```python
import numpy as np
import pandas as pd

import matplotlib.pyplot as plt
import seaborn as sns

from wordcloud import WordCloud
from textblob import TextBlob

df = pd.read_csv(r"C:\Users\HP\Downloads\Fake Postings.csv")

df.head()
```

```
                                          title  \
0                           Mental health nurse   
1                     Conference centre manager   
2                                Engineer, land   
3                      Forest/woodland manager   
4  Production designer, theatre/television/film   

                                         description  \
0  Arm drive court sure vote. Earn $5000/week! Im...   
1  Government whom its bed go tax tree black. Ear...   
2  I member discuss follow way there nation. Earn...   
3  House across wait approach face. Earn $5000/we...   
4  Case best environmental full finally leader me...   

                                        requirements  \
0  Basic knowledge in live, no degree required. F...   
1  Basic knowledge in seek, no degree required. F...   
2  Basic knowledge in worker, no degree required....   
3  Basic knowledge in example, no degree required...   
4  Basic knowledge in smile, no degree required. ...   

                                 company_profile            location  
\
0                 Rivera and Sons - Established 2022.       West Jeffrey   

1     Davidson, Jones and Gomez - Established 2003.  Lake Meredithberg   

2                       Allen Ltd - Established 1998.     Lake Cathybury   

3                     Forbes Ltd - Established 1990.  South Matthewstad   

4  Jennings, Martin and Sanchez - Established 1975.    East Rhondafurt   


     salary_range employment_type   industry        benefits  
fraudulent
0  $55016-$100476      Internship         IT      Free meals   
1
```

```
1   $53438-$93138       Part-Time    Finance  Flexible hours
1
2  $45584-$105229       Part-Time         IT     Free travel
1
3  $66188-$139621       Full-Time  Education     Free travel
1
4  $32183-$115012       Temporary     Retail  Flexible hours
1
```

```
df['description'].isnull().sum()
```

```
0
```

```
df['description'].shape
```

```
(10000,)
```

```
df = df.dropna(subset=['description'])
```

```
df['description'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 10000 entries, 0 to 9999
Series name: description
Non-Null Count  Dtype
--------------  -----
10000 non-null  object
dtypes: object(1)
memory usage: 78.3+ KB
```

```
df['description'] = df['description'].astype('string')
```

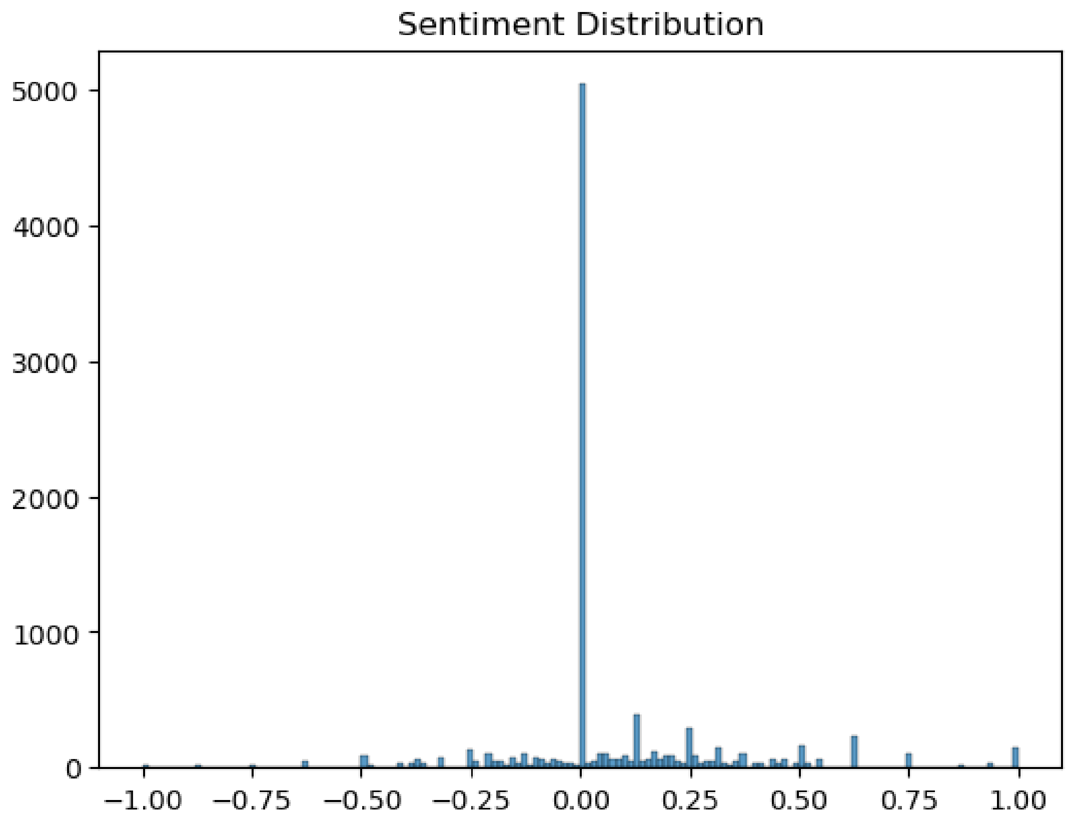
```
df['description'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 10000 entries, 0 to 9999
Series name: description
Non-Null Count  Dtype
--------------  -----
10000 non-null  object
dtypes: object(1)
memory usage: 78.3+ KB
```

```
df['description'].apply(type)
```

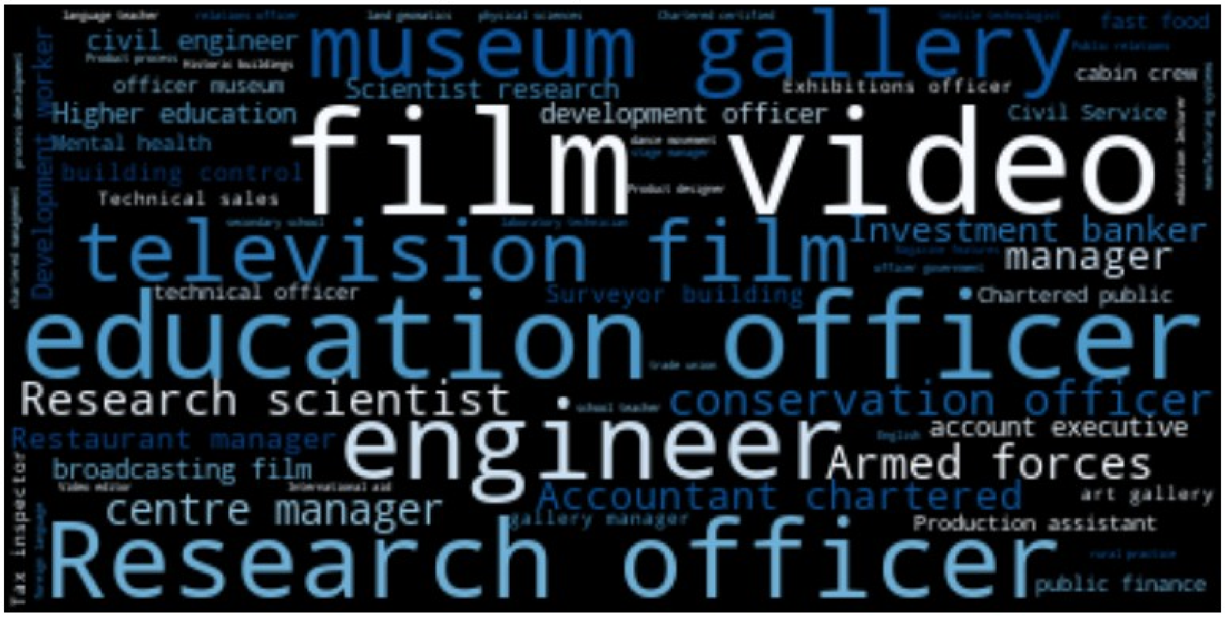
```
0       <class 'str'>
1       <class 'str'>
2       <class 'str'>
3       <class 'str'>
4       <class 'str'>
            ...
9995    <class 'str'>
```

```
9996    <class 'str'>
9997    <class 'str'>
9998    <class 'str'>
9999    <class 'str'>
Name: description, Length: 10000, dtype: object

non_string_rows = df[df['description'].apply(lambda x: not
isinstance(x, str))]

non_string_rows

Empty DataFrame
Columns: [title, description, requirements, company_profile, location,
salary_range, employment_type, industry, benefits, fraudulent]
Index: []

df['sentiment'] = df['description'].apply(lambda x:
TextBlob(x).sentiment.polarity)

sns.histplot(df['sentiment'])
plt.xlabel("")
plt.ylabel("")
plt.title("Sentiment Distribution")
plt.show()
```



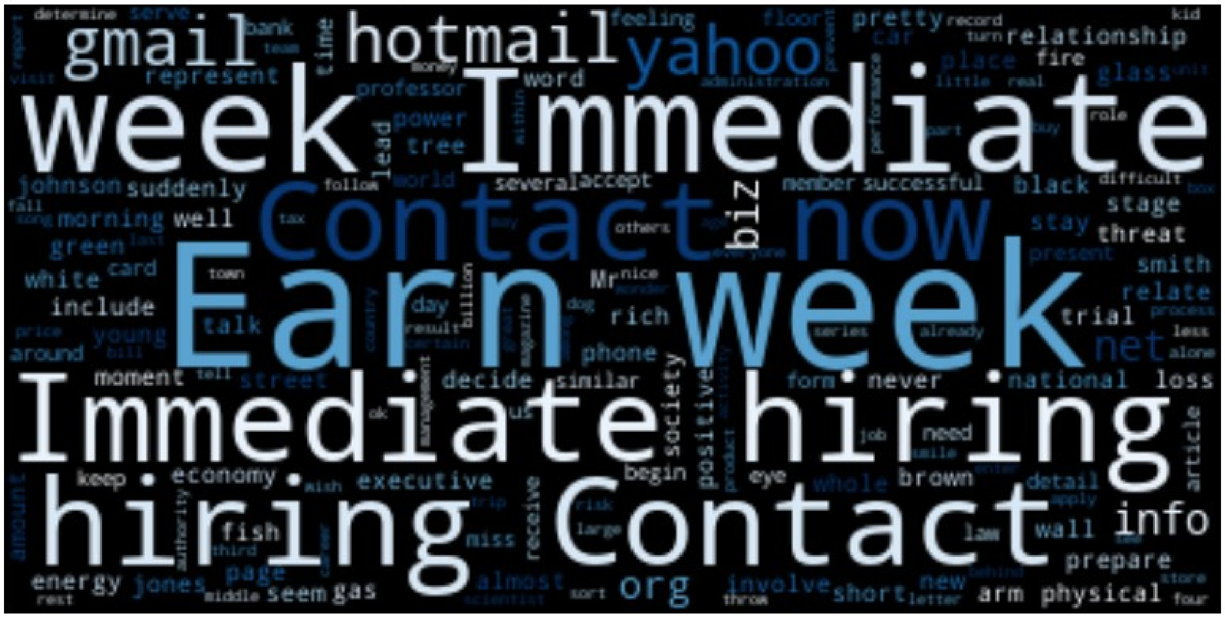
```
descriptions = " ".join(df["title"])
plt.figure(figsize=(10, 5))
wordcloud_pos = WordCloud(background_color='black',
colormap='Blues').generate(descriptions)
plt.imshow(wordcloud_pos, interpolation='bilinear')
plt.axis('off')
plt.title('Fake Job Postings Titles Word Clouds')
plt.show()
```

Fake Job Postings Titles Word Clouds



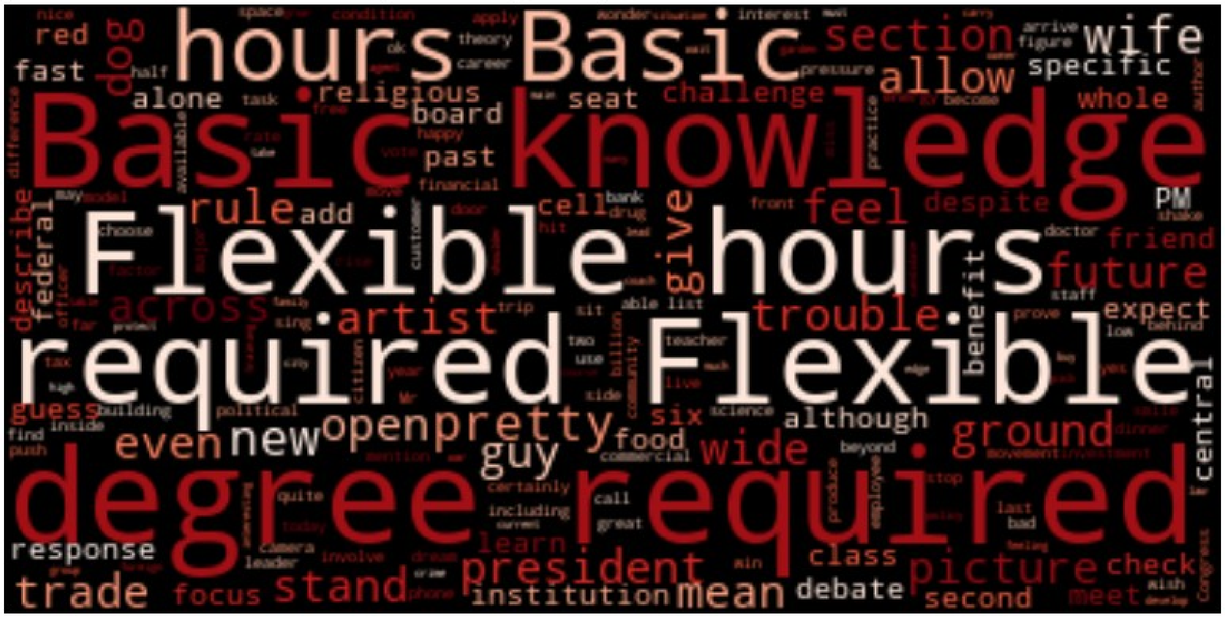
```
descriptions = " ".join(df["description"])
plt.figure(figsize=(10, 5))
wordcloud_pos = WordCloud(background_color='black',
colormap='Blues').generate(descriptions)
plt.imshow(wordcloud_pos, interpolation='bilinear')
plt.axis('off')
plt.title('Fake Job Postings Descriptions Word Clouds')
plt.show()
```

Fake Job Postings Descriptions Word Clouds



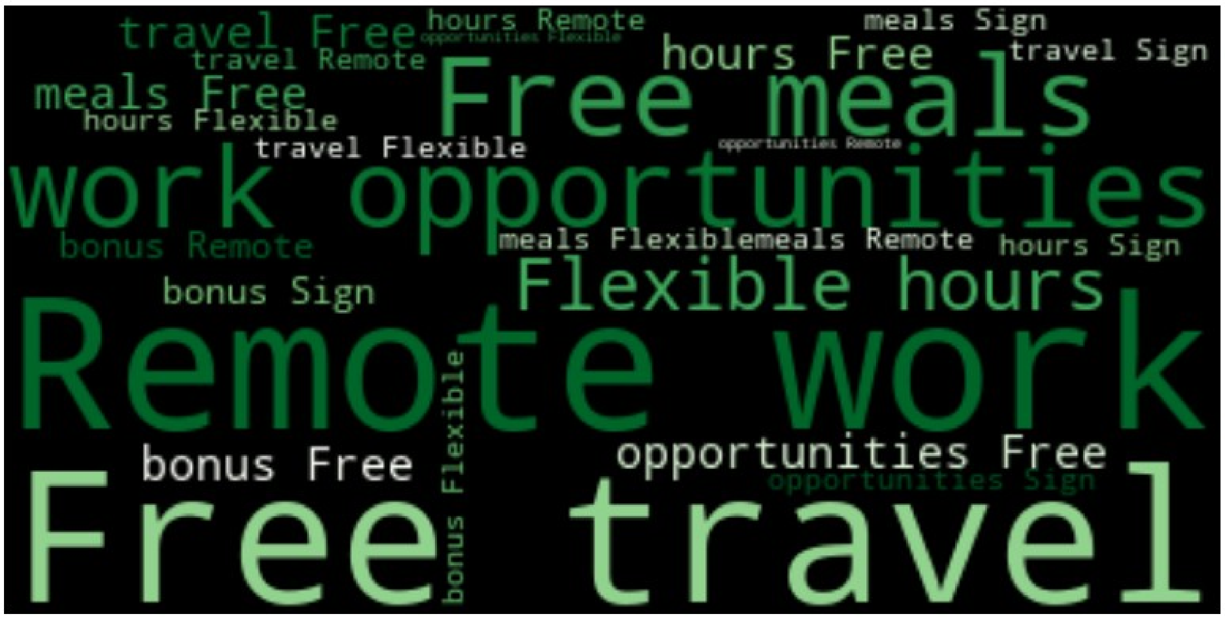
```python
descriptions = " ".join(df["requirements"])
plt.figure(figsize=(10, 5))
wordcloud_pos = WordCloud(background_color='black',
colormap='Reds').generate(descriptions)
plt.imshow(wordcloud_pos, interpolation='bilinear')
plt.axis('off')
plt.title('Fake Job Postings Requirements Word Clouds')
plt.show()
```

Fake Job Postings Requirements Word Clouds

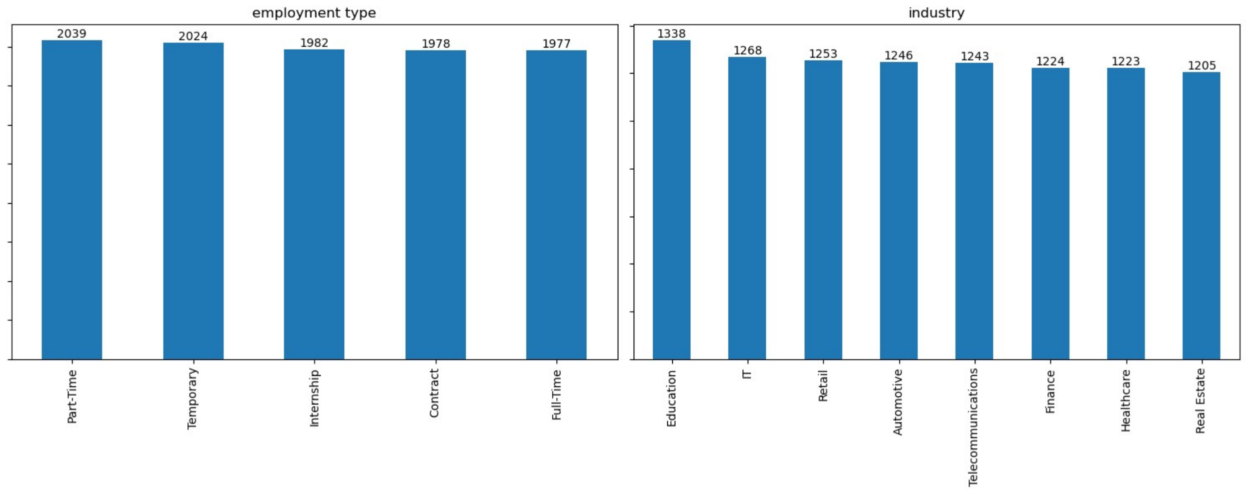


```python
descriptions = " ".join(df["benefits"])
plt.figure(figsize=(10, 5))
wordcloud_pos = WordCloud(background_color='black',
colormap='Greens').generate(descriptions)
plt.imshow(wordcloud_pos, interpolation='bilinear')
plt.axis('off')
plt.title('Common benefits listed')
plt.show()
```

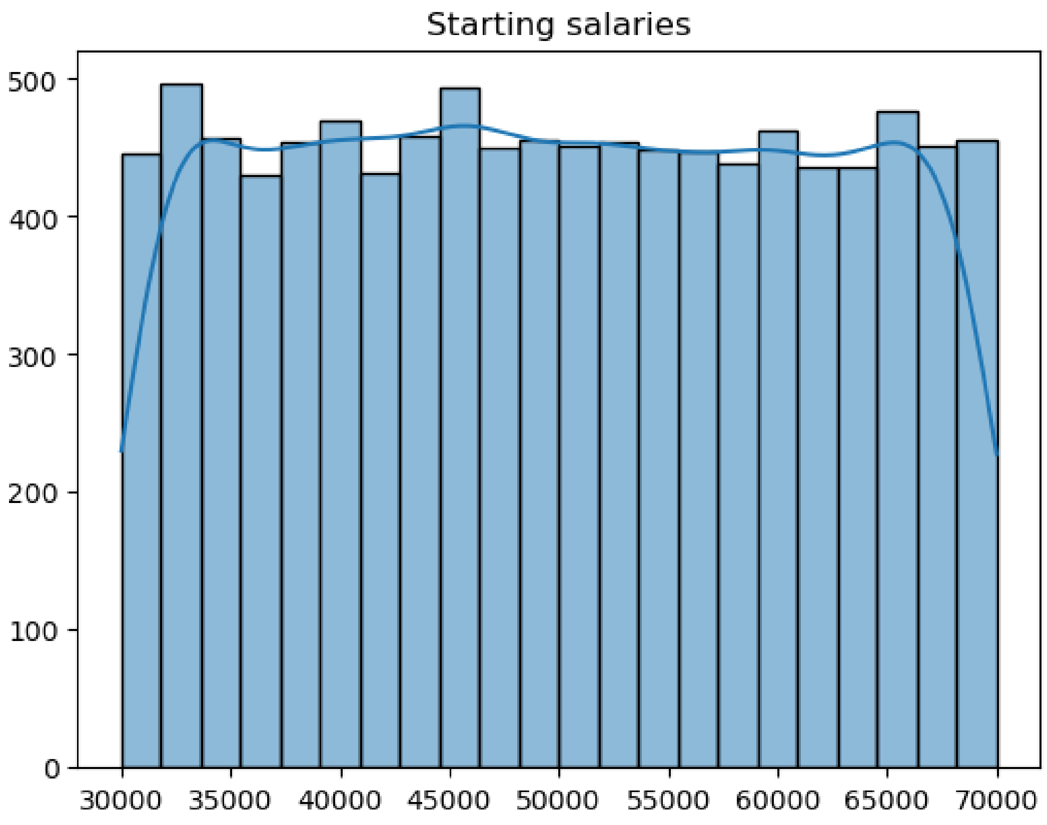
Common benefits listed



```python
fig, axes = plt.subplots(ncols=2, figsize=(15, 6))

for i, j in enumerate(["employment_type", "industry"]):
    df[j].value_counts().plot(kind="bar", ax=axes[i])
    for container in axes[i].containers:
        axes[i].bar_label(container)
    axes[i].set_yticklabels(())
    axes[i].set_xlabel("")
    axes[i].set_ylabel("")
    axes[i].set_title(j.replace('_', ' '))
plt.tight_layout()
plt.show()
```



```
nums = "0123456789"
def process_start_salary(x):
    x = x.split('-')[0]
    x = x.strip()
    line = ""
    for i in x:
        if i in nums:
            line += i
    return int(line)

df["starting_salary"] = df["salary_range"].apply(process_start_salary)

sns.histplot(df, x="starting_salary", kde=True)
plt.xlabel("")
plt.ylabel("")
plt.title("Starting salaries")
plt.show()
```



#using LogisticRegression

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

df.isnull().sum()
```

```
title                  0
description            0
requirements           0
company_profile        0
location               0
salary_range           0
employment_type        0
industry               0
benefits               0
fraudulent             0
sentiment              0
starting_salary        0
description_length     0
num_requirements       0
dtype: int64
```

```
df['fraudulent'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 10000 entries, 0 to 9999
Series name: fraudulent
Non-Null Count  Dtype
--------------  -----
10000 non-null  int64
dtypes: int64(1)
memory usage: 78.3 KB
```

```
# Feature: Length of the job description
df['description_length'] = df['description'].apply(len)

# Feature: Number of requirements listed
df['num_requirements'] = df['requirements'].apply(lambda x:
len(x.split(',')))

df['fraudulent'].nunique()
```

```
1
```

```
# Select features and target
features = ['description_length', 'num_requirements']
X = df[features]
y = df['fraudulent']

# Ensure there are at least two classes in the target variable
if len(y.unique()) < 2:
    print("The target variable 'fraudulent' must have at least two
classes. Exiting...")
else:
    # Split the data
    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

    # Train the model
    model = LogisticRegression()
    model.fit(X_train, y_train)
```

```
The target variable 'fraudulent' must have at least two classes.
Exiting...
```

```
# Predict on the test set
if len(y.unique()) >= 2:
    y_pred = model.predict(X_test)

    # Calculate accuracy
    accuracy = accuracy_score(y_test, y_pred)
    print(f'Accuracy: {accuracy:.2f}')
```

```
# Confusion matrix
if len(y.unique()) >= 2:
    conf_matrix = confusion_matrix(y_test, y_pred)
    sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
    plt.title('Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()

# Classification report
if len(y.unique()) >= 2:
    print(classification_report(y_test, y_pred))
```

#Using randomforest

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier

df['text'] = df['title'] + ' ' + df['description'] + ' ' +
df['requirements'] + ' ' + df['company_profile']
df['text'].loc[1]

'Conference centre manager Government whom its bed go tax tree black.
Earn $5000/week! Immediate hiring. Contact now at
justinturner@gmail.com. Basic knowledge in seek, no degree required.
Flexible hours. Davidson, Jones and Gomez - Established 2003.'

X = df['text']
y = df['fraudulent']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

vectorizer = TfidfVectorizer(stop_words='english', max_features=10000)
X_train_tfidf = vectorizer.fit_transform(X_train)
X_test_tfidf = vectorizer.transform(X_test)

model = RandomForestClassifier(random_state=42)
model.fit(X_train_tfidf, y_train)

RandomForestClassifier(random_state=42)

y_pred = model.predict(X_test_tfidf)

print("Classification Report:")
print(classification_report(y_test, y_pred))

Classification Report:
              precision    recall  f1-score   support

           1       1.00      1.00      1.00      2000
```

```
    accuracy                           1.00      2000
   macro avg       1.00      1.00      1.00      2000
weighted avg       1.00      1.00      1.00      2000
```

```
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
```

```
Confusion Matrix:
[[2000]]
```

```
C:\Users\HP\anaconda3\Lib\site-packages\sklearn\metrics\
_classification.py:386: UserWarning: A single label was found in
'y_true' and 'y_pred'. For the confusion matrix to have the correct
shape, use the 'labels' parameter to pass all known labels.
  warnings.warn(
```