What Life Looks Like in Data!

Lecture1_Intro.pptx

Download Lecture1_Intro.pptx (11.9 MB)

Page 16 of 46 ZOOM

Data

"Data is the new oil."

"It's valuable, but if unrefined, it cannot really be used. It has to be changed into gas, plastic, chemicals, etc., to create a valuable entity that drives profitable activity; so data must be broken down, analyzed for it to have value."

Clive Humby 2006

15

The Goal?

For someone who is looking for a career path in mastering the refining, analysis, and/or deployment/modeling of data, we are trying to determine what life could look like through modeling methods we've learned in class.

- What does the market look like?/ What would you need to know?
- What could your work schedule look like?
- Where might you geographically be?

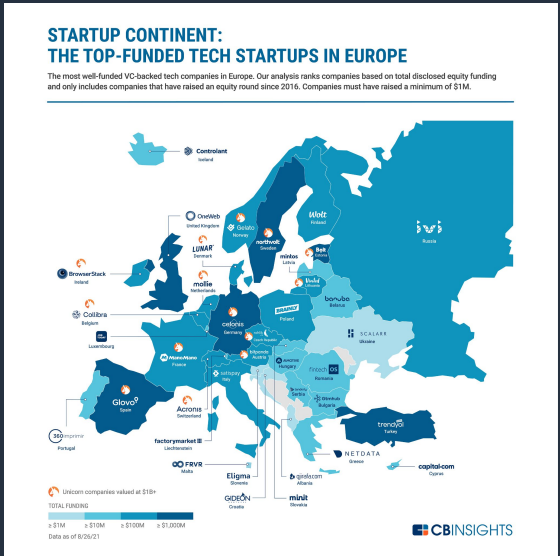
Intro. To Our Data

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6025 entries, 0 to 6024
Data columns (total 11 columns):
 #   Column          Non-Null Count  Dtype 
---  ------          --------------  ----- 
 0   job_title       6025 non-null   object
 1   company         6025 non-null   object
 2   job_location    6025 non-null   object
 3   job_link        6025 non-null   object
 4   first_seen      6025 non-null   object
 5   search_city     6025 non-null   object
 6   search_country  6025 non-null   object
 7   job level       6025 non-null   object
 8   job_type        6025 non-null   object
 9   job_summary     5665 non-null   object
 10  job_skills      4960 non-null   object
dtypes: object(11)
memory usage: 517.9+ KB
```

```
[41] df.shape
```

```
(6025, 11)
```

Predictions



Clustering Job Postings by Location

K Means & Agglomerative Clustering

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

```
array(['Bloomington', 'Del Rio', 'Inverell', 'Northampton', 'Spokane',
       'Cincinnati', 'Garland', 'Silver Spring', 'Bridgeport', 'Exeter',
       'Dundee', 'Beverly', 'Fort Collins', 'Macon', 'Norristown',
       'Defiance', 'Little Rock', 'Germantown', 'Flushing', 'Oceanside',
       'High Wycombe', 'Ipswich', 'Baytown', 'Lebanon', 'Seminole',
       'Providence', 'Corinth', 'Collinsville', 'Herrin', 'Leavenworth',
       'Manti', 'Ohio', 'Gloucester', 'Hollywood', 'Asbury Park',
       'Litchfield', 'Perth', 'Haverhill', 'Glendale', 'West Seneca',
       'Pasadena', 'Atlanta', 'Cambridge', 'Bethany', 'North Carolina',
       'Elizabethton', 'Ephrata', 'San Juan Capistrano', 'La Habra',
       'Layton', 'Lawton', 'Savannah', 'Van Buren', 'San Luis Obispo',
       'Santa Clara', 'Arlington', 'Compton', 'Eastchester', 'Wisconsin',
       'Cleveland Heights', 'Las Vegas', 'Calexico', 'Sainte-Foy',
       'Durham', 'Peoria', 'Santa Rosa', 'Wenatchee', 'Baltimore',
       'Denton', 'Gastonia', 'Greater London', 'Georgia', 'Medicine Hat',
       'El Dorado', 'Novato', 'Greenville', 'State College',
       'Mount Vernon', 'Stockbridge', 'Huntington', 'Cardiff',
       'Fitzgerald', 'Dickinson', 'Avondale', 'Blackburn', 'San Felipe',
       'Washington', 'Montpelier', 'Tyneside', 'Jackson', 'Monroe',
       'Ogden', 'Brantford', 'Covington', 'Sarnia-Clearwater', 'Oneonta',
       'Bound Brook', 'Raleigh', 'New Jersey', 'Union City', 'Ishpeming',
       'Bayonne', 'Dallas', 'Puyallup', 'Lexington', 'Cleveland',
       'Saskatchewan', 'Weirton', 'Malden', 'Palmdale', 'Maryland',
       'Beechworth', 'Temiskaming Shores', 'Oregon', 'Levittown',
       'Rutland', 'Grand Island', 'Tallahassee', 'Arkansas',
       'Gainesville', 'El Cerrito', 'Pompano Beach', 'Shelby',
       'New Westminster', 'Dedham', 'Santa Barbara', 'Marysville',
       'Winslow', 'Steubenville', 'South Carolina', 'Herkimer',
       'Glens Falls', 'Decatur', 'Nome', 'Arthur', 'Highland Park',
       'Watertown', 'Pendleton', 'Fairbanks', 'Largo', 'Little Falls',
       'Charleston', 'Lincoln', 'Canberra', 'Waynesboro', 'Vandalia',
       'Nashville', 'Homer', 'Hamilton', 'Charles City', 'Yarmouth',
       'Norfolk', 'Whitman', 'Nyack', 'Norwich', 'Chicago Heights',
       'Laurentian Hills', 'Saint Albert', 'Palatka', 'Nebraska',
       'Laurel', 'Climax', 'Stoke-on-Trent', 'Massachusetts', 'Canyon',
       'Chickasaw', 'Plains', 'Lake Forest', 'Heber City', 'Burnaby',
       'Penrith', 'Louisiana', 'Montrose', 'San Antonio', 'Yellowknife',
       'Kentucky', 'East Lansing', 'Boston', 'Oswego', 'Jonquière',
       'San Rafael', 'Dorval', 'Edinburgh', 'Oyster Bay', 'Enterprise',
       'San Leandro', 'Coral Gables', 'East Aurora', 'Stillwater',
       'Tamworth', 'Crawley', 'Red Deer', 'Worcester', 'Summit',
       'Oak Ridge', 'Brunswick', 'Gosford', 'North Chicago', 'Pomona',
       'Manassas', 'Vernon', 'Brighton and Hove', 'Chicago', 'Barrie',
       'Hopewell', 'Quesnel', 'Delta', 'Bend', 'Attleboro', 'Sunderland',
```

667 Unique Search Cities

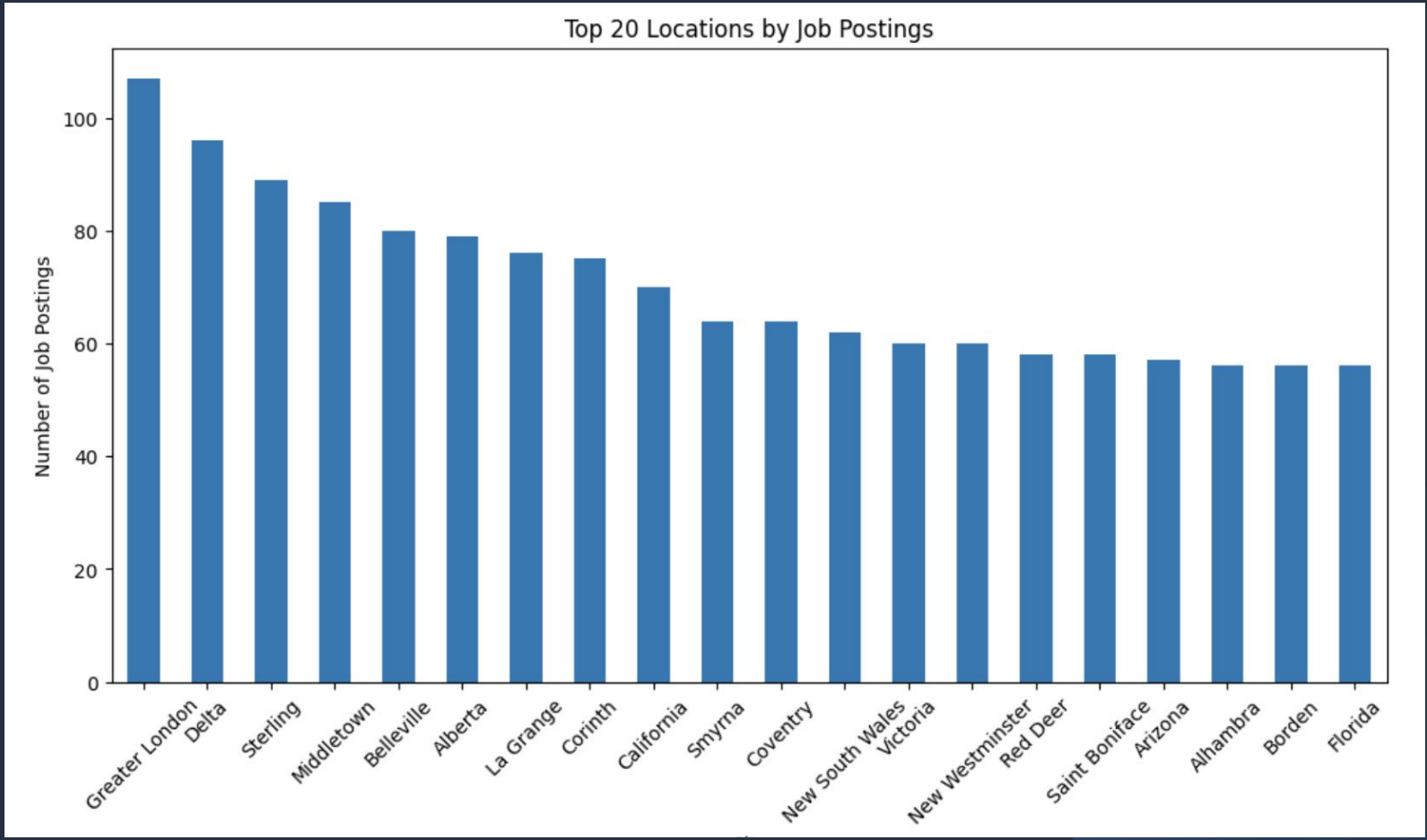
```
df.isna().sum()
```

	0
job_title	0
company	0
job_location	0
job_link	0
first_seen	0
search_city	0
search_country	0
job level	0
job_type	0
job_summary	360
job_skills	1065

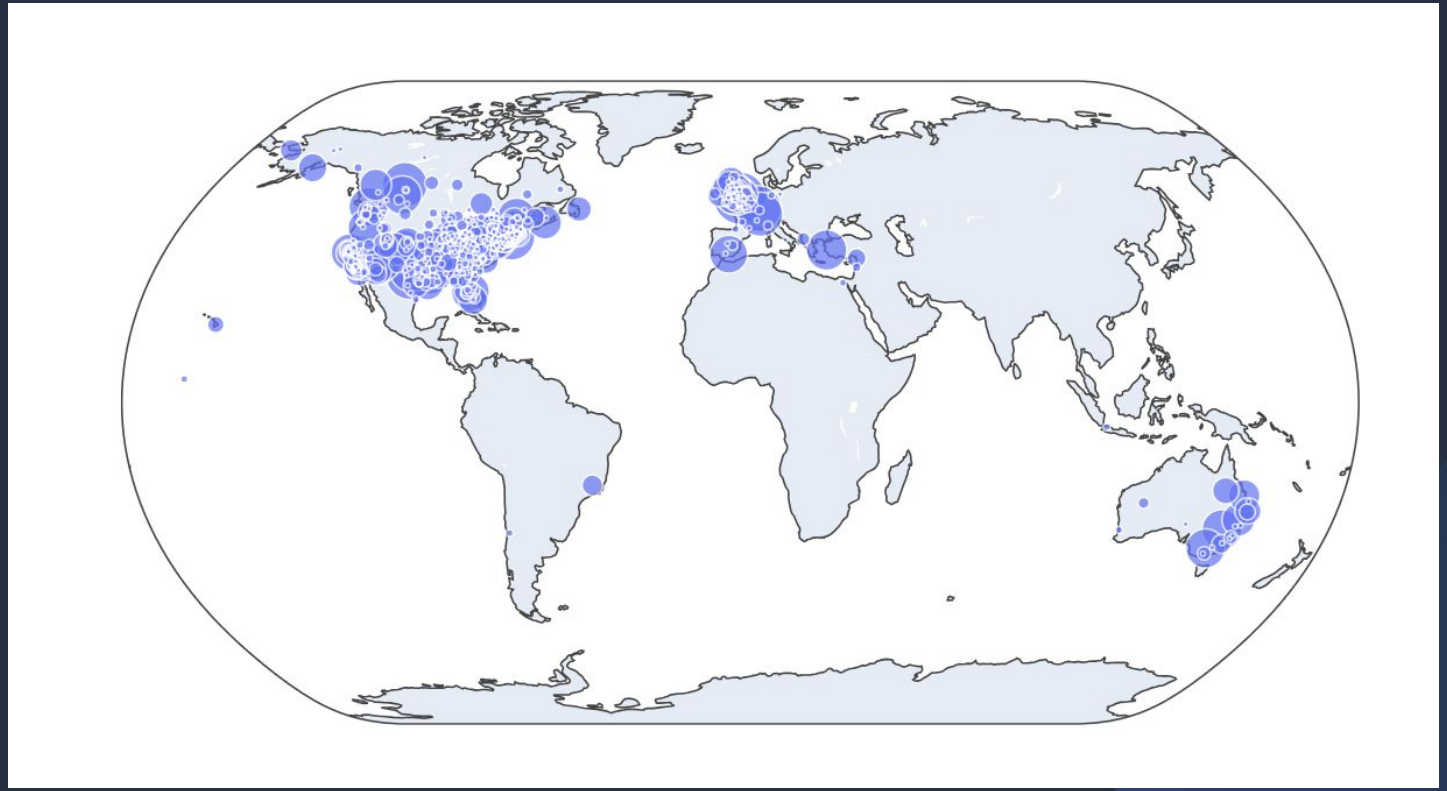
```
[7] df['job_summary'] = df['job_summary'].fillna('')
    df['job_skills'] = df['job_skills'].fillna('')
    df['job_skills'] = df['job_skills'].fillna('No skills listed')
    df['job_summary'] = df['job_summary'].fillna('No summary listed')
```

```
df.isna().sum()
```

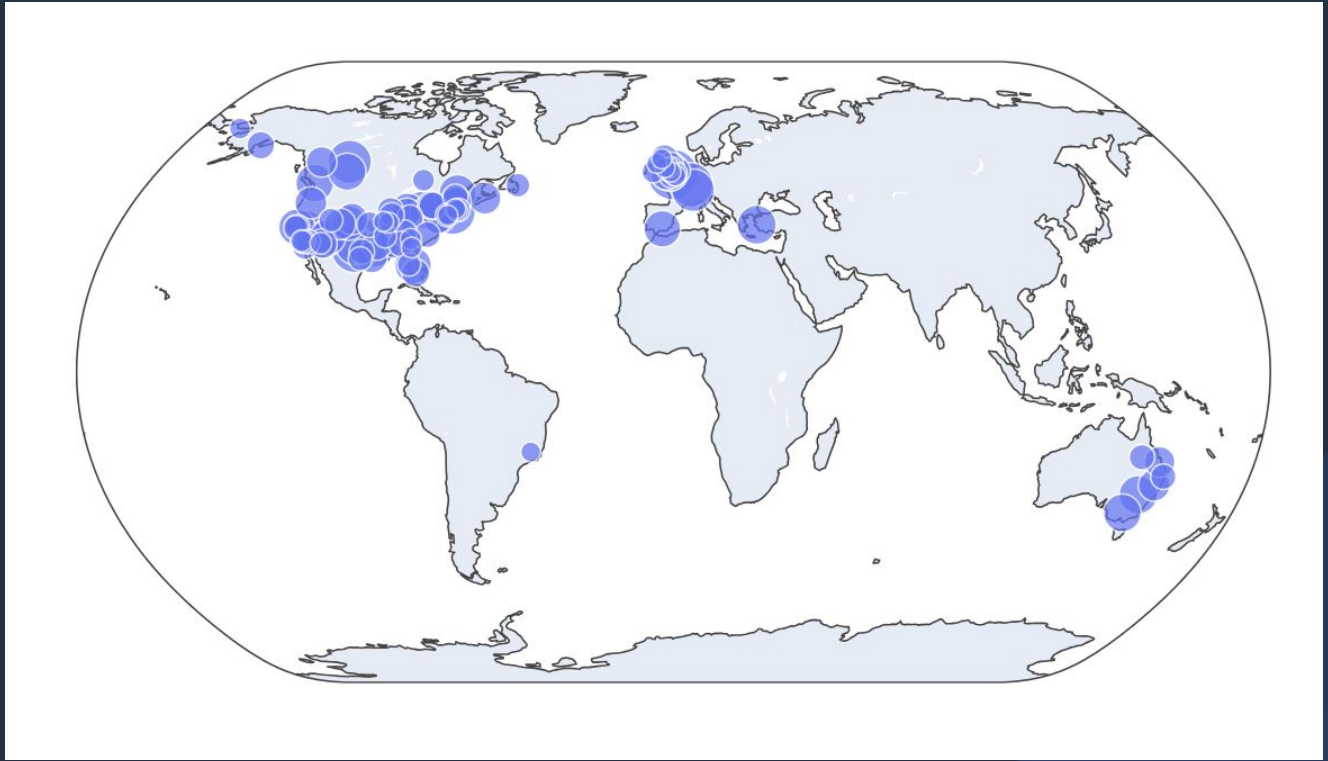
	0
job_title	0
company	0
job_location	0
job_link	0
first_seen	0
search_city	0
search_country	0
job level	0
job_type	0
job_summary	0
job_skills	0


Top 20 Locations by Job Postings
Number of Job Postings
0
20
40
60
80
100
Greater London
Delta
Sterling
Middletown
Belleville
Alberta
La Grange
Corinth
California
Smyrna
Coventry
New South Wales
Victoria
New Westminster
Red Deer
Saint Boniface
Arizona
Alhambra
Borden
Florida

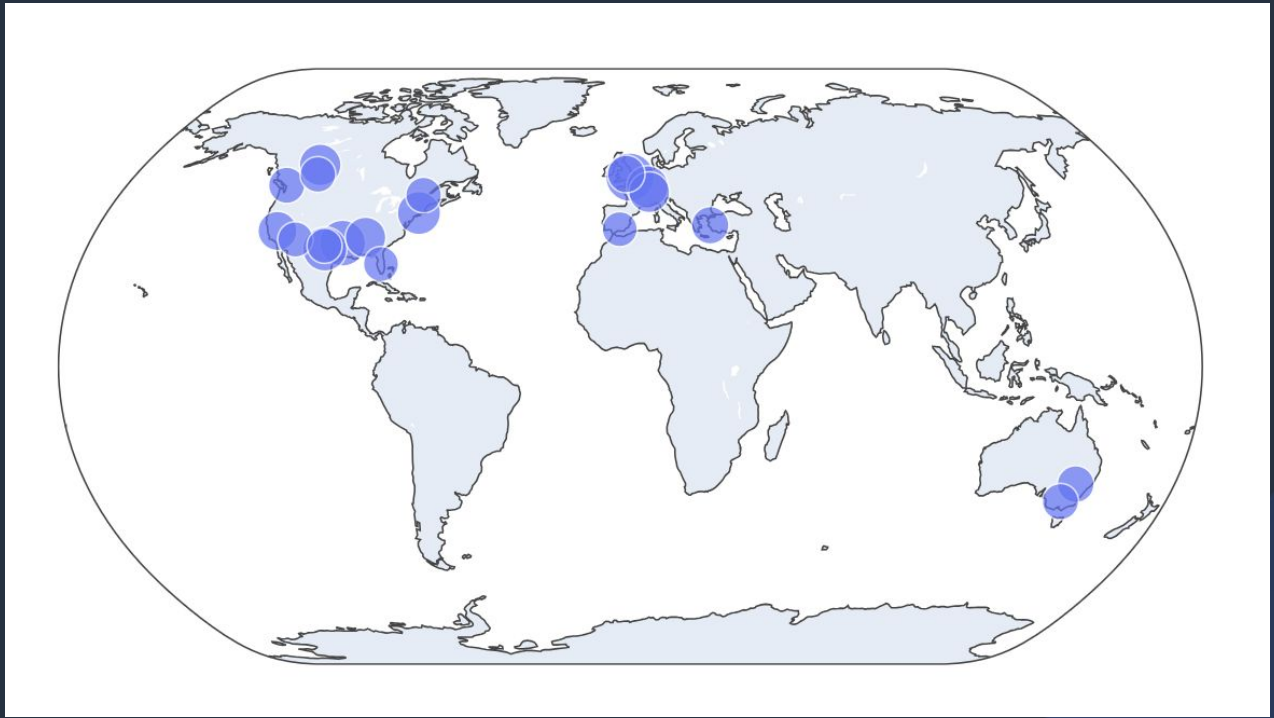
All Locations Mapped
Larger size = more postings



Top 100 Locations mapped



Top 20 Locations mapped



Top 20 Locations mapped;
Larger size = more postings

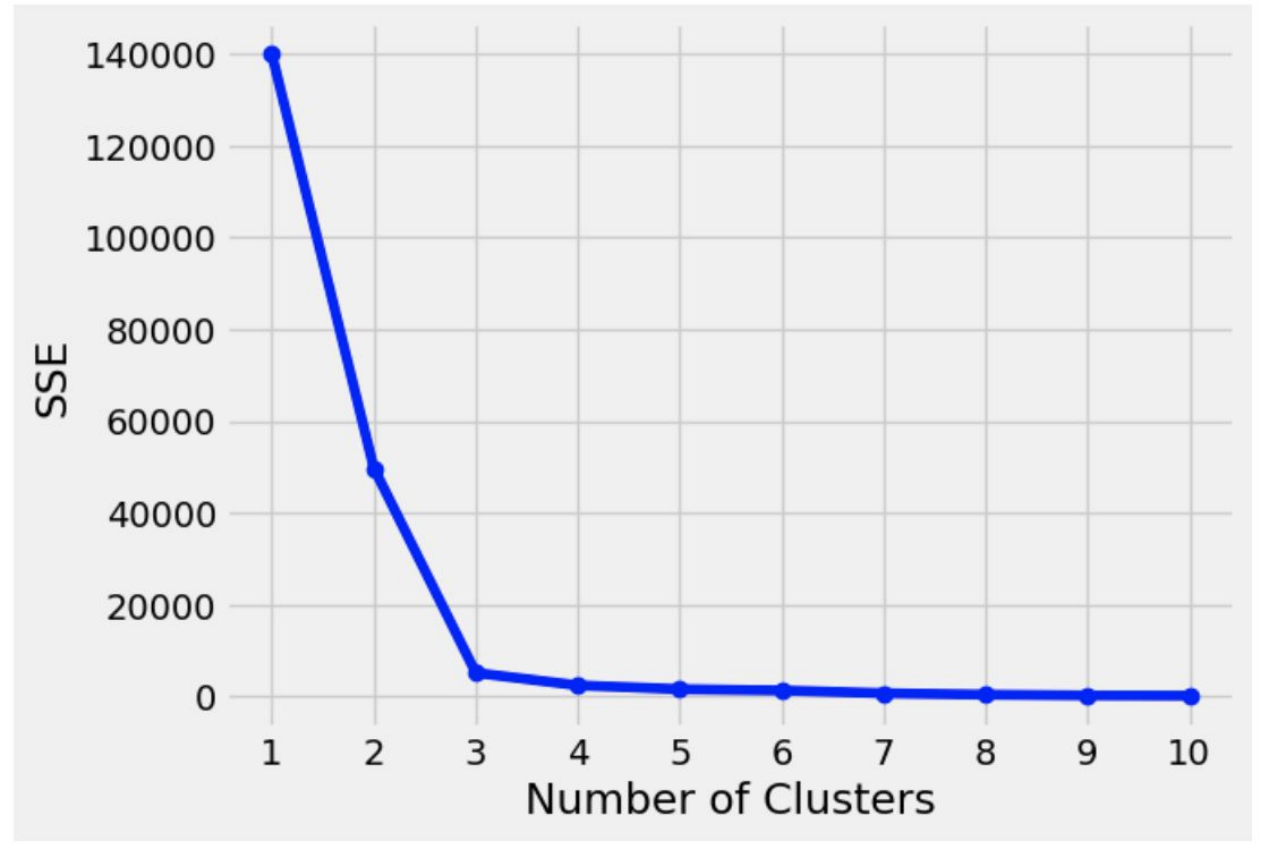
Technical Aspects

```
from geopy.geocoders import Nominatim
from geopy.extra.rate_limiter import RateLimiter
```

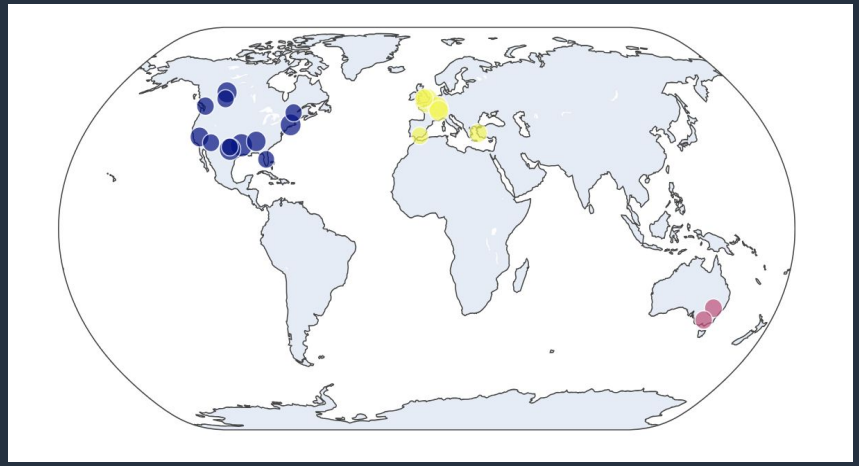
```
geolocator = Nominatim(user_agent="job_posting_locator")
geocode = RateLimiter(geolocator.geocode, min_delay_seconds=1)

def get_coords(city):
    try:
        loc = geocode(city)
        if loc:
            return loc.latitude, loc.longitude
        else:
            return None, None
    except:
        return None, None
```

```
fig = px.scatter_geo(city_counts,
                     lat='lat',
                     lon='lon',
                     hover_name='city',
                     size='count',
                     projection="natural earth",
                     title='Top 20 Cities by Job Postings')
```

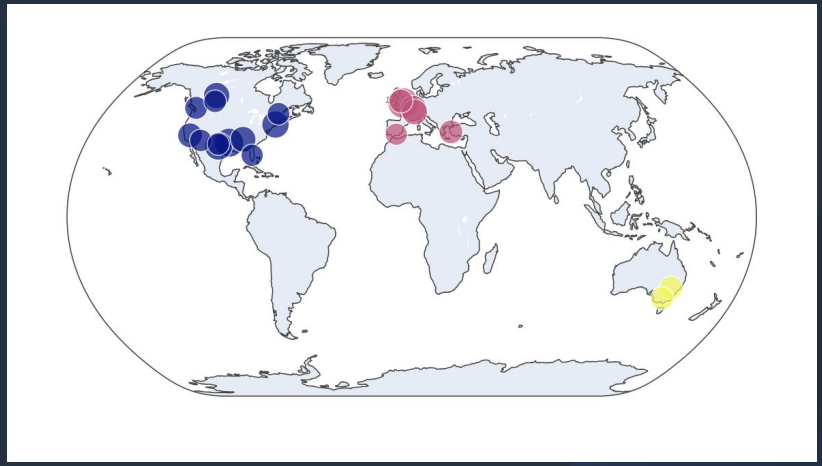
K Means Clustering



K-Means Clustering



Agglomerative Clustering



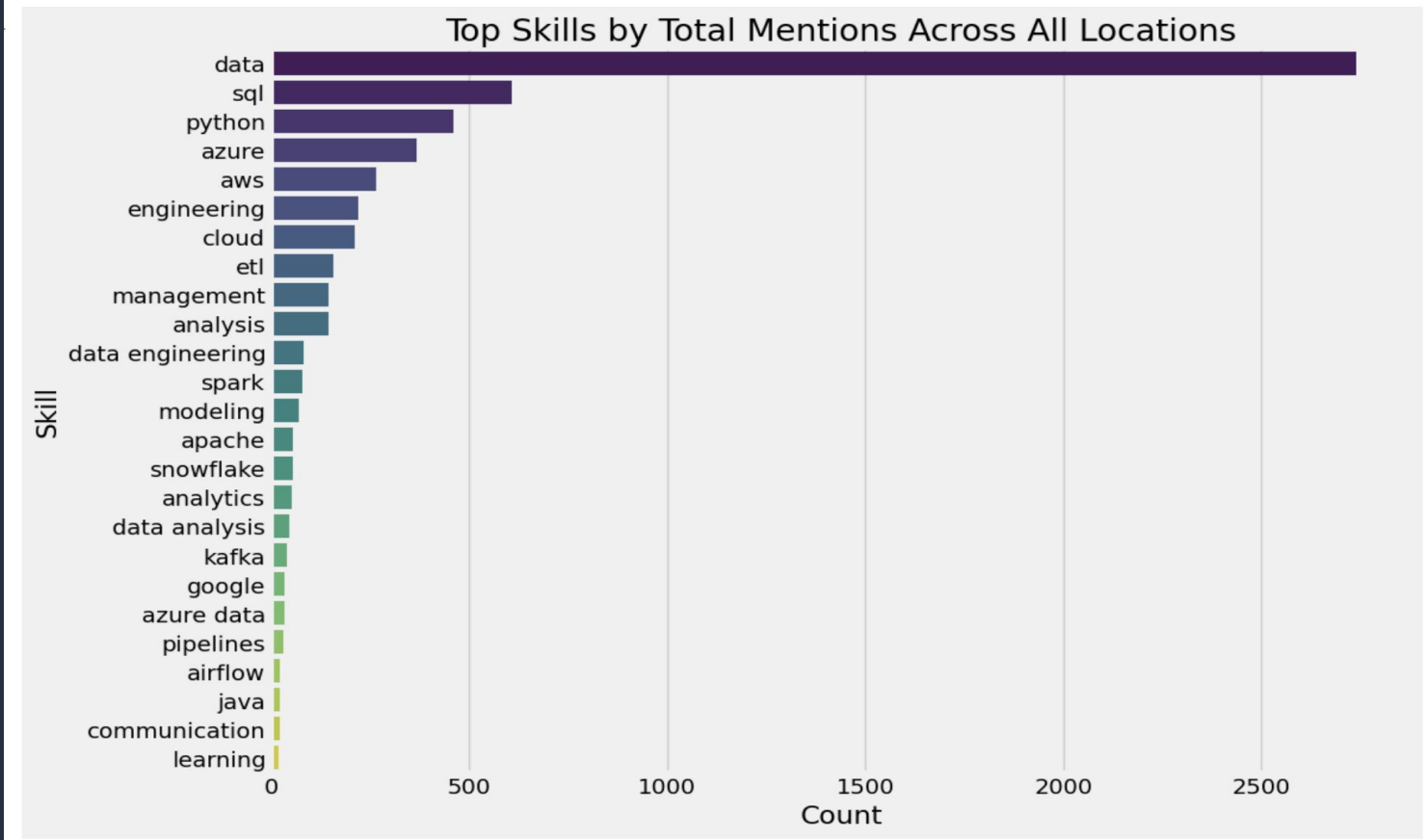
```
Silhouette Score for k=3: 0.807
```

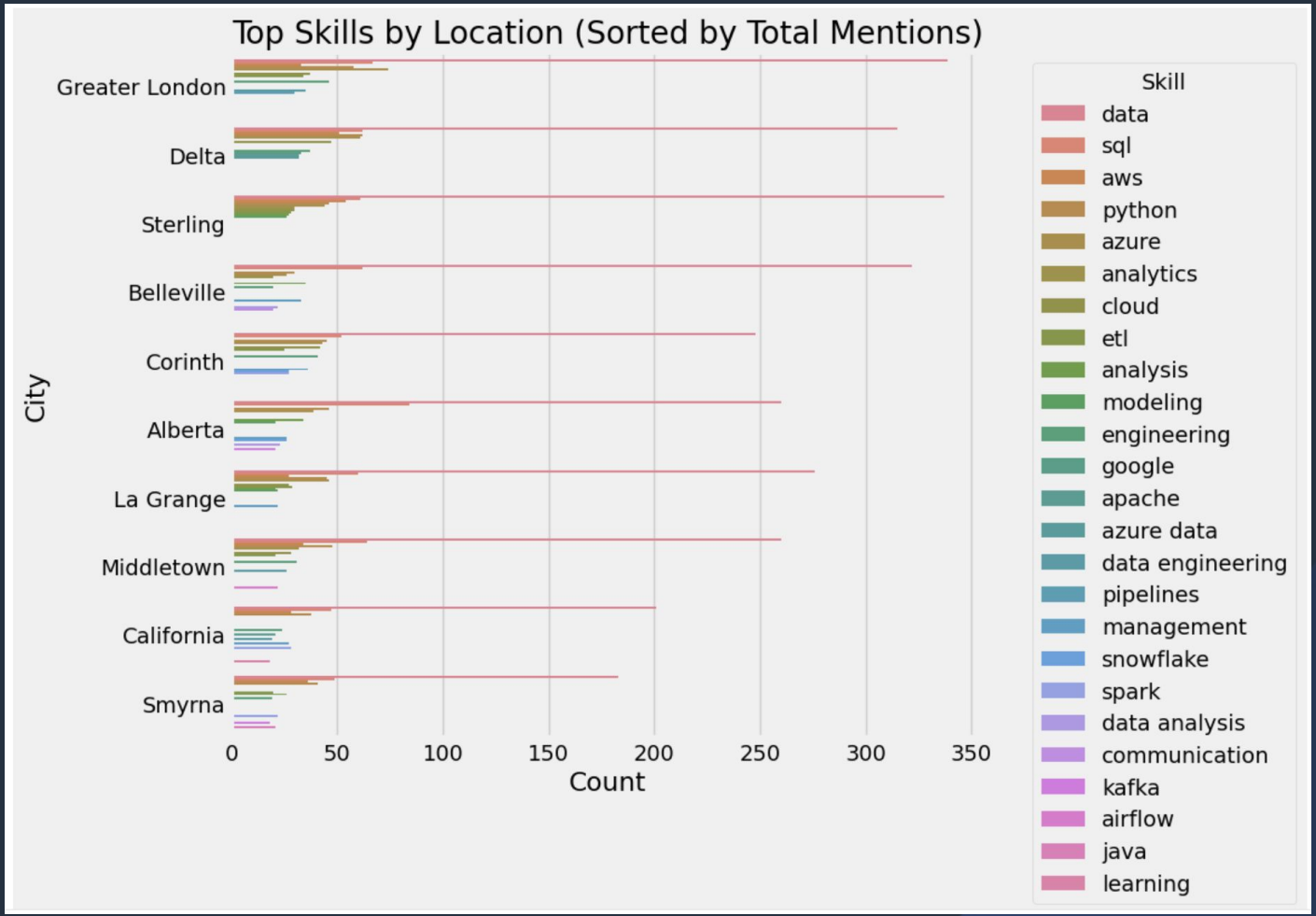
Part 2: Text Analysis

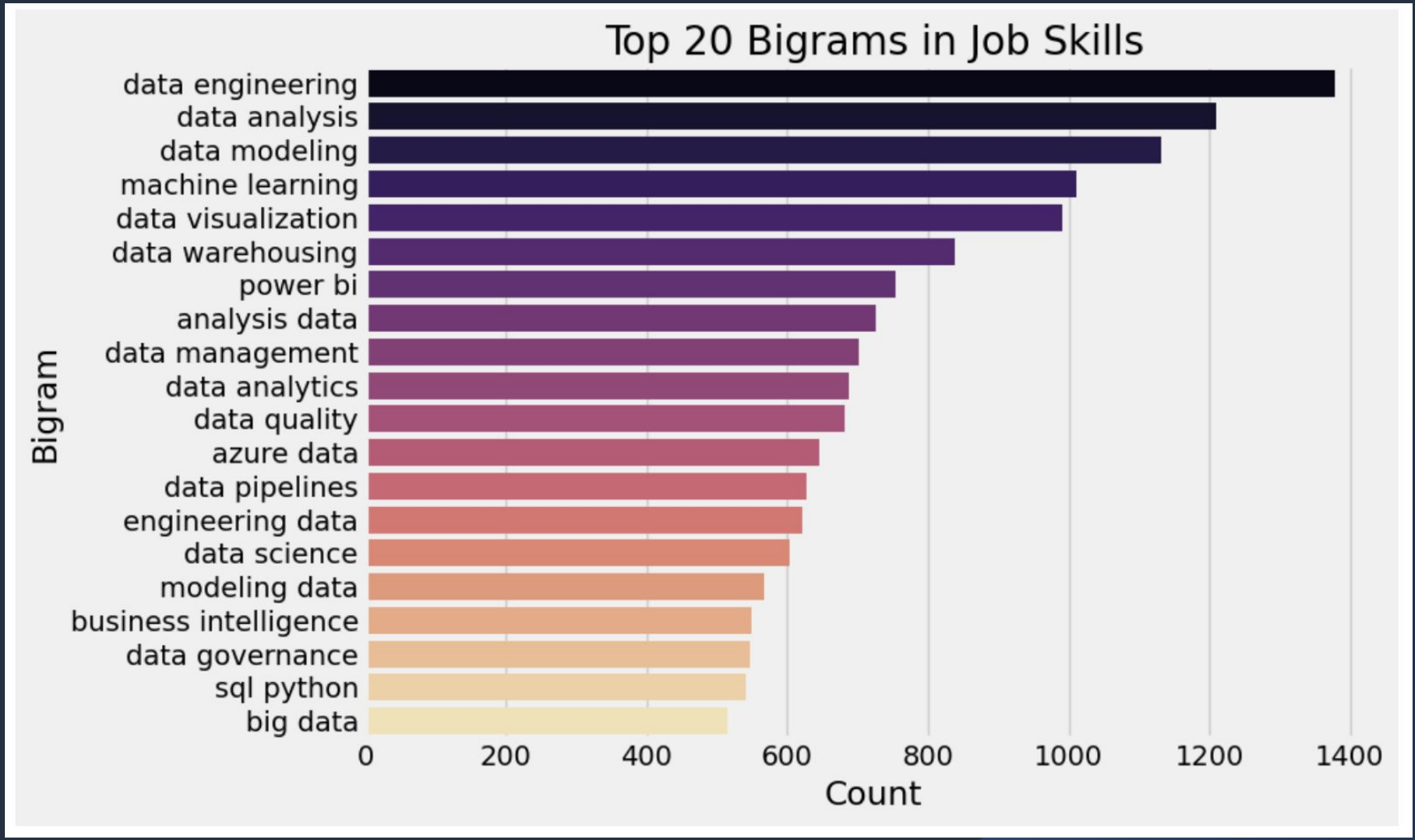
Finding Trends & Similarities within Job Postings

“Job Skills” Example:

Python, Snowflake, Airflow, Kubernetes, Docker, Helm, Spark, pySpark, SQL, TDD, Pair Programming, Continuous Integration, automated testing, Kafka, Storm, SparkStreaming, dimensional data modeling, ETL, data management, data classification


Top Skills by Total Mentions Across All Locations
Skill
data
sql
python
azure
aws
engineering
cloud
etl
management
analysis
data engineering
spark
modeling
apache
snowflake
analytics
data analysis
kafka
google
azure data
pipelines
airflow
java
communication
learning
0
500
1000
1500
2000
2500
Count


Top Skills by Location (Sorted by Total Mentions)
City
Greater London
Delta
Sterling
Belleville
Corinth
Alberta
La Grange
Middletown
California
Smyrna
0
50
100
150
200
250
300
350
Count
Skill
data
sql
aws
python
azure
analytics
cloud
etl
analysis
modeling
engineering
google
apache
azure data
data engineering
pipelines
management
snowflake
spark
data analysis
communication
kafka
airflow
java
learning


Top 20 Bigrams in Job Skills
data engineering
data analysis
data modeling
machine learning
data visualization
data warehousing
power bi
analysis data
data management
data analytics
data quality
azure data
data pipelines
engineering data
data science
modeling data
business intelligence
data governance
sql python
big data
Bigram
0
200
400
600
800
1000
1200
1400
Count

Clustering Job Summaries U
TF-IDF & KMeans

Goal: Automatically group job descriptions by skill and role type to uncover key hiring patterns

Approach:

- Cleaned and preprocessed text (lowercase, removed noise)
- Used TF-IDF to convert summaries into keyword-based vectors
- Applied KMeans (K=4) to cluster job descriptions into groups

```python
import pandas as pd
import re
import nltk
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans

# Download stopwords
nltk.download('stopwords')
from nltk.corpus import stopwords

# Load English stopwords
stop_words = set(stopwords.words('english'))

# Step 1: Fill missing summaries
df['job_summary'] = df['job_summary'].fillna('')

# Step 2: Advanced text cleaning
def clean_text(text):
    text = text.lower()
    text = re.sub(r'https?://\S+', '', text)  # Remove URLs
    text = re.sub(r'\b(?:canada|usa|uk|united states|california|london|chicago|new york|san francisco)\b', '', text)
    text = re.sub(r'[^a-z\s]', '', text)  # Remove punctuation and digits
    text = re.sub(r'\s+', ' ', text).strip()  # Normalize spaces
    text = ' '.join([word for word in text.split() if word not in stop_words])  # Remove stopwords
    return text

df['job_summary_clean'] = df['job_summary'].apply(clean_text)

# Step 3: TF-IDF vectorization with tuned parameters
vectorizer = TfidfVectorizer(stop_words='english', max_df=0.9, min_df=5, max_features=1000)
X_tfidf = vectorizer.fit_transform(df['job_summary_clean'])

# Step 4: KMeans clustering
kmeans = KMeans(n_clusters=4, random_state=42)
df['summary_cluster'] = kmeans.fit_predict(X_tfidf)

# Step 5: View sample summaries from each cluster
for i in range(4):
    print(f"\nCluster {i} sample summaries:")
    print(df[df['summary_cluster'] == i]['job_summary'].head(2).values)
```

```
Cluster 0 sample summaries:
['Overview\nThe Data Engineer develops, implements and documents data systems that provide the technical solutions to meet specifications and busine
 "Who is Recruiting from Scratch :\nRecruiting from Scratch is a premier talent firm that focuses on placing the best product managers, software, an

Cluster 1 sample summaries:
["Join a recognized leader in promoting health and well-being! We are currently seeking a skilled and experienced professional to fill a remote posi
 "For database engineer we have 2 roles.\nFor 1 we need SQL server and Sybase\nfor 2nd we need oracle and DB2.\nFor primary vs secondary out of both

Cluster 2 sample summaries:
["This is for a client of Recruiting from Scratch.\nWho is Recruiting from Scratch:\nRecruiting from Scratch is a premier talent firm that focuses o
 'This is for a client of Recruiting from Scratch.\nWho is Recruiting from Scratch:\nRecruiting from Scratch is a premier talent firm that focuses o

Cluster 3 sample summaries:
["Overview\nWe are seeking a talented Azure Cloud and Fabric Data Platform Engineer to play a pivotal role in ensuring the availability, performance
 'An Artificial Intelligence/National Defense start-up is searching for Data Scientists to join their team. You would be solving complex problems by
```

Interpreting Job Clusters via Top Keywords (TF-IDF + KMeans)

Approach

- TF-IDF Vectorization: Converted job summaries into a numerical matrix emphasizing unique, high-impact words.
- KMeans Clustering (k=4): Grouped job summaries by keyword similarity.
- Keyword Extraction: For each cluster, top 10 keywords were retrieved from cluster centroids to characterize the cluster.

```python
import numpy as np

# Get feature names (words)
terms = vectorizer.get_feature_names_out()

# For each cluster, get top keywords
def print_top_keywords_per_cluster(kmeans_model, terms, n_terms=10):
    for i, cluster_center in enumerate(kmeans_model.cluster_centers_):
        top_indices = cluster_center.argsort()[-n_terms:][::-1]  # top n keywords
        top_terms = [terms[ind] for ind in top_indices]
        print(f"\nCluster {i} top keywords:")
        print(", ".join(top_terms))

print_top_keywords_per_cluster(kmeans, terms)
```

Cluster 0 top keywords:
data, experience, business, azure, work, pipelines, engineering, team, solutions, skills

Cluster 1 top keywords:
database, experience, data, oracle, sql, databases, systems, work, server, management

Cluster 2 top keywords:
data, experience, work, spark, years, client, engineer, skills, etl, sql

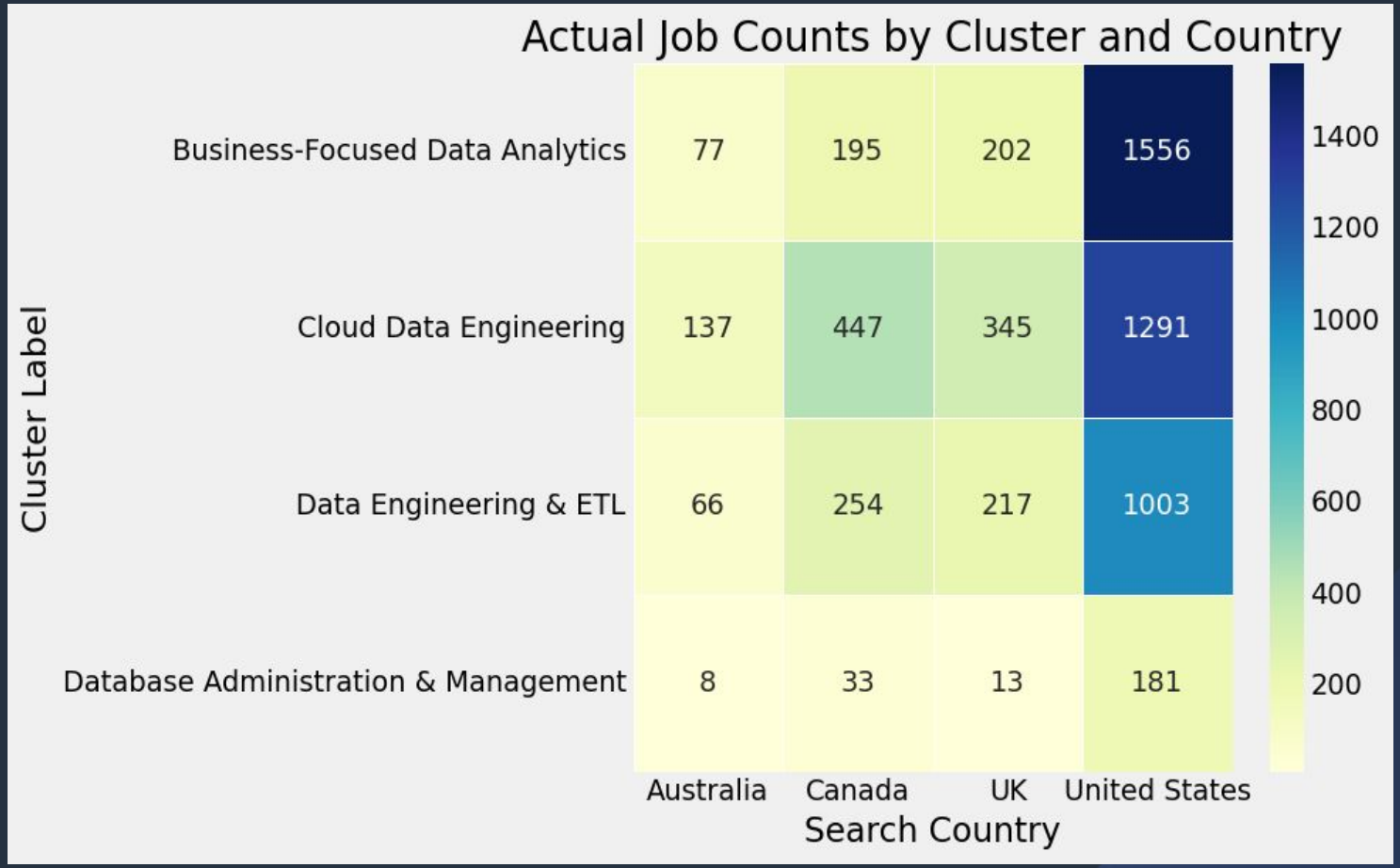
Cluster 3 top keywords:
data, experience, work, team, business, status, skills, company, support, development

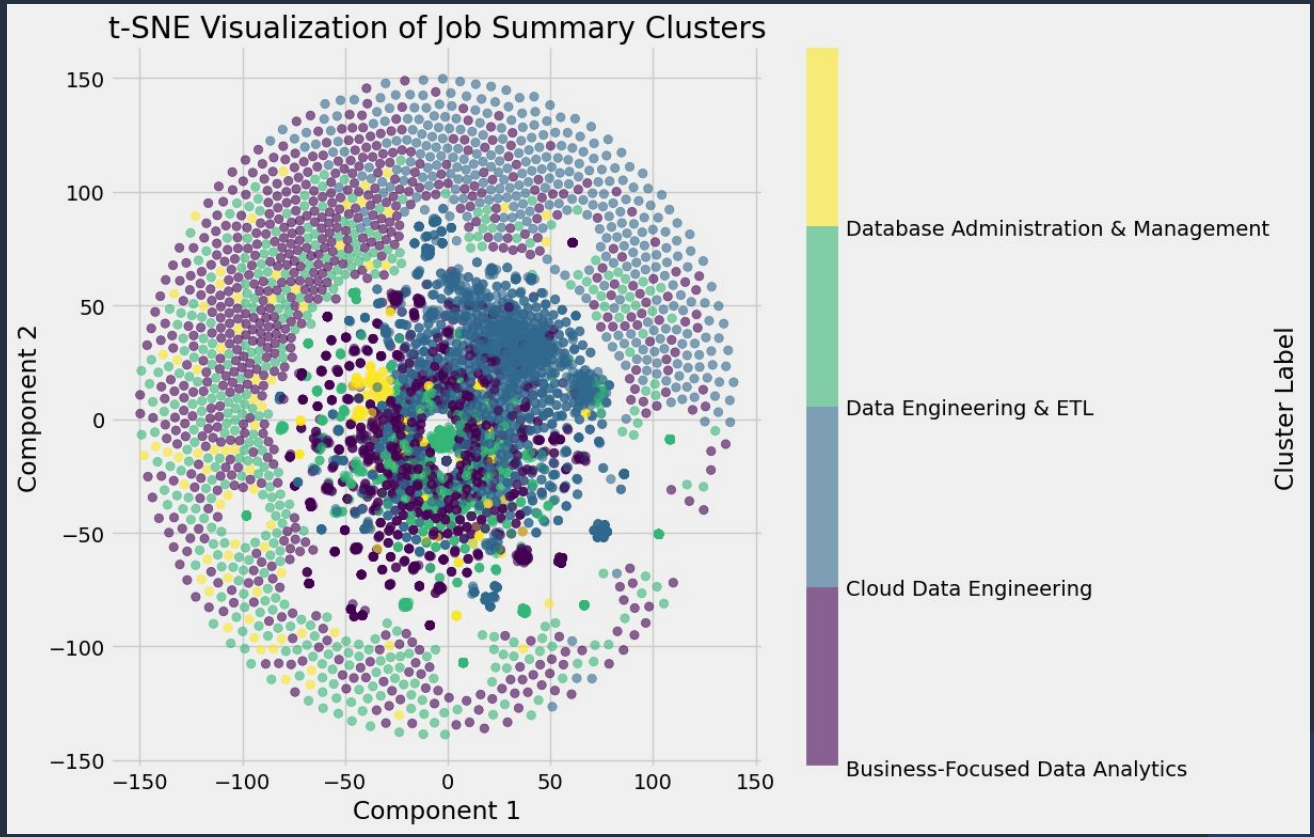
```
cluster_labels = {
    0: 'Cloud Data Engineering',
    1: 'Database Administration & Management',
    2: 'Data Engineering & ETL',
    3: 'Business-Focused Data Analytics'
}

df['cluster_label'] = df['summary_cluster'].map(cluster_labels)
```

```
df.groupby('cluster_label')['job_type'].value_counts(normalize=True)
df.groupby('cluster_label')['search_country'].value_counts(normalize=True)
```

cluster_label	search_country	proportion
Business-Focused Data Analytics	United States	0.766502
	United Kingdom	0.099507
	Canada	0.096059
	Australia	0.037931
Cloud Data Engineering	United States	0.581532
	Canada	0.201351
	United Kingdom	0.155405
	Australia	0.061712
Data Engineering & ETL	United States	0.651299
	Canada	0.164935
	United Kingdom	0.140909
	Australia	0.042857
Database Administration & Management	United States	0.770213
	Canada	0.140426
	United Kingdom	0.055319
	Australia	0.034043

	cluster_label	search_country	proportion	total_jobs	actual_jobs
0	Business-Focused Data Analytics	United States	0.766502	2030	1556
1	Business-Focused Data Analytics	United Kingdom	0.099507	2030	202
2	Business-Focused Data Analytics	Canada	0.096059	2030	195
3	Business-Focused Data Analytics	Australia	0.037931	2030	77
4	Cloud Data Engineering	United States	0.581532	2220	1291
5	Cloud Data Engineering	Canada	0.201351	2220	447
6	Cloud Data Engineering	United Kingdom	0.155405	2220	345
7	Cloud Data Engineering	Australia	0.061712	2220	137
8	Data Engineering & ETL	United States	0.651299	1540	1003
9	Data Engineering & ETL	Canada	0.164935	1540	254
10	Data Engineering & ETL	United Kingdom	0.140909	1540	217
11	Data Engineering & ETL	Australia	0.042857	1540	66
12	Database Administration & Management	United States	0.770213	235	181
13	Database Administration & Management	Canada	0.140426	235	33
14	Database Administration & Management	United Kingdom	0.055319	235	13
15	Database Administration & Management	Australia	0.034043	235	8


Actual Job Counts by Cluster and Country
Cluster Label
Business-Focused Data Analytics
Cloud Data Engineering
Data Engineering & ETL
Database Administration & Management
77
195
202
1556
137
447
345
1291
66
254
217
1003
8
33
13
181
1400
1200
1000
800
600
400
200
Australia
Canada
UK
United States
Search Country




```
[78] from sklearn.metrics import silhouette_score

     # Calculate silhouette score using TF-IDF vectors and assigned labels
     score = silhouette_score(X_tfidf, df['summary_cluster'])

     print(f"Silhouette Score for TF-IDF + KMeans (k=4): {score:.3f}")
```

```
Silhouette Score for TF-IDF + KMeans (k=4): 0.034
```

Conclusions and Implications

	cluster_label	search_country	proportion	total_jobs	actual_jobs
0	Business-Focused Data Analytics	United States	0.766502	2030	1556
1	Business-Focused Data Analytics	United Kingdom	0.099507	2030	202
2	Business-Focused Data Analytics	Canada	0.096059	2030	195
3	Business-Focused Data Analytics	Australia	0.037931	2030	77
4	Cloud Data Engineering	United States	0.581532	2220	1291
5	Cloud Data Engineering	Canada	0.201351	2220	447
6	Cloud Data Engineering	United Kingdom	0.155405	2220	345
7	Cloud Data Engineering	Australia	0.061712	2220	137
8	Data Engineering & ETL	United States	0.651299	1540	1003
9	Data Engineering & ETL	Canada	0.164935	1540	254
10	Data Engineering & ETL	United Kingdom	0.140909	1540	217
11	Data Engineering & ETL	Australia	0.042857	1540	66
12	Database Administration & Management	United States	0.770213	235	181
13	Database Administration & Management	Canada	0.140426	235	33
14	Database Administration & Management	United Kingdom	0.055319	235	13
15	Database Administration & Management	Australia	0.034043	235	8