In [5]:
```
#importing relevent libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy import stats as st
```

In [2]:
```
df = pd.read_csv("C:\\Users\\91939\\Downloads\\data.xlsx - Sheet1.csv")
```

In [3]:
```
df.head()
```

Out[3]:

	Unnamed: 0	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DOB	10per
0	train	203097	420000.0	6/1/12 0:00	present	senior quality engineer	Bangalore	f	2/19/90 0:00	
1	train	579905	500000.0	9/1/13 0:00	present	assistant manager	Indore	m	10/4/89 0:00	
2	train	810601	325000.0	6/1/14 0:00	present	systems engineer	Chennai	f	8/3/92 0:00	
3	train	267447	1100000.0	7/1/11 0:00	present	senior software engineer	Gurgaon	m	12/5/89 0:00	
4	train	343523	200000.0	3/1/14 0:00	3/1/15 0:00	get	Manesar	m	2/27/91 0:00	

5 rows × 39 columns

In [6]:
```
#shape of the dataset
df.shape
```

Out[6]:
```
(3998, 39)
```

In [7]:
```
#size of the dataset
df.size
```

Out[7]:
```
155922
```

In [9]:
```
#removing the unknown column
df.drop("Unnamed: 0",axis = 1,inplace = True)
```

In [10]:
```
df.head()
```

Out[10]:

	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DOB	10percentage	
0	203097	420000.0	6/1/12 0:00	present	senior quality engineer	Bangalore	f	2/19/90 0:00	84.3	of edu
1	579905	500000.0	9/1/13 0:00	present	assistant manager	Indore	m	10/4/89 0:00	85.4	
2	810601	325000.0	6/1/14 0:00	present	systems engineer	Chennai	f	8/3/92 0:00	85.0	
3	267447	1100000.0	7/1/11 0:00	present	senior software engineer	Gurgaon	m	12/5/89 0:00	85.6	
4	343523	200000.0	3/1/14 0:00	3/1/15 0:00	get	Manesar	m	2/27/91 0:00	78.0	

5 rows × 38 columns

In [11]:
```
#checking the info of the dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3998 entries, 0 to 3997
Data columns (total 38 columns):
 #   Column                 Non-Null Count  Dtype  
---  ------                 --------------  -----  
 0   ID                     3998 non-null   int64  
 1   Salary                 3998 non-null   float64
 2   DOJ                    3998 non-null   object 
 3   DOL                    3998 non-null   object 
 4   Designation            3998 non-null   object 
 5   JobCity                3998 non-null   object 
 6   Gender                 3998 non-null   object 
 7   DOB                    3998 non-null   object 
 8   10percentage           3998 non-null   float64
 9   10board                3998 non-null   object 
 10  12graduation           3998 non-null   int64  
 11  12percentage           3998 non-null   float64
 12  12board                3998 non-null   object 
 13  CollegeID              3998 non-null   int64  
 14  CollegeTier            3998 non-null   int64  
 15  Degree                 3998 non-null   object 
 16  Specialization         3998 non-null   object 
 17  collegeGPA             3998 non-null   float64
 18  CollegeCityID          3998 non-null   int64  
 19  CollegeCityTier        3998 non-null   int64  
 20  CollegeState           3998 non-null   object 
 21  GraduationYear         3998 non-null   int64  
 22  English                3998 non-null   int64  
 23  Logical                3998 non-null   int64  
 24  Quant                  3998 non-null   int64  
 25  Domain                 3998 non-null   float64
 26  ComputerProgramming    3998 non-null   int64  
 27  ElectronicsAndSemicon  3998 non-null   int64  
 28  ComputerScience        3998 non-null   int64  
 29  MechanicalEngg         3998 non-null   int64  
 30  ElectricalEngg         3998 non-null   int64  
 31  TelecomEngg            3998 non-null   int64  
 32  CivilEngg              3998 non-null   int64  
 33  conscientiousness      3998 non-null   float64
 34  agreeableness          3998 non-null   float64
 35  extraversion           3998 non-null   float64
 36  nueroticism            3998 non-null   float64
 37  openess_to_experience  3998 non-null   float64
dtypes: float64(10), int64(17), object(11)
memory usage: 1.2+ MB
```

In [12]:
```
#columns
df.columns
```

Out[12]:
```
Index(['ID', 'Salary', 'DOJ', 'DOL', 'Designation', 'JobCity', 'Gender', 'DOB',
       '10percentage', '10board', '12graduation', '12percentage', '12board',
       'CollegeID', 'CollegeTier', 'Degree', 'Specialization', 'collegeGPA',
       'CollegeCityID', 'CollegeCityTier', 'CollegeState', 'GraduationYear',
       'English', 'Logical', 'Quant', 'Domain', 'ComputerProgramming',
       'ElectronicsAndSemicon', 'ComputerScience', 'MechanicalEngg',
       'ElectricalEngg', 'TelecomEngg', 'CivilEngg', 'conscientiousness',
       'agreeableness', 'extraversion', 'nueroticism',
       'openess_to_experience'],
      dtype='object')
```

In [13]:
```
#fixing the datatypes
df['DOJ'] = pd.to_datetime(df['DOJ'])
```

In [65]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3998 entries, 0 to 3997
Data columns (total 38 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   ID                     3998 non-null   int64
 1   Salary                 3998 non-null   float64
 2   DOJ                    3998 non-null   datetime64[ns]
 3   DOL                    3998 non-null   object
 4   Designation            3998 non-null   object
 5   JobCity                3998 non-null   object
 6   Gender                 3998 non-null   object
 7   DOB                    3998 non-null   object
 8   10percentage           3998 non-null   float64
 9   10board                3998 non-null   object
 10  12graduation           3998 non-null   int64
 11  12percentage           3998 non-null   float64
 12  12board                3998 non-null   object
 13  CollegeID              3998 non-null   int64
 14  CollegeTier            3998 non-null   int64
 15  Degree                 3998 non-null   object
 16  Specialization         3998 non-null   object
 17  collegeGPA             3998 non-null   float64
 18  CollegeCityID          3998 non-null   int64
 19  CollegeCityTier        3998 non-null   int64
 20  CollegeState           3998 non-null   object
 21  GraduationYear         3998 non-null   int64
 22  English                3998 non-null   int64
 23  Logical                3998 non-null   int64
 24  Quant                  3998 non-null   int64
 25  Domain                 3998 non-null   float64
 26  ComputerProgramming    3998 non-null   int64
 27  ElectronicsAndSemicon  3998 non-null   int64
 28  ComputerScience        3998 non-null   int64
 29  MechanicalEngg         3998 non-null   int64
 30  ElectricalEngg         3998 non-null   int64
 31  TelecomEngg            3998 non-null   int64
 32  CivilEngg              3998 non-null   int64
 33  conscientiousness      3998 non-null   float64
 34  agreeableness          3998 non-null   float64
 35  extraversion           3998 non-null   float64
 36  nueroticism            3998 non-null   float64
 37  openess_to_experience  3998 non-null   float64
dtypes: datetime64[ns](1), float64(10), int64(17), object(10)
memory usage: 1.2+ MB
```

In [66]:
```
#checking missing values
df.isna().sum()
```

Out[66]:
```
ID                        0
Salary                    0
DOJ                       0
DOL                       0
Designation               0
JobCity                   0
Gender                    0
DOB                       0
10percentage              0
10board                   0
12graduation              0
12percentage              0
12board                   0
CollegeID                 0
CollegeTier               0
Degree                    0
Specialization            0
collegeGPA                0
CollegeCityID             0
CollegeCityTier           0
CollegeState              0
GraduationYear            0
English                   0
Logical                   0
Quant                     0
Domain                    0
ComputerProgramming       0
ElectronicsAndSemicon     0
ComputerScience           0
MechanicalEngg            0
ElectricalEngg            0
TelecomEngg               0
CivilEngg                 0
conscientiousness         0
agreeableness             0
extraversion              0
nueroticism               0
openess_to_experience     0
dtype: int64
```

In [67]:
```
#checking duplicate values
df.duplicated().sum()
```

Out[67]:
```
0
```

Univariate Analysis

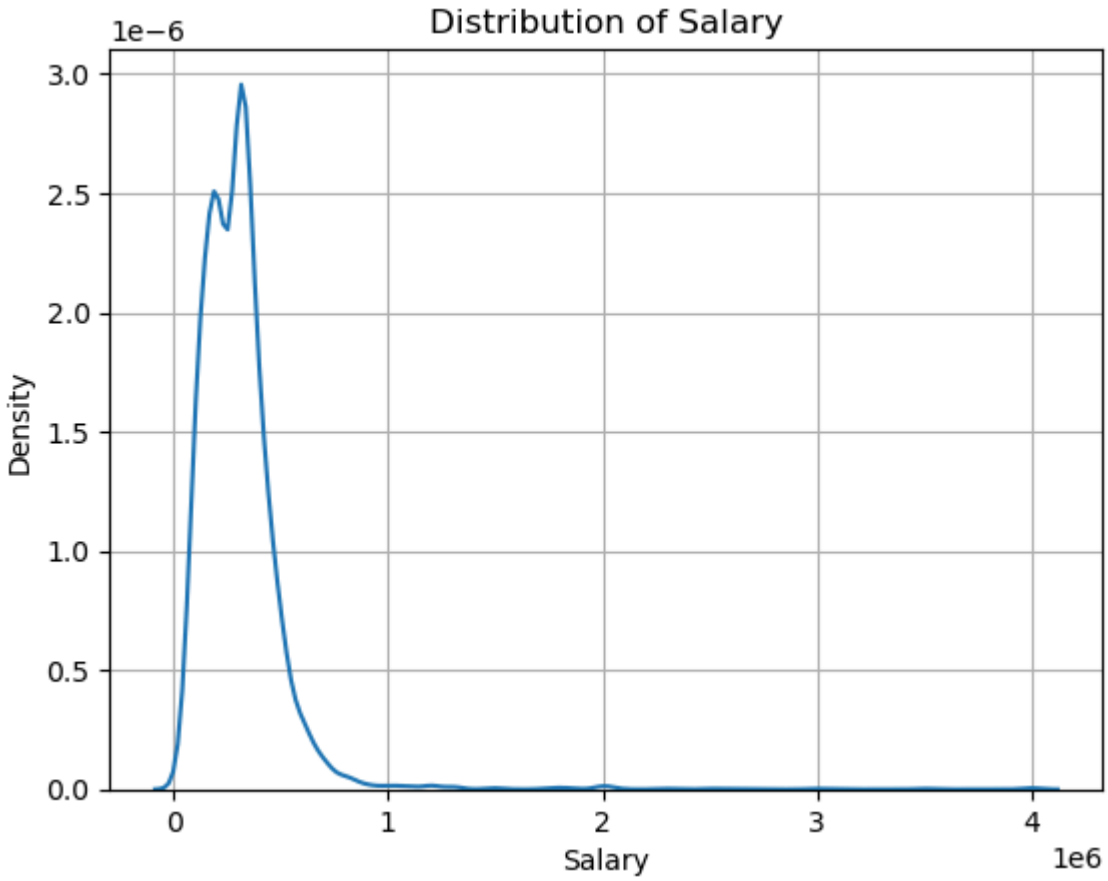
Analysing the data using single variable

In [68]:
```
#distribution of target variable(salary)
pd.DataFrame(df["Salary"].describe())
```

Out[68]:

	Salary
count	3.998000e+03
mean	3.076998e+05
std	2.127375e+05
min	3.500000e+04
25%	1.800000e+05
50%	3.000000e+05
75%	3.700000e+05
max	4.000000e+06

In [69]:
```
#plotting kde plot
sns.kdeplot(data = df["Salary"])
plt.grid()
plt.title("Distribution of Salary")
plt.show()
```



observations

In between 0 to 100000 the salaries are more compare to other salaries

After 300000 salaries are less

In [70]:
```
df["collegeGPA"].mean()
```

Out[70]:
```
71.48617058529268
```

In [77]:
```
pd.DataFrame(df["JobCity"].value_counts())
```

Out[77]:

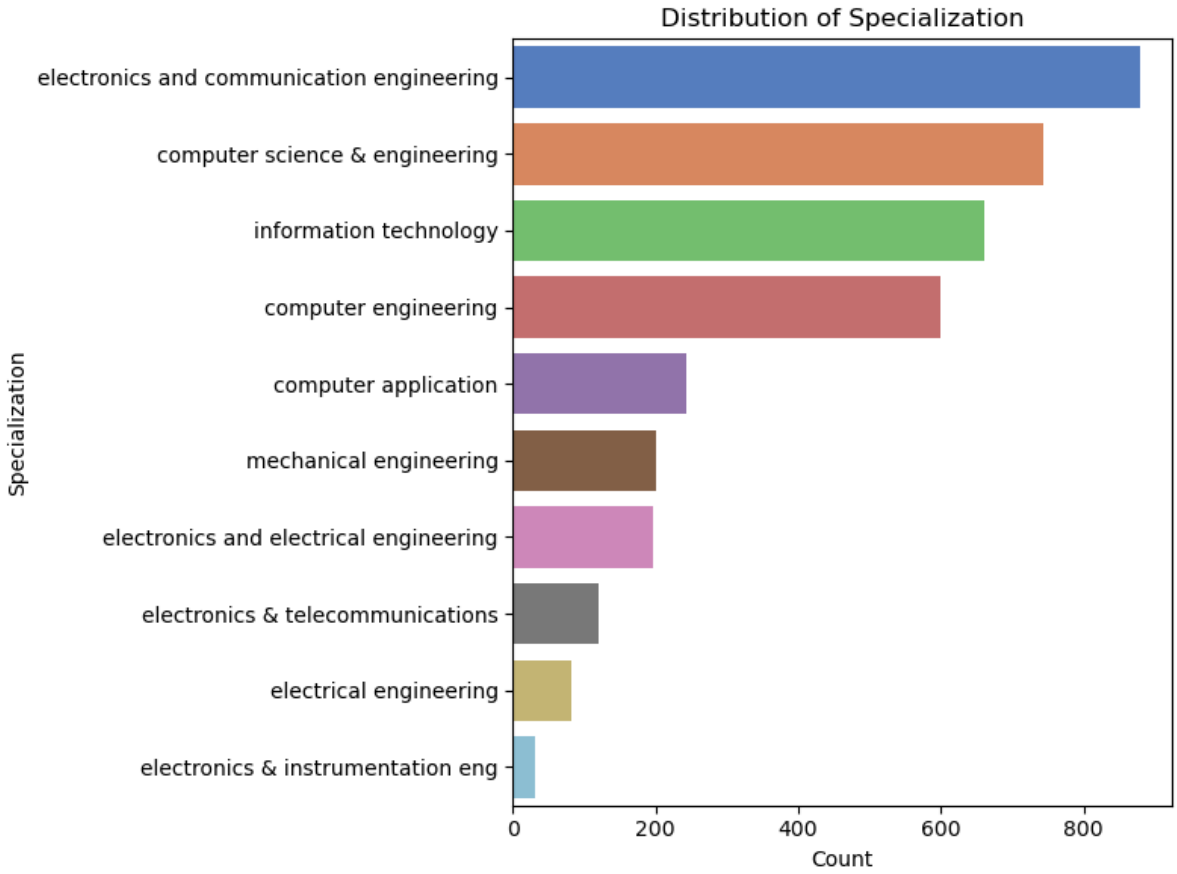
	JobCity
Bangalore	627
-1	461
Noida	368
Hyderabad	335
Pune	290
...	...
Tirunelvelli	1
Ernakulam	1
Nanded	1
Dharmapuri	1
Asifabadbanglore	1

339 rows × 1 columns

In [85]:
```
#finding which specialization is most common
specialization_counts=df["Specialization"].value_counts().head(10)
d1 = pd.DataFrame(specialization_counts)
d1.columns = ["Count"]
d1 = d1.reset_index()
d1.columns = ['Specialization','Count']
print(d1)
```

```
                            Specialization  Count
0  electronics and communication engineering    880
1              computer science & engineering    744
2                      information technology    660
3                        computer engineering    600
4                        computer application    244
5                      mechanical engineering    201
6      electronics and electrical engineering    196
7            electronics & telecommunications    121
8                      electrical engineering     82
9           electronics & instrumentation eng     32
```

In [89]:
```
#barplot
plt.figure(figsize=(8,6))
sns.barplot(y=d1['Specialization'],x=d1['Count'],palette="muted")
plt.title("Distribution of Specialization")
plt.xlabel("Count")
plt.ylabel("Specialization")
plt.tight_layout()
plt.show()
```



observations:

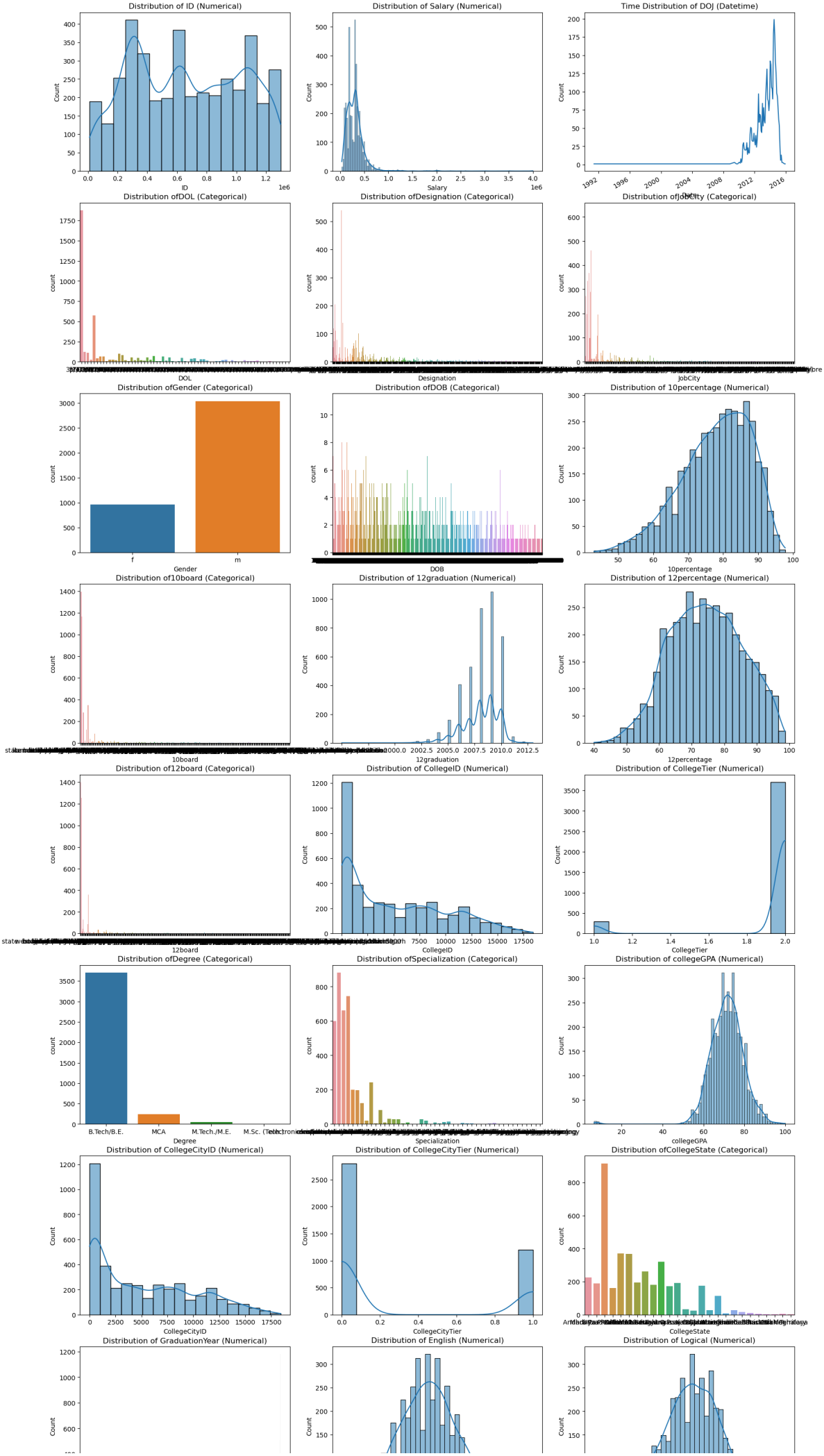
electronics and communication engineering students are more

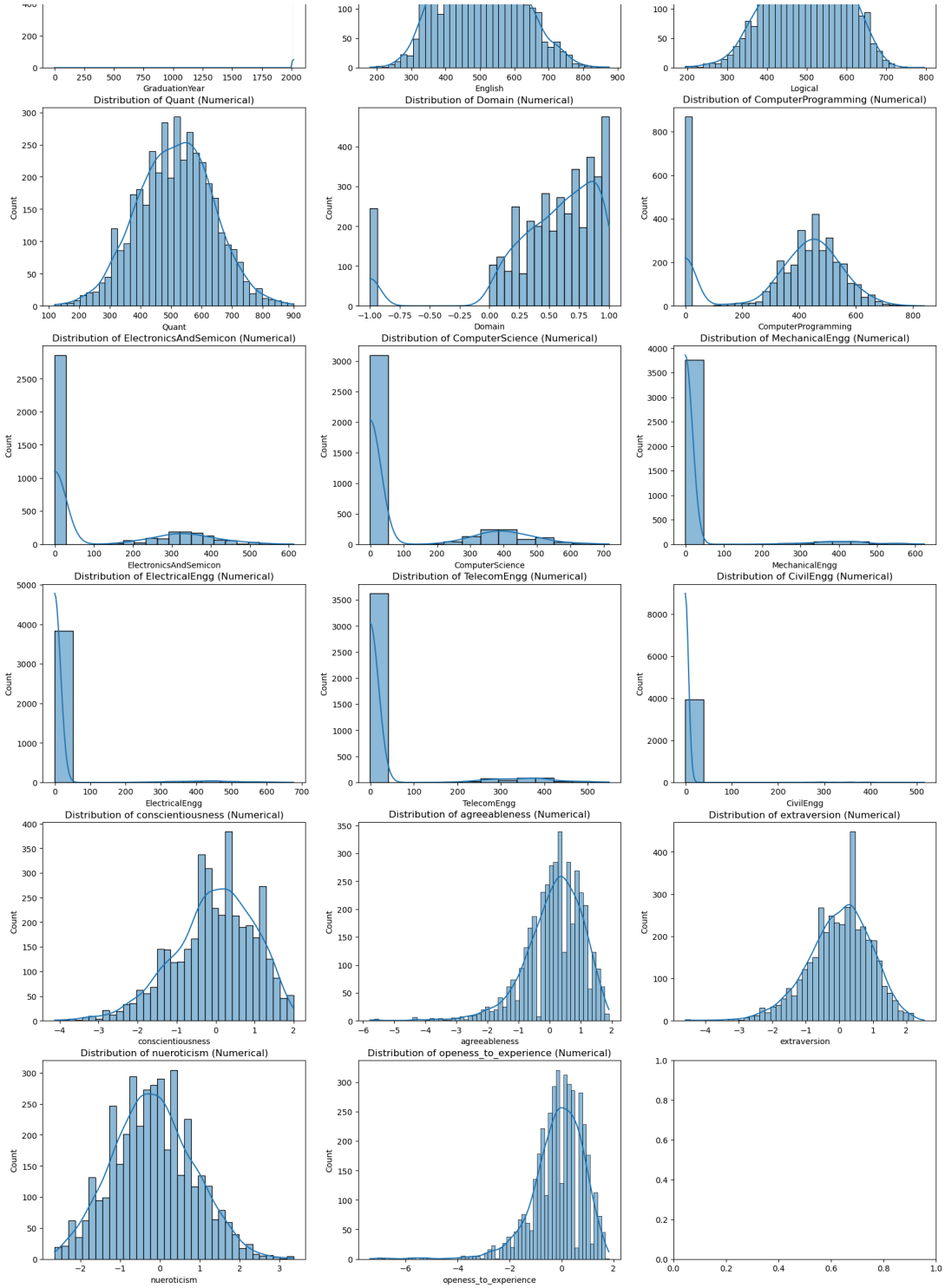
electronics & instrumentation eng are less

In [90]:
```
#plotting the graphs on columns
c_columns = len(df.columns)
#choosing 3columns per row
r_rows = int(np.ceil(c_columns / 3))
```

In [96]:
```
fig, axes = plt.subplots(r_rows, 3, figsize=(20, r_rows*6))
axes = axes.flatten()
#iterating over each column in the dataframe
for i, columns in enumerate(df.columns):
    #if the column is categorical
    if df[columns].dtype == "object" or df[columns].dtype == "category":
        sns.countplot(x=columns, data=df, ax=axes[i])
        axes[i].set_title(f'Distribution of{columns} (Categorical)')
        #if column is datetime
    elif pd.api.types.is_datetime64_any_dtype(df[columns]):
        df[columns] = pd.to_datetime(df[columns])
        df[columns].value_counts().sort_index().plot(ax=axes[i])
        axes[i].set_title(f'Time Distribution of {columns} (Datetime)')
        axes[i].set_xlabel("Date")
        axes[i].set_ylabel("Count")
        #if column is numeric
    elif pd.api.types.is_numeric_dtype(df[columns]):
        sns.histplot(df[columns], kde=True, ax=axes[i])
        axes[i].set_title(f'Distribution of {columns} (Numerical)')
    if i>=c_columns:
        axes[i].axes("off")
```

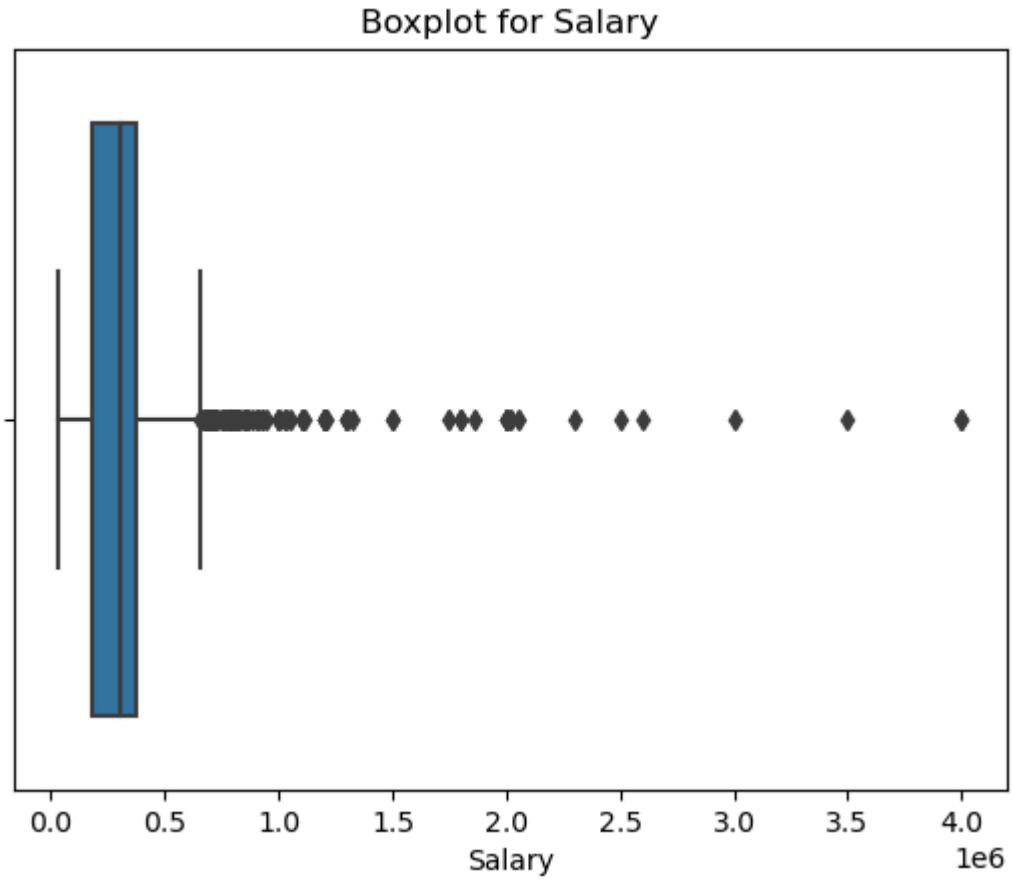
```
plt.show()
```

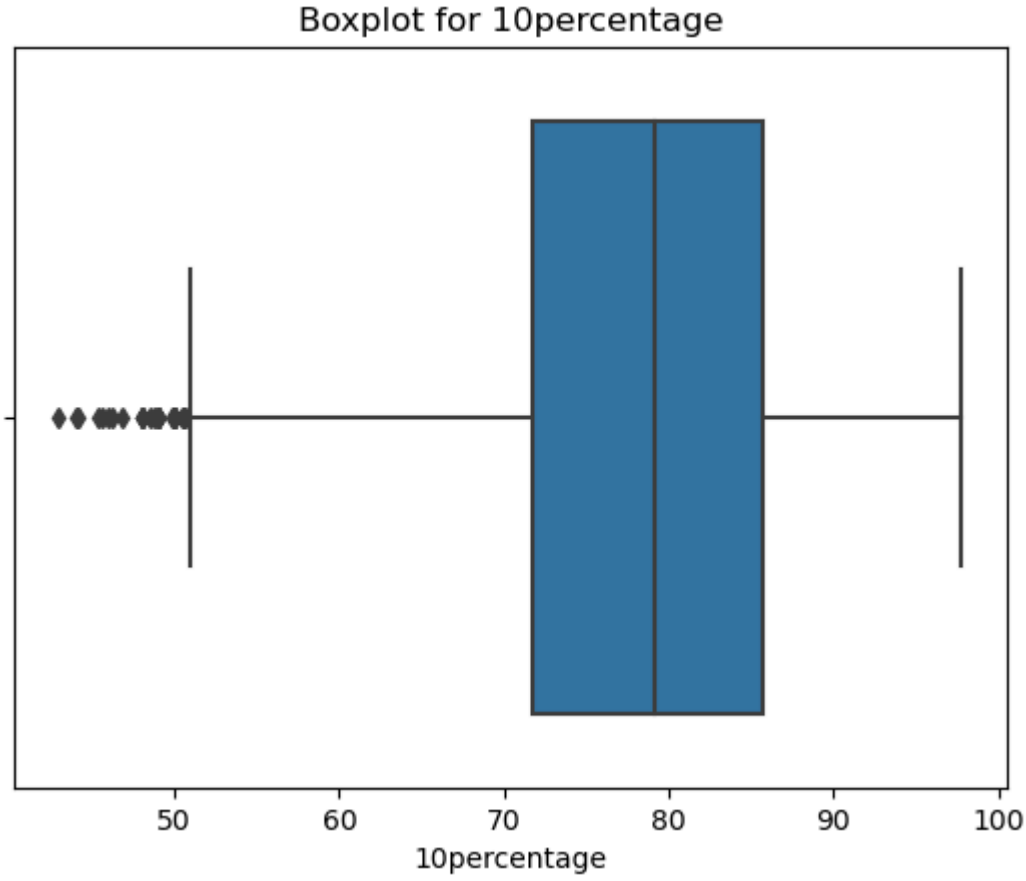

Distribution of ID (Numerical)
Distribution of Salary (Numerical)
Time Distribution of DOJ (Datetime)
Distribution ofDOL (Categorical)
Distribution ofDesignation (Categorical)
Distribution ofJobCity (Categorical)
Distribution ofGender (Categorical)
Distribution ofDOB (Categorical)
Distribution of 10percentage (Numerical)
Distribution of10board (Categorical)
Distribution of 12graduation (Numerical)
Distribution of 12percentage (Numerical)
Distribution of12board (Categorical)
Distribution of CollegeID (Numerical)
Distribution of CollegeTier (Numerical)
Distribution ofDegree (Categorical)
Distribution ofSpecialization (Categorical)
Distribution of collegeGPA (Numerical)
Distribution of CollegeCityID (Numerical)
Distribution of CollegeCityTier (Numerical)
Distribution ofCollegeState (Categorical)
Distribution of GraduationYear (Numerical)
Distribution of English (Numerical)
Distribution of Logical (Numerical)

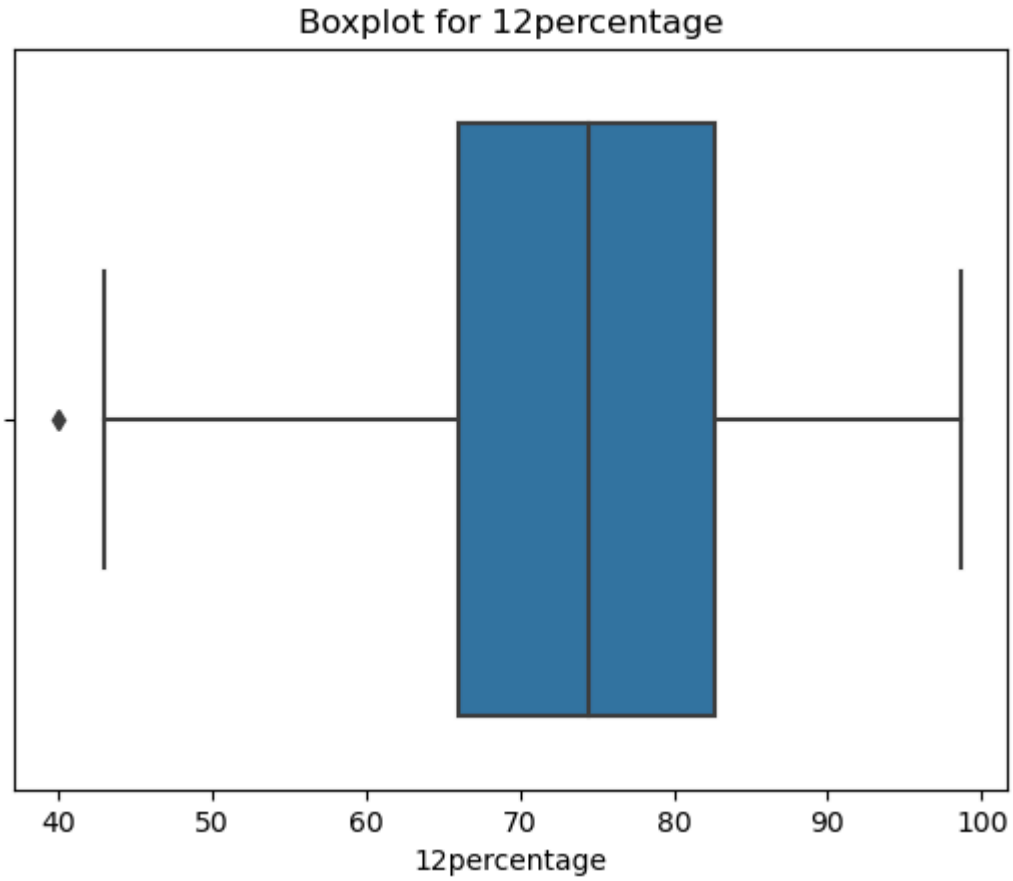


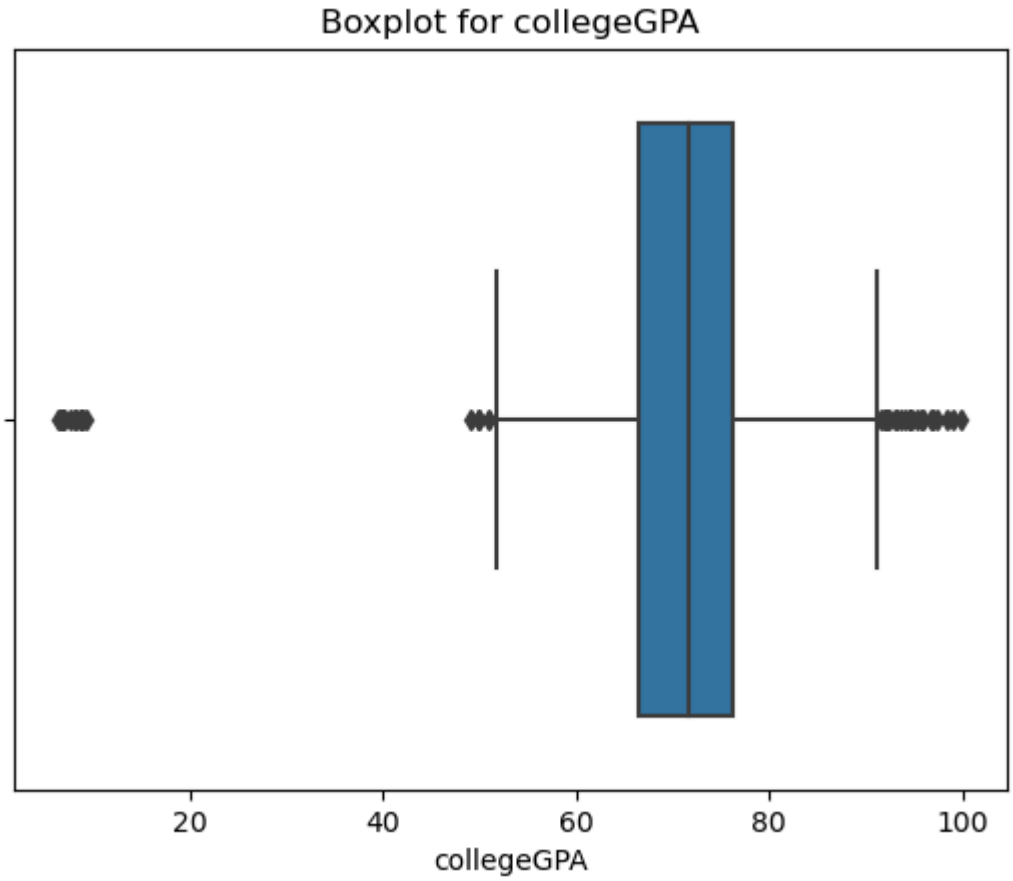
outliers in each numerical column:

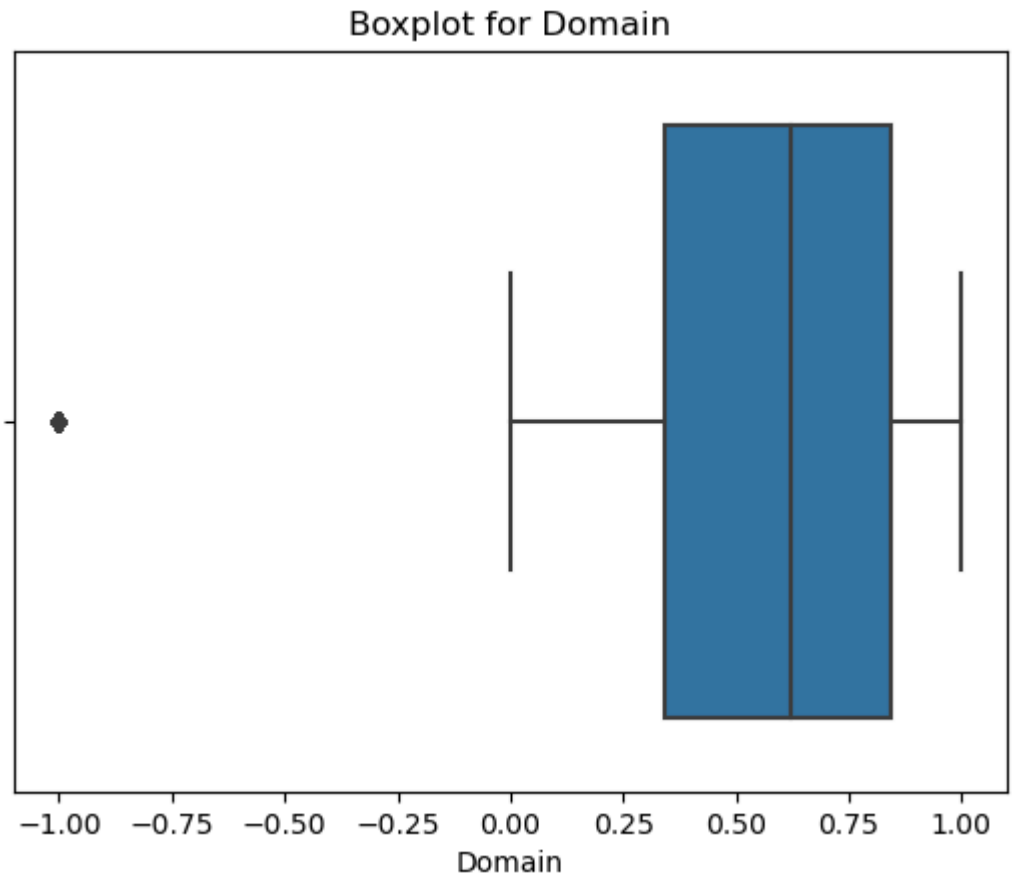
In [97]:
```
for i in df.columns:
    if df[i].dtype=="int" or df[i].dtype=="float":
        sns.boxplot(x=df[i])
        plt.title("Boxplot for {}".format(i))
        plt.show()
```

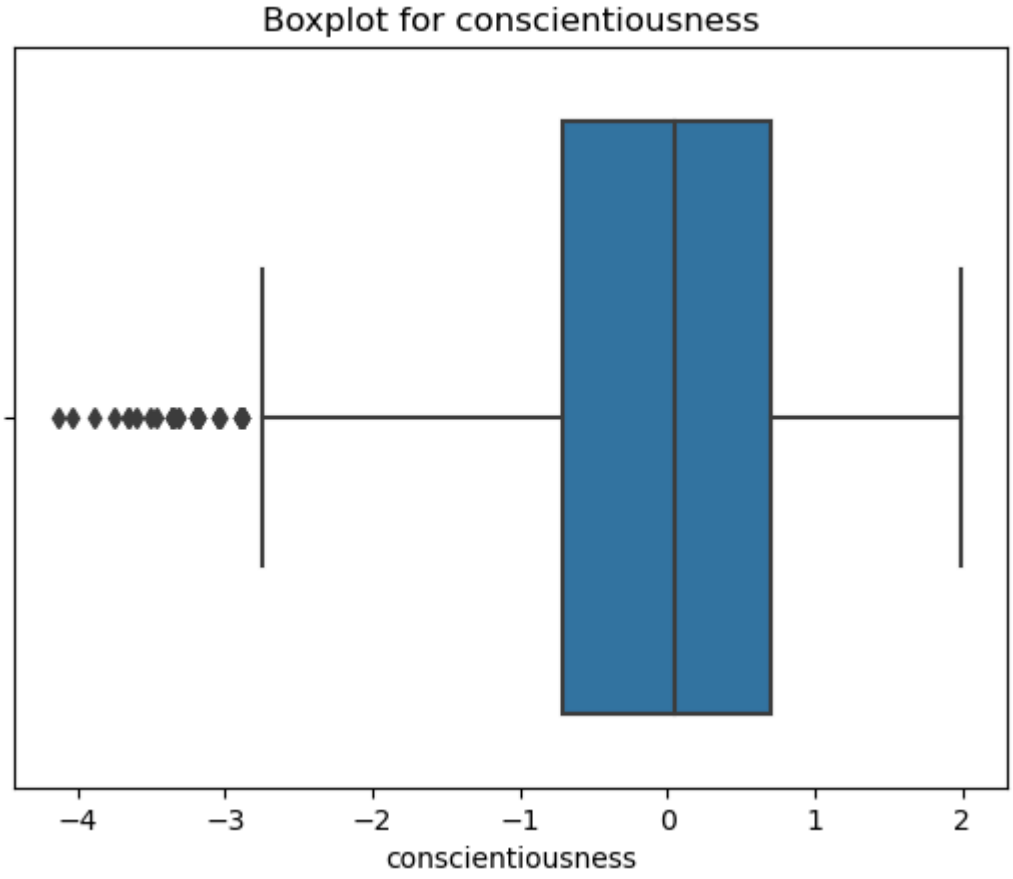

Boxplot for Salary
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Salary
1e6

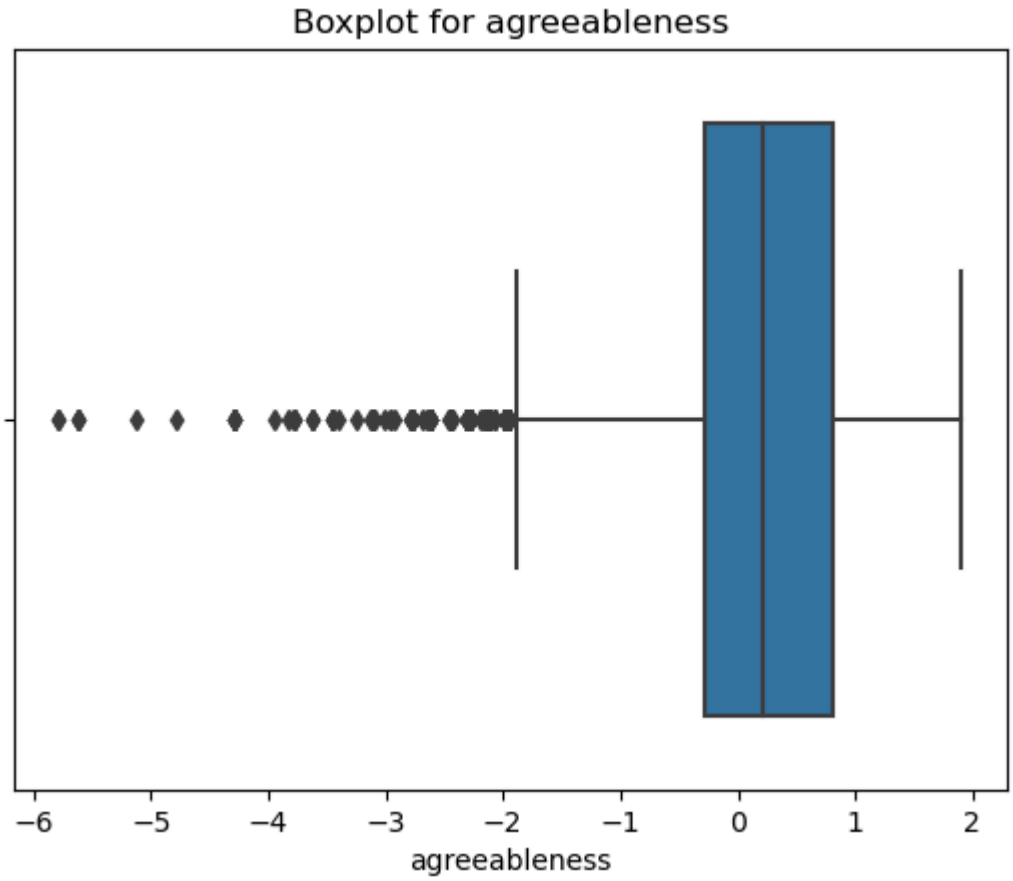

Boxplot for 10percentage
50
60
70
80
90
100
10percentage

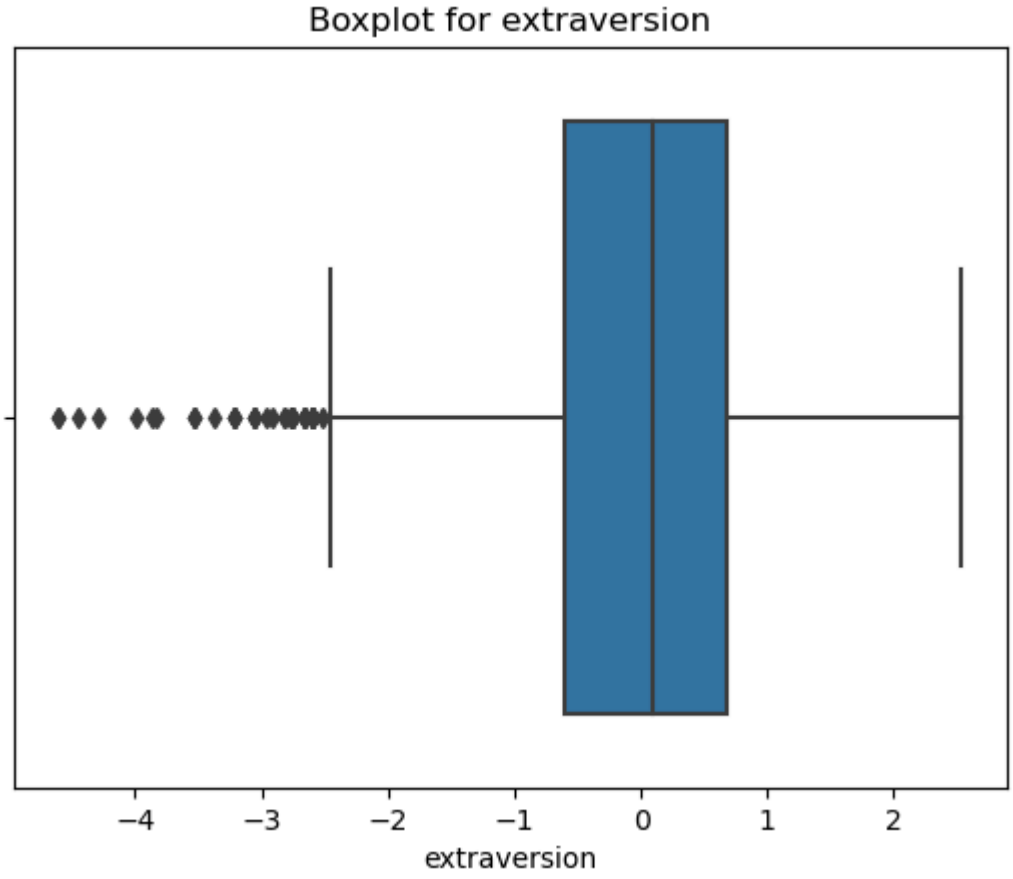

Boxplot for 12percentage
40
50
60
70
80
90
100
12percentage

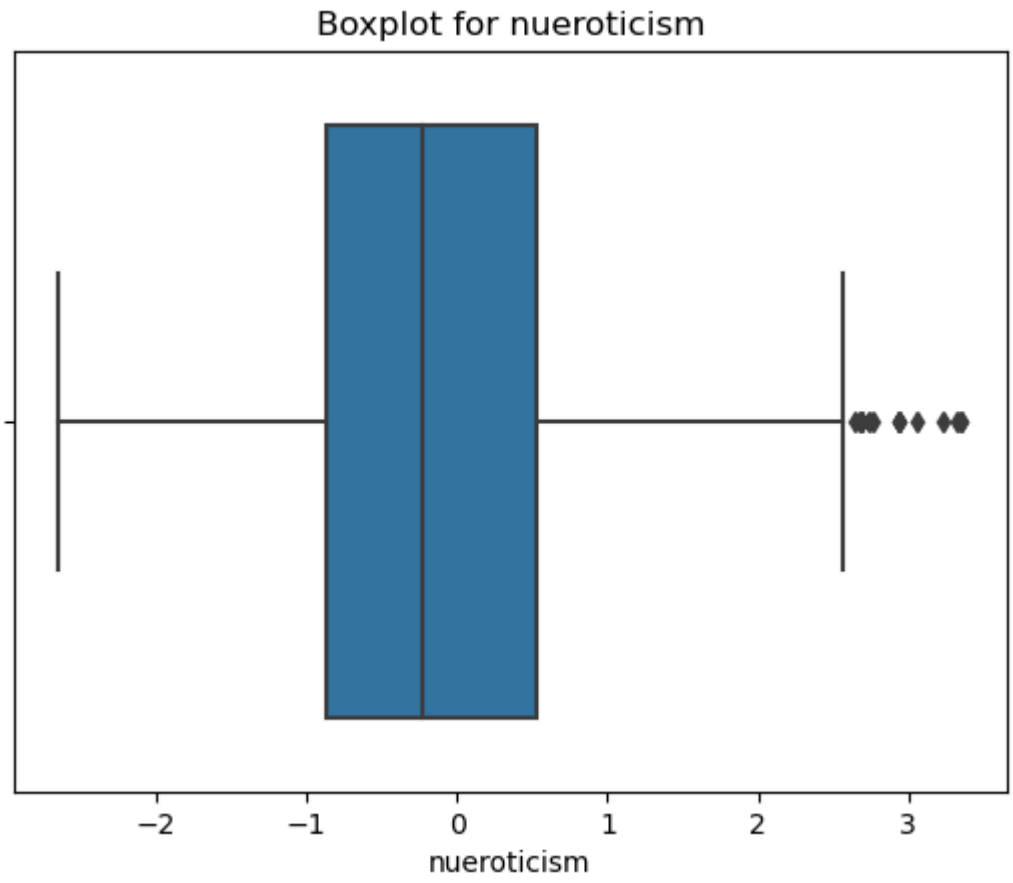

Boxplot for collegeGPA
20
40
60
80
100
collegeGPA

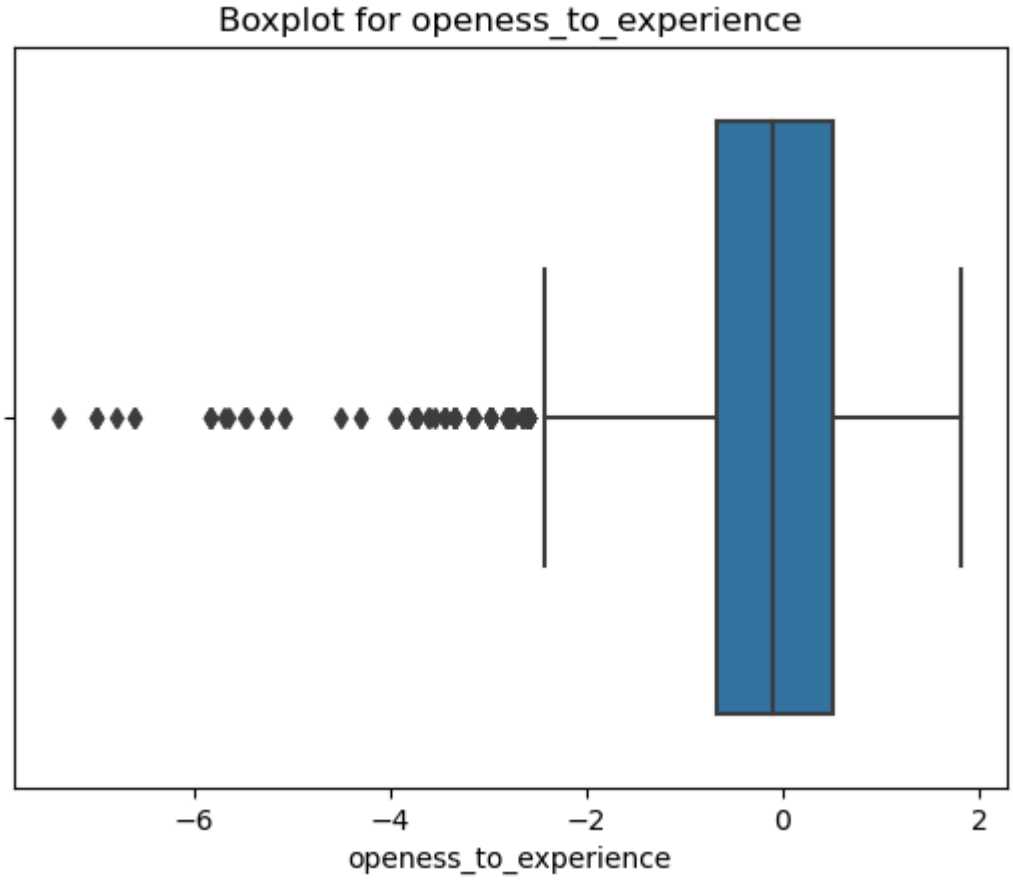

Boxplot for Domain
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
Domain


Boxplot for conscientiousness
−4
−3
−2
−1
0
1
2
conscientiousness


Boxplot for agreeableness
−6
−5
−4
−3
−2
−1
0
1
2
agreeableness


Boxplot for extraversion
−4
−3
−2
−1
0
1
2
extraversion

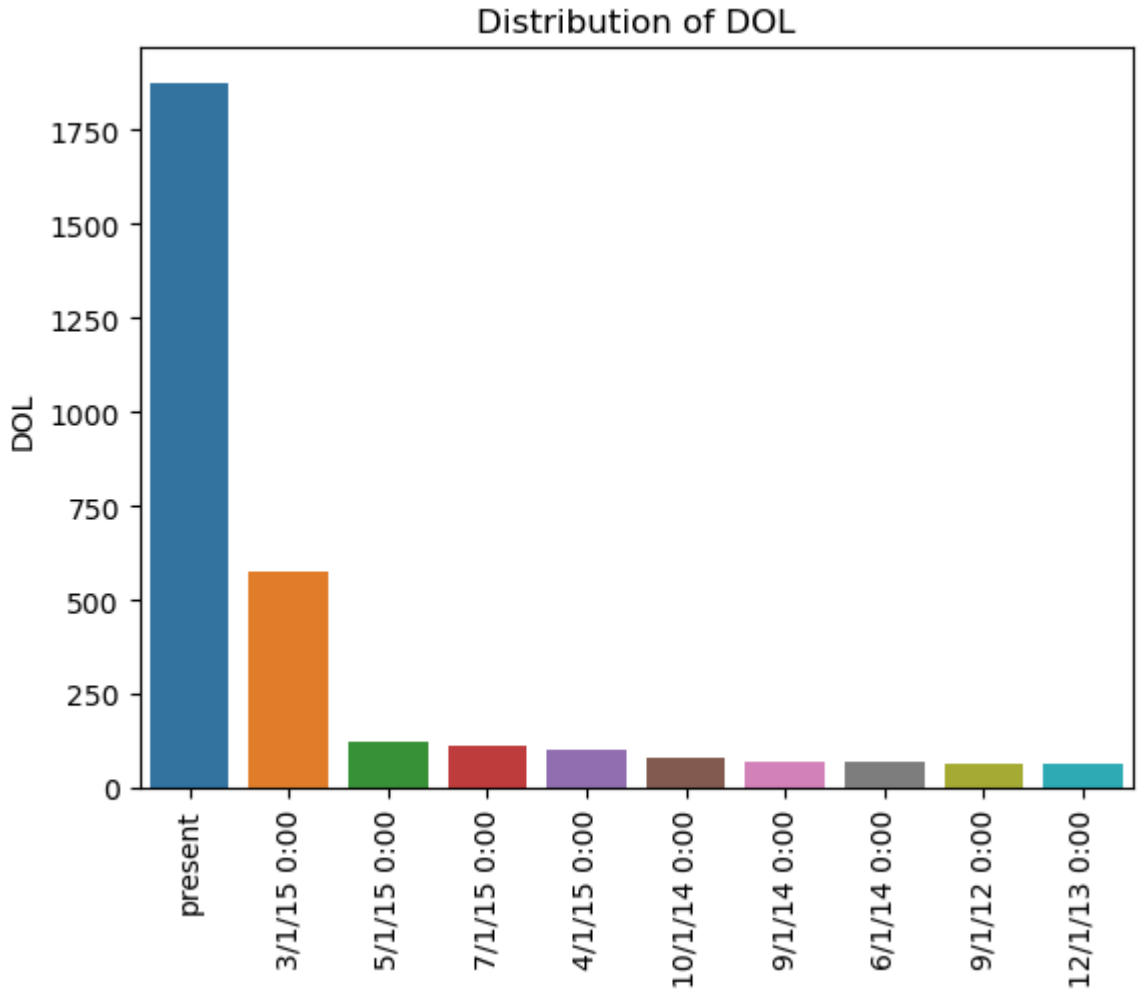


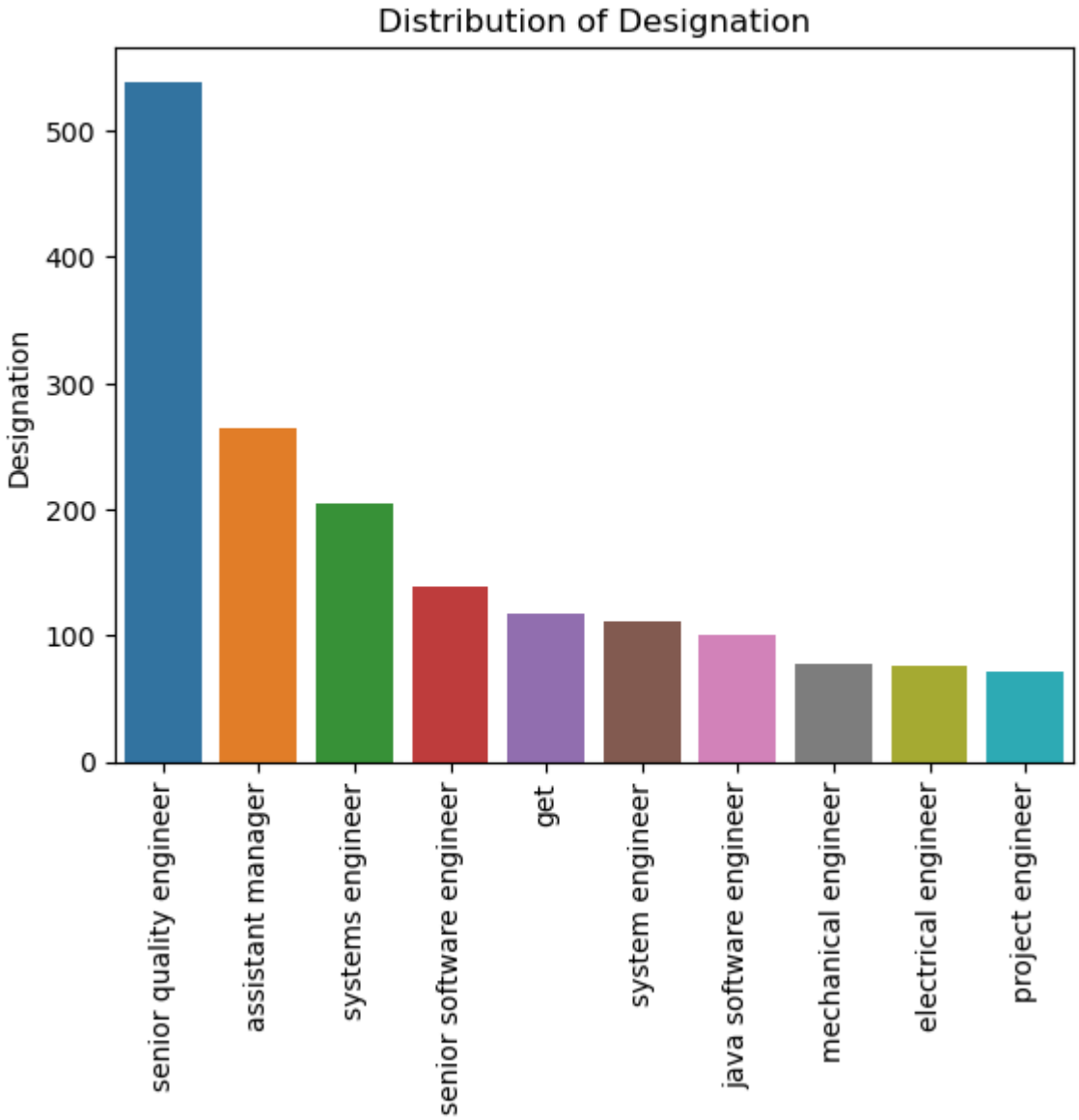


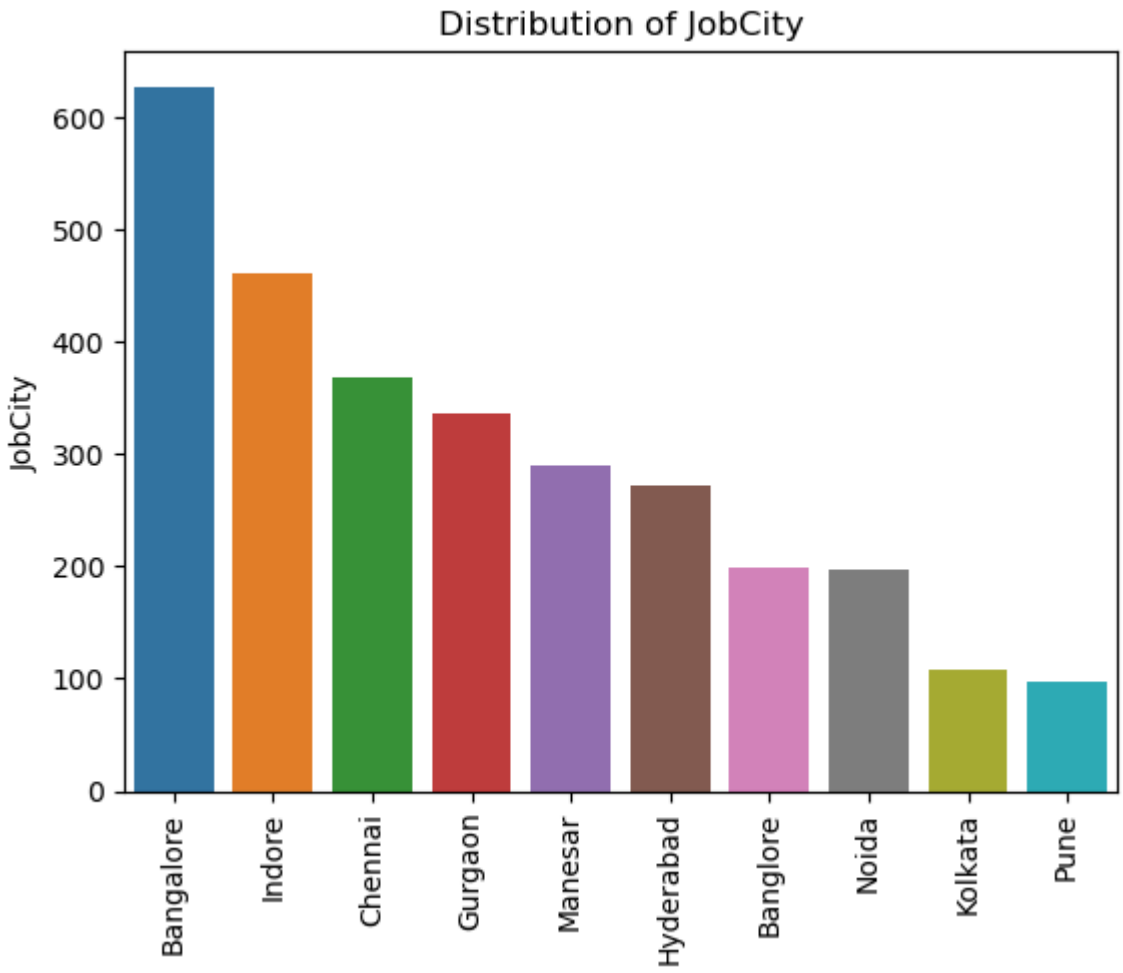
The frequency distribution of each categorical Variable

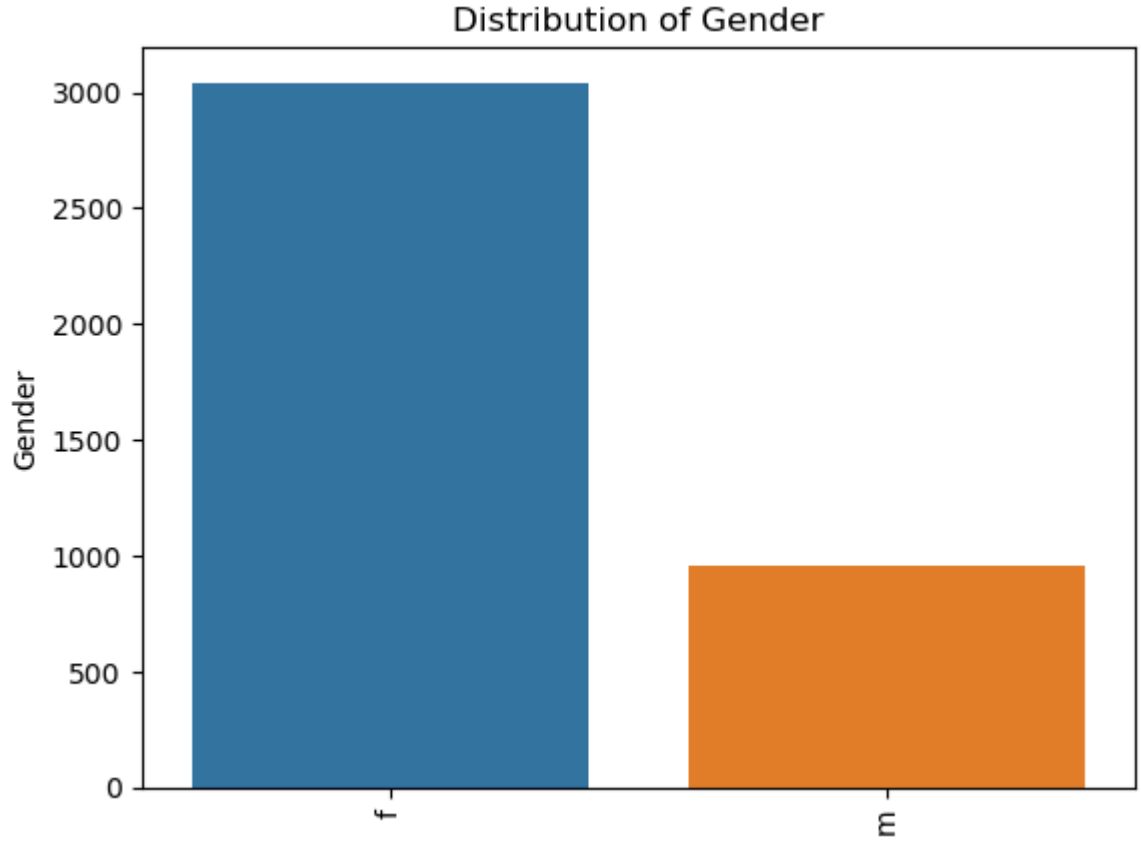
In [99]:
```
for i in df.columns:
    if df[i].dtype=="object":
        sns.barplot(x=df[i].unique()[:10],y=df[i].value_counts()[:10])
        plt.title("Distribution of {}".format(i))
```

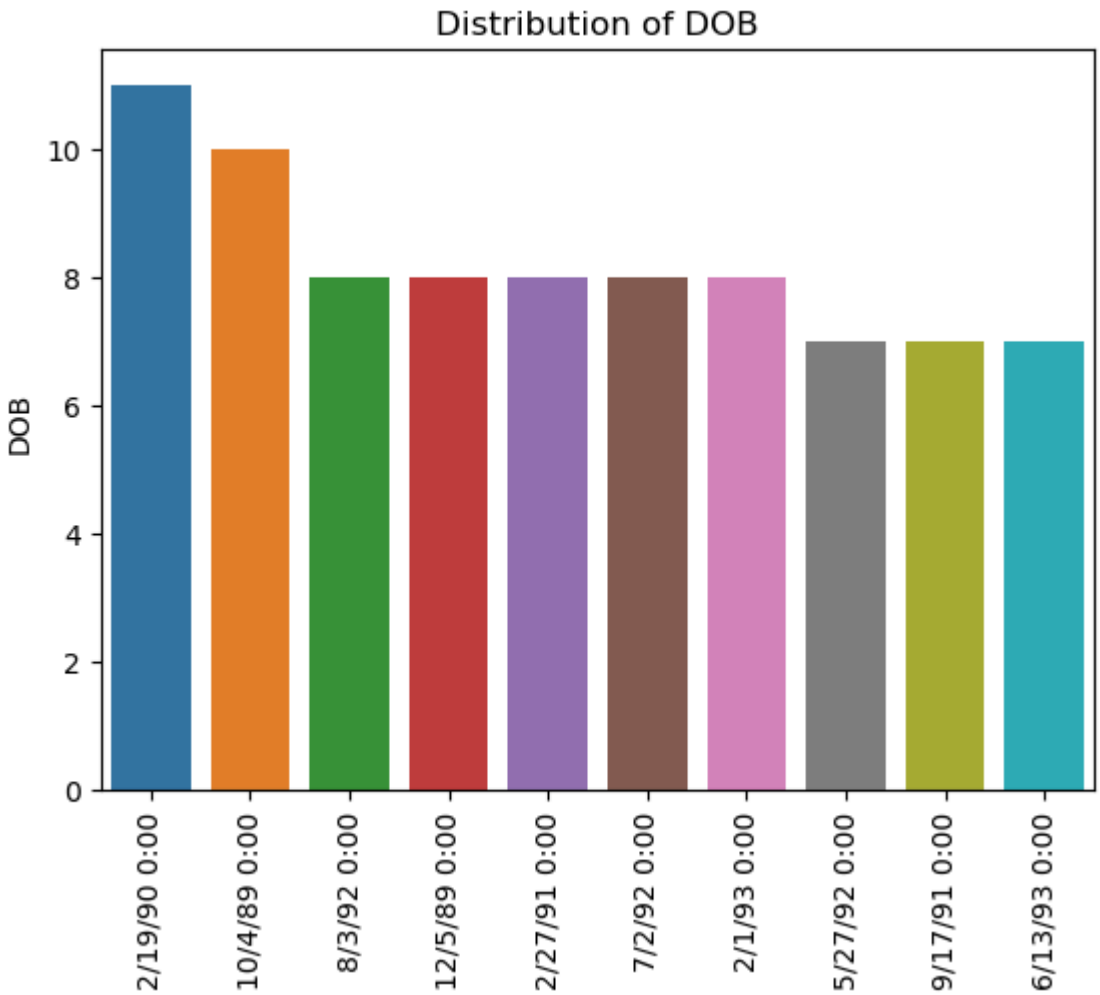
```
        plt.xticks(rotation=90)
        plt.show()
```

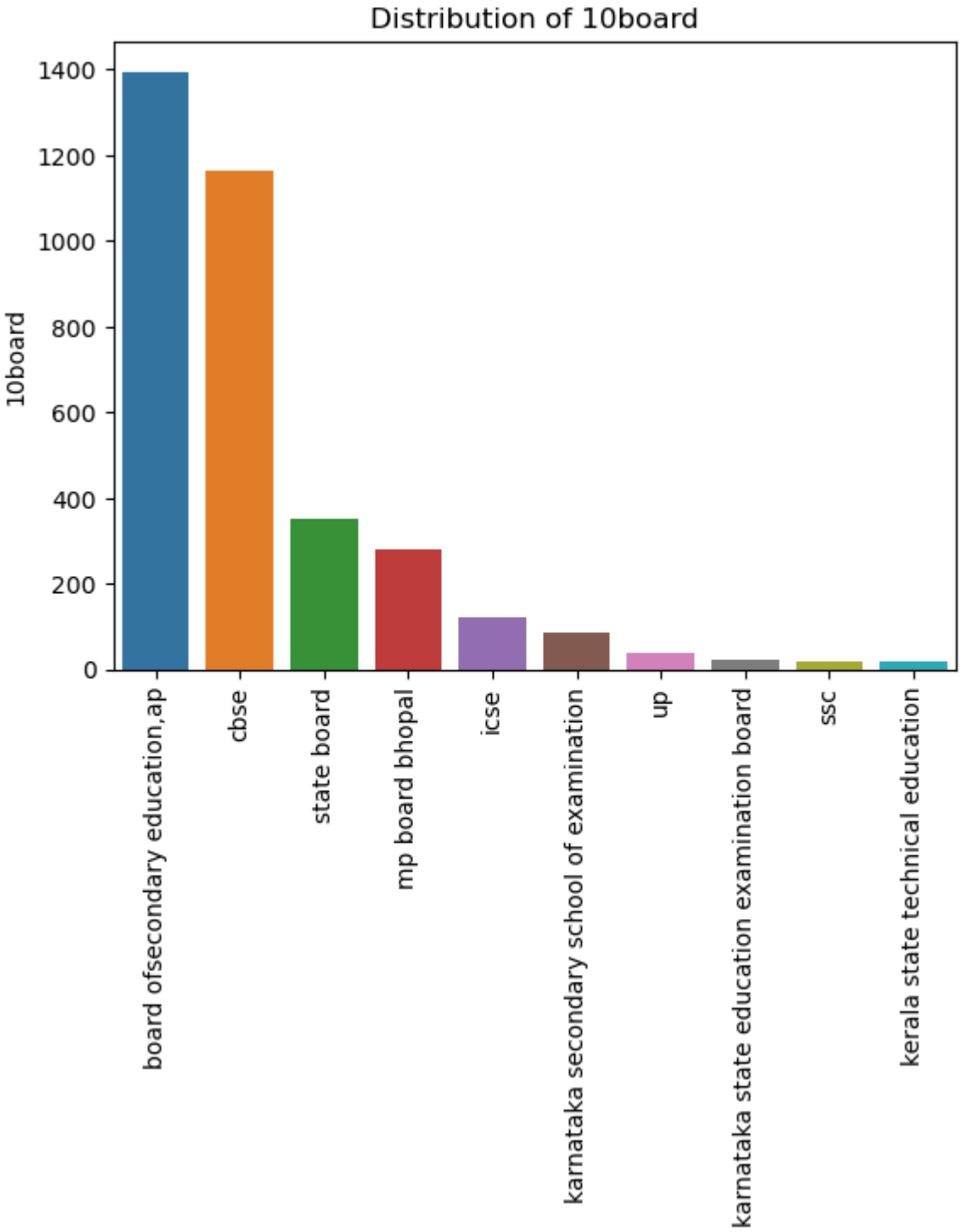


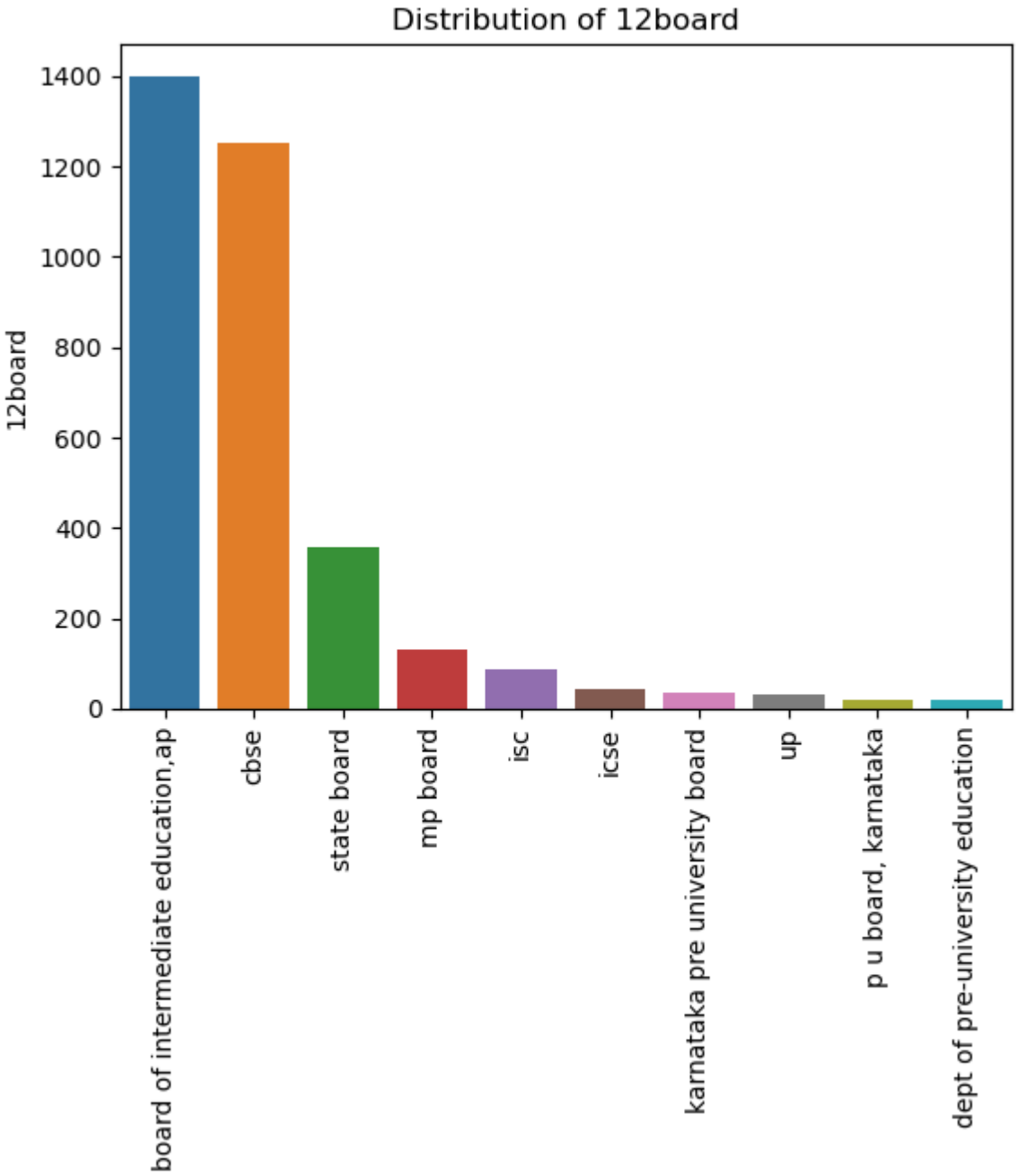

Distribution of Designation
Designation
0
100
200
300
400
500
senior quality engineer
assistant manager
systems engineer
senior software engineer
get
system engineer
java software engineer
mechanical engineer
electrical engineer
project engineer

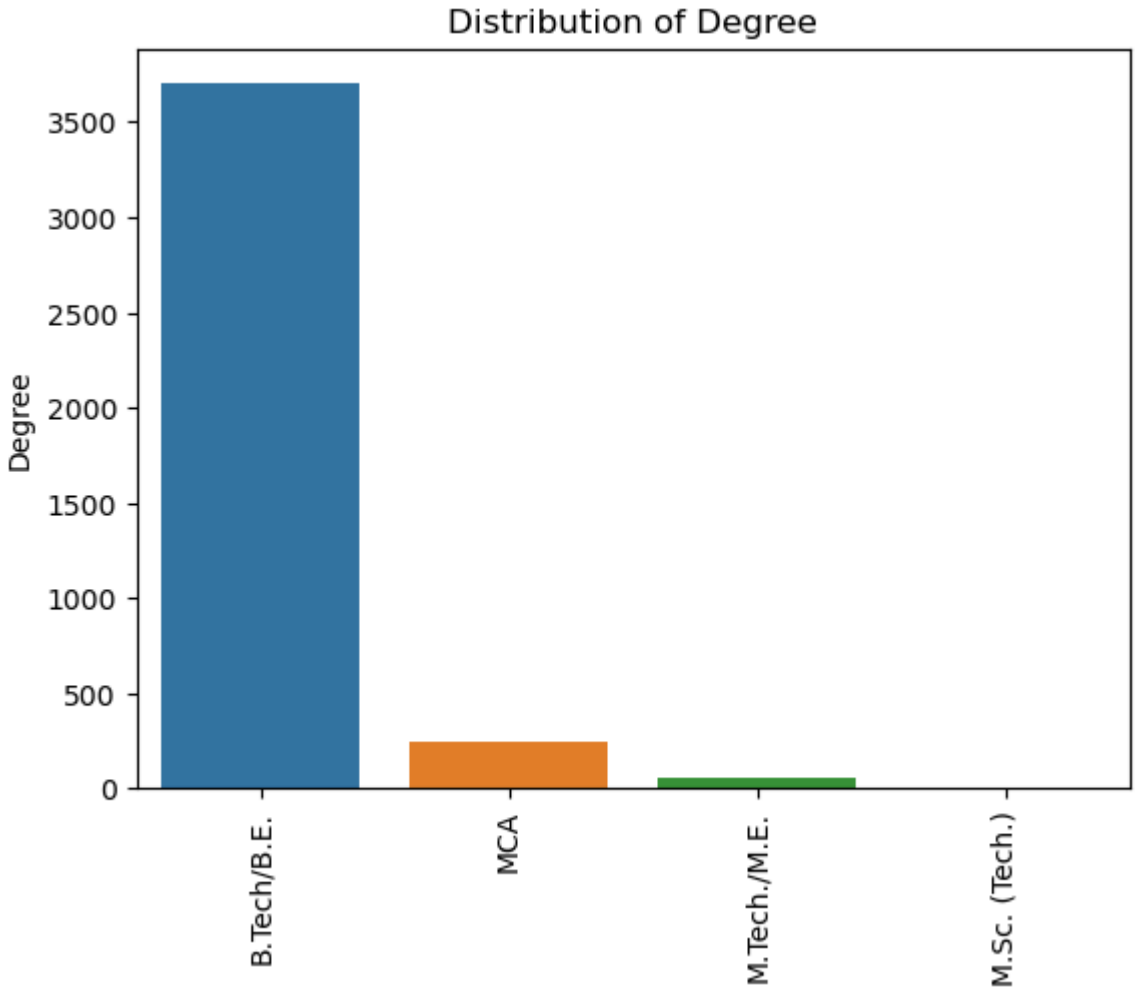

Distribution of JobCity
JobCity
0
100
200
300
400
500
600
Bangalore
Indore
Chennai
Gurgaon
Manesar
Hyderabad
Banglore
Noida
Kolkata
Pune

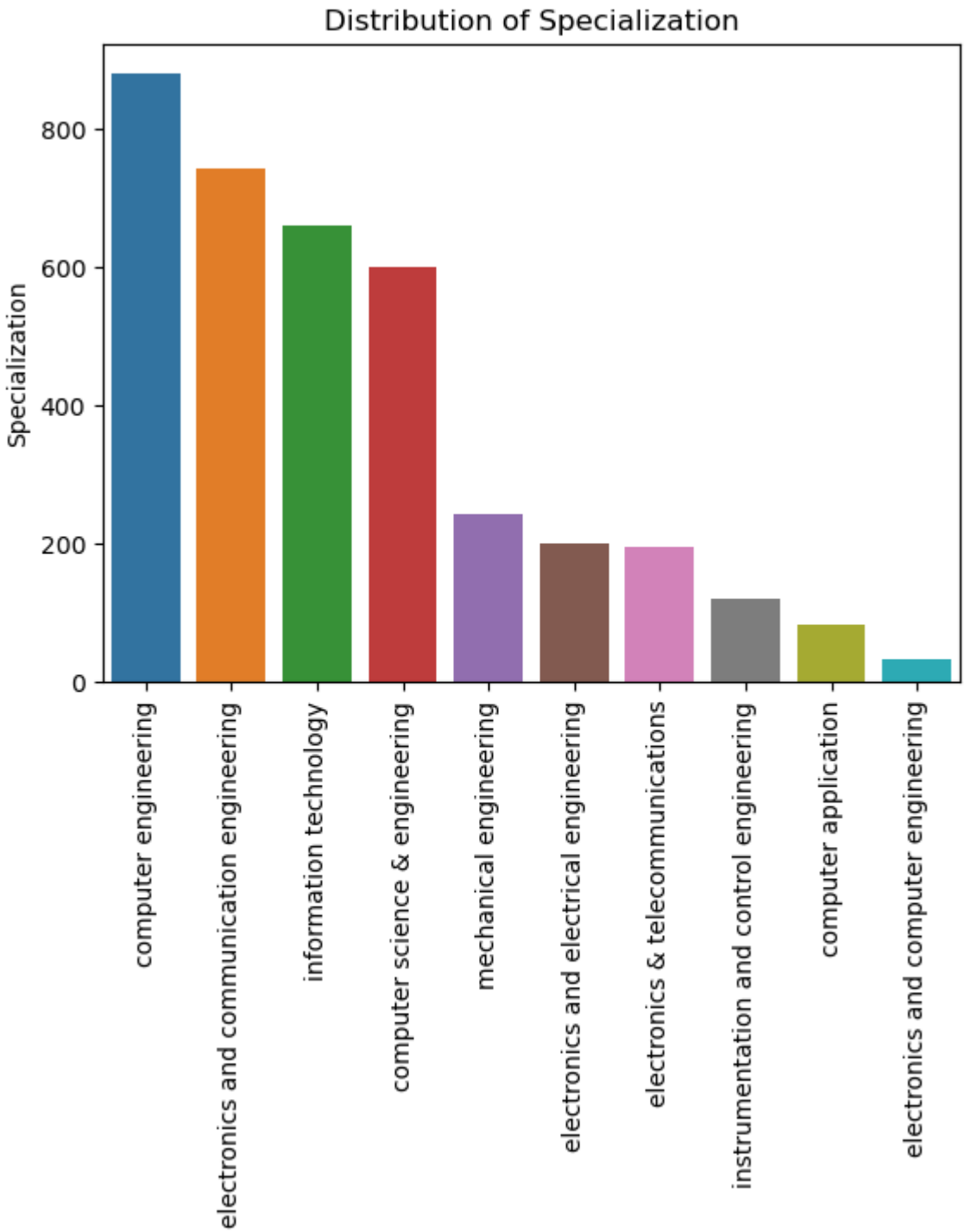

Distribution of Gender
Gender
0
500
1000
1500
2000
2500
3000
f
m

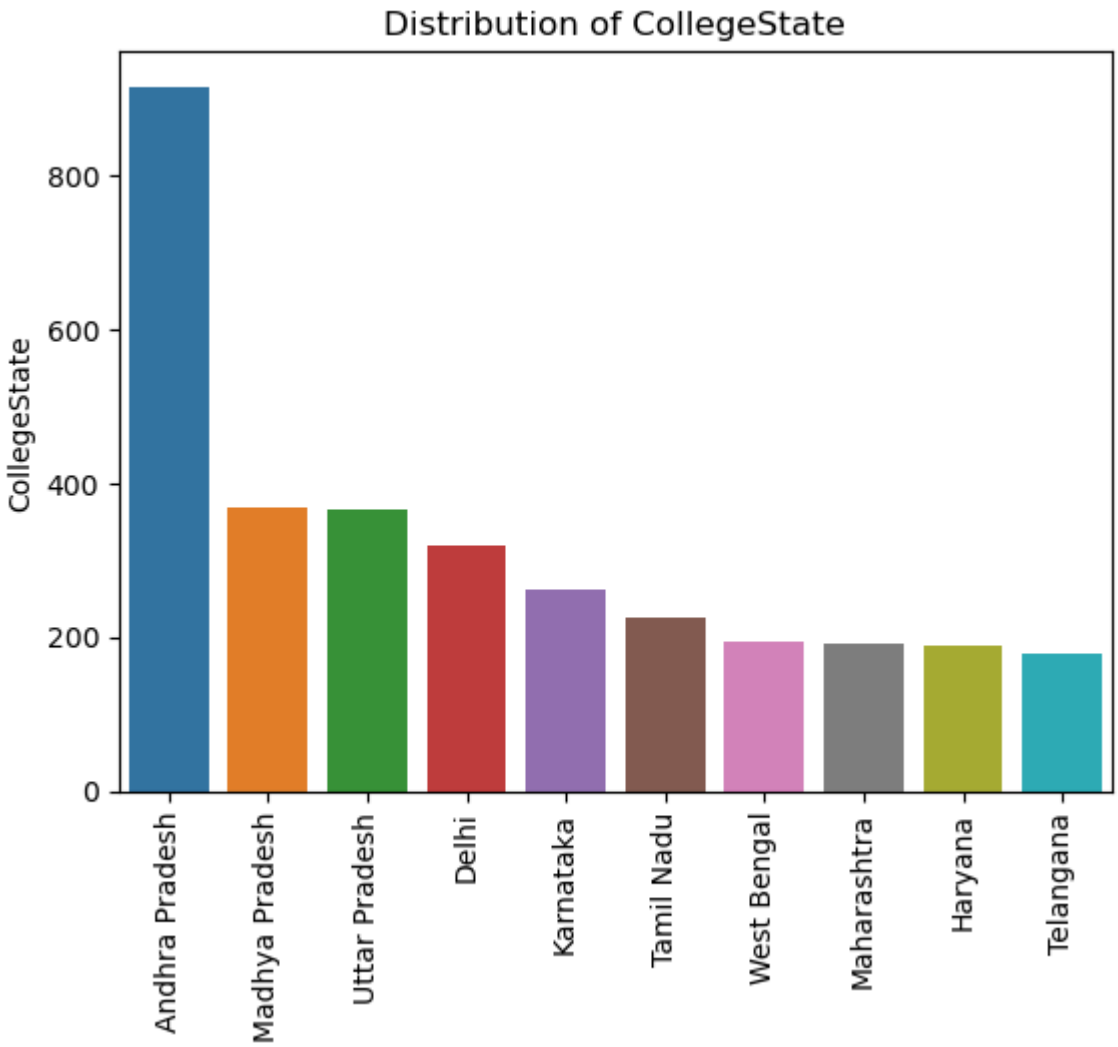

Distribution of DOB
DOB
0
2
4
6
8
10
2/19/90 0:00
10/4/89 0:00
8/3/92 0:00
12/5/89 0:00
2/27/91 0:00
7/2/92 0:00
2/1/93 0:00
5/27/92 0:00
9/17/91 0:00
6/13/93 0:00


Distribution of 10board
10board
1400
1200
1000
800
600
400
200
0
board ofsecondary education,ap
cbse
state board
mp board bhopal
icse
karnataka secondary school of examination
up
karnataka state education examination board
ssc
kerala state technical education


Distribution of 12board
12board
1400
1200
1000
800
600
400
200
0
board of intermediate education,ap
cbse
state board
mp board
isc
icse
karnataka pre university board
up
p u board, karnataka
dept of pre-university education


Distribution of Degree
Degree
3500
3000
2500
2000
1500
1000
500
0
B.Tech/B.E.
MCA
M.Tech./M.E.
M.Sc. (Tech.)


Distribution of Specialization
Specialization
0
200
400
600
800
computer engineering
electronics and communication engineering
information technology
computer science & engineering
mechanical engineering
electronics and electrical engineering
electronics & telecommunications
instrumentation and control engineering
computer application
electronics and computer engineering



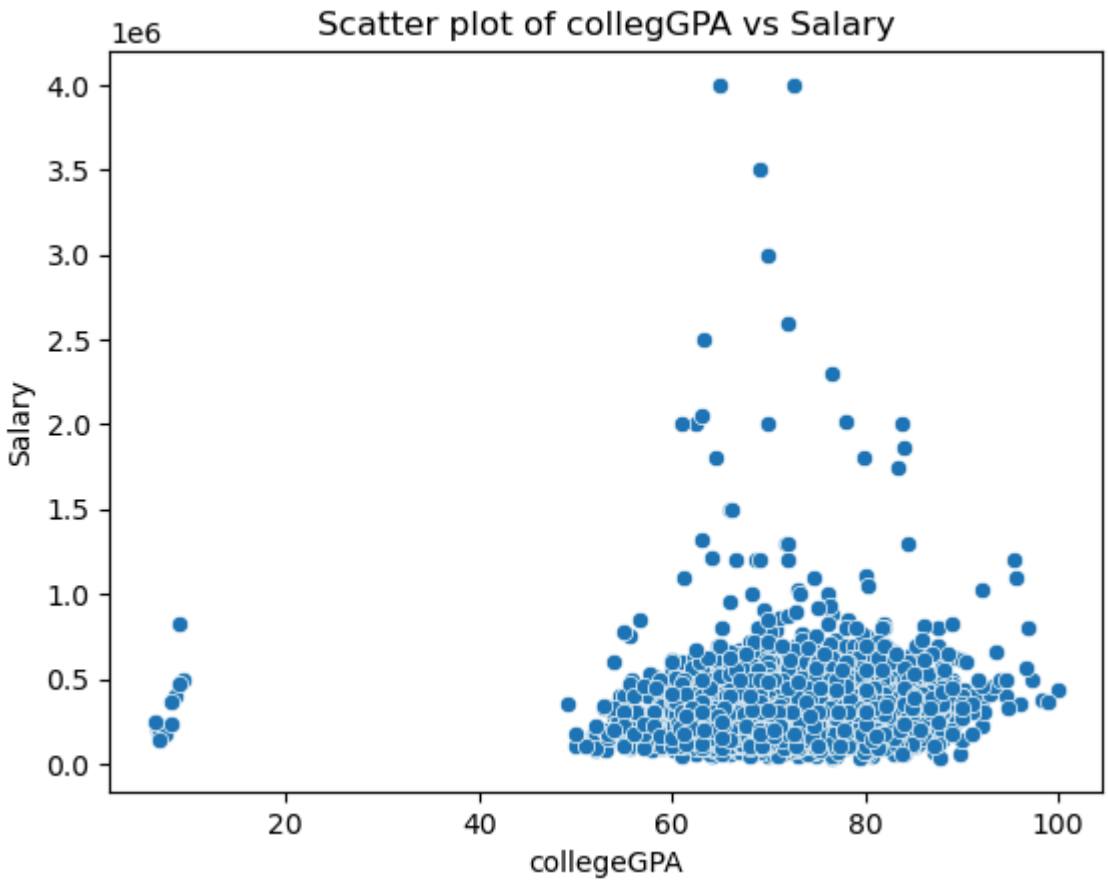
Bivariate Analysis

Analysing data using two variables

The relationships between numerical columns using scatter plot:

In [103…

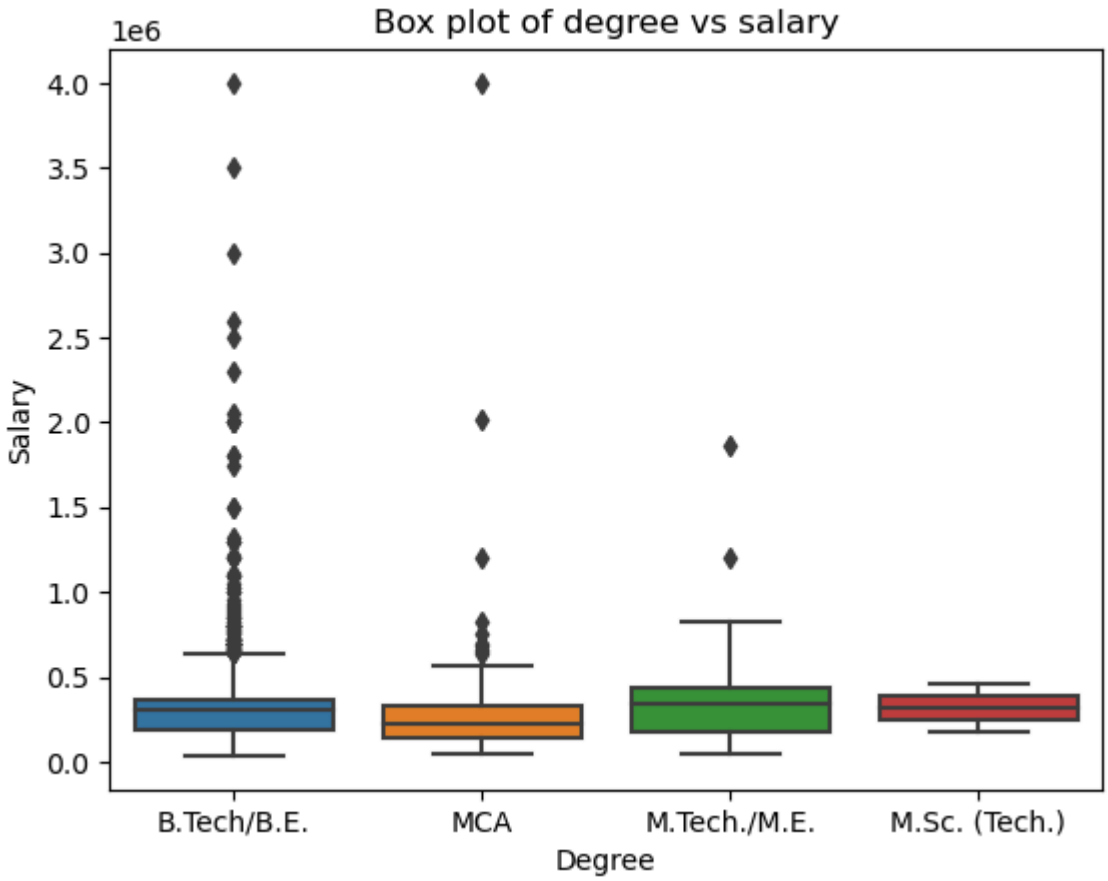
```
sns.scatterplot(x="collegeGPA", y="Salary",data=df)
plt.title("Scatter plot of collegGPA vs Salary")
plt.show()
```



categorical vs numerical columns using box plot:

In [106…

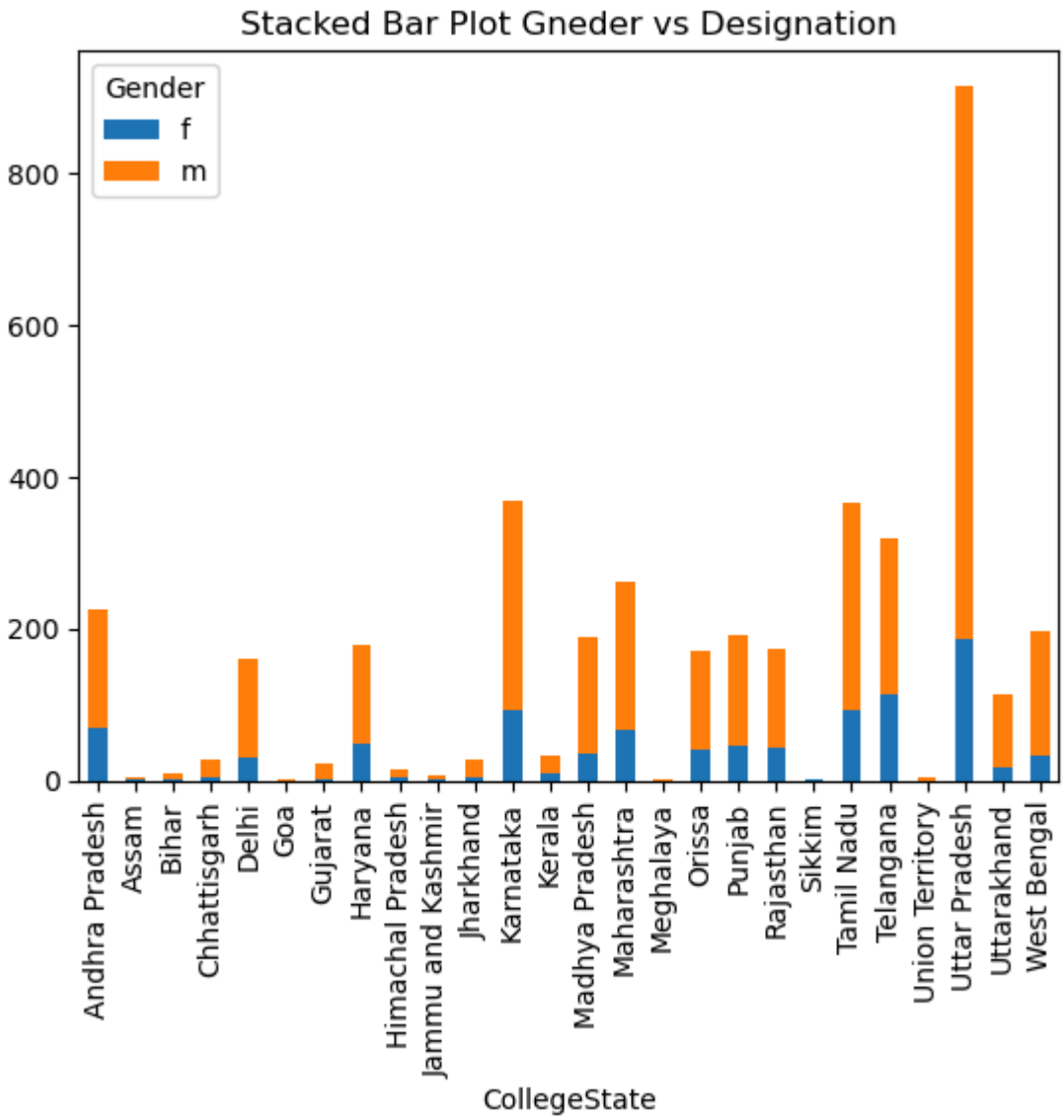
```
sns.boxplot(x="Degree", y="Salary", data = df)
plt.title("Box plot of degree vs salary")
plt.show()
```



Relationships between categorical vs categorical columns using stacked bar plots:

In [109…

```
pd.crosstab(df['CollegeState'], df['Gender']).plot(kind='bar', stacked=True)
plt.title('Stacked Bar Plot Gneder vs Designation')
plt.show()
```



Research Questions

In [123…

```
from scipy import stats

# Filter the data for relevant job roles
relevant_roles = ['programmer Analyst', 'software engineer', 'hardware engineer', '
df_filtered = df[df['Designation'].isin(relevant_roles)]
salary_data = df_filtered["Salary"]
claimed_mean_salary = 2.75*100000

t_stat, p_value = stats.ttest_1samp(salary_data, claimed_mean_salary)

print(f"Mean Salary of Selected Roles: {salary_data.mean():.2f}")
print(f"Claimed Mean Salary: {claimed_mean_salary:.2f}")


print(f"T-statistic: {t_stat:.2f}")
print(f"p-value: {p_value:.4f}")

alpha = 0.05

if p_value < alpha:
      print("Reject the null hypothesis: The average salary is significantly differ
else:
      print("Fail to reject the null hypothesis: There is no significant difference
```

```
Mean Salary of Selected Roles: 339792.04
Claimed Mean Salary: 275000.00
T-statistic: 10.55
p-value: 0.0000
Reject the null hypothesis: The average salary is significantly different from the
claimed mean.
```

In [125…

```
from scipy import stats as st
from scipy.stats import chi2_contingency
# Create a contingency table
contingency_table = pd.crosstab(index = df['Specialization'], columns = df['Gender'

# Chi-square test of independence
chi2_stat, p_val, dof, expected = chi2_contingency(contingency_table)
if p_value < alpha:
    print("Reject the null hypothesis: There is a significant difference between th
else:
    print("Fail to reject the null hypothesis: There is no significant difference b
```

```
Reject the null hypothesis: There is a significant difference between the gender a
nd speciliazation.
```

In [ ]: