```
from google.colab import files
uploaded = files.upload()
```

```
Choose Files No file chosen          Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving fake_job_postings.csv to fake_job_postings.csv
```

```
import pandas as pd
df = pd.read_csv('fake_job_postings.csv')
```

```
import os
uploaded_files = os.listdir()
uploaded_files
```

```
['.config', 'fake_job_postings.csv', 'drive', 'sample_data']
```

```
import pandas as pd
df = pd.read_csv('fake_job_postings.csv')
```

```
df.head()
```

	job_id	title	location	department	salary_range	company_profile	description	requirements	benefits	tele
0	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	
1	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...	
2	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	
3	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 ...	
4	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	

```
df.info()
```

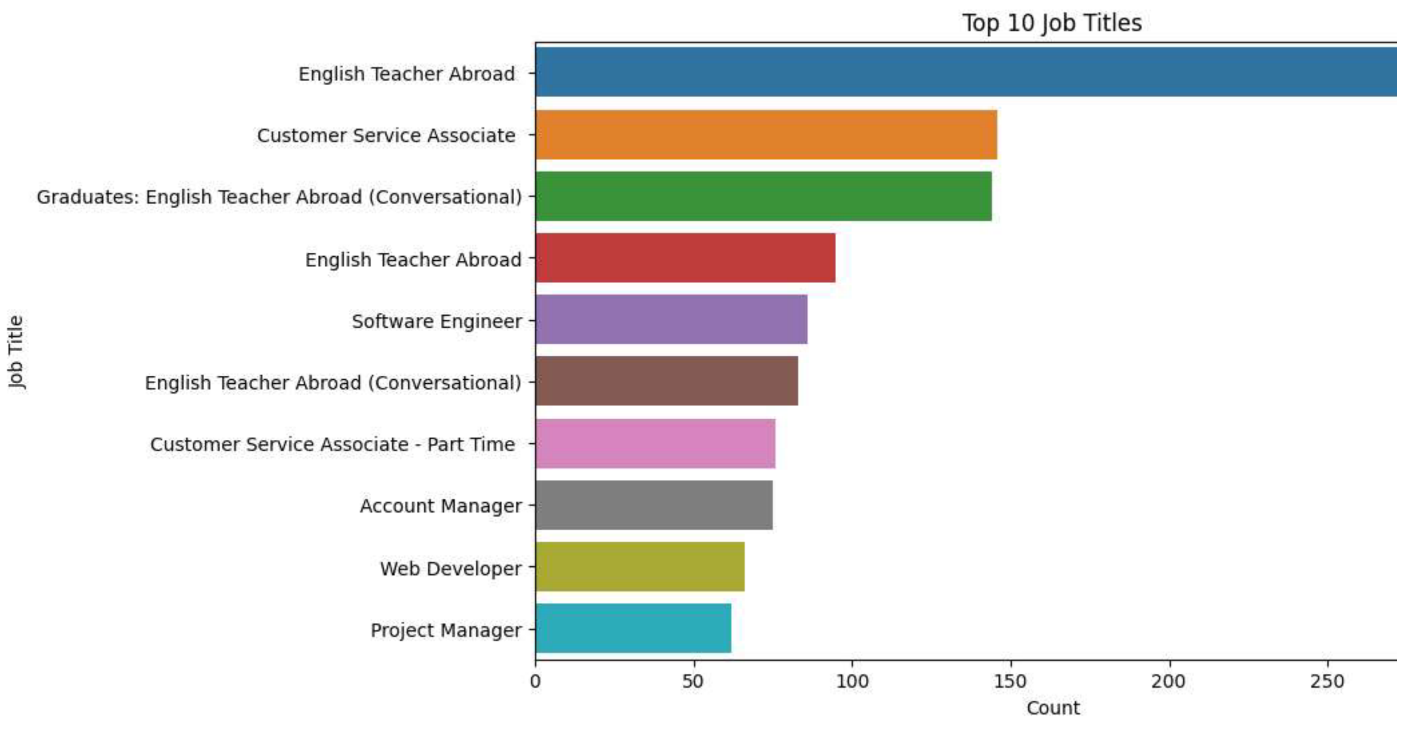
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 17880 entries, 0 to 17879
Data columns (total 18 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   job_id               17880 non-null  int64 
 1   title                17880 non-null  object
 2   location             17534 non-null  object
 3   department           6333 non-null   object
 4   salary_range         2868 non-null   object
 5   company_profile      14572 non-null  object
 6   description          17879 non-null  object
 7   requirements         15185 non-null  object
 8   benefits             10670 non-null  object
 9   telecommuting        17880 non-null  int64 
 10  has_company_logo     17880 non-null  int64 
 11  has_questions        17880 non-null  int64 
 12  employment_type      14409 non-null  object
 13  required_experience  10830 non-null  object
 14  required_education   9775 non-null   object
 15  industry             12977 non-null  object
 16  function             11425 non-null  object
 17  fraudulent           17880 non-null  int64
dtypes: int64(5), object(13)
memory usage: 2.5+ MB
```

```
from sklearn.preprocessing import LabelEncoder

# Create a LabelEncoder instance
label_encoder = LabelEncoder()

# Fit and transform the column to numeric values
df['salary_range'] = label_encoder.fit_transform(df['salary_range'])
```

```
df.head()
```

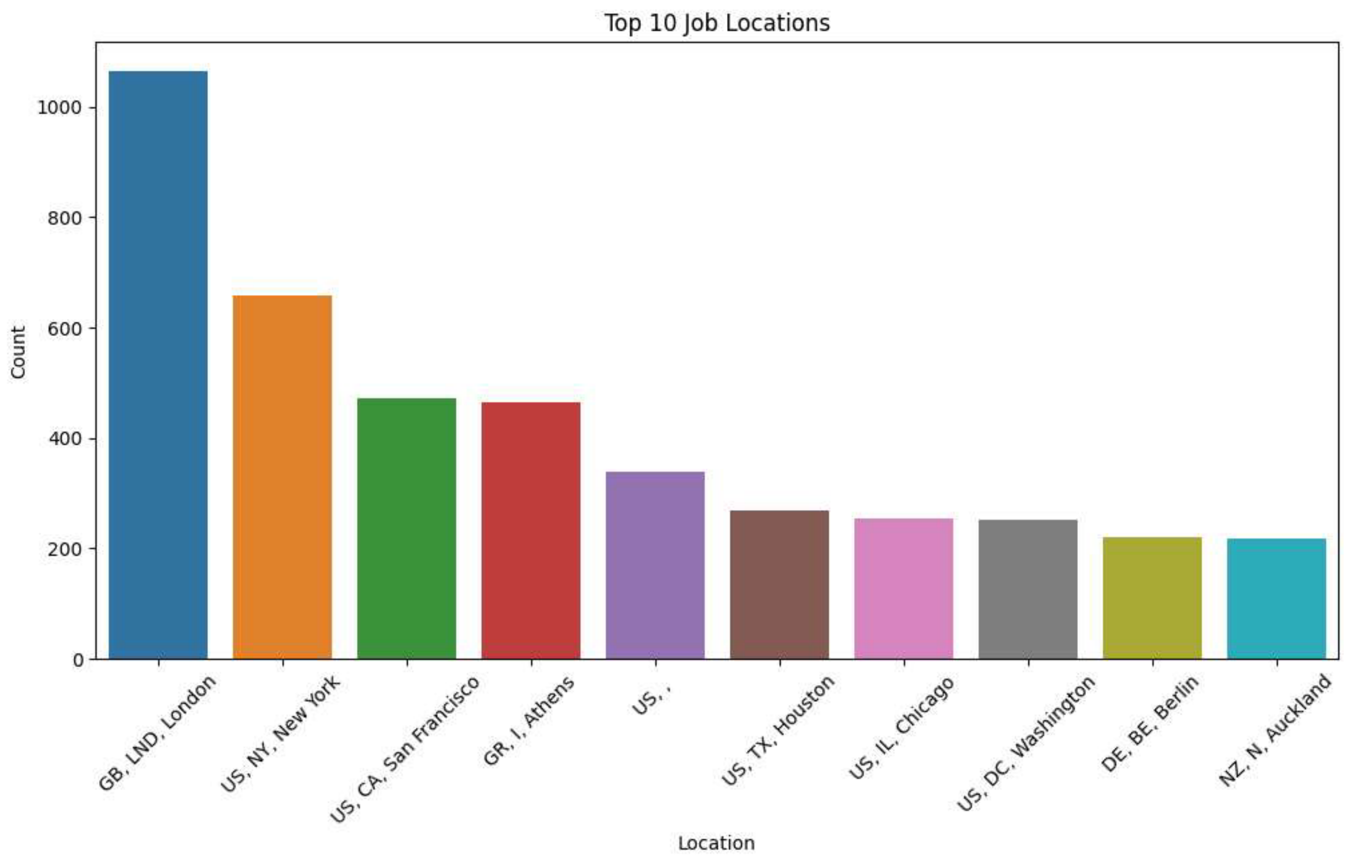
	title	location	department	salary_range	company_profile	
0	Marketing Intern	US, NY, New York	Marketing	874	We're Food52, and we've created a groundbreaki...	
1	Customer Service - Cloud Video Production	NZ, , Auckland	Success	874	90 Seconds, the worlds Cloud Video Production ...	Custo
2	Commissioning Machinery Assistant (CMA)	US, IA, Wever	NaN	874	Valor Services provides Workforce Solutions th...	
3	Account Executive - Washington DC	US, DC, Washington	Sales	874	Our passion for improving quality of life thro...	
4	Bill Review Manager	US, FL, Fort Worth	NaN	874	SpotSource Solutions LLC is a Global Human Cap...	Health Ca

```
# datavisualization barplot on job title
import matplotlib.pyplot as plt
import seaborn as sns

common_job_titles = df['title'].value_counts().head(10)
plt.figure(figsize=(10, 6))
sns.barplot(x=common_job_titles.values, y=common_job_titles.index, orient="h")
plt.xlabel('Count')
plt.ylabel('Job Title')
plt.title('Top 10 Job Titles')
plt.show()
```



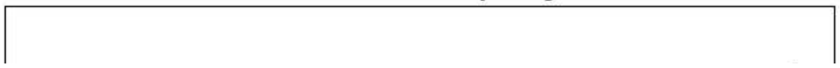
```
#countpot on job loactions
plt.figure(figsize=(12, 6))
```

```
sns.countplot(x='location', data=df, order=df['location'].value_counts().index[:10])
plt.xticks(rotation=45)
plt.xlabel('Location')
plt.ylabel('Count')
plt.title('Top 10 Job Locations')
plt.show()
```



```
#boxplot on salaryranges
plt.figure(figsize=(8, 6))
sns.boxplot(x='salary_range', data=df, orient="h")
plt.xlabel('Salary Range')
plt.title('Distribution of Salary Ranges')
plt.show()
```

Distribution of Salary Ranges



```
 us_jobs = df[df['location'].str.contains('US', case=False, na=False)]

# Count job titles
common_job_titles = us_jobs['title'].value_counts().head(10)

# Display the most common job titles
print(common_job_titles)
```

```
     English Teacher Abroad                                295
     Graduates: English Teacher Abroad (Conversational)    144
     Customer Service Associate                            136
     English Teacher Abroad                                 89
     English Teacher Abroad (Conversational)                83
     Customer Service Associate - Part Time                 74
     Graduates: English Teacher Abroad                      55
     Software Engineer                                      47
     Customer Service Representative                        43
     Administrative Assistant                               42
     Name: title, dtype: int64
```

```
# Filter the dataset to include only fraudulent (fake) job listings
fraudulent_jobs = df[df['fraudulent'] == 1]

# Count the occurrences of each department in fraudulent job listings
fraudulent_department_counts = fraudulent_jobs['department'].value_counts()

# Find the department with the most fake jobs
most_common_fraudulent_department = fraudulent_department_counts.idxmax()

# Display the department with the most fake jobs
print("Department with the most fake jobs:", most_common_fraudulent_department)
```

```
     Department with the most fake jobs: Sales
```

```
uk_jobs = df[df['location'].str.contains('UK')]

# Step 2: Group the data by department or function and calculate the average salary
average_salary_by_department = uk_jobs.groupby('department')['salary_range'].mean()
average_salary_by_function = uk_jobs.groupby('function')['salary_range'].mean()

# Step 3: Find the department or function with the highest average salary
highest_paying_department = average_salary_by_department.idxmax()
highest_paying_function = average_salary_by_function.idxmax()

# Step 4: Print the results
print("Department with the highest average salary_range in the UK:", highest_paying_department)
print("Function with the highest average salary_range in the UK:", highest_paying_function)
```

```
     Department with the highest average salary_range in the UK: CS
     Function with the highest average salary_range in the UK: Customer Service
```

```
from sklearn.preprocessing import LabelEncoder

# Create a LabelEncoder instance
label_encoder = LabelEncoder()

# Transform each column separately
df['title'] = label_encoder.fit_transform(df['title'])
df['location'] = label_encoder.fit_transform(df['location'])
df['department'] = label_encoder.fit_transform(df['department'])
df['company_profile'] = label_encoder.fit_transform(df['company_profile'])
df['function']=label_encoder.fit_transform(df['function'])
```

```
df.head()
```

	title	location	department	salary_range	company_profile	function	fraudulent
0	6043	2535	758	874	1526	22	0
1	2183	1073	1161	874	50	7	0
2	1763	1867	1054	874	1389	18	0

```
#feature selection
all_features=list(df.columns)
```

```
all_features.remove('fraudulent')
```

```
input_features=all_features
```

```
x=df[input_features]
y=df['fraudulent']
```

```
#training-testing split
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,Y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=42)
```

```
x_train.shape,x_test.shape,Y_train.shape,y_test.shape
```

```
((14304, 6), (3576, 6), (14304,), (3576,))
```

```
#standardscaler for stability and convergence of algorithms
from sklearn.preprocessing import StandardScaler
scaler=StandardScaler()
x_train=scaler.fit_transform(x_train)
x_test=scaler.transform(x_test)
```

```
#k-NearestNeighboring algorithm1
from sklearn.neighbors import KNeighborsClassifier
```

```
k=2
knn_classifier=KNeighborsClassifier(n_neighbors=k)
```

```
knn_classifier.fit(x_train,Y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier(n_neighbors=2)
```

```
y_pred1=knn_classifier.predict(x_test)
```

```
y_pred1
```

```
array([0, 0, 0, ..., 0, 0, 0])
```

```
nb=KNeighborsClassifier(n_neighbors=2)
nb.fit(x_train,Y_train)
print("score",nb.score(x_test,y_test))
```

```
score 0.9611297539149888
```

```
from sklearn.metrics import confusion_matrix
print(confusion_matrix(y_test,y_pred1))
```

```
[[3371   24]
 [ 115   66]]
```

```
#support vector machine algorithm2
from sklearn.svm import SVC
model=SVC()
```

```
from sklearn.metrics import accuracy_score
model.fit(x_train,Y_train)
y_pred2=model.predict(x_test)
accuracy_SVM=accuracy_score(y_test,y_pred2)
```

```
accuracy_SVM
```

```
0.9507829977628636
```

```
#RandomForestClassifier algorithm3
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,Y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

```
y_pred3=rfc.predict(x_test)
accuracy_rfc=accuracy_score(y_test,y_pred3)
```

```
accuracy_rfc
```

```
0.9742729306487695
```

```
new_df=pd.DataFrame({'actual':y_test,'predicted':y_pred3})
```

```
new_df
```

	actual	predicted
4708	0	0
11079	0	0
12357	0	0
14511	0	0
16691	0	0
...	...	...
10855	0	0
9827	0	0
4903	0	0
6723	0	0
16899	0	0

3576 rows × 2 columns