```
from google.colab import files

uploaded = files.upload()
```

Choose Files gsearch_jobs.csv

- **gsearch_jobs.csv**(text/csv) - 172727065 bytes, last modified: 11/26/2023 - 100% done

```
Saving gsearch_jobs.csv to gsearch_jobs.csv
```

```
import pandas as pd

# Assuming the file name is 'gsearch_jobs.csv'
file_name = 'gsearch_jobs.csv'

# Load data into Pandas DataFrame
df = pd.read_csv(file_name)

# Display the first few rows of the DataFrame
df.head()
```

	Unnamed: 0	index	title	company_name	location	via	description	extensions	
0	0	0	Data Analyst	Meta	Anywhere	via LinkedIn	In the intersection of compliance and analytic...	['15 hours ago', '101K–143K a year', 'Work fro...	eyJqb2JfdGl0bGUiO
1	1	1	Data Analyst	ATC	United States	via LinkedIn	Job Title: Entry Level Business Analyst / Prod...	['12 hours ago', 'Full-time', 'Health insurance']	eyJqb2JfdGl0bGUiO
2	2	2	Aeronautical Data Analyst	Garmin International, Inc.	Olathe, KS	via Indeed	Overview:\n\nWe are seeking a full-time...\nAe...	['18 hours ago', 'Full-time']	eyJqb2JfdGl0bGUiOiJE
3	3	3	Data Analyst - Consumer Goods - Contract to Hire	Upwork	Anywhere	via Upwork	Enthusiastic Data Analyst for processing sales...	['12 hours ago', '15–25 an hour', 'Work from h...	eyJqb2JfdGl0bGUiOiJE
4	4	4	Data Analyst | Workforce Management	Krispy Kreme	United States	via LinkedIn	Overview of Position\n\nThis position will be ...	['7 hours ago', '90K–110K a year', 'Contractor']	eyJqb2JfdGl0bGUiOiJ

5 rows × 27 columns

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36051 entries, 0 to 36050
Data columns (total 27 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Unnamed: 0          36051 non-null  int64
 1   index               36051 non-null  int64
 2   title               36051 non-null  object
 3   company_name        36051 non-null  object
 4   location            36025 non-null  object
 5   via                 36042 non-null  object
 6   description          36051 non-null  object
 7   extensions           36051 non-null  object
 8   job_id               36051 non-null  object
 9   thumbnail            20443 non-null  object
 10  posted_at            36051 non-null  object
 11  schedule_type        35874 non-null  object
 12  work_from_home       16025 non-null  object
 13  salary               6309 non-null   object
 14  search_term          36051 non-null  object
 15  date_time            36051 non-null  object
 16  search_location      36051 non-null  object
 17  commute_time         0 non-null      float64
 18  salary_pay           6309 non-null   object
 19  salary_rate          6309 non-null   object
 20  salary_avg           6309 non-null   float64
 21  salary_min           5952 non-null   float64
 22  salary_max           5952 non-null   float64
 23  salary_hourly        4167 non-null   float64
 24  salary_yearly        2129 non-null   float64
 25  salary_standardized  6309 non-null   float64
 26  description_tokens   36051 non-null  object
dtypes: float64(7), int64(2), object(18)
memory usage: 7.4+ MB
```

```
df.describe()
```

	Unnamed: 0	index	commute_time	salary_avg	salary_min	salary_max	salary_hourly	salar
count	36051.000000	36051.000000	0.0	6309.000000	5952.000000	5952.000000	4167.000000	212
mean	18025.000000	1117.465729	NaN	34479.365127	29067.529400	40728.531630	42.456192	10205
std	10407.171614	696.771915	NaN	51428.069516	43427.331132	60779.792689	22.815874	3073
min	0.000000	0.000000	NaN	7.250000	8.000000	10.000000	7.250000	2928
25%	9012.500000	530.000000	NaN	30.500000	18.330000	45.000000	25.000000	8379
50%	18025.000000	1080.000000	NaN	57.500000	40.000000	75.000000	35.000000	965
75%	27037.500000	1643.000000	NaN	85000.000000	74752.500000	100000.000000	57.500000	1140
max	36050.000000	3275.000000	NaN	288000.000000	230000.000000	346000.000000	300.000000	2880

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

```
Unnamed: 0                0
index                     0
title                     0
company_name              0
location                 26
via                       9
description               0
extensions                0
job_id                    0
thumbnail             15608
posted_at                 0
schedule_type           177
work_from_home        20026
salary                29742
search_term               0
date_time                 0
search_location           0
commute_time          36051
salary_pay            29742
salary_rate           29742
salary_avg            29742
salary_min            30099
```

```
salary_max              30099
salary_hourly           31884
salary_yearly           33922
salary_standardized     29742
description_tokens          0
dtype: int64
```

```
df['company_name'].unique()
```

```
array(['Meta', 'ATC', 'Garmin International, Inc.', ...,
       'Applied Memetics L.L.C', 'Global Enterprise Partners',
       'Techdash Telecom'], dtype=object)
```

```
df['title'].unique()
```

```
array(['Data Analyst', 'Aeronautical Data Analyst',
       'Data Analyst - Consumer Goods - Contract to Hire', ...,
       'Data Analyst - Business Intelligence',
       'COOP - Senior Data Analyst', 'Lead FP&A Analyst- Remote, US'],
      dtype=object)
```

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
nltk.download('stopwords')
nltk.download('punkt')
```

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

```
# Tokenize the job descriptions
df['description_tokens'] = df['description'].apply(lambda x: word_tokenize(str(x).lower()))

# Remove stop words and non-alphabetic tokens
stop_words = set(stopwords.words('english'))
df['description_tokens'] = df['description_tokens'].apply(lambda x: [word for word in x if word.isalpha() and word r
```

```
# List of known skills
known_skills = ["python", "java", "data analysis", "sql", "excel", "machine learning", "statistics", "r", "snowflake
```

```
# Display the first few rows to check the result
df[['description', 'description_tokens']].head()
```

	description	description_tokens
0	In the intersection of compliance and analytic...	[intersection, compliance, analytics, seeking,...
1	Job Title: Entry Level Business Analyst / Prod...	[job, title, entry, level, business, analyst, ...
2	Overview:\n\nWe are seeking a full-time...\nAe...	[overview, seeking, aeronautical, data, analys...
3	Enthusiastic Data Analyst for processing sales...	[enthusiastic, data, analyst, processing, sale...
4	Overview of Position\n\nThis position will be ...	[overview, position, position, primary, person...

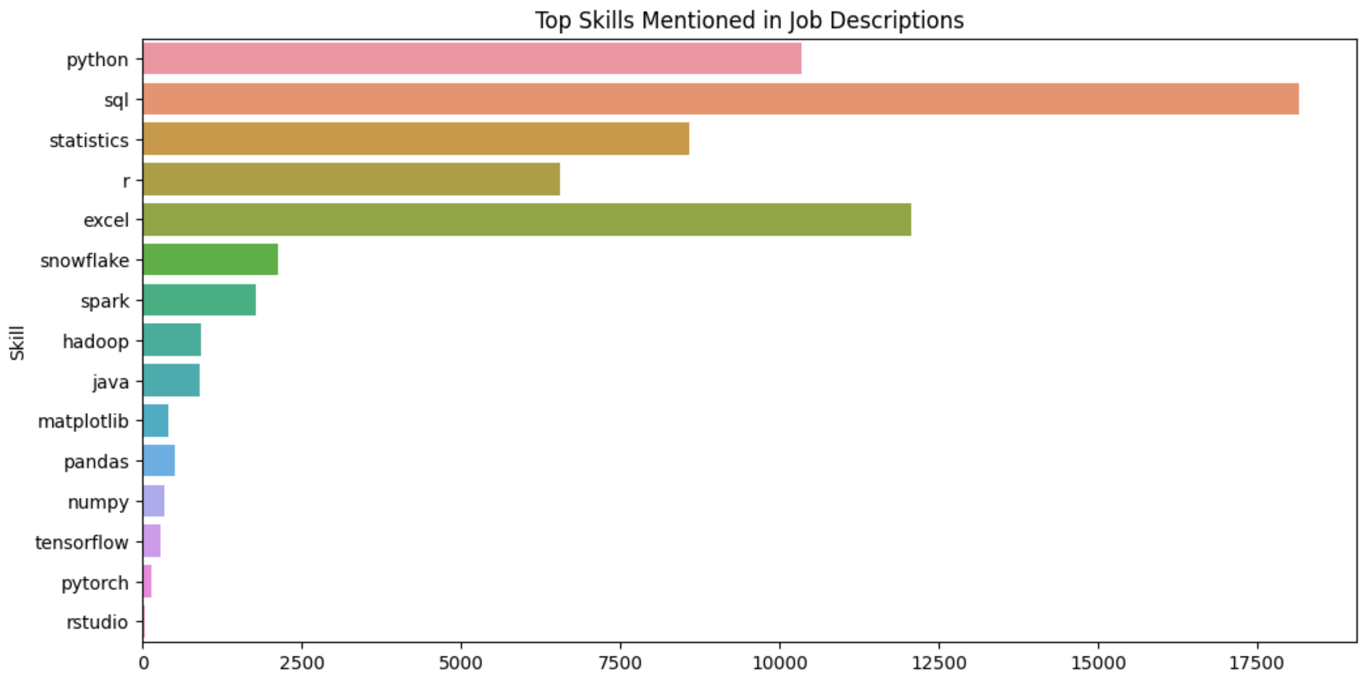
```
# Create a new column in the DataFrame to store the extracted skills
df['extracted_skills'] = ""
```

```
# Loop through each row and tokenize the words to check for known skills
for index, row in df.iterrows():
    description_tokens = row['description_tokens']

    if description_tokens:
        extracted_skills = [skill for skill in known_skills if skill in description_tokens]
        df.at[index, 'extracted_skills'] = extracted_skills
```

```
from collections import Counter

# Combine all extracted skills into a single list
all_skills = [skill for skills_list in df['extracted_skills'] for skill in skills_list]

# Count the frequency of each skill
skill_counts = Counter(all_skills)

# Display the top 15 most common skills
top_skills = skill_counts.most_common(15)
print(top_skills)
```

```
[('sql', 18156), ('excel', 12061), ('python', 10350), ('statistics', 8590), ('r', 6550), ('snowflake', 2137), ('
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# Convert skill counts to a DataFrame for easier plotting
skill_counts_df = pd.DataFrame(skill_counts.items(), columns=['Skill', 'Count'])

# Plot the top 15 skills
plt.figure(figsize=(12, 6))
sns.barplot(x='Count', y='Skill', data=skill_counts_df.head(15))
plt.title('Top Skills Mentioned in Job Descriptions')
plt.show()
```



```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.feature_extraction.text import CountVectorizer
```

```
# Create a subset (10% of the data)
subset_df = df.sample(frac=0.1, random_state=42)


# Check the shape of the subset to ensure it's a reasonable size
print("Subset shape:", subset_df.shape)
```

```
    Subset shape: (3605, 28)
```

```
X = subset_df['description']
y = subset_df['title']



# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)



#Use CountVectorizer to convert 'skills' into numerical features
from sklearn.feature_extraction.text import CountVectorizer


vectorizer = CountVectorizer()
X_train_vec = vectorizer.fit_transform(X_train)
X_test_vec = vectorizer.transform(X_test)


# Train a RandomForestClassifier
clf = RandomForestClassifier(random_state=42)
clf.fit(X_train_vec, y_train)
```

```
     ▾        RandomForestClassifier
     RandomForestClassifier(random_state=42)
```

```
# Make predictions on the test set
y_pred = clf.predict(X_test_vec)



# Display actual labels and predicted labels side by side
predictions_comparison = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred})
# Display the first 10 rows of the predictions_comparison DataFrame
N = 10
print(predictions_comparison.head(N))
```

```
                                                  Actual  \
     35860                                LEAD DATA ANALYST
     29610                                   Data Scientist
     14041      Senior Data Analyst (Transportation Strategy)
     15731                 Data Engineer, Planning & Analysis
     17319          Systems Test Engineer - Test Data Analyst
     34649  (USA) Analyst II, Merchandising Business Analy...
     27998                           Data Analyst with Claims
     21553                                     Data Analyst
     33150                             Healthcare Analyst III
     30552  Senior Financial Data Analyst (Remote) - Medic...

                                               Predicted
     35860                                LEAD DATA ANALYST
     29610                              Senior Data Analyst
```

```
14041                                  Senior Data Analyst
15731                Data Engineer, Planning & Analysis
17319                                         Data Analyst
34649  (USA) Analyst II, Merchandising Business Analy...
27998                                         Data Analyst
21553                 Data Analyst at COVID TESTING LLC
33150                                Analyst III - REMOTE
30552  Senior Financial Data Analyst (Remote) - Medic...
```

```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

```
Accuracy: 0.30235783633841884
```