Resume Selection

IMPORTING LIBRARIES

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud, STOPWORDS
import nltk
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize, sent_tokenize
import gensim
from gensim.utils import simple_preprocess
from gensim.utils import simple_preprocess
from gensim.parsing.preprocessing import STOPWORDS
from sklearn.metrics import classification_report, confusion_matrix
```

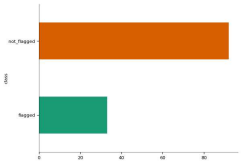
LOADING THE DATASET

```
resume_df = pd.read_csv('/content/resume_data.csv', encoding = 'latin-1')
resume_df
```

	resume_id	class	resume_text
0	resume_1	not_flagged	\rCustomer Service Supervisor/Tier - Isabella ...
1	resume_2	not_flagged	\rEngineer / Scientist - IBM Microelectronics ...
2	resume_3	not_flagged	\rLTS Software Engineer Computational Lithogra...
3	resume_4	not_flagged	TUTOR\rWilliston VT - Email me on Indeed: ind...
4	resume_5	flagged	\rIndependent Consultant - Self-employed\rBurl...
...	...	...	...
120	resume_121	not_flagged	\rBrattleboro VT - Email me on Indeed: indeed....
121	resume_122	not_flagged	\rResearch and Teaching Assistant - University...
122	resume_123	not_flagged	\rMedical Coder - Highly Skilled - Entry Level...
123	resume_124	flagged	\rWaterbury VT - Email me on Indeed: indeed.co...
124	resume_125	not_flagged	\rResearch and Development Scientist - Burling...

125 rows × 3 columns

Categorical distributions



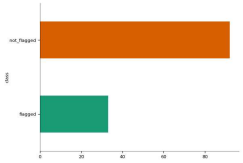
Next steps: Generate code with `resume_df` View recommended plots

```
resume_df = resume_df[['resume_text', 'class']]
resume_df
```

	resume_text	class
0	\rCustomer Service Supervisor/Tier - Isabella ...	not_flagged
1	\rEngineer / Scientist - IBM Microelectronics ...	not_flagged
2	\rLTS Software Engineer Computational Lithogra...	not_flagged
3	TUTOR\rWilliston VT - Email me on Indeed: ind...	not_flagged
4	\rIndependent Consultant - Self-employed\rBurl...	flagged
...	...	...
120	\rBrattleboro VT - Email me on Indeed: indeed....	not_flagged
121	\rResearch and Teaching Assistant - University...	not_flagged
122	\rMedical Coder - Highly Skilled - Entry Level...	not_flagged
123	\rWaterbury VT - Email me on Indeed: indeed.co...	flagged
124	\rResearch and Development Scientist - Burling...	not_flagged

125 rows × 2 columns

Categorical distributions



Next steps: Generate code with resume_df View recommended plots

PERFORMING EXPLORATORY DATA ANALYSIS:

```
resume_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 125 entries, 0 to 124
Data columns (total 2 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   resume_text  125 non-null    object
 1   class        125 non-null    object
dtypes: object(2)
memory usage: 2.1+ KB
```

```
resume_df['class'].value_counts()
```

```
class
not_flagged    92
flagged        33
Name: count, dtype: int64
```

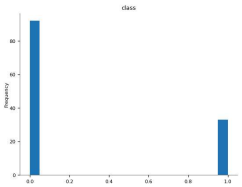
```
# HERE WE OBSERVE, WE HAVE NO NULL POINTS IN OUR DATASET
resume_df['class'] = resume_df['class'].apply(lambda x:1 if x == 'flagged' else 0)
resume_df
```

```
<ipython-input-8-a97fb2daf353>:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user
  resume_df['class'] = resume_df['class'].apply(lambda x:1 if x == 'flagged' else 0)
```

	resume_text	class
0	\rCustomer Service Supervisor/Tier - Isabella ...	0
1	\rEngineer / Scientist - IBM Microelectronics ...	0
2	\rLTS Software Engineer Computational Lithogra...	0
3	TUTOR\rWilliston VT - Email me on Indeed: ind...	0
4	\rIndependent Consultant - Self-employed\rBurl...	1
...	...	...
120	\rBrattleboro VT - Email me on Indeed: indeed....	0
121	\rResearch and Teaching Assistant - University...	0
122	\rMedical Coder - Highly Skilled - Entry Level...	0
123	\rWaterbury VT - Email me on Indeed: indeed.co...	1
124	\rResearch and Development Scientist - Burling...	0

125 rows × 2 columns

Distributions



Values

Next steps: Generate code with resume_df | View recommended plots

PERFORMING DATA CLEANING:

```
# PREMOVING UNNECESSARY WORDS FROM DATASET

resume_df['resume_text'] = resume_df['resume_text'].apply(lambda x: x .replace('\r', ''))

nltk.download('punkt')
nltk.download('stopwords')

from nltk.corpus import stopwords

stop_words = stopwords.words('english')

stop_words.extend(['from', 'subject', 'edu', 're', 'use', 'email', 'com'])

def preprocess(text):
    result = []
    for token in gensim.utils.simple_preprocess(text):
        if token not in gensim.parsing.preprocessing.STOPWORDS and len(token) > 2 and token not in stop_words:
            result.append(token)
    return ' '.join(result)
```

```
<ipython-input-9-b910c1193183>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-c
  resume_df['resume_text'] = resume_df['resume_text'].apply(lambda x: x .replace('\r', ''))
```

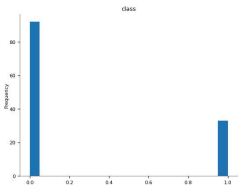
```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
resume_df
```

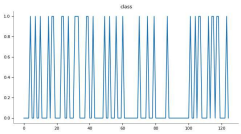
	resume_text	class
0	Customer Service Supervisor/Tier - Isabella Ca...	0
1	Engineer / Scientist - IBM Microelectronics Di...	0
2	LTS Software Engineer Computational Lithograph...	0
3	TUTORWilliston VT - Email me on Indeed: indee...	0
4	Independent Consultant - Self-employedBurlingt...	1
...	...	...
120	Brattleboro VT - Email me on Indeed: indeed.co...	0
121	Research and Teaching Assistant - University o...	0
122	Medical Coder - Highly Skilled - Entry LevelSu...	0
123	Waterbury VT - Email me on Indeed: indeed.com/...	1
124	Research and Development Scientist - Burlingto...	0

125 rows × 2 columns

Distributions



Values



Next steps: Generate code with `resume_df` View recommended plots

```
resume_df['cleaned'] = resume_df['resume_text'].apply(preprocess)
```

```
resume_df
```

	resume_text	class	cleaned
0	Customer Service Supervisor/Tier - Isabella Ca...	0	customer service supervisor tier isabella cata...
1	Engineer / Scientist - IBM Microelectronics Di...	0	engineer scientist ibm albert gregoritsch ecaw...
2	LTS Software Engineer Computational Lithograph...	0	lts software engineer computational lithograph...
3	TUTORWilliston VT - Email me on Indeed: indee...	0	tutorwilliston alec schwartz awork college bio...
4	Independent Consultant - Self-employedBurlingt...	1	independent consultant self alex reutter fefwo...
...	...	...	...
120	Brattleboro VT - Email me on Indeed: indeed.co...	0	brattleboro bcc skilled presenter trainer micr...
121	Research and Teaching Assistant - University o...	0	research teaching assistant university cdd gra...
122	Medical Coder - Highly Skilled - Entry LevelSu...	0	medical coder highly skilled entry levelsudbur...
123	Waterbury VT - Email me on Indeed: indeed.com/...	1	waterbury bec fcwilling relocate work employer...
124	Research and Development Scientist - Burlingto...	0	research development scientist burlington cda ...

125 rows × 3 columns

Distributions

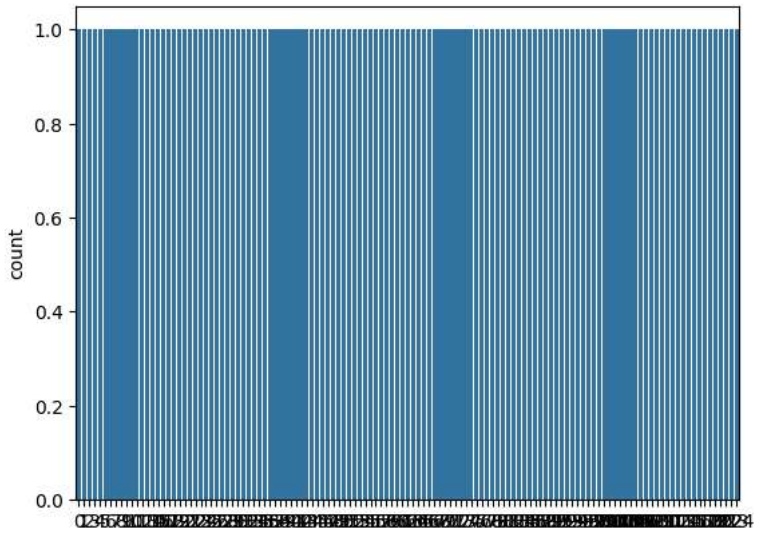
Next steps: Generate code with `resume_df` View recommended plots

```
resume_df['cleaned'][0]
```

```
'customer service supervisor tier isabella catalog companysouth burlington aecf work s
ervice supervisor tierisabella catalog company shelburne august present customer servi
ce visual set display website maintenance supervise customer service team popular cata
log company manage day day issues resolution customer upset ensure customer satisfacti
on troubleshoot order shipping issues lost transit order errors damages manage resolve
escalated customer calls ensure customer satisfaction assist customers order placing c
ross selling upselling catalog merchandise set display sample merchandise catalog libr
ary customer pick area facility website clean adding images type product information a
```

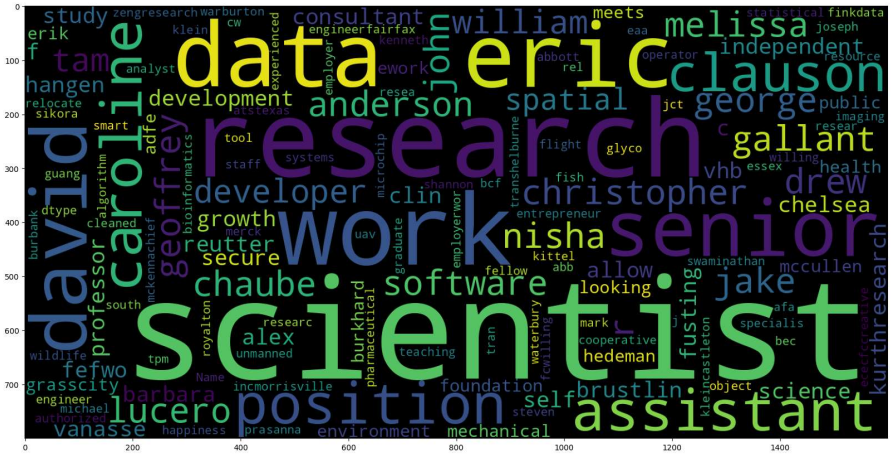
VISUALIZING CLEANED DATASETS

```
# PLOTTING COUNTS OF SAMPLE LABELLED AS 1 AND 0
sns.countplot(resume_df['class'], label = 'Count Plot')
plt.show()
```



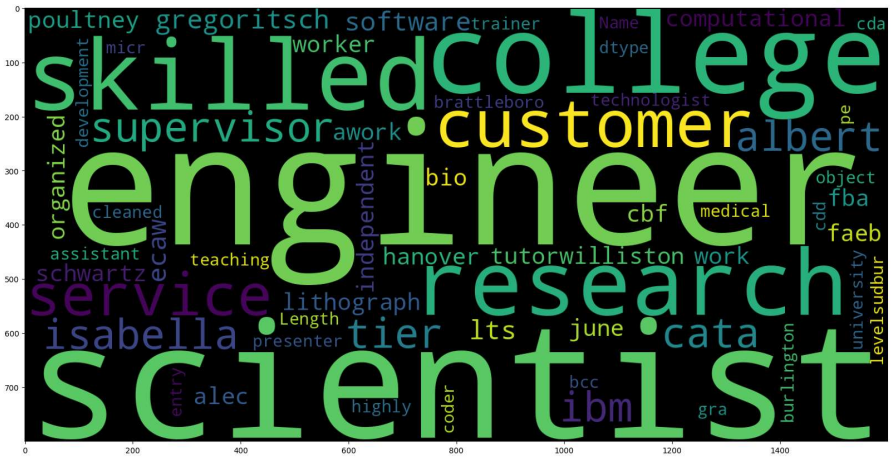

```
# PLOTTING THE WORDCLOUD:

# 1) FOR CLASS 1:

%matplotlib inline

plt.figure(figsize = (20, 20))

wc = WordCloud(max_words = 2000, width = 1600, height = 800, stopwords = stop_words).generate(str(resume_df[resume_df['class']==1].cleaned))

plt.imshow(wc)
```

```
<matplotlib.image.AxesImage at 0x7b6598129510>
```



```
#1) FOR CLASS 0:

%matplotlib inline

plt.figure(figsize = (20, 20))

wc = WordCloud(max_words = 2000, width = 1600, height = 800, stopwords = stop_words).generate(str(resume_df[resume_df['class']==0].cleaned)

plt.imshow(wc)
```

```
<matplotlib.image.AxesImage at 0x7b659825d4e0>
```



PREPARING THE DATA BY APPLYING COUNT VECTORIZATION

```
# Import CountVectorizer from sklearn
from sklearn.feature_extraction.text import CountVectorizer

# Initialize the vectorizer
vectorizer = CountVectorizer()

# Fit and transform the cleaned column
countvectorizer = vectorizer.fit_transform(resume_df['cleaned'])

# Print the feature names using the new method
print(vectorizer.get_feature_names_out())
```

```
['aaalac' 'aabb' 'aac' ... 'ãæcomputer' 'ètravel' 'ô_torrent']
```

```
# PROCESSED DATA:
print(countvectorizer.toarray())
```

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
```

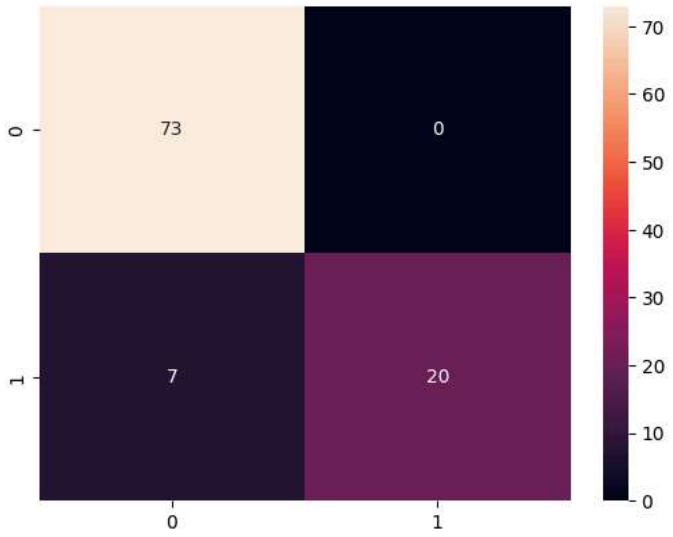
TRAINING A NAIVE BAYES CLASSIFER

```
X = countvectorizer
y = resume_df['class']

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)


from sklearn.naive_bayes import MultinomialNB

Bayes_clf = MultinomialNB(alpha = 3)
Bayes_clf.fit(X_train, y_train)  ## Training the model
```

```
MultinomialNB
MultinomialNB(alpha=3)
```

ASSESING THE TRAINED MODEL

```
%matplotlib inline

# PLOTTING CONFUSION MATRIX:

# 1) FOR TRAINING DATA

y_pred_train = Bayes_clf.predict(X_train)

cm = confusion_matrix(y_train, y_pred_train)

sns.heatmap(cm, annot=True)

plt.show()
```




```
%matplotlib inline

# WE CAN SEE OUR MODEL PERFORMED REALLY WELL ON TRAINING DATA: IT CLASSFIED ALL OF THE POINTS CORRECTLY

# 2) FOR TEST DATA:

y_pred_test = Bayes_clf.predict(X_test)

cm = confusion_matrix(y_test, y_pred_test)

sns.heatmap(cm, annot=True)

plt.show()
```