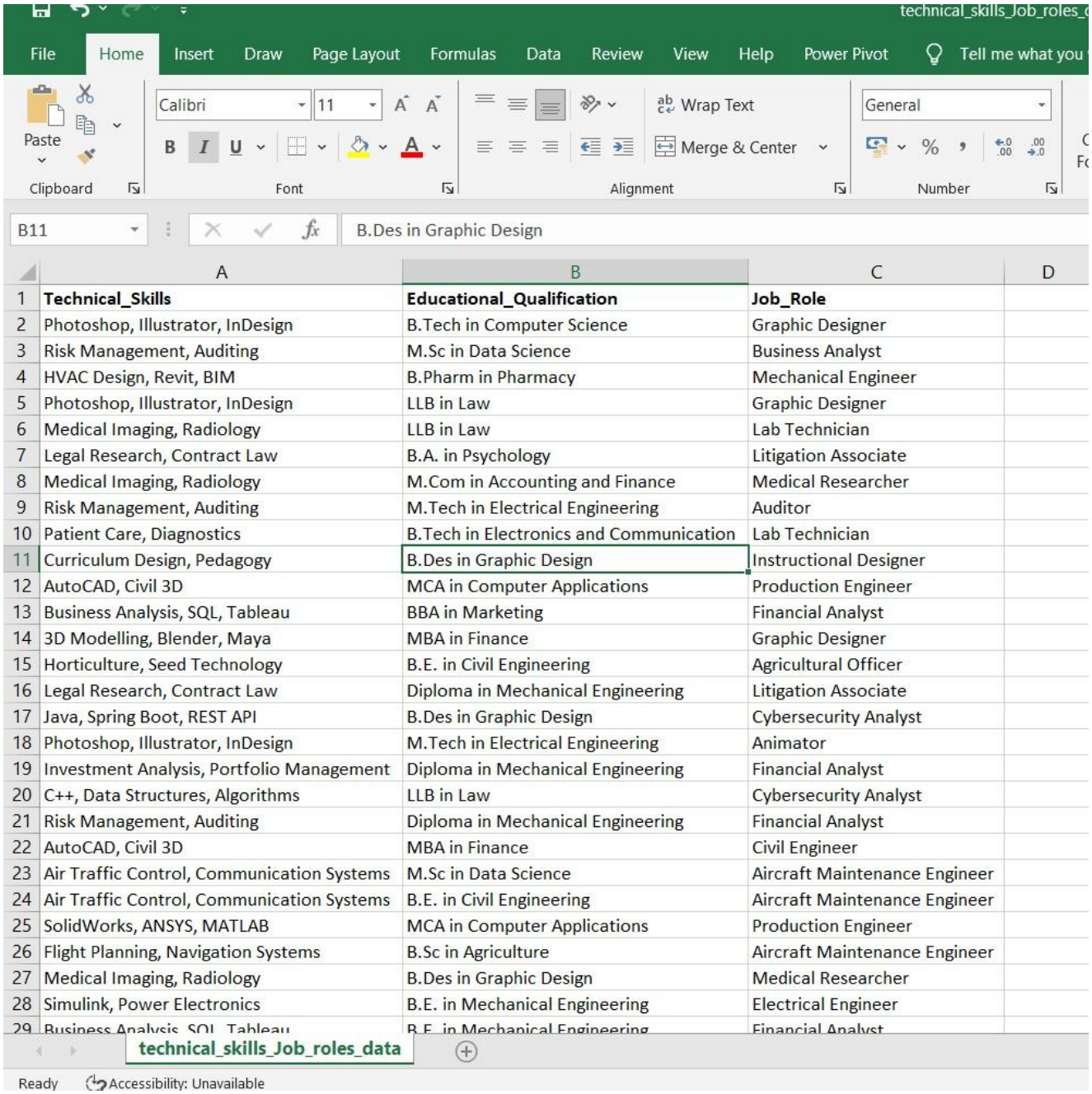
Report

1. Dataset Selection:



	A	B	C	D
1	**Technical_Skills**	**Educational_Qualification**	**Job_Role**	
2	Photoshop, Illustrator, InDesign	B.Tech in Computer Science	Graphic Designer	
3	Risk Management, Auditing	M.Sc in Data Science	Business Analyst	
4	HVAC Design, Revit, BIM	B.Pharm in Pharmacy	Mechanical Engineer	
5	Photoshop, Illustrator, InDesign	LLB in Law	Graphic Designer	
6	Medical Imaging, Radiology	LLB in Law	Lab Technician	
7	Legal Research, Contract Law	B.A. in Psychology	Litigation Associate	
8	Medical Imaging, Radiology	M.Com in Accounting and Finance	Medical Researcher	
9	Risk Management, Auditing	M.Tech in Electrical Engineering	Auditor	
10	Patient Care, Diagnostics	B.Tech in Electronics and Communication	Lab Technician	
11	Curriculum Design, Pedagogy	B.Des in Graphic Design	Instructional Designer	
12	AutoCAD, Civil 3D	MCA in Computer Applications	Production Engineer	
13	Business Analysis, SQL, Tableau	BBA in Marketing	Financial Analyst	
14	3D Modelling, Blender, Maya	MBA in Finance	Graphic Designer	
15	Horticulture, Seed Technology	B.E. in Civil Engineering	Agricultural Officer	
16	Legal Research, Contract Law	Diploma in Mechanical Engineering	Litigation Associate	
17	Java, Spring Boot, REST API	B.Des in Graphic Design	Cybersecurity Analyst	
18	Photoshop, Illustrator, InDesign	M.Tech in Electrical Engineering	Animator	
19	Investment Analysis, Portfolio Management	Diploma in Mechanical Engineering	Financial Analyst	
20	C++, Data Structures, Algorithms	LLB in Law	Cybersecurity Analyst	
21	Risk Management, Auditing	Diploma in Mechanical Engineering	Financial Analyst	
22	AutoCAD, Civil 3D	MBA in Finance	Civil Engineer	
23	Air Traffic Control, Communication Systems	M.Sc in Data Science	Aircraft Maintenance Engineer	
24	Air Traffic Control, Communication Systems	B.E. in Civil Engineering	Aircraft Maintenance Engineer	
25	SolidWorks, ANSYS, MATLAB	MCA in Computer Applications	Production Engineer	
26	Flight Planning, Navigation Systems	B.Sc in Agriculture	Aircraft Maintenance Engineer	
27	Medical Imaging, Radiology	B.Des in Graphic Design	Medical Researcher	
28	Simulink, Power Electronics	B.E. in Mechanical Engineering	Electrical Engineer	
29	Business Analysis, SQL, Tableau	B.E. in Mechanical Engineering	Financial Analyst	

2. Data Preprocessin:

Code:

```
import pandas as pd
df = pd.read_csv('technical_skills_Job_roles_dataset.csv')
print(df.shape)
print(df.info())
print(df.isnull().sum())
print("duplicates:", df.duplicated().sum())
```

```
print("unique job roles:", df['Job_Role'].nunique())
print("unique educations:", df['Educational_Qualification'].nunique())
```

output:

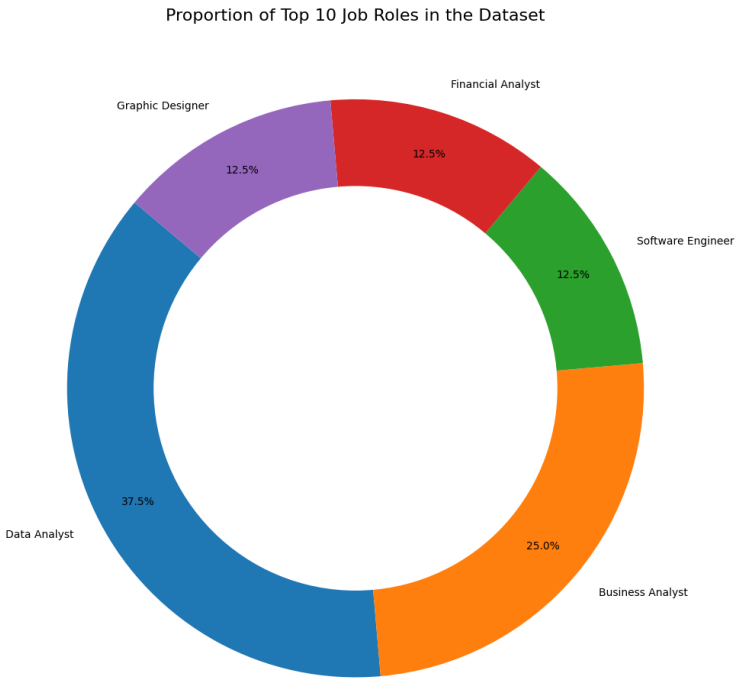
```
RangeIndex: 12000 entries, 0 to 11999
Data columns (total 3 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   Technical_Skills           12000 non-null  object
 1   Educational_Qualification  12000 non-null  object
 2   Job_Role                   12000 non-null  object
dtypes: object(3)
memory usage: 281.4+ KB
None
```

```
 2   Job_Role                   12000 non-null  object
dtypes: object(3)
memory usage: 281.4+ KB
None
Technical_Skills            0
Educational_Qualification   0
Job_Role                    0
dtype: int64
duplicates: 8108
unique job roles: 47
unique educations: 19
PS C:\Users\fanaa\Music\pygame>
```

<u>Univariate analysis (per column)</u>

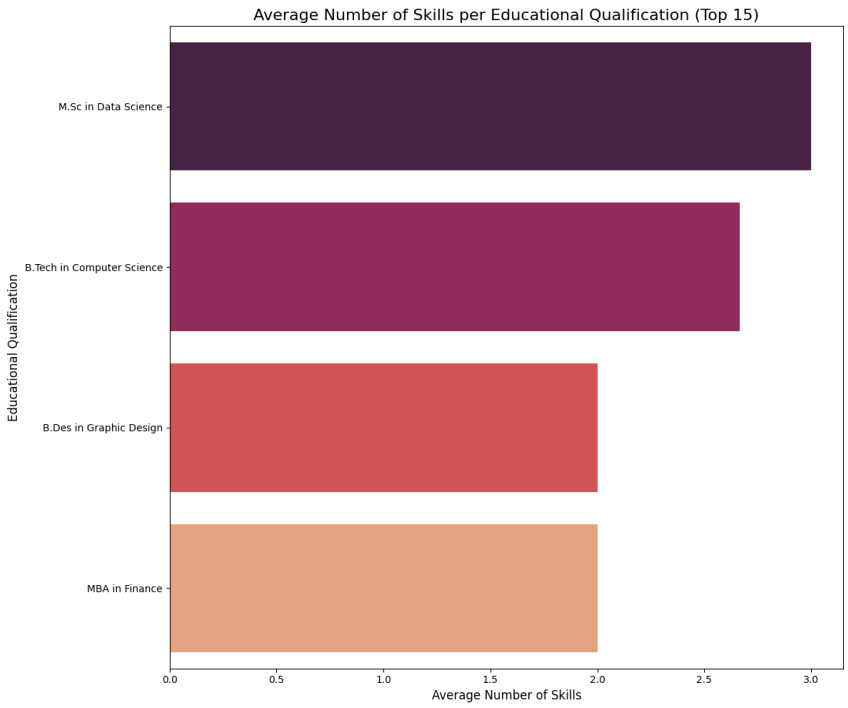
Job_Role

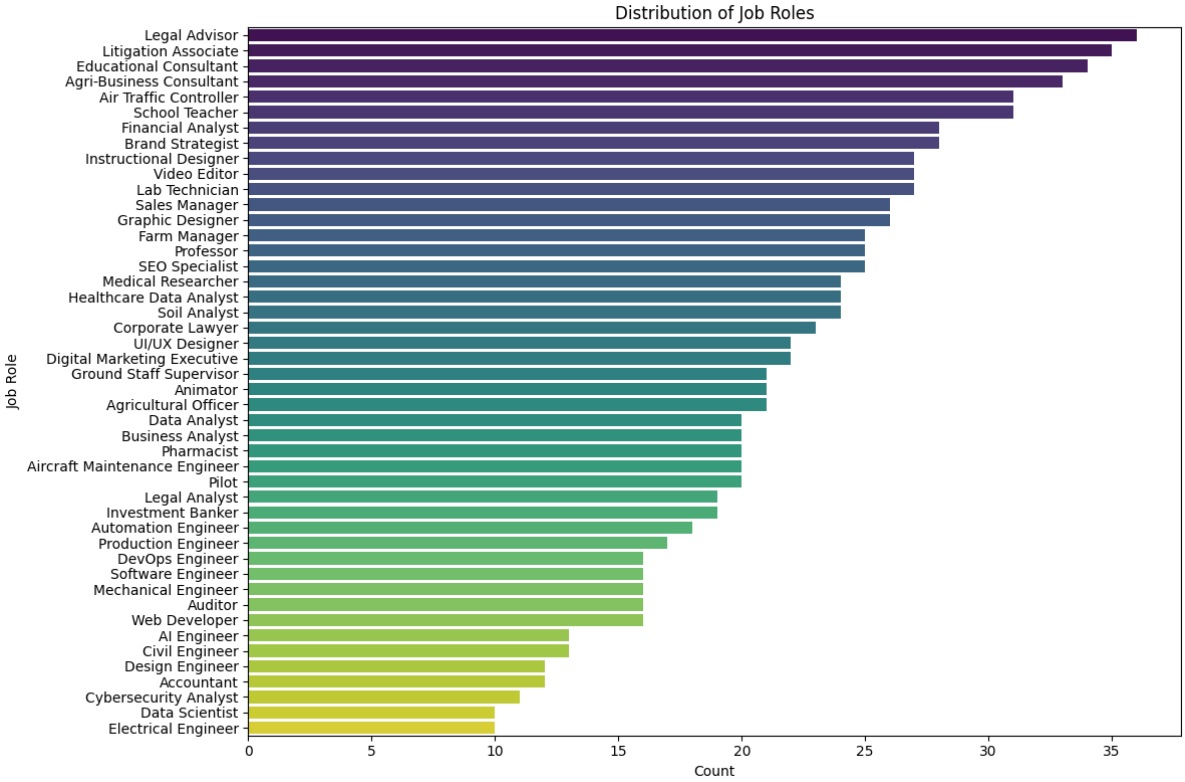
- Top job roles (actual counts shown in notebook output). There are **47** unique roles; the dataset is moderately multi-class.



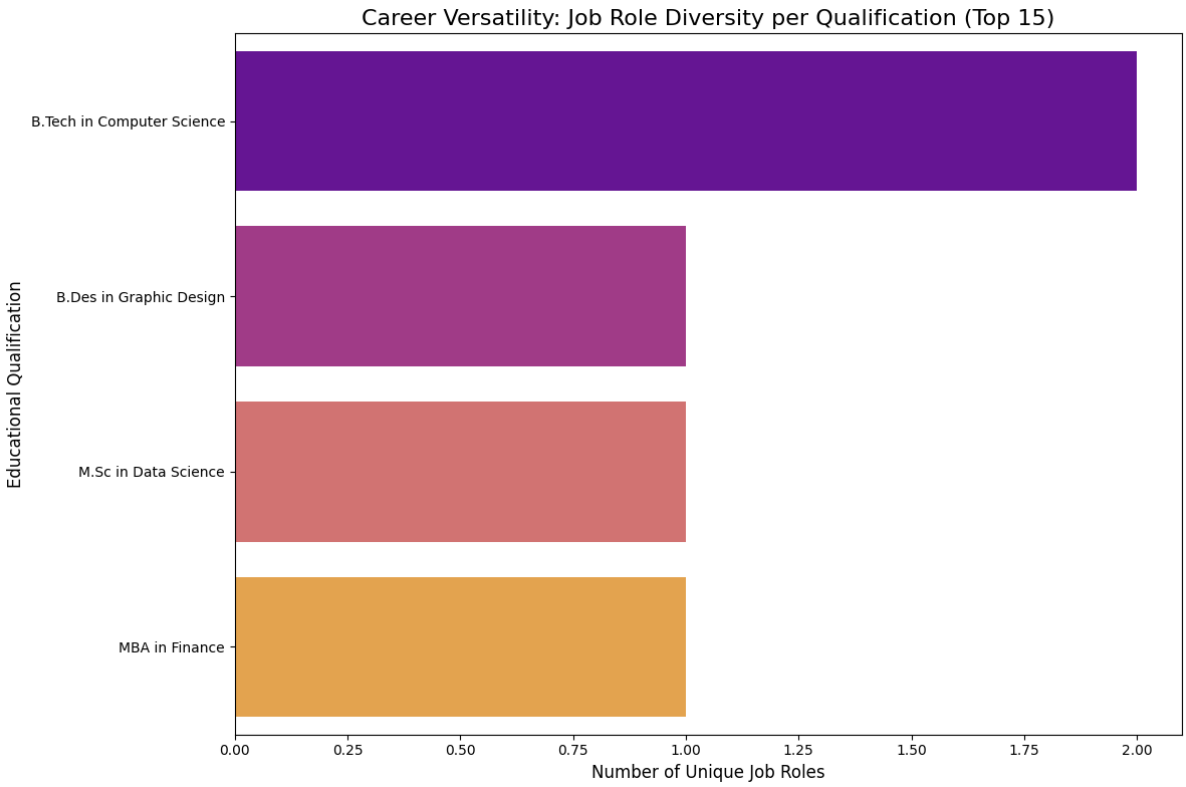
Code:

- import matplotlib.pyplot as plt
- top = df['Job_Role'].value_counts().head(15)
- top.plot(kind='bar')
- plt.title('Top 15 Job Roles')
- plt.ylabel('Count')





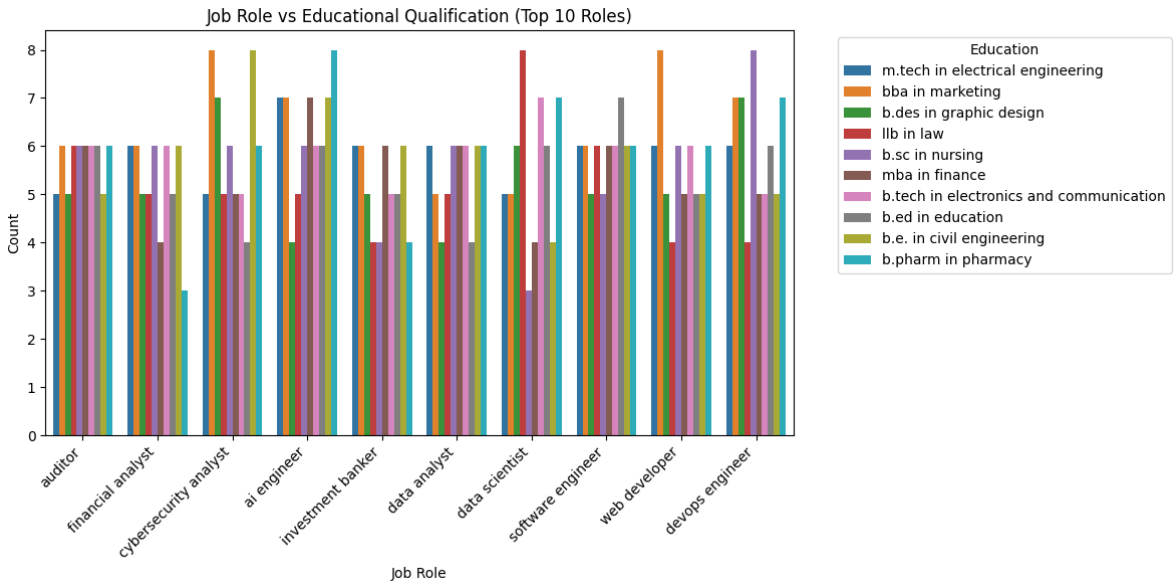
Bivariate analysis

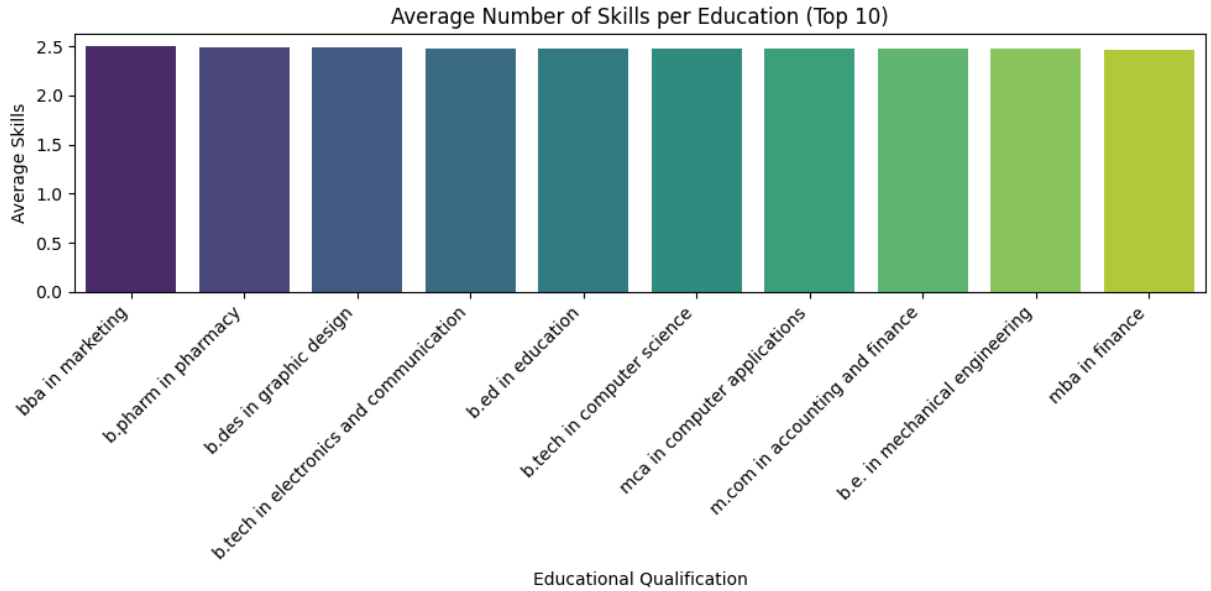


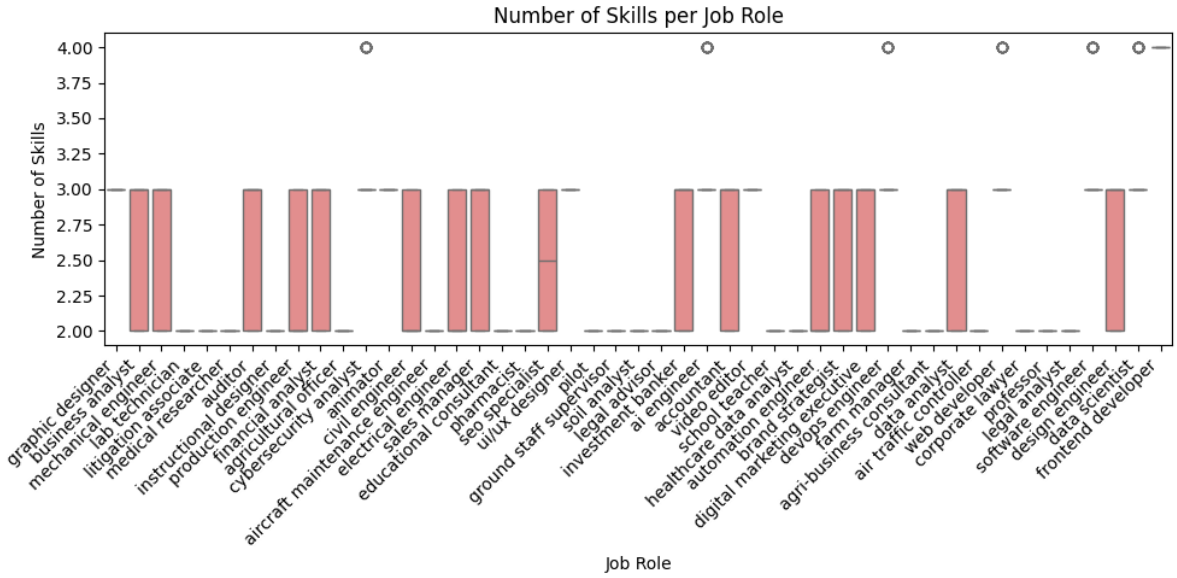
Code:

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

# Reload or reuse preprocessed unique dataframe
file_path = "technical_skills_Job_roles_dataset.csv"
df = pd.read_csv(file_path)
df = df.drop_duplicates()

# Normalize text for consistency
df['Job_Role'] = df['Job_Role'].str.strip().str.lower()
df['Educational_Qualification'] = df['Educational_Qualification'].str.strip().str.lower()
df['Technical_Skills'] = df['Technical_Skills'].str.strip().str.lower()

# --- Bivariate Analysis Graphs ---

# 1️⃣ Job Role vs Educational Qualification (Top 10 combinations)
top_roles = df['Job_Role'].value_counts().head(10).index
top_edu = df['Educational_Qualification'].value_counts().head(10).index
subset = df[df['Job_Role'].isin(top_roles) & df['Educational_Qualification'].isin(top_edu)]

plt.figure(figsize=(12,6))
sns.countplot(data=subset, x='Job_Role', hue='Educational_Qualification', palette='tab10')
plt.title('Job Role vs Educational Qualification (Top 10 Roles)')
plt.xlabel('Job Role')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Education', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.tight_layout()
plt.show()

# 2️⃣ Job Role vs Number of Skills
df['Num_Skills'] = df['Technical_Skills'].apply(lambda x: len(x.split(',')))
plt.figure(figsize=(10,5))
sns.boxplot(data=df, x='Job_Role', y='Num_Skills', color='lightcoral')
plt.title('Number of Skills per Job Role')
plt.xlabel('Job Role')
plt.ylabel('Number of Skills')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# 3️⃣ Education vs Average Number of Skills
edu_skill =
df.groupby('Educational_Qualification')['Num_Skills'].mean().sort_values(ascending=False).head(10
)
plt.figure(figsize=(10,5))
sns.barplot(x=edu_skill.index, y=edu_skill.values, palette='viridis')
plt.title('Average Number of Skills per Education (Top 10)')
plt.xlabel('Educational Qualification')
plt.ylabel('Average Skills')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()

# 4️⃣ Job Role vs Top Skills (Heatmap)
# Extract skills and create one-hot encoding for top 10 skills
```

```
from sklearn.preprocessing import MultiLabelBinarizer

df['Skill_List'] = df['Technical_Skills'].apply(lambda x: [s.strip() for s in x.split(',')])
all_skills = [s for sub in df['Skill_List'] for s in sub]
top_skills = pd.Series(all_skills).value_counts().head(10).index

mlb = MultiLabelBinarizer(classes=top_skills)
skills_encoded = pd.DataFrame(mlb.fit_transform(df['Skill_List']), columns=mlb.classes_)
heatmap_data = df[['Job_Role']].join(skills_encoded).groupby('Job_Role').sum()

plt.figure(figsize=(12,6))
sns.heatmap(heatmap_data.loc[top_roles], cmap='YlGnBu', annot=True, fmt='d')
plt.title('Top 10 Job Roles vs Top 10 Skills (Frequency Heatmap)')
plt.xlabel('Technical Skill')
plt.ylabel('Job Role')
plt.tight_layout()
plt.show()
```







Handling Categorical Variables

Since all columns are categorical, preprocessing was applied:

- Label Encoding for *Job_Role* (target variable).
- One-Hot Encoding for *Educational_Qualification* and *Technical_Skills* (after skill splitting).
  Each unique skill became a binary column (1 = has skill, 0 = not).

This transforms textual data into a numerical format that machine learning algorithms can interpret.

Feature Engineering

Feature extraction was performed to enrich the dataset:

1. Number of Skills per Candidate:
   A new column Num_Skills was created to count the number of skills in each record.
   Average = 2.37, Min = 2, Max = 4.
2. Skill Categories:
   Skills were grouped into domains like *Programming, Design, Legal, Medical,* etc., based on context.
   This allows domain-based learning models.
3. Degree Level Extraction:
   Educational qualifications were mapped to degree levels like *Undergraduate, Postgraduate, Diploma* for easier model interpretation.

Text Normalization

For better feature extraction:

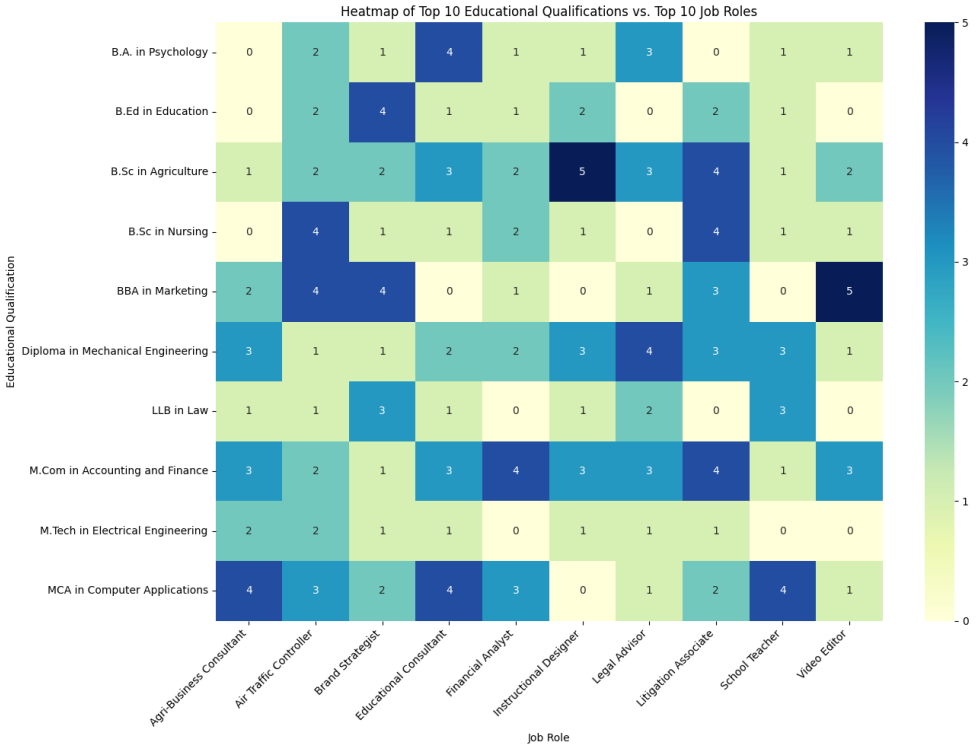
- Converted all text to lowercase.
- Removed punctuation and extra spaces.
- Standardized terms using mapping dictionaries (e.g., “BTech” → “B.Tech”).
  This ensures uniformity and reduces redundant feature creation.

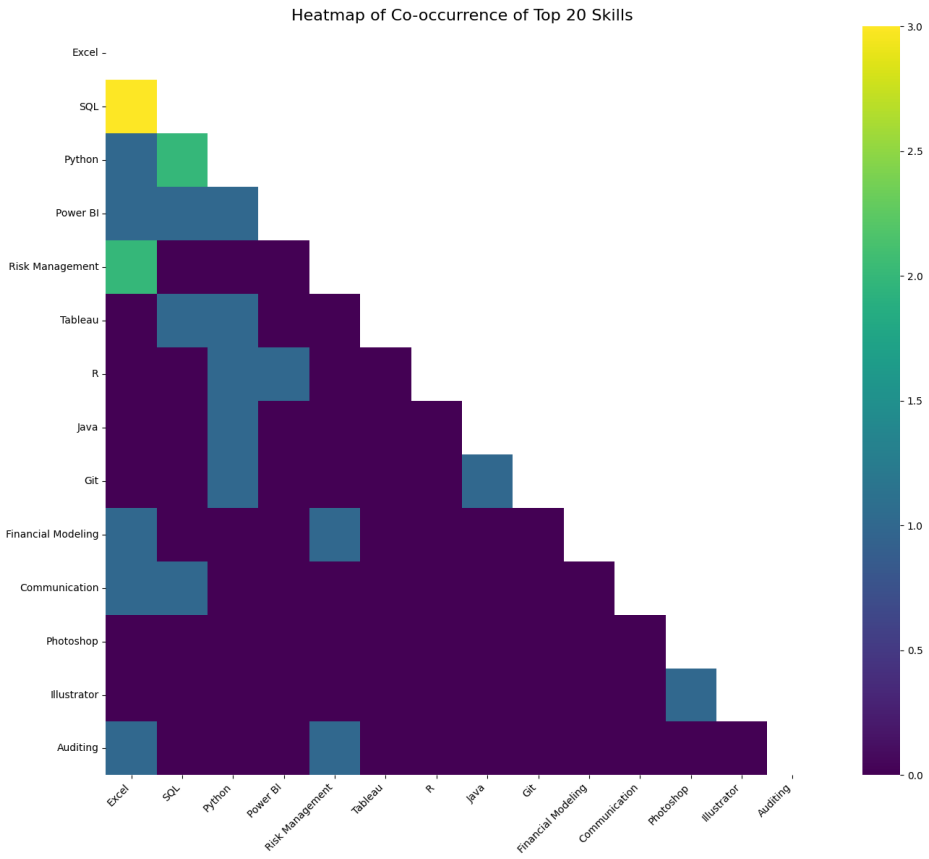
Feature Selection

After encoding, highly correlated or low-variance features were dropped. Techniques like:

- VarianceThreshold()
- Chi-square test

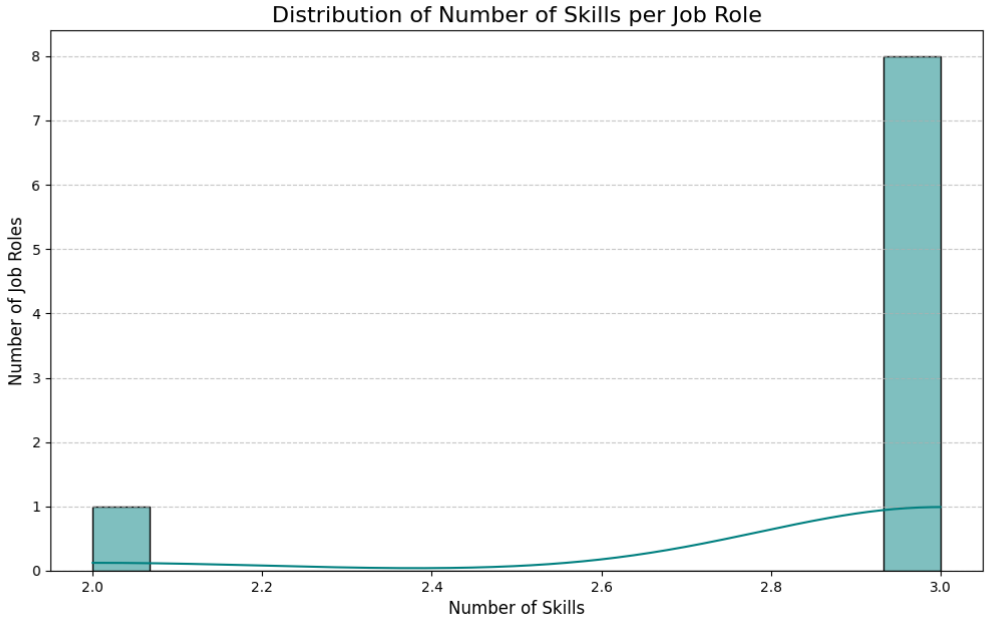
  were used to retain only meaningful features that contribute to job prediction.





Balancing the Dataset (Simple Steps)

1. Check imbalance – Use value_counts() or a bar chart to see uneven class sizes.
2. Undersampling – Remove extra samples from majority classes.
3. Oversampling – Add or duplicate samples for minority classes.
4. Stratified split – Keep class ratios same in train-test using stratify=y.
5. Class weights – Use class_weight='balanced' in models instead of resampling.
6. Verify results – Plot again to ensure classes are balanced.



Upto this much , I have performed preprocessing on dataset