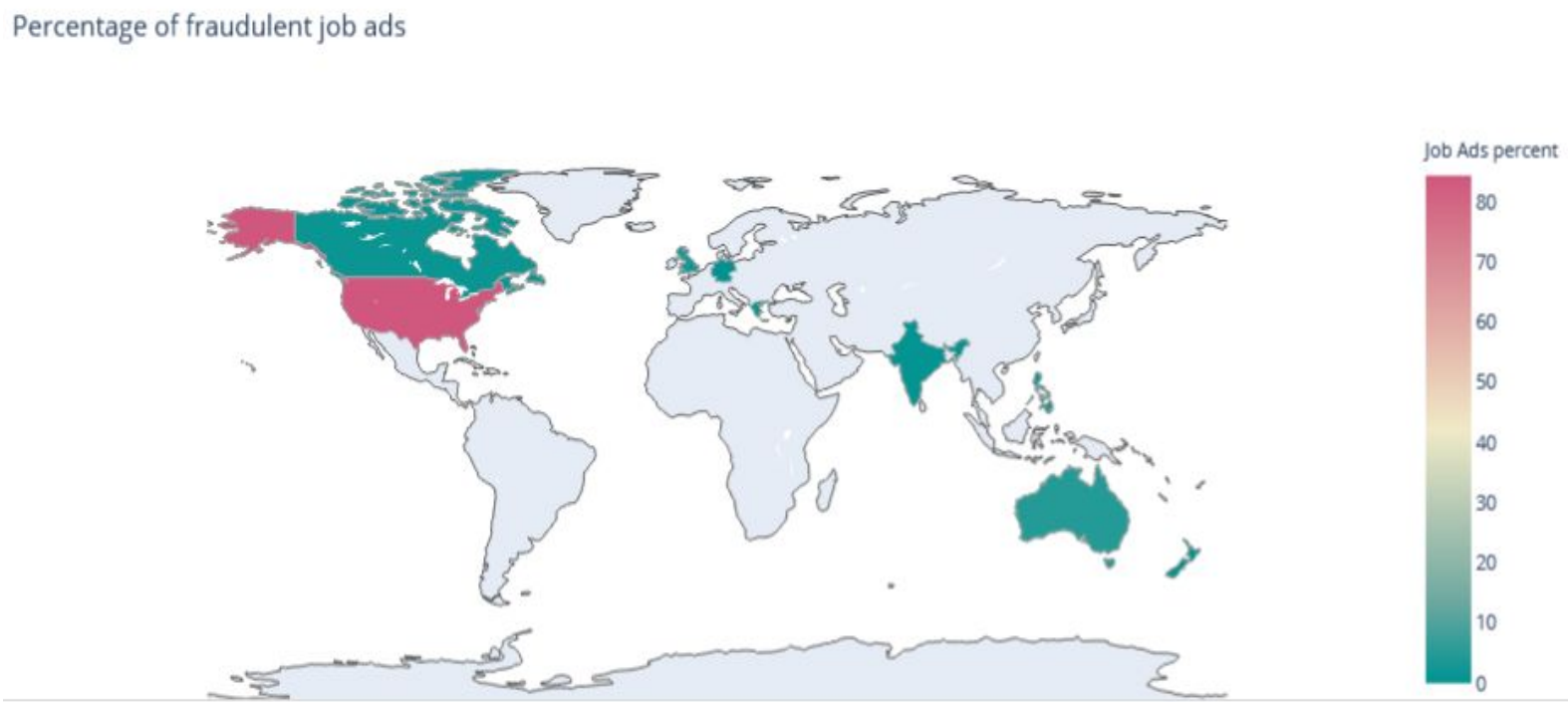
Top 3 countries

Fake Jobs- USA, Australia, Great Britain

Real Jobs - USA, Great Britain, Greece



Countries with top Fraudulent Job posts



DEFINING PERFORMANCE METRICS

- Accuracy
- Confusion Matrix
- Precision
- Recall
- ROC Curve
- AUC

Accuracy

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Indicator of percentage of correct predictions

CONFUSION MATRIX

CONFUSION MATRIX		*Predicted*	
		Fraudulent	Non-Fraudulent
Target	Fraudulent	True Positive (TP)	False Negative (FN)
	Non-Fraudulent	False Positive (FP)	True Negative (TN)

- Shows a matrix of predicted values against target values
- True Positives & True Negatives - correct predictions of respective target classes
- False Negative - refers to fraudulent jobs wrongly classified as Non-fraudulent
- False Positive - refers to Non-fraudulent jobs wrongly classified as fraudulent
- Many evaluation metrics are calculated from the Confusion Matrix

PRECISION

$$\text{Precision} = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

- Indicates percentage of Fraudulent jobs classified correctly out of all those classified as Fraudulent

RECALL

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative}$$

- Indicates percentage of Fraudulent jobs classified correctly out of all those jobs that are actually Fraudulent

F1 Score

$$recision = \frac{TP}{TP + FP}$$

$$\frac{2 \text{ x (Precision x Recall)}}{\text{Precision + Recall}}$$

- Combined measure of precision and recall having values between [0,1]
- Harmonic mean of Precision and Recall
- Values =1 indicate perfect precision and recall
- Values =0 indicate either precision or recall to be zero

TRUE POSITIVITY RATE (TPR)

$$\text{TPR (sensitivity)} = \frac{TP}{TP + FN}$$

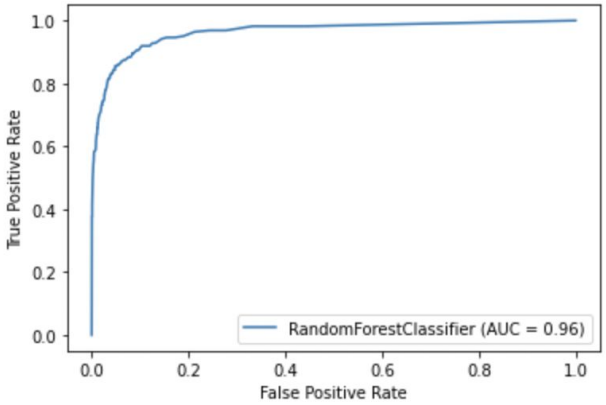
- Indicates percentage of Fraudulent jobs classified correctly out of all those jobs that are actually Fraudulent
- Same as Recall

FALSE POSITIVITY RATE (FPR)

$$\text{FPR (1-specificity)} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

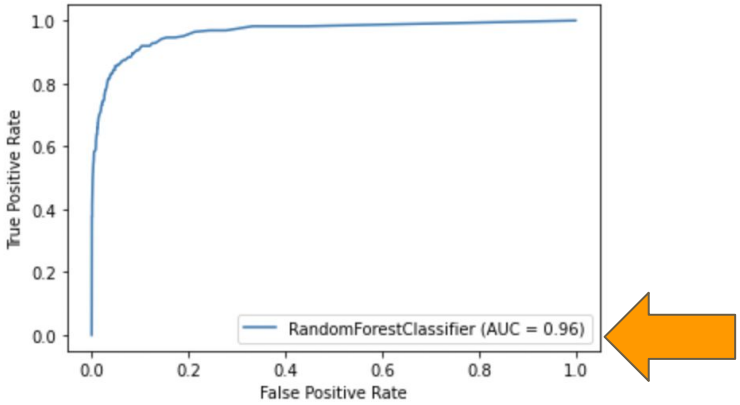
- Indicates percentage of Non-Fraudulent jobs classified wrongly as Fraudulent out of all those jobs that are actually Non-Fraudulent

Receiver Operating Curve (ROC)



- Plotted with True Positive Rate(TPR) against False Positive Rate(FPR) for various threshold values
- Can be used to check which threshold classifier gives a good balance between TPR and FPR

Area Under the Curve(AUC)



- It is an performance metric aggregated across all possible classification thresholds.

MODELING

- Feature Engineering
- K-fold validation
- Model selection, comparison, tuning and evaluation

Models Used for Prediction

- Linear Support Vector Classification
- Gradient Boosting
- Gaussian Naive Bayes
- Logistic Regression
- KNeighbors
- Random Forest
- XGBoost

Dataset is split into training, validation and test

TRAINING - 60%

```
X_train.shape    y_train.shape
(10728, 17)      (10728, 1)
```

Used for training the model

Used to check if sampling will make a difference

VALIDATION- 20%

```
X_val.shape      y_val.shape
(3576, 17)       (3576, 1)
```

Validating model

Used in Feature Engineering

Hyperparameter optimization

TESTING- 20%

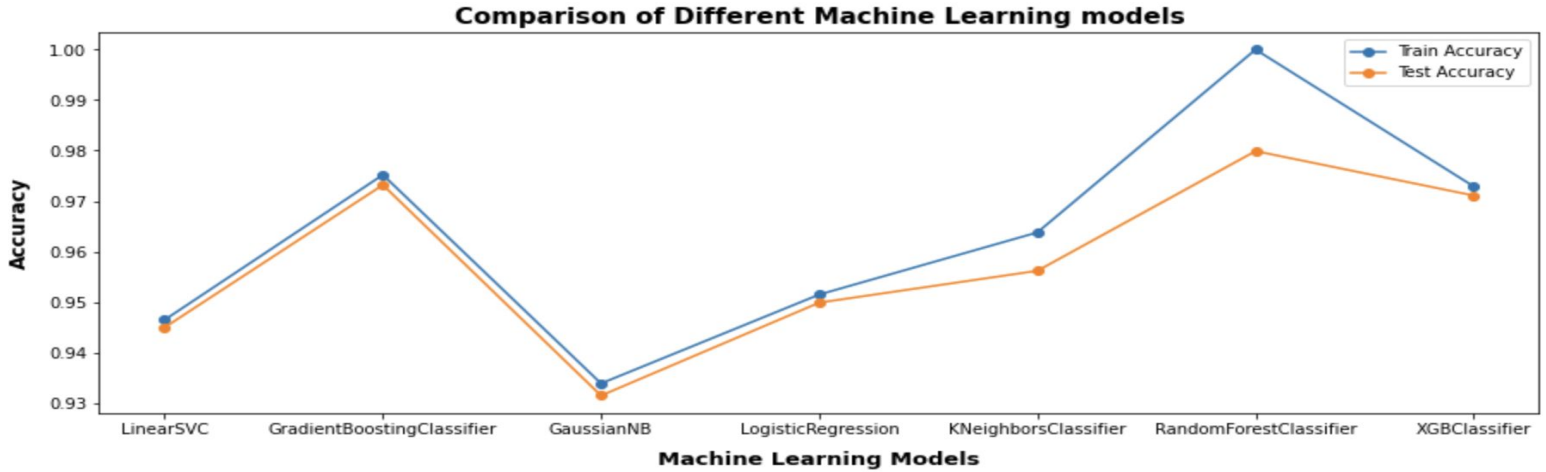
```
X_test.shape     y_test.shape
(3576, 17)       (3576, 1)
```

Testing the final model with optimal features and hyperparameters

Before Sampling

Model Prediction Summary

	Model Names	Train Accuracy	Test Accuracy	Precision	Recall	F1 Score
0	LinearSVC	0.9465	0.945	0.0673	0.283	0.1087
1	GradientBoostingClassifier	0.9752	0.9732	0.4978	0.9328	0.6491
2	GaussianNB	0.9339	0.9315	0.3139	0.3139	0.3139
3	LogisticRegression	0.9515	0.9499	0.0045	0.3333	0.0088
4	KNeighborsClassifier	0.9638	0.9562	0.3049	0.6239	0.4096
5	RandomForestClassifier	1	0.9799	0.6233	0.9586	0.7554
6	XGBClassifier	0.9729	0.9711	0.4484	0.9434	0.6079



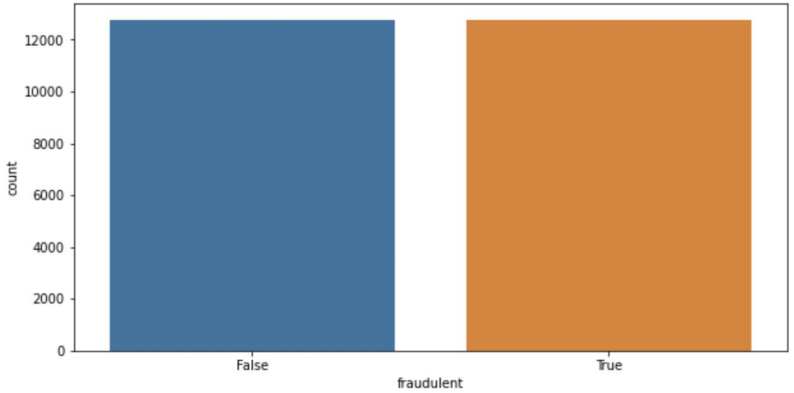
OverSampling of data using SMOTE

- Transformed the categorical features to numerical features using **label.fit_transform()**
- Data is split to Train and Test data in the ratio 75:25
- Performed oversampling of the data (**imblearn.over_sampling**) using SMOTE

```
Shape of X before SMOTE : (17880, 17)
 Shape of X after SMOTE : (25534, 17)

Balance of positive and negative classes (%):
50.0
50.0
```

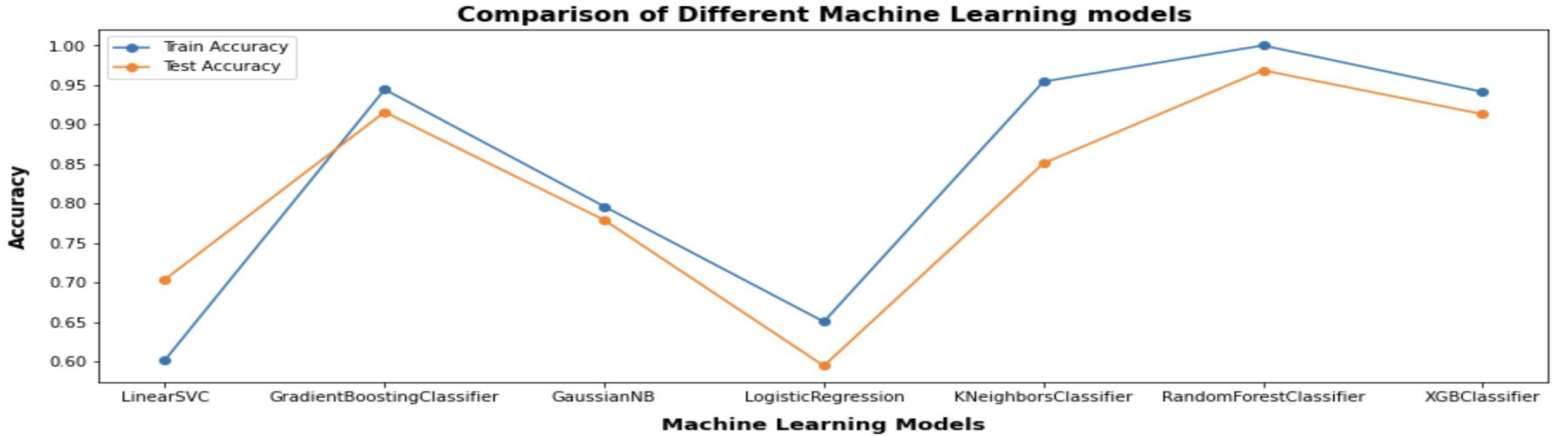
```
plt.figure(figsize=(10,5))
sns.countplot(x='fraudulent',data=y_sm_df)
plt.show()
```



After SMOTE OverSampling - Considering all the features

Model Prediction Summary

	Model Names	Train Accuracy	Test Accuracy	Precision	Recall	F1 Score
0	LinearSVC	0.6019	0.704	0.5381	0.0896	0.1536
1	GradientBoostingClassifier	0.9443	0.9157	0.8296	0.3531	0.4953
2	GaussianNB	0.7962	0.779	0.6682	0.1402	0.2317
3	LogisticRegression	0.6503	0.5944	0.722	0.0842	0.1508
4	KNeighborsClassifier	0.9542	0.8508	0.6054	0.1691	0.2882
5	RandomForestClassifier	1	0.9685	0.7534	0.6614	0.7044
6	XGBClassifier	0.9412	0.913	0.8206	0.344	0.4848



Feature Engineering

- Performed feature engineering using RandomForestClassifer() and XGBoostClassifier() method
- Extracting the importance feature using **model.feature_importances_**
- Comparison is done by combining both the RandomForest and XGBoostClassifier importance features

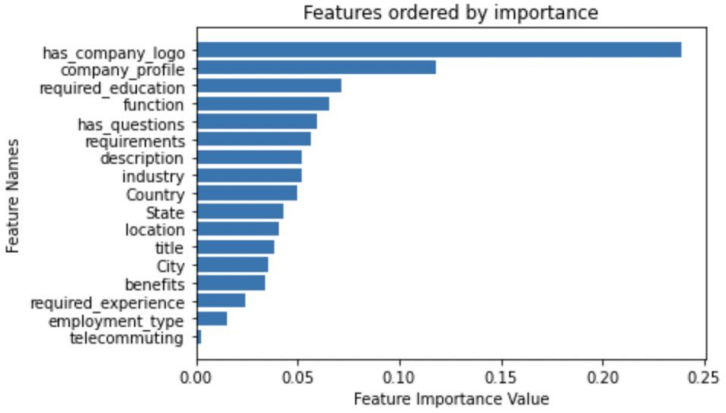
Feature Importance Using RandomForest and XGBoost Classifier

Feature Importance using : RandomForestClassifier Classifier

Confusion Matrix

```
[[4168   56]
 [  79  167]]
```

```
    Recall Score : 0.6789
 Precision Score : 0.7489
  Accuracy Score : 0.9688
        F1 Score : 0.7122
```

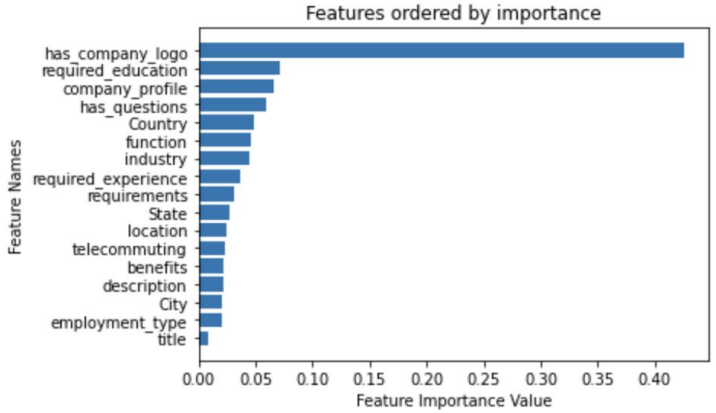


Feature Importance using : XGBoost Classifier

Confusion Matrix

```
[[3898   40]
 [ 349  183]]
```

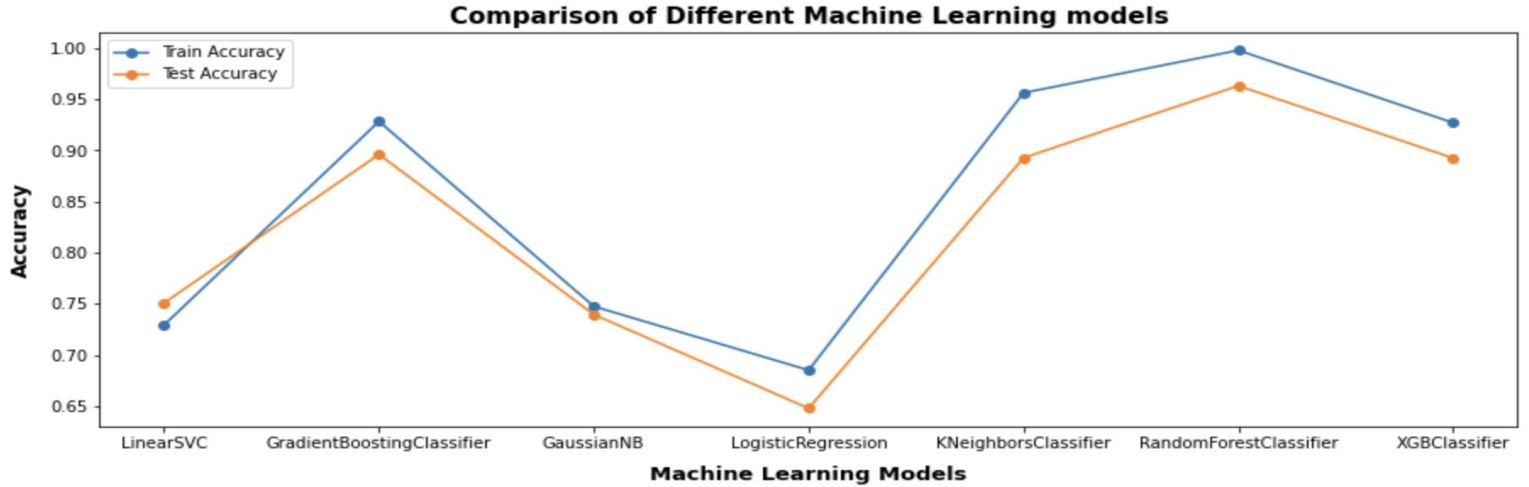
```
    Recall Score : 0.3440
 Precision Score : 0.8206
  Accuracy Score : 0.9130
        F1 Score : 0.4848
```



After selecting the important features

Model Prediction Summary

	Model Names	Train Accuracy	Test Accuracy	Precision	Recall	F1 Score
0	LinearSVC	0.7293	0.7508	0.6592	0.1241	0.2088
1	GradientBoostingClassifier	0.9283	0.896	0.8565	0.3061	0.451
2	GaussianNB	0.7474	0.7394	0.704	0.125	0.2123
3	LogisticRegression	0.685	0.6477	0.704	0.0942	0.1662
4	KNeighborsClassifier	0.9562	0.8926	0.7623	0.2848	0.4146
5	RandomForestClassifier	0.9978	0.9631	0.7848	0.5993	0.6796
6	XGBClassifier	0.927	0.8924	0.8565	0.2984	0.4426



K-fold Validation- another way to validate model

- Without sampling, for imbalance dataset, we used k-fold validation to validate the correct model
- It reduces overfitting
- Randomly Splits the data into 10 folds
- Process repeats 10 times ,each fold tests the dataset

K-fold Cross : Prediction Summary

	Model Names	Accuracy	Precision	Recall	F1 Score
0	RandomForestClassifier	0.97792	0.45995	0.981	0.61456
1	XGBClassifier	0.97195	0.29471	0.94718	0.43667

Hyperparameter optimization

- We used these parameters to tune the Random Forest Model
- Best parameters after hyperparameter tuning will be selected for evaluating final model
- K-fold cross validation was checked for k=3,k=5 and k=10

```
# Number of trees in random forest
n_estimators = [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]

# Number of features to consider at every split
max_features = ['auto', 'sqrt']

# Maximum number of levels in tree
max_depth = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, None]

# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]

# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]

# Method of selecting samples for training each tree
bootstrap = [True, False]
```

Best parameters - without important features

```
rf_random.best_params_
```

```
{'bootstrap': False,
 'max_depth': 60,
 'max_features': 'auto',
 'min_samples_leaf': 2,
 'min_samples_split': 2,
 'n_estimators': 600}
```

Best parameters - with important features

```
rf_random_imp.best_params_
```

```
{'bootstrap': False,
 'max_depth': 50,
 'max_features': 'auto',
 'min_samples_leaf': 2,
 'min_samples_split': 2,
 'n_estimators': 2000}
```

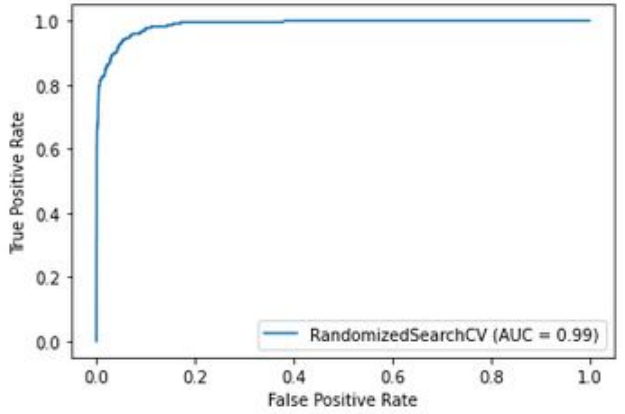
Metrics with model- all features + tuned hyperparameters

Classification Report :

	precision	recall	f1-score	support
False	0.98	1.00	0.99	4247
True	0.97	0.64	0.77	223
accuracy			0.98	4470
macro avg	0.97	0.82	0.88	4470
weighted avg	0.98	0.98	0.98	4470

Prediction for Tuned model

Train Dataset Accuracy : 0.9999
Test Dataset Accuracy : 0.981
ROC AUC Score : 0.82



Confusion Matrix

[[4242 80]
[5 143]]

FINAL MODEL EVALUATION

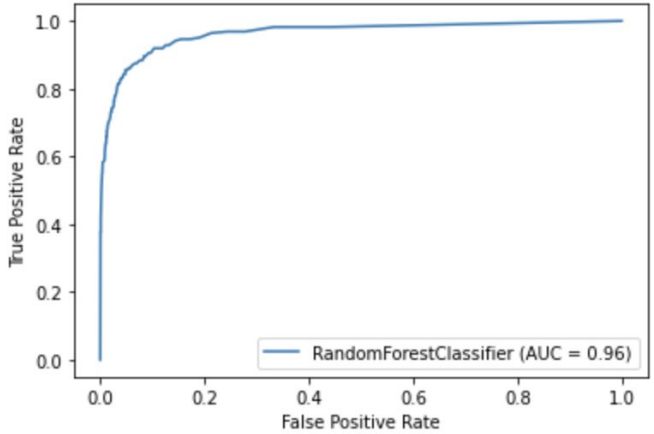
- Performance Tuning
- Test and Evaluate on final model

Final Model Evaluation

With IMPORTANT features

Prediction for RandomForestClassifier

Train Dataset Accuracy : 0.9979
Test Dataset Accuracy : 0.9624
ROC AUC Score : 0.874



Confusion Matrix

[[4129 50]
[118 173]]

Classification Report :

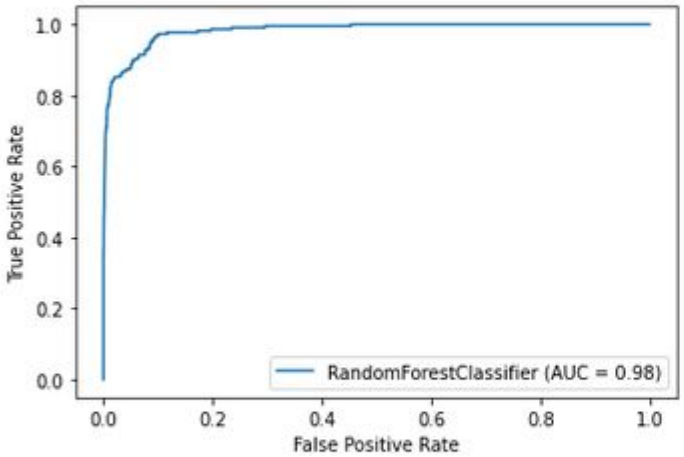
	precision	recall	f1-score	support
False	0.99	0.97	0.98	4247
True	0.59	0.78	0.67	223
accuracy			0.96	4470
macro avg	0.79	0.87	0.83	4470
weighted avg	0.97	0.96	0.96	4470

Final Model Evaluation

With Optimal Hyperparameters and IMPORTANT features

```
Prediction for FINAL Tuned model WITH IMPORTANT FEATURES & TUNED PARAMETER

Train Dataset Accuracy : 0.9955
 Test Dataset Accuracy : 0.9803
 ROC AUC Score         : 0.8324
```



Confusion Matrix

```
[[4233   74]
 [  14  149]]
```

Classification Report :

	precision	recall	f1-score	support
False	0.98	1.00	0.99	4247
True	0.91	0.67	0.77	223
accuracy			0.98	4470
macro avg	0.95	0.83	0.88	4470
weighted avg	0.98	0.98	0.98	4470

CONCLUSION

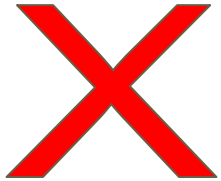
- Results
- Future work

So- What worked for our problem?

Random Forest Classifier

Feature Selection

Hyperparameter optimization



LogisticRegression, LInear SVC, Gaussian Naive Bayes, XGBoost, KNN

Random Undersampling

Oversampling with SMOTE

Random Undersampling + SMOTE Oversampling

Conclusion

Random Forest Classifier performed the best with a F1 score of 0.77 while using important features selected and tuned Hyper parameters

Selected Important Features:

```
['has_company_logo','company_profile','required_education',

'function','has questions' ,'Country',
'required_experience','requirements','industry']
```

Optimal Hyperparameters:

```
 {'bootstrap': False, 'max depth': 50, 'max features': 'auto',
'min_samples_leaf': 2,'min_samples_split': 2, 'n_estimators': 2000}
```

Future Work

Create an app where users can upload a job description and use the ML model to classify it as fraudulent or not with a probability score

Thank You