WebScraping

May 9, 2024

1 Companies and Jobs

1.1 Step 1: Importing Required Libraries

[2]:
```
pip install PyGithub
```

```
Collecting PyGithub
  Downloading PyGithub-2.3.0-py3-none-any.whl (354 kB)
     |                  | 354 kB 38.6 MB/s
Collecting Deprecated
  Downloading Deprecated-1.2.14-py2.py3-none-any.whl (9.6 kB)
Collecting typing-extensions>=4.0.0
  Downloading typing_extensions-4.7.1-py3-none-any.whl (33 kB)
Collecting pyjwt[crypto]>=2.4.0
  Downloading PyJWT-2.8.0-py3-none-any.whl (22 kB)
Requirement already satisfied: requests>=2.14.0 in
/opt/conda/lib/python3.7/site-packages (from PyGithub) (2.23.0)
Collecting urllib3>=1.26.0
  Downloading urllib3-2.0.7-py3-none-any.whl (124 kB)
     |                  | 124 kB 113.3 MB/s
Collecting pynacl>=1.4.0
  Downloading PyNaCl-1.5.0-cp36-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.
manylinux_2_24_x86_64.whl (856 kB)
     |                  | 856 kB 96.3 MB/s
Collecting cryptography>=3.4.0
  Downloading
cryptography-42.0.7-cp37-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl
(3.8 MB)
     |                  | 3.8 MB 80.0 MB/s
Requirement already satisfied: cffi>=1.4.1 in
/opt/conda/lib/python3.7/site-packages (from pynacl>=1.4.0->PyGithub) (1.14.0)
Requirement already satisfied: chardet<4,>=3.0.2 in
/opt/conda/lib/python3.7/site-packages (from requests>=2.14.0->PyGithub) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /opt/conda/lib/python3.7/site-
packages (from requests>=2.14.0->PyGithub) (2.9)
Requirement already satisfied: certifi>=2017.4.17 in
/opt/conda/lib/python3.7/site-packages (from requests>=2.14.0->PyGithub)
```

```
(2020.4.5.1)
Collecting requests>=2.14.0
  Downloading requests-2.31.0-py3-none-any.whl (62 kB)
     |                 | 62 kB 7.3 MB/s
Collecting charset-normalizer<4,>=2
  Downloading charset_normalizer-3.3.2-cp37-cp37m-manylinux_2_17_x86_64.manylinu
x2014_x86_64.whl (136 kB)
     |                 | 136 kB 105.7 MB/s
Collecting wrapt<2,>=1.10
  Downloading wrapt-1.16.0-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.man
ylinux_2_17_x86_64.manylinux2014_x86_64.whl (77 kB)
     |                 | 77 kB 33.0 MB/s
Requirement already satisfied: pycparser in /opt/conda/lib/python3.7/site-
packages (from cffi>=1.4.1->pynacl>=1.4.0->PyGithub) (2.20)
Installing collected packages: typing-extensions, wrapt, urllib3, pyjwt,
cryptography, charset-normalizer, requests, pynacl, Deprecated, PyGithub
  Attempting uninstall: typing-extensions
    Found existing installation: typing-extensions 3.7.4.2
    Uninstalling typing-extensions-3.7.4.2:
      Successfully uninstalled typing-extensions-3.7.4.2
  Attempting uninstall: urllib3
    Found existing installation: urllib3 1.25.9
    Uninstalling urllib3-1.25.9:
      Successfully uninstalled urllib3-1.25.9
  Attempting uninstall: pyjwt
    Found existing installation: PyJWT 1.7.1
    Uninstalling PyJWT-1.7.1:
      Successfully uninstalled PyJWT-1.7.1
  Attempting uninstall: cryptography
    Found existing installation: cryptography 2.9.2
    Uninstalling cryptography-2.9.2:
      Successfully uninstalled cryptography-2.9.2
  Attempting uninstall: requests
    Found existing installation: requests 2.23.0
    Uninstalling requests-2.23.0:
      Successfully uninstalled requests-2.23.0
Successfully installed Deprecated-1.2.14 PyGithub-2.3.0 charset-normalizer-3.3.2
cryptography-42.0.7 pyjwt-2.8.0 pynacl-1.5.0 requests-2.31.0 typing-
extensions-4.7.1 urllib3-2.0.7 wrapt-1.16.0
WARNING: You are using pip version 21.3.1; however, version 24.0 is

available.

You should consider upgrading via the '/opt/conda/bin/python -m pip install

--upgrade pip' command.
Note: you may need to restart the kernel to use updated packages.
```

[3]:
```
from github import Github
import pandas as pd
import io  # Import io module for StringIO

# GitHub authentication
access_token = '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'  # Replace␣
↪'your-access-token' with your personal access token
github = Github(access_token)

# Repository name and file paths
repository_name = 'sombir1/Projext-X'  # Your repository name
file_paths = [
    'final_ai_jobs.csv',
]

# Initialize an empty DataFrame to store the merged data
merged_df = pd.DataFrame()

# Loop through each CSV file path
for file_path in file_paths:
    # Get file content from GitHub
    repo = github.get_repo(repository_name)
    file_content = repo.get_contents(file_path)

    # Decode the content
    decoded_content = file_content.decoded_content.decode('utf-8')

    # Read CSV file into DataFrame
    df = pd.read_csv(io.StringIO(decoded_content))

    # Merge DataFrame with previous ones
    merged_df = pd.concat([merged_df, df], ignore_index=True)

# Write the merged DataFrame to a CSV file
merged_df.to_csv('merged.csv', index=False)
```

[4]:
```
import pandas as pd

# Read the merged CSV file into a DataFrame
merged_df = pd.read_csv('merged.csv')

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

[4]:

	job_link \
0	https://www.indeed.com/rc/clk?jk=3a9047b41601a…
1	https://www.indeed.com/rc/clk?jk=b1b60d57adfbb…

```
2  https://www.indeed.com/rc/clk?jk=a0f33cc71b175…
3  https://www.indeed.com/rc/clk?jk=ac4bbf72deed7…
4  https://www.indeed.com/rc/clk?jk=7f8e658852335…

                                          job_title  company_name  \
0        UX Researcher, Qualitative - Generative AI          Meta
1  Fundamental Language Research Scientist - Gene…          Meta
2           Data Scientist, Product - Generative AI  Facebook App
3                  Product Designer - Generative AI          Meta
4  Director, Product Technical Program Management…          Meta

  company_location  salary       job_type rating  \
0     New York, NY  184500      Full-time    4.1
1     New York, NY  173500  Not available    4.1
2     New York, NY  164000  Not available    4.1
3     New York, NY  158000  Not available    4.1
4     New York, NY  265000  Not available    4.1

                                    job_description   searched_job  \
0  Work closely with product and business teams t…  Generative AI
1  Direct experience in responsible ,, , ,, and L…  Generative AI
2  You will collaborate on a wide array of produc…  Generative AI
3  We utilize a full range of product design skil…  Generative AI
4  We are uniquely positioned to adopt an end-to-…  Generative AI

  searched_location
0          New York
1          New York
2          New York
3          New York
4          New York
```

[5]:
```python
import pandas as pd

# Read the merged CSV file into a DataFrame
merged_df = pd.read_csv('merged.csv')

# Display the column headings
print(merged_df.columns)
```

```
Index(['job_link', 'job_title', 'company_name', 'company_location', 'salary',
       'job_type', 'rating', 'job_description', 'searched_job',
       'searched_location'],
      dtype='object')
```

[6]:
```python
import pandas as pd
```

```
# Read the merged CSV file into a DataFrame
merged_df = pd.read_csv('merged.csv')

# Display the column headings
print(merged_df.columns)
```

```
Index(['job_link', 'job_title', 'company_name', 'company_location', 'salary',
       'job_type', 'rating', 'job_description', 'searched_job',
       'searched_location'],
      dtype='object')
```

[7]:
```
common_job_titles = merged_df['job_title'].value_counts().head(10)
print("Top 10 most common job titles:\n", common_job_titles)
```

```
Top 10 most common job titles:
 Senior Research Scientist, Generative AI
3
Digital Contact Solutions Manager
3
Software Engineer, Product - Generative AI
2
Business Intelligence Manager, Finance
2
Engineering Manager
2
Product Designer - Generative AI
2
UX Researcher, Qualitative - Generative AI
2
Senior Director of Public Relations and Corporate Communications (USA REMOTE)
2
Fundamental Language Research Scientist - Generative AI
2
Product Technical Program Manager - Generative AI
2
Name: job_title, dtype: int64
```

[8]:
```
print(merged_df['rating'].dtype)
```

```
object
```

[9]:
```
merged_df['rating'] = pd.to_numeric(merged_df['rating'], errors='coerce')
```

[10]:
```
highest_rated_companies = merged_df.groupby('company_name')['rating'].mean().
 ↪nlargest(10)
print("\nTop 10 companies with highest-rated job postings:\n",␣
 ↪highest_rated_companies)
```

```
Top 10 companies with highest-rated job postings:
 company_name
Revvity                          5.0
Quizlet                          4.8
Grammarly                        4.6
DiDi Labs                        4.5
Kasisto                          4.5
SecurityScorecard                4.4
Tencent                          4.4
Adobe                            4.3
Argonne National Laboratory      4.3
Discord                          4.3
Name: rating, dtype: float64
```

[11]:
```
print(merged_df['salary'].dtype)
```

```
object
```

[12]:
```
merged_df['salary'] = pd.to_numeric(merged_df['salary'], errors='coerce')
```

[13]:
```
average_salaries = merged_df.groupby('job_title')['salary'].mean().nlargest(10)
print("\nAverage salaries for top 10 job titles:\n", average_salaries)
```

```
Average salaries for top 10 job titles:
 job_title
Technical Director, Machine Learning (Individual Contributor)     412050.0
Technical Product Manager                                         410000.0
Vice President of Product and Software Security                   292950.0
Executive Director, Corporate Affairs- Operations                 279997.0
Director of Product                                               266500.0
Director, Product Technical Program Management - Generative AI    265000.0
Sr. Director Security Engagement                                  265000.0
Senior Director of Product                                        258400.0
Machine Learning Engineer (Mobile Team)                           256500.0
Senior Research Engineer                                          254750.0
Name: salary, dtype: float64
```

[14]:
```
job_type_distribution = merged_df['job_type'].value_counts()
print("\nDistribution of job types:\n", job_type_distribution)
```

```
Distribution of job types:
 Full-time                131
Not available             118
Part-time                   2
Contract                    2
```

```
Choose your own hours      2
Tenure track               1
Overtime                   1
Extended hours             1
Internship                 1
Name: job_type, dtype: int64
```

[15]:
```
print(merged_df.columns)
```

```
Index(['job_link', 'job_title', 'company_name', 'company_location', 'salary',
       'job_type', 'rating', 'job_description', 'searched_job',
       'searched_location'],
      dtype='object')
```

[ ]:
```
```

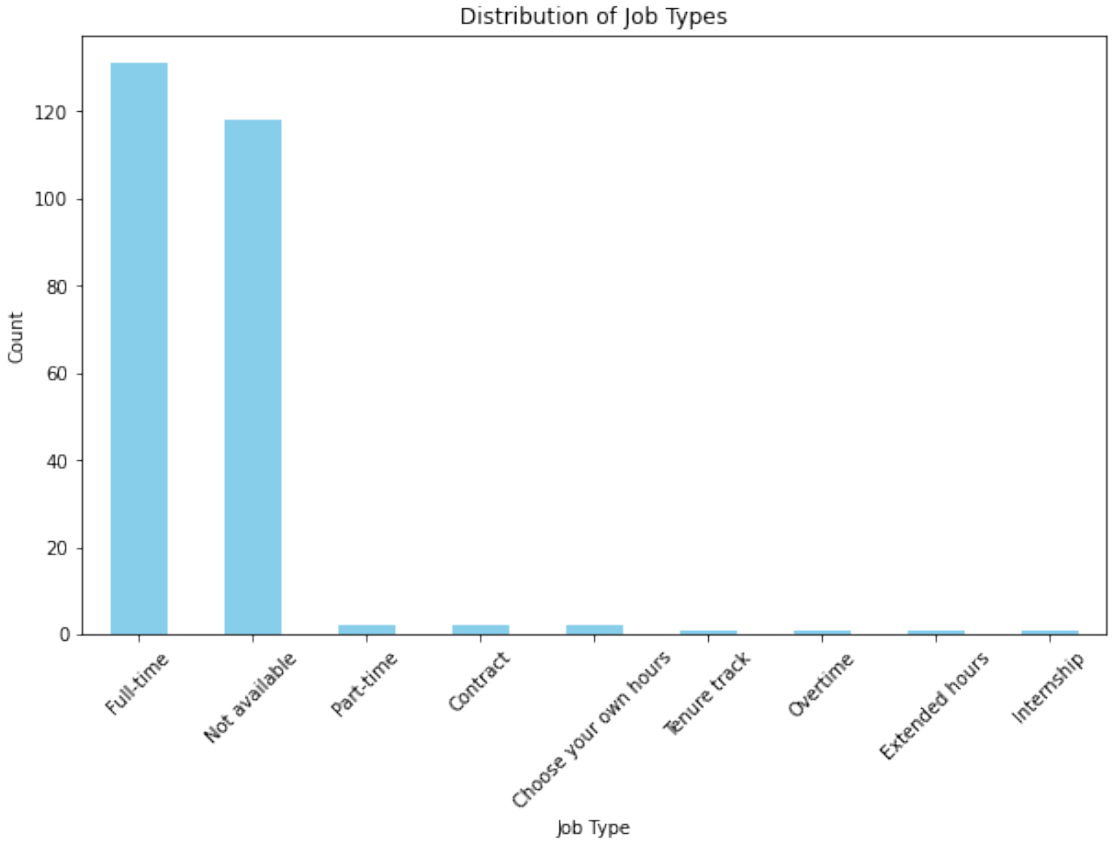
[16]:
```
correlation = merged_df[['rating', 'salary']].dropna().corr()
print("\nCorrelation between company ratings and offered salaries:\n",␣
 ↪correlation)
```

```
Correlation between company ratings and offered salaries:
           rating    salary
rating  1.000000  0.348204
salary  0.348204  1.000000
```

[17]:
```
frequent_searched_job_titles = merged_df['searched_job'].value_counts().head(10)
print("\nTop 10 most frequently searched job titles:\n",␣
 ↪frequent_searched_job_titles)

frequent_searched_locations = merged_df['searched_location'].value_counts().
 ↪head(10)
print("\nTop 10 most frequently searched locations:\n",␣
 ↪frequent_searched_locations)
```

```
Top 10 most frequently searched job titles:
 Generative AI    259
Name: searched_job, dtype: int64

Top 10 most frequently searched locations:
 California       139
New York           57
San Francisco      40
Chicago            14
Los Angeles         9
Name: searched_location, dtype: int64
```

[18]:
```
import matplotlib.pyplot as plt

# Plot the distribution of job types
plt.figure(figsize=(10, 6))
merged_df['job_type'].value_counts().plot(kind='bar', color='skyblue')
plt.title('Distribution of Job Types')
plt.xlabel('Job Type')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



[19]:
```
# Extract job descriptions from the 'job_description' column
job_descriptions = merged_df['job_description']

# Display the first few job descriptions
for idx, job_desc in enumerate(job_descriptions[:5]):
    print(f"Job Description {idx + 1}:")
    print(job_desc)
    print("-------------------------")
```

```
Job Description 1:
```

```
Work closely with product and business teams to identify research topics.,Act as
a thought leader in the domain of research, while advocating for the people who…
------------------------
Job Description 2:
Direct experience in responsible ,, , ,, and LLM research.,We are looking for
recognized experts in responsible NLP, with experience in areas like…
------------------------
Job Description 3:
You will collaborate on a wide array of product and business problems with a
diverse set of cross-functional partners across Product, Engineering, Research,…
------------------------
Job Description 4:
We utilize a full range of product design skills to contribute to high-level
strategic decisions with the rest of the product and executive teams.
------------------------
Job Description 5:
We are uniquely positioned to adopt an end-to-end approach to , , that few
organizations can offer, through building breakthrough product experiences…
------------------------
```

[20]:
```
pip install nltk
```

```
Collecting nltk
  Downloading nltk-3.8.1-py3-none-any.whl (1.5 MB)
     |                  | 1.5 MB 62.3 MB/s
Collecting regex>=2021.8.3
  Downloading
regex-2024.4.16-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (761
kB)
     |                  | 761 kB 92.1 MB/s
Requirement already satisfied: joblib in /opt/conda/lib/python3.7/site-
packages (from nltk) (0.14.1)
Requirement already satisfied: click in /opt/conda/lib/python3.7/site-packages
(from nltk) (7.1.2)
Requirement already satisfied: tqdm in /opt/conda/lib/python3.7/site-packages
(from nltk) (4.45.0)
Installing collected packages: regex, nltk
Successfully installed nltk-3.8.1 regex-2024.4.16
WARNING: You are using pip version 21.3.1; however, version 24.0 is

available.

You should consider upgrading via the '/opt/conda/bin/python -m pip install

--upgrade pip' command.
Note: you may need to restart the kernel to use updated packages.
```

[21]:
```
import nltk
from nltk.corpus import stopwords
```

```
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer

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

# Initialize stopwords and lemmatizer
stop_words = set(stopwords.words('english'))
lemmatizer = WordNetLemmatizer()

# Preprocess function
def preprocess_text(text):
    # Tokenization and lowercase conversion
    tokens = word_tokenize(text.lower())
    # Lemmatization and stopword removal
    filtered_tokens = [lemmatizer.lemmatize(token) for token in tokens if token.
 ↪isalnum() and token not in stop_words]
    return filtered_tokens

# Preprocess job descriptions
preprocessed_job_descriptions = [preprocess_text(job_desc) for job_desc in␣
 ↪job_descriptions]

# Display the preprocessed job descriptions
for idx, tokens in enumerate(preprocessed_job_descriptions[:5]):
    print(f"Preprocessed Job Description {idx + 1}:")
    print(tokens)
    print("------------------------")
```

```
[nltk_data] Downloading package stopwords to /home/jovyan/nltk_data…
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /home/jovyan/nltk_data…
[nltk_data]   Unzipping tokenizers/punkt.zip.
[nltk_data] Downloading package wordnet to /home/jovyan/nltk_data…
```

```
Preprocessed Job Description 1:
['work', 'closely', 'product', 'business', 'team', 'identify', 'research',
'act', 'thought', 'leader', 'domain', 'research', 'advocating', 'people']
------------------------
Preprocessed Job Description 2:
['direct', 'experience', 'responsible', 'llm', 'looking', 'recognized',
'expert', 'responsible', 'nlp', 'experience', 'area']
------------------------
Preprocessed Job Description 3:
['collaborate', 'wide', 'array', 'product', 'business', 'problem', 'diverse',
'set', 'partner', 'across', 'product', 'engineering', 'research']
------------------------
```

```
Preprocessed Job Description 4:
['utilize', 'full', 'range', 'product', 'design', 'skill', 'contribute',
'strategic', 'decision', 'rest', 'product', 'executive', 'team']
------------------------
Preprocessed Job Description 5:
['uniquely', 'positioned', 'adopt', 'approach', 'organization', 'offer',
'building', 'breakthrough', 'product']
------------------------
```

[22]:
```python
from sklearn.feature_extraction.text import TfidfVectorizer

# Convert preprocessed job descriptions back to text
preprocessed_job_descriptions_text = [" ".join(tokens) for tokens in␣
 ↪preprocessed_job_descriptions]

# Initialize TF-IDF vectorizer
tfidf_vectorizer = TfidfVectorizer()

# Fit and transform the preprocessed job descriptions
tfidf_matrix = tfidf_vectorizer.
 ↪fit_transform(preprocessed_job_descriptions_text)

# Display the shape of the TF-IDF matrix
print("Shape of TF-IDF matrix:", tfidf_matrix.shape)
```

```
Shape of TF-IDF matrix: (259, 1057)
```

[23]:
```python
from sklearn.cluster import KMeans

# Initialize K-means clustering with k=5
kmeans = KMeans(n_clusters=5, random_state=42)

# Fit K-means clustering to the TF-IDF matrix
kmeans.fit(tfidf_matrix)

# Get cluster labels for each job description
cluster_labels = kmeans.labels_

# Display the cluster labels
print("Cluster Labels:", cluster_labels)
```

```
Cluster Labels: [4 0 4 4 4 4 1 2 2 1 4 2 1 0 4 0 1 1 0 4 0 0 1 0 0 2 0 0 1 2 0 4
0 0 0 3 0
 0 2 3 3 0 3 1 2 3 0 0 1 2 3 0 1 0 0 3 0 0 0 2 4 2 2 4 0 2 1 0 0 2 4 4 2 4
 0 4 2 4 2 0 4 1 4 1 0 1 0 2 0 4 0 0 0 4 4 1 2 0 1 0 4 0 0 2 0 2 0 2 0 2 2
 1 0 1 4 4 4 2 1 0 0 2 1 0 0 2 4 2 2 4 0 0 4 2 4 1 0 2 1 0 4 2 2 2 4 0 0 2
 2 0 0 4 0 4 3 2 0 0 0 0 4 1 0 4 0 4 2 0 2 0 2 2 2 0 4 1 0 0 0 0 2 0 0 2 2
 0 0 0 0 4 0 1 4 0 0 0 0 0 1 3 0 2 0 2 0 0 1 1 0 0 0 0 0 0 0 1 0 4 1 0 0 1 0
```

```
 1 1 0 0 0 0 0 0 4 0 0 0 2 0 0 4 0 0 0 0 3 2 0 1 4 0 4 0 0 3 0 2 2 1 1 3 4]
```

[24]:
```python
# Create a dictionary to store job descriptions for each cluster
cluster_job_descriptions = {i: [] for i in range(5)}

# Group job descriptions by cluster
for idx, label in enumerate(cluster_labels):
    cluster_job_descriptions[label].append(job_descriptions[idx])

# Print job descriptions for each cluster
for cluster, descriptions in cluster_job_descriptions.items():
    print(f"\nCluster {cluster} Job Descriptions:")
    for idx, desc in enumerate(descriptions[:5]):  # Print the first 5␣
 ↪descriptions for brevity
        print(f"Job Description {idx + 1}: {desc}")
```

```
Cluster 0 Job Descriptions:
Job Description 1: Direct experience in responsible ,, , ,, and LLM research.,We
are looking for recognized experts in responsible NLP, with experience in areas
like…
Job Description 2: We focus on a collection of organisational technology
capabilities, including business intelligence, data management, and data
assurance that help our clients…
Job Description 3: We are looking for a Lead Engineer for implementing and
operating , , technologies applied to developer productivity e.g. GitHub
Copilot, Amazon…
Job Description 4: Experience with NLP platforms & familiarity with , , and
Large Language Models (LLMs).,This role will be responsible for driving key
automation &…
Job Description 5: We are looking for an experienced Business Intelligence Data
Manager to scale a critically important function responsible for building and
maintaining systems,…

Cluster 1 Job Descriptions:
Job Description 1: Full stack web/mobile application development with a variety
of coding languages.,Create consumer products and features using internal
programming language Hack…
Job Description 2: We rely on sophisticated in-house tools and new technologies
such as , , and automated feedback systems to build learning content at scale.
Job Description 3: Continually work to improve the design and functioning of
resources, including incorporating , artificial intelligence (AI) technologies
and suggesting…
Job Description 4: While you are currently permitted to work remotely in New
York and the San Francisco Bay Area, you must remain able to work in the office
as business needs…
Job Description 5: Knowledge of how marketing campaigns work, including content
generation as well as the potential applications of , , to these tasks.
```

Cluster 2 Job Descriptions:
Job Description 1: The ideal candidate will have a minimum of 5 years of finance or industry experience in sectors that include: Machine Learning, Data Science, and Artificial…
Job Description 2: The ideal candidate will have industry experience working on a range of classification and optimization problems, e.g. payment fraud, click-through rate…
Job Description 3: Bachelor's degree or equivalent practical experience.,6 years of experience in marketing.,Knowledge of product marketing with a track record in launching…
Job Description 4: 15+ years of experience in , ,, including hands-on experience with state-of-the-art , models and techniques.
Job Description 5: Experience with modern , methods, such as probabilistic ML, Bayesian reasoning, sampling algorithms, and , , models.

Cluster 3 Job Descriptions:
Job Description 1: Kasisto is the leader in conversational , for financial institutions, with a mission to humanize digital experiences.,What you need for this position.
Job Description 2: Aisera is the , , space focused on creating a ChatGPT experience for Enterprise applications.,Aisera fast tracks the digital transformation journey…
Job Description 3: Our Digital Experience Platform, KAI, is designed to enable financial institutions to offer personalized and intelligent digital assistants to their consumers,…
Job Description 4: Aisera is the , , space focused on creating a ChatGPT experience for Enterprise applications.,Aisera fast tracks the digital transformation journey…
Job Description 5: Kasisto is the leader in conversational , for financial institutions, with a mission to humanize digital experiences.

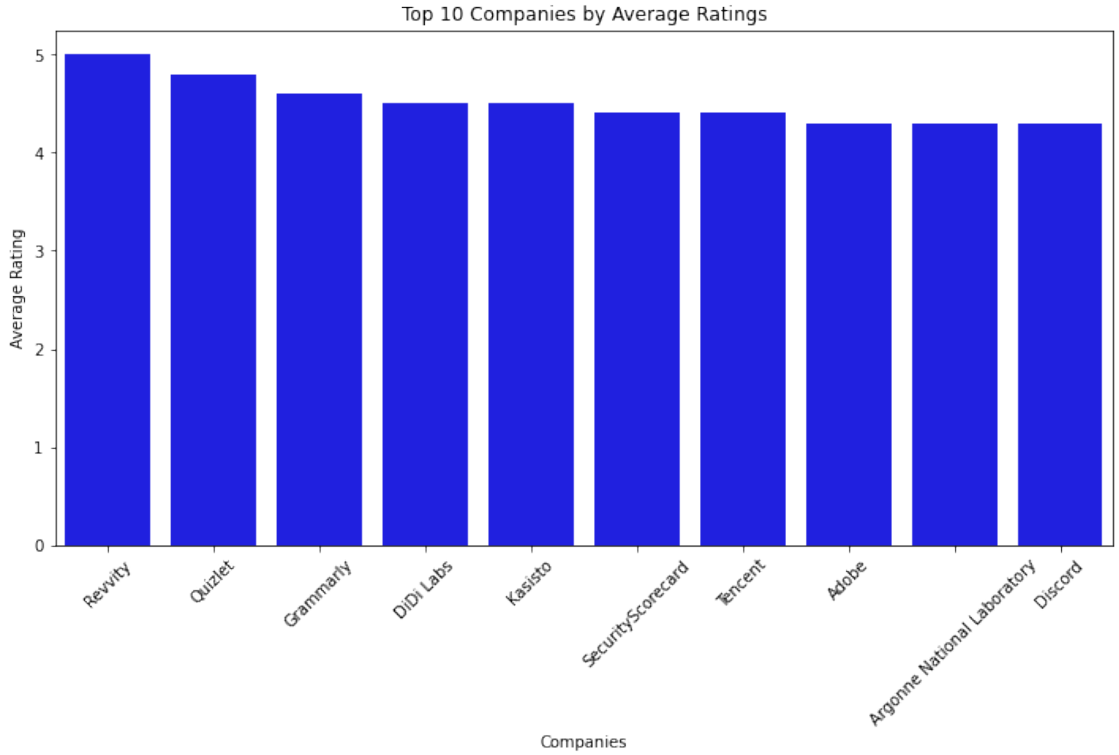
Cluster 4 Job Descriptions:
Job Description 1: Work closely with product and business teams to identify research topics.,Act as a thought leader in the domain of research, while advocating for the people who…
Job Description 2: You will collaborate on a wide array of product and business problems with a diverse set of cross-functional partners across Product, Engineering, Research,…
Job Description 3: We utilize a full range of product design skills to contribute to high-level strategic decisions with the rest of the product and executive teams.
Job Description 4: We are uniquely positioned to adopt an end-to-end approach to , , that few organizations can offer, through building breakthrough product experiences…
Job Description 5: Product TPMs work at the cross-section between technical execution and business strategy and are expected to partner closely with Engineering and Product teams.

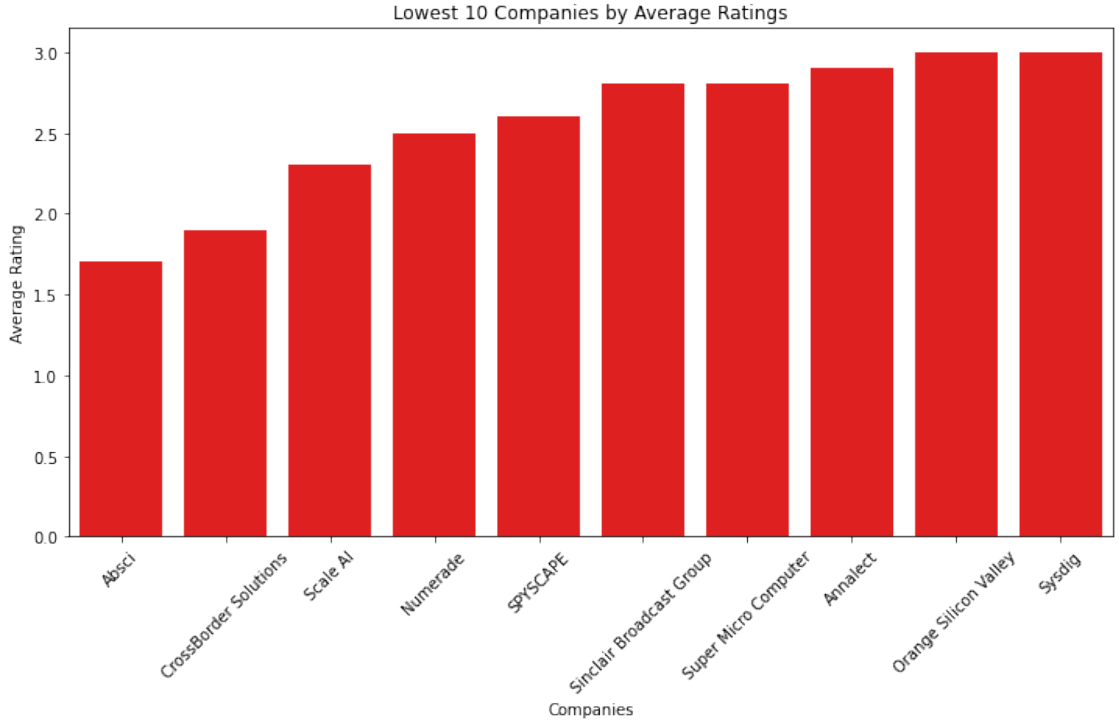
[25]:
```
pip install matplotlib seaborn
```

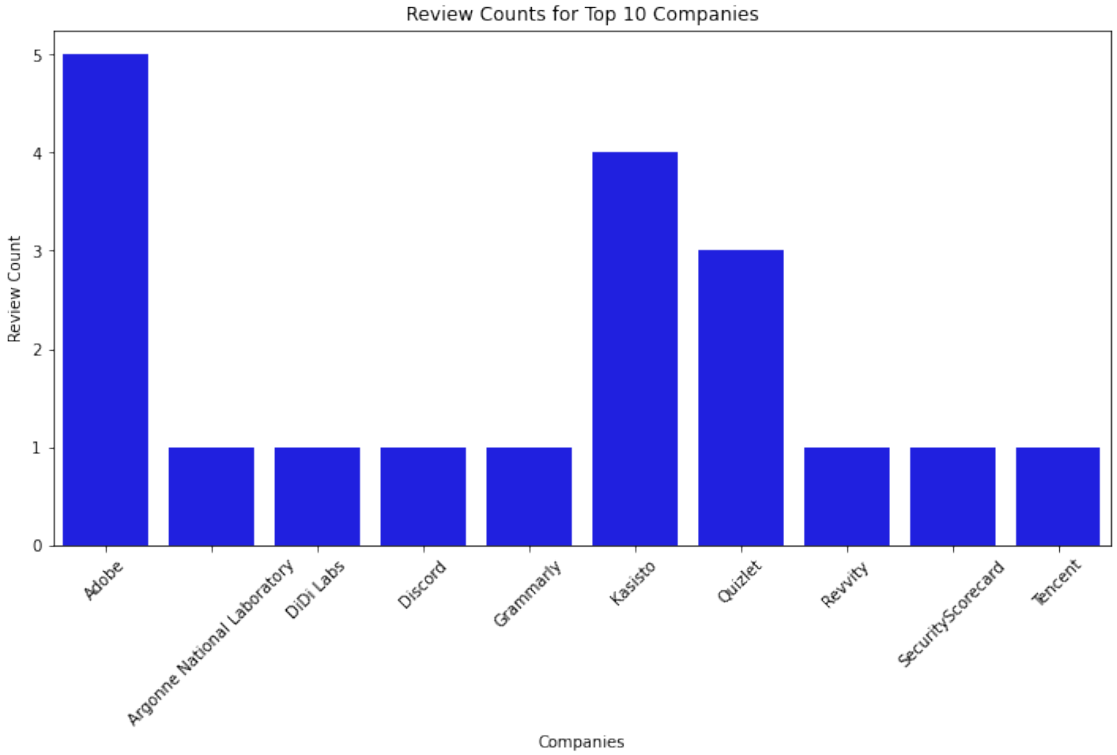
```
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.7/site-
packages (3.2.1)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.7/site-packages
(0.10.1)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.7/site-
packages (from matplotlib) (0.10.0)
Requirement already satisfied: numpy>=1.11 in /opt/conda/lib/python3.7/site-
packages (from matplotlib) (1.18.4)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib) (1.2.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib) (2.4.7)
Requirement already satisfied: python-dateutil>=2.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: pandas>=0.22.0 in /opt/conda/lib/python3.7/site-
packages (from seaborn) (1.0.3)
Requirement already satisfied: scipy>=1.0.1 in /opt/conda/lib/python3.7/site-
packages (from seaborn) (1.4.1)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from cycler>=0.10->matplotlib) (1.14.0)
Requirement already satisfied: pytz>=2017.2 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.22.0->seaborn) (2020.1)
WARNING: You are using pip version 21.3.1; however, version 24.0 is
available.
You should consider upgrading via the '/opt/conda/bin/python -m pip install
--upgrade pip' command.
Note: you may need to restart the kernel to use updated packages.
```

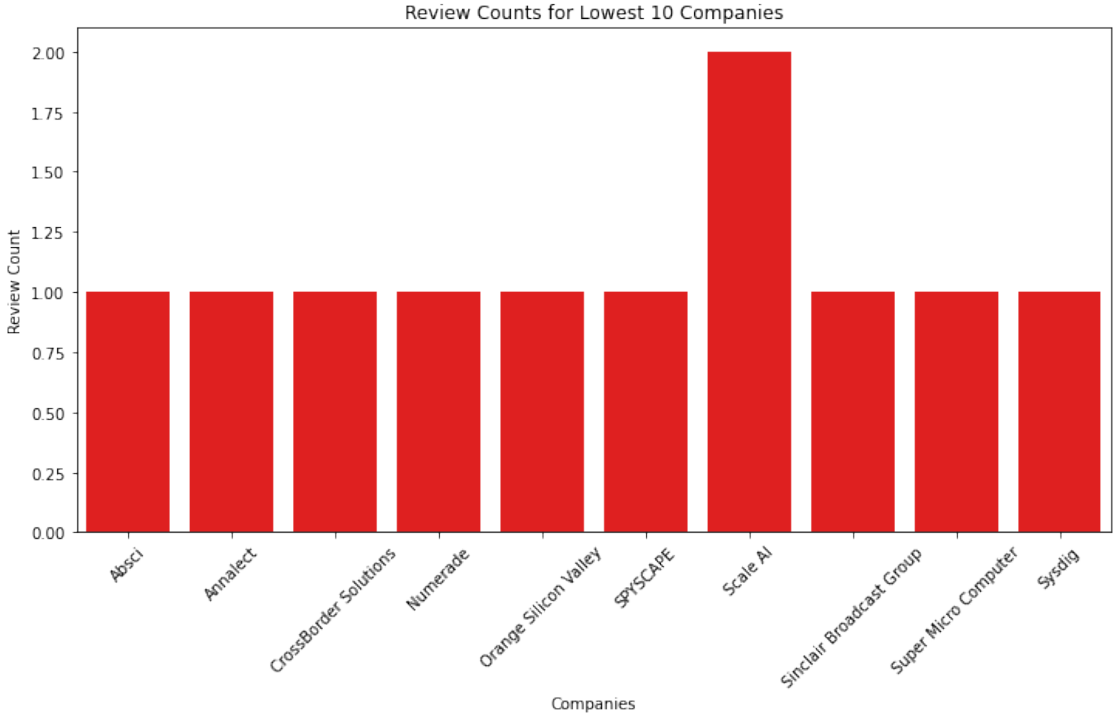
[35]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Get the top 10 and lowest 10 companies based on average ratings
top_10_companies = merged_df.groupby('company_name')['rating'].mean().
 ↪nlargest(10)
lowest_10_companies = merged_df.groupby('company_name')['rating'].mean().
 ↪nsmallest(10)

# Get review counts for the top 10 and lowest 10 companies
top_10_review_counts = merged_df[merged_df['company_name'].
 ↪isin(top_10_companies.index)].groupby('company_name')['rating'].count()
lowest_10_review_counts = merged_df[merged_df['company_name'].
 ↪isin(lowest_10_companies.index)].groupby('company_name')['rating'].count()
```

```
# Plot for top 10 companies
plt.figure(figsize=(12, 6))
sns.barplot(x=top_10_companies.index, y=top_10_companies.values, color='blue')
plt.title('Top 10 Companies by Average Ratings')
plt.xlabel('Companies')
plt.ylabel('Average Rating')
plt.xticks(rotation=45)
plt.show()

# Plot for lowest 10 companies
plt.figure(figsize=(12, 6))
sns.barplot(x=lowest_10_companies.index, y=lowest_10_companies.values,␣
↪color='red')
plt.title('Lowest 10 Companies by Average Ratings')
plt.xlabel('Companies')
plt.ylabel('Average Rating')
plt.xticks(rotation=45)
plt.show()

# Plot for review counts of top 10 companies
plt.figure(figsize=(12, 6))
sns.barplot(x=top_10_review_counts.index, y=top_10_review_counts.values,␣
↪color='blue')
plt.title('Review Counts for Top 10 Companies')
plt.xlabel('Companies')
plt.ylabel('Review Count')
plt.xticks(rotation=45)
plt.show()

# Plot for review counts of lowest 10 companies
plt.figure(figsize=(12, 6))
sns.barplot(x=lowest_10_review_counts.index, y=lowest_10_review_counts.values,␣
↪color='red')
plt.title('Review Counts for Lowest 10 Companies')
plt.xlabel('Companies')
plt.ylabel('Review Count')
plt.xticks(rotation=45)
plt.show()
```


Top 10 Companies by Average Ratings
Average Rating
0
1
2
3
4
5
Revvity
Quizlet
Grammarly
DiDi Labs
Kasisto
SecurityScorecard
Tencent
Adobe
Argonne National Laboratory
Discord
Companies


Lowest 10 Companies by Average Ratings
Average Rating
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Absci
CrossBorder Solutions
Scale AI
Numerade
SPYSCAPE
Sinclair Broadcast Group
Super Micro Computer
Annalect
Orange Silicon Valley
Sysdig
Companies

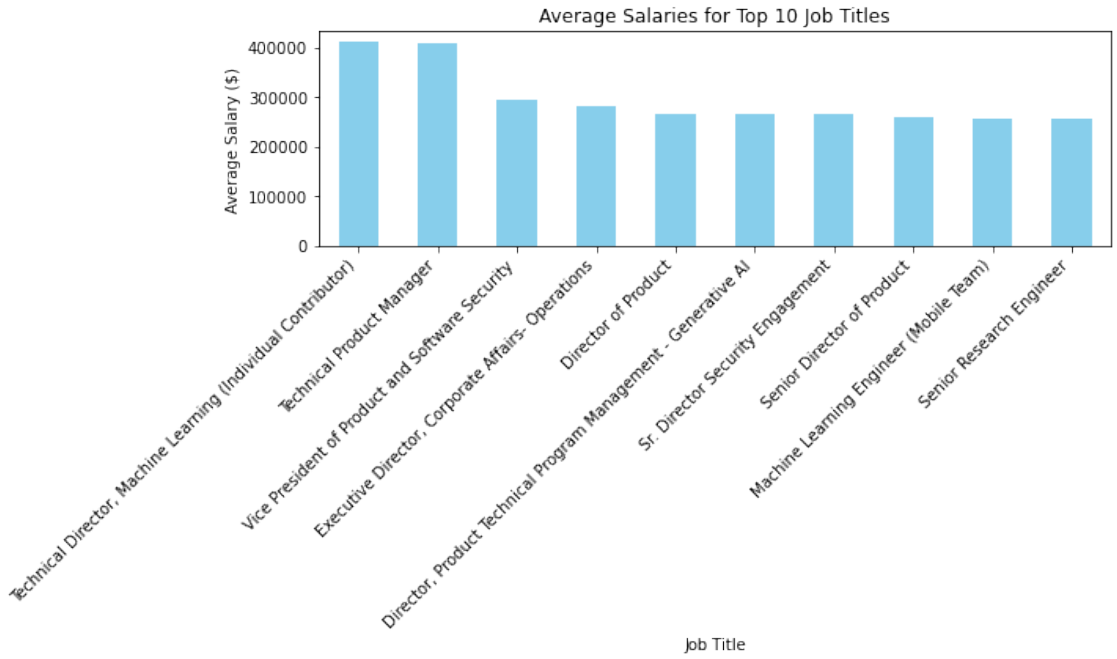

Review Counts for Top 10 Companies
Review Count
0
1
2
3
4
5
Adobe
Argonne National Laboratory
DiDi Labs
Discord
Grammarly
Kasisto
Quizlet
Revvity
SecurityScorecard
Tencent
Companies


Review Counts for Lowest 10 Companies
Review Count
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Absci
Annalect
CrossBorder Solutions
Numerade
Orange Silicon Valley
SPYSCAPE
Scale AI
Sinclair Broadcast Group
Super Micro Computer
Sysdig
Companies

[28]:
```
# Calculate average salaries for different job titles
average_salaries = merged_df.groupby('job_title')['salary'].mean().nlargest(10)
print("\nAverage salaries for top 10 job titles:\n", average_salaries)
```

```
Average salaries for top 10 job titles:
 job_title
Technical Director, Machine Learning (Individual Contributor)     412050.0
Technical Product Manager                                         410000.0
Vice President of Product and Software Security                   292950.0
Executive Director, Corporate Affairs- Operations                 279997.0
Director of Product                                               266500.0
Director, Product Technical Program Management - Generative AI    265000.0
Sr. Director Security Engagement                                  265000.0
Senior Director of Product                                        258400.0
Machine Learning Engineer (Mobile Team)                           256500.0
Senior Research Engineer                                          254750.0
Name: salary, dtype: float64
```

[30]:
```
import matplotlib.pyplot as plt

# Plot average salaries for top 10 job titles
plt.figure(figsize=(10, 6))
average_salaries.plot(kind='bar', color='skyblue')
plt.title('Average Salaries for Top 10 Job Titles')
plt.xlabel('Job Title')
plt.ylabel('Average Salary ($)')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

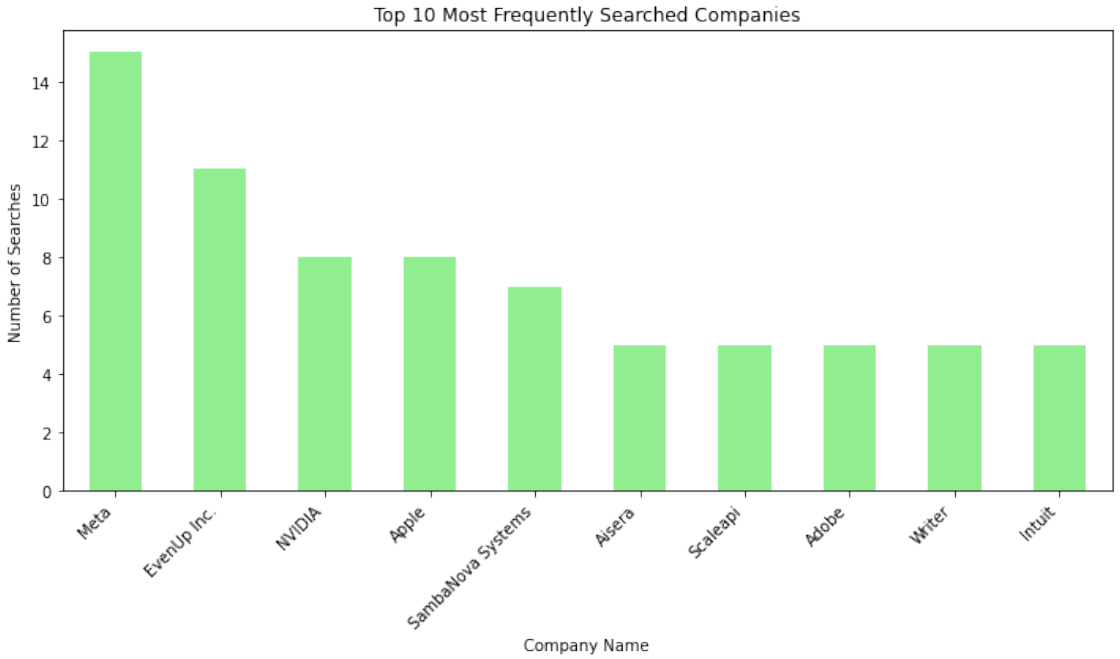


[29]:
```
# Determine which companies are most frequently searched for
most_searched_companies = merged_df['company_name'].value_counts().nlargest(10)
print("\nTop 10 most frequently searched companies:\n", most_searched_companies)
```

```
Top 10 most frequently searched companies:
 Meta                15
EvenUp Inc.          11
NVIDIA                8
Apple                 8
SambaNova Systems     7
Aisera                5
Scaleapi              5
Adobe                 5
Writer                5
Intuit                5
Name: company_name, dtype: int64
```

[31]:
```
# Plot top 10 most frequently searched companies
plt.figure(figsize=(10, 6))
most_searched_companies.plot(kind='bar', color='lightgreen')
plt.title('Top 10 Most Frequently Searched Companies')
plt.xlabel('Company Name')
plt.ylabel('Number of Searches')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
```

```
plt.show()
```


Top 10 Most Frequently Searched Companies
Number of Searches
0
2
4
6
8
10
12
14
Meta
EvenUp Inc.
NVIDIA
Apple
SambaNova Systems
Aisera
Scaleapi
Adobe
Writer
Intuit
Company Name

[ ]: