AMCAT Case-study

```
# Univariate Analysis -> PDF, Histograms, Boxplots, Countplots, etc..
# Find the outliers in each numerical column
# Understand the probability and frequency distribution of each numerical column
# Understand the frequency distribution of each categorical Variable/Column
# # Mention observations after each plot.
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import numpy as np
from scipy import stats
```

```
from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving data.xlsx to data (1).xlsx

```
for filename in uploaded.keys():
    print(f'User uploaded file "{filename}"')
```

```
User uploaded file "data (1).xlsx"
```

```
df = pd.read_excel("data.xlsx")
df
```

	Unnamed: 0	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DOB	10percentage	...	ComputerScience	Mechanic
0	train	203097	420000	2012-06-01	present	senior quality engineer	Bangalore	f	1990-02-19	84.30	...	-1	
1	train	579905	500000	2013-09-01	present	assistant manager	Indore	m	1989-10-04	85.40	...	-1	
2	train	810601	325000	2014-06-01	present	systems engineer	Chennai	f	1992-08-03	85.00	...	-1	
3	train	267447	1100000	2011-07-01	present	senior software engineer	Gurgaon	m	1989-12-05	85.60	...	-1	
4	train	343523	200000	2014-03-01	2015-03-01 00:00:00	get	Manesar	m	1991-02-27	78.00	...	-1	
...	...	...	...	...	...	...	...	...	...	...	...	...	
3993	train	47916	280000	2011-10-01	2012-10-01 00:00:00	software engineer	New Delhi	m	1987-04-15	52.09	...	-1	
3994	train	752781	100000	2013-07-01	2013-07-01 00:00:00	technical writer	Hyderabad	f	1992-08-27	90.00	...	-1	
3995	train	355888	320000	2013-07-01	present	associate software engineer	Bangalore	m	1991-07-03	81.86	...	-1	
3996	train	947111	200000	2014-07-01	2015-01-01 00:00:00	software developer	Asifabadbanglore	f	1992-03-20	78.72	...	438	
3997	train	324966	400000	2013-02-01	present	senior systems engineer	Chennai	f	1991-02-26	70.60	...	-1	

3998 rows × 39 columns

```
df.head(2)
```

	Unnamed: 0	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DOB	10percentage	...	ComputerScience	MechanicalEngg	Elec
0	train	203097	420000	2012-06-01	present	senior quality engineer	Bangalore	f	1990-02-19	84.3	...	-1	-1	
1	train	579905	500000	2013-09-01	present	assistant manager	Indore	m	1989-10-04	85.4	...	-1	-1	

2 rows × 39 columns

```
df.columns
```

```
Index(['Unnamed: 0', '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')
```

```
#Fix the column
df = df.drop('Unnamed: 0',axis=1)
```

```
df.head()
```

	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DOB	10percentage	10board	...	ComputerScience	MechanicalEngg
0	203097	420000	2012-06-01	present	senior quality engineer	Bangalore	f	1990-02-19	84.3	board ofsecondary education,ap	...	-1	-1
1	579905	500000	2013-09-01	present	assistant manager	Indore	m	1989-10-04	85.4	cbse	...	-1	-1
2	810601	325000	2014-06-01	present	systems engineer	Chennai	f	1992-08-03	85.0	cbse	...	-1	-1
3	267447	1100000	2011-07-01	present	senior software engineer	Gurgaon	m	1989-12-05	85.6	cbse	...	-1	-1
4	343523	200000	2014-03-01	2015-03-01 00:00:00	get	Manesar	m	1991-02-27	78.0	cbse	...	-1	-1

5 rows × 38 columns

```
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   int64
 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   datetime64[ns]
 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](2), float64(9), int64(18), object(9)
memory usage: 1.2+ MB
```

```
unique_cities = df['JobCity'].unique()
unique_cities
```

```
array(['Bangalore', 'Indore', 'Chennai', 'Gurgaon', 'Manesar',
       'Hyderabad', 'Banglore', 'Noida', 'Kolkata', 'Pune', -1, 'mohali',
       'Jhansi', 'Delhi', 'Hyderabad ', 'Bangalore ', 'noida', 'delhi',
       'Bhubaneswar', 'Navi Mumbai', 'Mumbai', 'New Delhi', 'Mangalore',
       'Rewari', 'Gaziabaad', 'Bhiwadi', 'Mysore', 'Rajkot',
       'Greater Noida', 'Jaipur', 'noida ', 'HYDERABAD', 'mysore',
       'THANE', 'Maharajganj', 'Thiruvananthapuram', 'Punchkula',
       'Bhubaneshwar', 'Pune ', 'coimbatore', 'Dhanbad', 'Lucknow',
```

```
            'Trivandrum', 'kolkata', 'mumbai', 'Gandhi Nagar', 'Una',
            'Daman and Diu', 'chennai', 'GURGOAN', 'vsakhapttnam', 'pune',
            'Nagpur', 'Bhagalpur', 'new delhi - jaisalmer', 'Coimbatore',
            'Ahmedabad', 'Kochi/Cochin', 'Bankura', 'Bengaluru', 'Mysore ',
            'Kanpur ', 'jaipur', 'Gurgaon ', 'bangalore', 'CHENNAI',
            'Vijayawada', 'Kochi', 'Beawar', 'Alwar', 'NOIDA', 'Greater noida',
            'Siliguri ', 'raipur', 'gurgaon', 'Bhopal', 'Faridabad', 'Jodhpur',
            'udaipur', 'Muzaffarpur', 'Kolkata`', 'Bulandshahar', 'Haridwar',
            'Raigarh', 'Visakhapatnam', 'Jabalpur', 'hyderabad', 'Unnao',
            'KOLKATA', 'Thane', 'Aurangabad', 'Belgaum', 'gurgoan', 'Dehradun',
            'Rudrapur', 'Jamshedpur', 'vizag', 'Nouda', 'Dharamshala',
            'Banagalore', 'Hissar', 'Ranchi', 'BANGALORE', 'Madurai', 'Gurga',
            'Chandigarh', 'Australia', ' Chennai', 'CHEYYAR', 'Mumbai ',
            'sonepat', 'Ghaziabad', 'Pantnagar', 'Siliguri', 'mumbai ',
            'Jagdalpur', 'Chennai ', 'angul', 'Baroda', ' ariyalur', 'Jowai',
            'Kochi/Cochin, Chennai and Coimbatore', 'bhubaneswar', 'Neemrana',
            'VIZAG', 'Tirupathi', 'Lucknow ', 'Ahmedabad ', 'Bhubneshwar',
            'Noida ', 'pune ', 'Calicut', 'Gandhinagar', 'LUCKNOW', 'Dubai',
            'bengaluru', 'MUMBAI', 'Ahmednagar', 'Nashik', 'New delhi',
            'Bellary', 'Ludhiana', 'New Delhi ', 'Muzaffarnagar', 'BHOPAL',
            'Gurgoan', 'Gagret', 'Indirapuram, Ghaziabad', 'Gwalior',
            'new delhi', 'TRIVANDRUM', 'Chennai & Mumbai', 'Rajasthan',
            'Sonipat', 'Bareli', 'Kanpur', 'Hospete', 'Miryalaguda', ' mumbai',
            'Dharuhera', 'lucknow', 'meerut', 'dehradun', 'Ganjam', 'Hubli',
            'bangalore ', 'NAVI MUMBAI', 'ncr', 'Agra', 'Trichy',
            'kudankulam ,tarapur', 'Ongole', 'Sambalpur', 'Pondicherry',
            'Bundi', 'SADULPUR,RAJGARH,DISTT-CHURU,RAJASTHAN', 'AM', 'Bikaner',
            'Vadodara', 'BAngalore', 'india', 'Asansol', 'Tirunelvelli',
            'Ernakulam', 'DELHI', 'Bilaspur', 'Chandrapur', 'Nanded',
            'Dharmapuri', 'Vandavasi', 'Rohtak', 'trivandrum', 'Nagpur ',
            'Udaipur', 'Patna', 'banglore', 'indore', 'Salem', 'Nasikcity',
            'Gandhinagar ', 'Technopark, Trivandrum', 'Bharuch', 'Tornagallu',
            'Raipur', 'Kolkata ', 'Jaspur', 'Burdwan', 'Bhubaneswar ',
            'Shimla', 'ahmedabad', 'Gajiabaad', 'Jammu', 'Shahdol',
            'Muvattupuzha', 'Al Jubail,Saudi Arabia', 'Kalmar, Sweden',
            'Secunderabad', 'A-64,sec-64,noida', 'Ratnagiri', 'Jhajjar',
            'Gulbarga', 'hyderabad(bhadurpally)', 'Nalagarh', 'Chandigarh ',
            'Jaipur ', 'Jeddah Saudi Arabia', ' Delhi', 'PATNA', 'SHAHDOL',
            'Chennai, Bangalore', 'Bhopal ', 'Jamnagar', 'PUNE', 'Tirupati',
            'Gonda', 'jamnagar', 'chennai ', 'orissa', 'kharagpur',
            'Trivandrum ', 'Navi Mumbai , Hyderabad', 'Joshimath',
            'chandigarh', 'Bathinda', 'Johannesburg', 'kala amb ', 'Karnal',
            'LONDON', 'Kota', 'Panchkula', 'Baddi HP', 'Nagari',
            'Mettur, Tamil Nadu ', 'Durgapur', 'pondi', 'Surat', 'Kurnool',
            'kolhapur', 'Madurai ', 'GREATER NOIDA', 'Bhilai', ' Pune',
            'hderabad', 'KOTA', 'thane', 'Vizag', 'Bahadurgarh',
            'Rayagada, Odisha', 'kakinada', 'GURGAON', 'Varanasi', 'punr',
            'Nellore', 'patna', 'Meerut', 'hyderabad ', 'Sahibabad', 'Howrah',
            'BHUBANESWAR', 'Trichur', 'Ambala', 'Khopoli', 'keral', 'Roorkee',
            'Greater NOIDA', 'Navi mumbai', 'ghaziabad', 'Allahabad'
```

```
city_mapping = {
'bangalore': 'Bangalore',
'banglore': 'Bangalore',
'banagalore': 'Bangalore',
'bengaluru': 'Bangalore',
'asifabadbanglore':'Bangalore',
'indore': 'Indore',
'chennai': 'Chennai',
'gurgaon': 'Gurgaon',
'gurgoan': 'Gurgaon',
'gurga': 'Gurgaon',
'manesar': 'Manesar',
'hyderabad': 'Hyderabad',
'hderabad': 'Hyderabad',
'hyderabad(bhadurpally)': 'Hyderabad',
'noida': 'Noida',
'nouda': 'Noida',
'kolkata': 'Kolkata',
'kolkata`': 'Kolkata',
'pune': 'Pune',
'-1': 'Unknown',
'mohali': 'Mohali',
'jhansi': 'Jhansi',
'delhi': 'Delhi',
'new delhi': 'New Delhi',
'bhubaneswar': 'Bhubaneswar',
'bhubaneshwar': 'Bhubaneswar',
'navi mumbai': 'Navi Mumbai',
'mumbai': 'Mumbai',
'mangalore': 'Mangalore',
```

```
'rewari': 'Rewari',
'gaziabaad': 'Ghaziabad',
'ghaziabad': 'Ghaziabad',
'bhiwadi': 'Bhiwadi',
'mysore': 'Mysore',
'rajkot': 'Rajkot',
'greater noida': 'Greater Noida',
'jaipur': 'Jaipur',
'thane': 'Thane',
'maharajganj': 'Maharajganj',
'thiruvananthapuram': 'Thiruvananthapuram',
'punchkula': 'Panchkula',
'coimbatore': 'Coimbatore',
'dhanbad': 'Dhanbad',
'lucknow': 'Lucknow',
'trivandrum': 'Thiruvananthapuram',
'gandhi nagar': 'Gandhinagar',
'una': 'Una',
'daman and diu': 'Daman and Diu',
'vsakhapttnam': 'Visakhapatnam',
'nagpur': 'Nagpur',
'bhagalpur': 'Bhagalpur',
'new delhi- jaisalmer': 'New Delhi',
'ahmedabad': 'Ahmedabad',
'kochi/cochin': 'Kochi',
'bankura': 'Bankura',
'kanpur': 'Kanpur',
'vijayawada': 'Vijayawada',
'kochi': 'Kochi',
'beawar': 'Beawar',
'alwar': 'Alwar',
'siliguri': 'Siliguri',
'raipur': 'Raipur',
'bhopal': 'Bhopal',
'faridabad': 'Faridabad',
'jodhpur': 'Jodhpur',
'udaipur': 'Udaipur',
'muzaffarpur': 'Muzaffarpur',
'bulandshahar': 'Bulandshahar',
'haridwar': 'Haridwar',
'raigarh': 'Raigarh',
'visakhapatnam': 'Visakhapatnam',
'jabalpur': 'Jabalpur',
'unnao': 'Unnao',
'aurangabad': 'Aurangabad',
'belgaum': 'Belgaum',
'dehradun': 'Dehradun',
'rudrapur': 'Rudrapur',
'jamshedpur': 'Jamshedpur',
'vizag': 'Visakhapatnam',
'nouda': 'Noida',
'dharamshala': 'Dharamshala',
'hissar': 'Hisar',
'ranchi': 'Ranchi',
'madurai': 'Madurai',
'chandigarh': 'Chandigarh',
'australia': 'Australia',
'cheyyar': 'Cheyyar',
'sonepat': 'Sonepat',
'pantnagar': 'Pantnagar',
'jagdalpur': 'Jagdalpur',
'angul': 'Angul',
'baroda': 'Vadodara',
'ariyalur': 'Ariyalur',
'jowai': 'Jowai',
'neemrana': 'Neemrana',
'tirupathi': 'Tirupati',
'bhubneshwar': 'Bhubaneswar',
'calicut': 'Kozhikode',
'gandhinagar': 'Gandhinagar',
'dubai': 'Dubai',
'ahmednagar': 'Ahmednagar',
'nashik': 'Nashik',
'bellary': 'Bellary',
'ludhiana': 'Ludhiana',
'muzaffarnagar': 'Muzaffarnagar',
'gagret': 'Gagret',
```

```
'indirapuram, ghaziabad': 'Ghaziabad',
'gwalior': 'Gwalior',
'chennai & mumbai': 'Chennai',
'rajasthan': 'Rajasthan',
'sonipat': 'Sonipat',
'bareli': 'Bareli',
'hospete': 'Hospete',
'miryalaguda': 'Miryalaguda',
'dharuhera': 'Dharuhera',
'meerut': 'Meerut',
'ganjam': 'Ganjam',
'hubli': 'Hubli',
'ncr': 'NCR',
'agra': 'Agra',
'trichy': 'Tiruchirappalli',
'kudankulam ,tarapur': 'Kudankulam',
'ongole': 'Ongole',
'sambalpur': 'Sambalpur',
'pondicherry': 'Puducherry',
'bundi': 'Bundi',
'sadulpur,rajgarh,distt-churu,rajasthan': 'Rajasthan',
'am': 'Am',
'bikaner': 'Bikaner',
'vadodara': 'Vadodara',
'india': 'India',
'asansol': 'Asansol',
'tirunelvelli': 'Tirunelveli',
'ernakulam': 'Ernakulam',
'bilaspur': 'Bilaspur',
'chandrapur': 'Chandrapur',
'nanded': 'Nanded',
'dharmapuri': 'Dharmapuri',
'vandavasi': 'Vandavasi',
'rohtak': 'Rohtak',
'patna': 'Patna',
'salem': 'Salem',
'nasikcity': 'Nashik',
'technopark, trivandrum': 'Trivandrum',
'bharuch': 'Bharuch',
'tornagallu': 'Tornagallu',
'jaspur': 'Jaspur',
'burdwan': 'Burdwan',
'shimla': 'Shimla',
'gajiabaad': 'Ghaziabad',
'jammu': 'Jammu',
'shahdol': 'Shahdol',
'muvattupuzha': 'Muvattupuzha',
'al jubail,saudi arabia': 'Al Jubail',
'kalmar, sweden': 'Kalmar',
'secunderabad': 'Secunderabad',
'a-64,sec-64,noida': 'Noida',
'ratnagiri': 'Ratnagiri',
'jhajjar': 'Jhajjar',
'gulbarga': 'Gulbarga',
'hyderabad(bhadurpally)': 'Hyderabad',
'nalagarh': 'Nalagarh',
'jeddah saudi arabia': 'Jeddah',
'chennai, bangalore': 'Chennai',
'jamnagar': 'Jamnagar',
'tirupati': 'Tirupati',
'gonda': 'Gonda',
'orissa': 'Odisha',
'kharagpur': 'Kharagpur',
'navi mumbai , hyderabad': 'Navi Mumbai',
'joshimath': 'Joshimath',
'bathinda': 'Bathinda',
'johannesburg': 'Johannesburg',
'kala amb': 'Kala Amb',
'karnal': 'Karnal',
'london': 'London',
'kota': 'Kota',
'dehraj': 'Dehradun',
'melbourne': 'Melbourne',
'moradabad': 'Moradabad',
'delhi-gurgaon': 'Delhi',
'ambala': 'Ambala',
'faridkot': 'Faridkot',
```

```
 'rohtak, haryana': 'Rohtak',
 'khammam': 'Khammam',
 'khurda': 'Khurda',
 'jhalawar': 'Jhalawar',
 'kaithal': 'Kaithal',
 'sonbhadra': 'Sonbhadra',
 'fatehgarh sahib': 'Fatehgarh Sahib',
 'kaithal-haryana': 'Kaithal',
 'bhilwara': 'Bhilwara',
 'coimbatore, tirupur': 'Coimbatore',
 'sri ganganagar': 'Sri Ganganagar',
 'manipal': 'Manipal',
 'tirupathi': 'Tirupati',
 'kharagpur, west bengal': 'Kharagpur',
 'kolkata': 'Kolkata',
 'trichy-tiruchirappalli': 'Tiruchirappalli',
 }


# df.jobcity = df.jobcity.str.strip().str.lower()
# unique_cities_cleaned = df['jobcity'].unique()
# print(unique_cities_cleaned)
# Use bracket notation to access the 'jobcity' column
df['JobCity'] = df['JobCity'].str.strip().str.lower()

# Get the unique cleaned cities
unique_cities_cleaned = df['JobCity'].unique()

# Print the unique cleaned cities
print(unique_cities_cleaned)
```

```
['bangalore' 'indore' 'chennai' 'gurgaon' 'manesar' 'hyderabad' 'banglore'
 'noida' 'kolkata' 'pune' nan 'mohali' 'jhansi' 'delhi' 'bhubaneswar'
 'navi mumbai' 'mumbai' 'new delhi' 'mangalore' 'rewari' 'gaziabaad'
 'bhiwadi' 'mysore' 'rajkot' 'greater noida' 'jaipur' 'thane'
 'maharajganj' 'thiruvananthapuram' 'punchkula' 'bhubaneshwar'
 'coimbatore' 'dhanbad' 'lucknow' 'trivandrum' 'gandhi nagar' 'una'
 'daman and diu' 'gurgoan' 'vsakhapttnam' 'nagpur' 'bhagalpur'
 'new delhi - jaisalmer' 'ahmedabad' 'kochi/cochin' 'bankura' 'bengaluru'
 'kanpur' 'vijayawada' 'kochi' 'beawar' 'alwar' 'siliguri' 'raipur'
 'bhopal' 'faridabad' 'jodhpur' 'udaipur' 'muzaffarpur' 'kolkata`'
 'bulandshahar' 'haridwar' 'raigarh' 'visakhapatnam' 'jabalpur' 'unnao'
 'aurangabad' 'belgaum' 'dehradun' 'rudrapur' 'jamshedpur' 'vizag' 'nouda'
 'dharamshala' 'banagalore' 'hissar' 'ranchi' 'madurai' 'gurga'
 'chandigarh' 'australia' 'cheyyar' 'sonepat' 'ghaziabad' 'pantnagar'
 'jagdalpur' 'angul' 'baroda' 'ariyalur' 'jowai'
 'kochi/cochin, chennai and coimbatore' 'neemrana' 'tirupathi'
 'bhubneshwar' 'calicut' 'gandhinagar' 'dubai' 'ahmednagar' 'nashik'
 'bellary' 'ludhiana' 'muzaffarnagar' 'gagret' 'indirapuram, ghaziabad'
 'gwalior' 'chennai & mumbai' 'rajasthan' 'sonipat' 'bareli' 'hospete'
 'miryalaguda' 'dharuhera' 'meerut' 'ganjam' 'hubli' 'ncr' 'agra' 'trichy'
 'kudankulam ,tarapur' 'ongole' 'sambalpur' 'pondicherry' 'bundi'
 'sadulpur,rajgarh,distt-churu,rajasthan' 'am' 'bikaner' 'vadodara'
 'india' 'asansol' 'tirunelvelli' 'ernakulam' 'bilaspur' 'chandrapur'
 'nanded' 'dharmapuri' 'vandavasi' 'rohtak' 'patna' 'salem' 'nasikcity'
 'technopark, trivandrum' 'bharuch' 'tornagallu' 'jaspur' 'burdwan'
 'shimla' 'gajiabaad' 'jammu' 'shahdol' 'muvattupuzha'
 'al jubail,saudi arabia' 'kalmar, sweden' 'secunderabad'
 'a-64,sec-64,noida' 'ratnagiri' 'jhajjar' 'gulbarga'
 'hyderabad(bhadurpally)' 'nalagarh' 'jeddah saudi arabia'
 'chennai, bangalore' 'jamnagar' 'tirupati' 'gonda' 'orissa' 'kharagpur'
 'navi mumbai , hyderabad' 'joshimath' 'bathinda' 'johannesburg'
 'kala amb' 'karnal' 'london' 'kota' 'panchkula' 'baddi hp' 'nagari'
 'mettur, tamil nadu' 'durgapur' 'pondi' 'surat' 'kurnool' 'kolhapur'
 'bhilai' 'hderabad' 'bahadurgarh' 'rayagada, odisha' 'kakinada'
 'varanasi' 'punr' 'nellore' 'sahibabad' 'howrah' 'trichur' 'ambala'
 'khopoli' 'keral' 'roorkee' 'allahabad' 'delhi/ncr' 'jalandhar' 'vapi'
 'pilani' 'muzzafarpur' 'ras al khaimah' 'bihar' 'singaruli' 'pondy'
 'phagwara' 'guragaon' 'baripada' 'yamuna nagar' 'shahibabad' 'sampla'
 'guwahati' 'rourkela' 'banaglore' 'vellore' 'dausa'
 'latur (maharashtra )' 'mainpuri' 'dammam' 'haldia' 'rae bareli'
 'patiala' 'gorakhpur' 'new dehli' 'ambala city' 'karad' 'rajpura'
 'haryana' 'asifabadbanglore']
```

```
df['DOL'] = df['DOL'].apply(lambda x: "Left" if x != "present" else x) # to find left and present numbers
```

```
df['DOL'].value_counts()
```

DOL	count
Left	2123
present	1875

dtype: int64

```
def numerical_univariets_analysis(numerical_data):
  for col_name in numerical_data:
    print("*"*10,col_name,"*"*10)
    print(numerical_data[col_name].agg(['min','max','mean','median','std','skew','kurt']))
    print()
```

```
df.head()
```

	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DOB	10percentage	10board	...	ComputerScience	MechanicalEngg
0	203097	420000	2012-06-01	present	senior quality engineer	bangalore	f	1990-02-19	84.3	board ofsecondary education,ap	...	-1	-1
1	579905	500000	2013-09-01	present	assistant manager	indore	m	1989-10-04	85.4	cbse	...	-1	-1
2	810601	325000	2014-06-01	present	systems engineer	chennai	f	1992-08-03	85.0	cbse	...	-1	-1
3	267447	1100000	2011-07-01	present	senior software engineer	gurgaon	m	1989-12-05	85.6	cbse	...	-1	-1
4	343523	200000	2014-03-01	Left	get	manesar	m	1991-02-27	78.0	cbse	...	-1	-1

5 rows × 38 columns

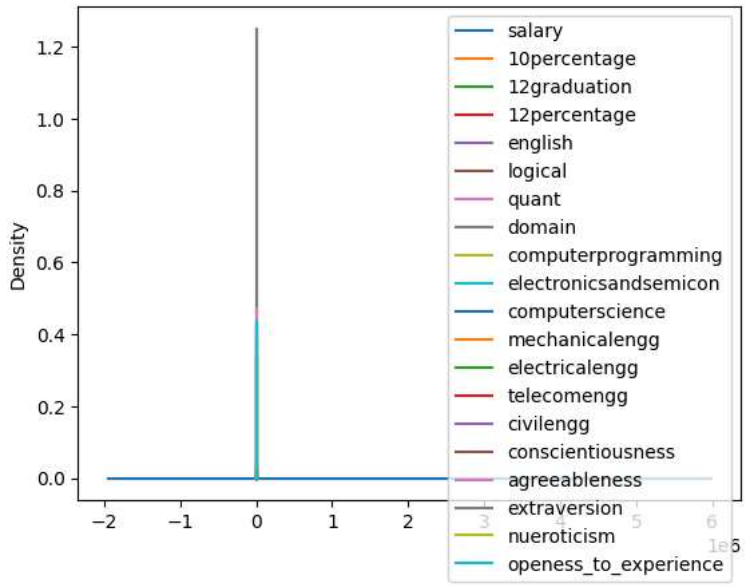
```
df.columns = df.columns.str.strip().str.lower()  # Remove spaces and convert to lowercase
numerical_univariets_analysis(df[['salary', '10percentage', '12graduation', '12percentage',
                                  'english', 'logical', 'quant', 'domain',
                                  'computerprogramming', 'electronicsandsemicon',
                                  'computerscience', 'mechanicalengg',
                                  'electricalengg', 'telecomengg',
                                  'civilengg', 'conscientiousness',
                                  'agreeableness', 'extraversion',
                                  'nueroticism', 'openess_to_experience']])
```

```
********** salary **********
min       3.500000e+04
max       4.000000e+06
mean      3.076998e+05
median    3.000000e+05
std       2.127375e+05
skew      6.451081e+00
kurt      8.093000e+01
Name: salary, dtype: float64

********** 10percentage **********
min       43.000000
max       97.760000
mean      77.925443
median    79.150000
std        9.850162
skew      -0.591019
kurt      -0.110284
Name: 10percentage, dtype: float64

********** 12graduation **********
min       1995.000000
max       2013.000000
mean      2008.087544
median    2008.000000
std          1.653599
skew        -0.964090
kurt         1.951164
Name: 12graduation, dtype: float64
```

```
********** 12percentage **********
min       40.000000
max       98.700000
mean      74.466366
median    74.400000
std       10.999933
skew      -0.032607
kurt      -0.630737
Name: 12percentage, dtype: float64

********** english **********
min       180.000000
max       875.000000
mean      501.649075
median    500.000000
std       104.940021
skew        0.191997
kurt       -0.254133
Name: english, dtype: float64

********** logical **********
min       195.000000
max       795.000000
mean      501.598799
median    505.000000
std        86.783297
skew       -0.216602
kurt       -0.224761
```

```
df[['salary','10percentage','12graduation', '12percentage',
       'english', 'logical', 'quant', 'domain', 'computerprogramming',
       'electronicsandsemicon', 'computerscience', 'mechanicalengg',
       'electricalengg', 'telecomengg', 'civilengg', 'conscientiousness',
       'agreeableness', 'extraversion', 'nueroticism',
       'openess_to_experience']].plot(kind="kde")
```

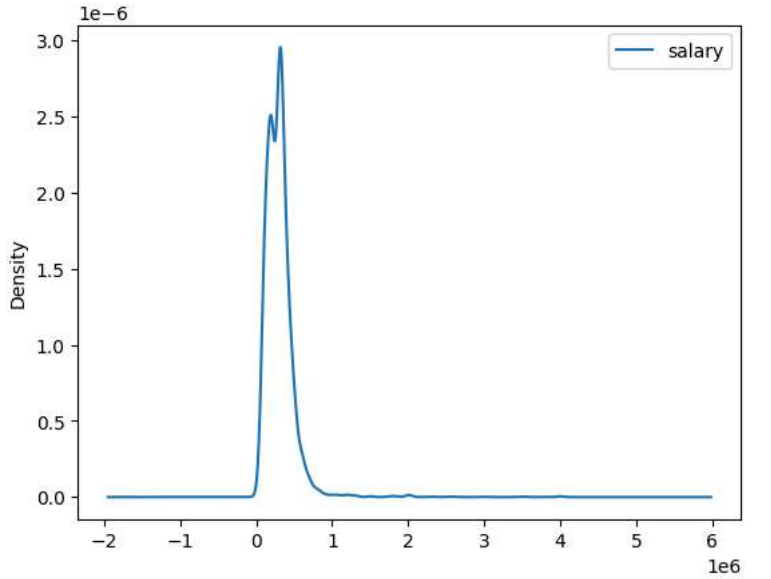
```
<Axes: ylabel='Density'>
```

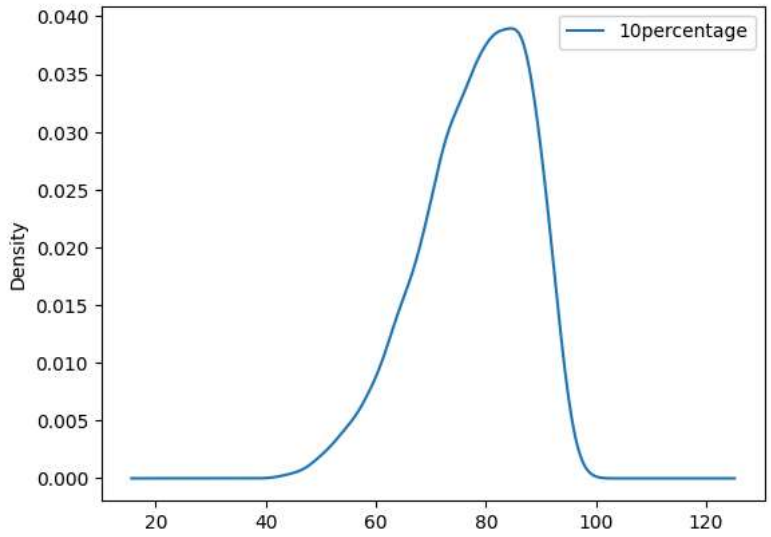



```
df[['salary']].plot(kind="kde")
df[['10percentage']].plot(kind="kde")
df[['12graduation']].plot(kind="kde")
df[['12percentage']].plot(kind="kde")
df[['english']].plot(kind="kde")
df[['logical']].plot(kind="kde")
df[['quant']].plot(kind="kde")
df[['domain']].plot(kind="kde")
df[['computerprogramming']].plot(kind="kde")
df[['electronicsandsemicon']].plot(kind="kde")
df[['computerscience']].plot(kind="kde")
df[['mechanicalengg']].plot(kind="kde")
df[['electricalengg']].plot(kind="kde")
df[['telecomengg']].plot(kind="kde")
df[['conscientiousness']].plot(kind="kde")
df[['10percentage']].plot(kind="kde")
```

```
df[['agreeableness']].plot(kind="kde")
df[['extraversion']].plot(kind="kde")
df[['nueroticism']].plot(kind="kde")
df[['openess_to_experience']].plot(kind="kde")
```

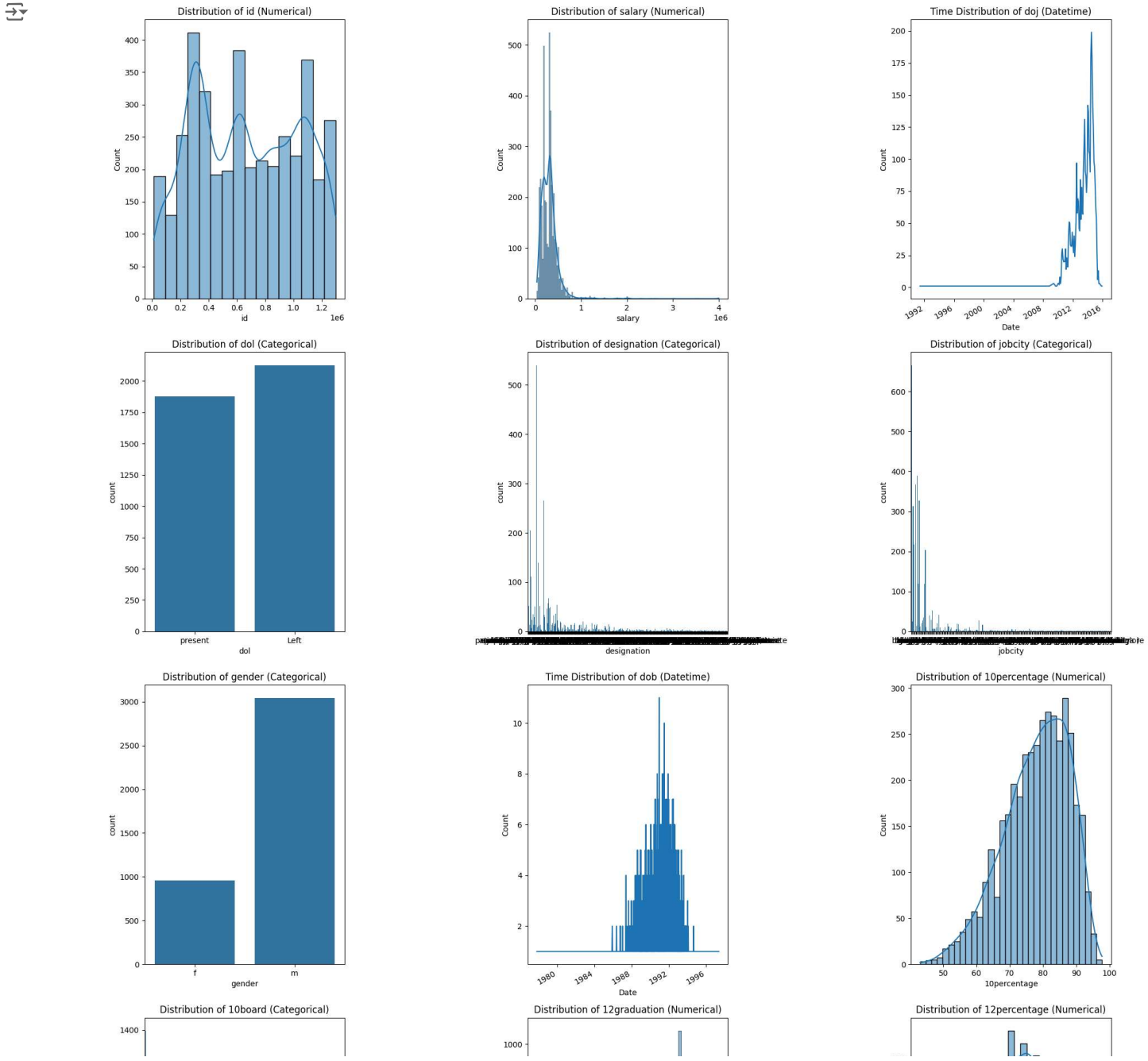
```
<Axes: ylabel='Density'>
```






```
n_cols = len(df.columns)
n_rows = int(np.ceil(n_cols / 3))  # 3 columns per row for better layout
fig, axes = plt.subplots(n_rows, 3, figsize=(20, n_rows * 6))
axes = axes.flatten()  # Flatten the axes array for easier indexing

for i, col in enumerate(df.columns):
    # Check if the column is categorical
    if df[col].dtype == 'object' or df[col].dtype.name == 'category':
        # Categorical column - use countplot
        sns.countplot(x=col, data=df, ax=axes[i])
        axes[i].set_title(f'Distribution of {col} (Categorical)')

    # Check if the column is datetime
    elif pd.api.types.is_datetime64_any_dtype(df[col]):
```

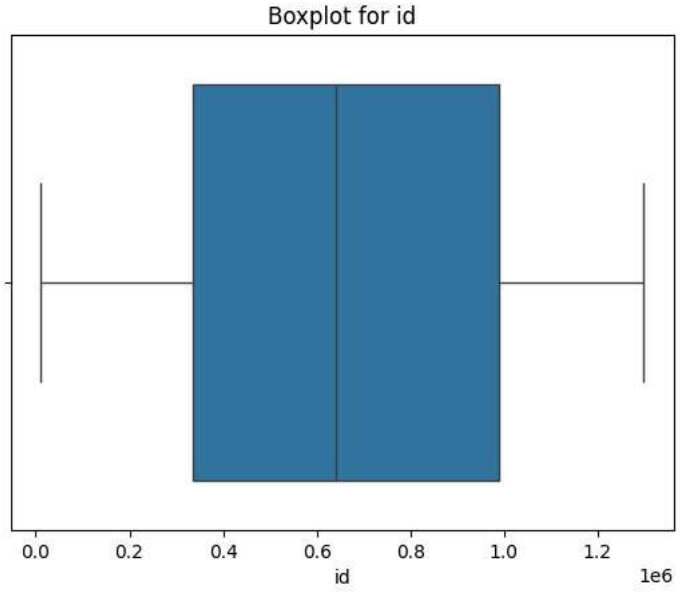
```
        # Datetime column - convert to datetime and plot time distribution
        df[col] = pd.to_datetime(df[col])
        df[col].value_counts().sort_index().plot(ax=axes[i])
        axes[i].set_title(f'Time Distribution of {col} (Datetime)')
        axes[i].set_xlabel('Date')
        axes[i].set_ylabel('Count')

    # Check if the column is numerical
    elif pd.api.types.is_numeric_dtype(df[col]):
        # Numerical column - use histplot
        sns.histplot(df[col], kde=True, ax=axes[i])
        axes[i].set_title(f'Distribution of {col} (Numerical)')

# Hide unused axes if fewer columns than subplots
for j in range(i + 1, len(axes)):
    axes[j].axis('off')

# Adjust layout for better spacing between subplots
plt.tight_layout()
plt.show()
```



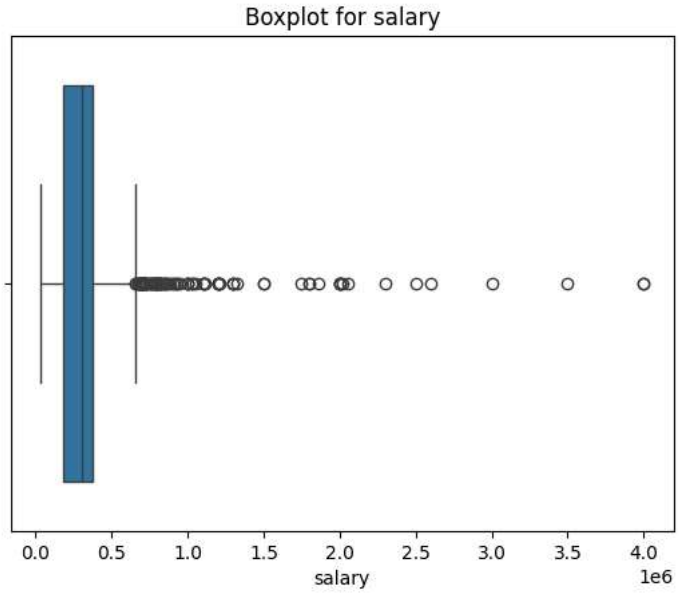

```
for i in df.columns:
  if df[i].dtype=="int" or df[i].dtype=="float":
```

```
    sns.boxplot(x=df[i])
    plt.title("Boxplot for {}".format(i))
    plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

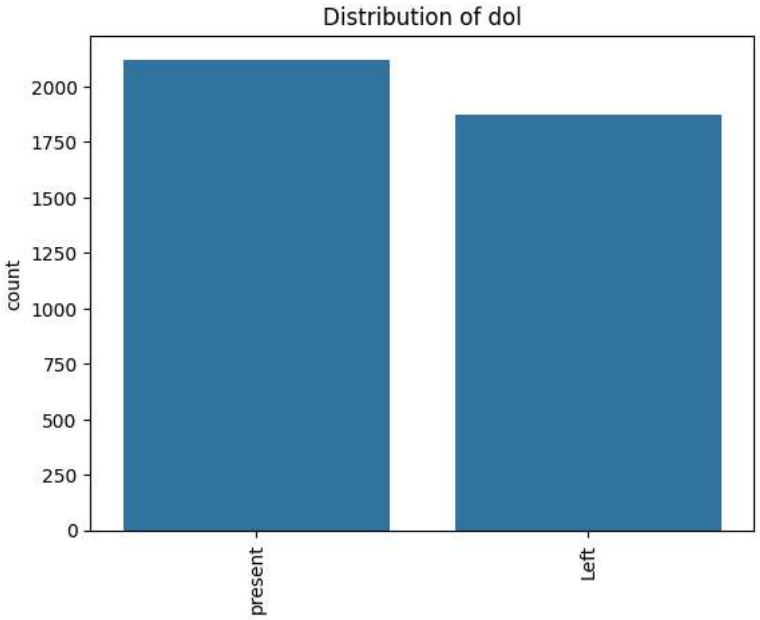


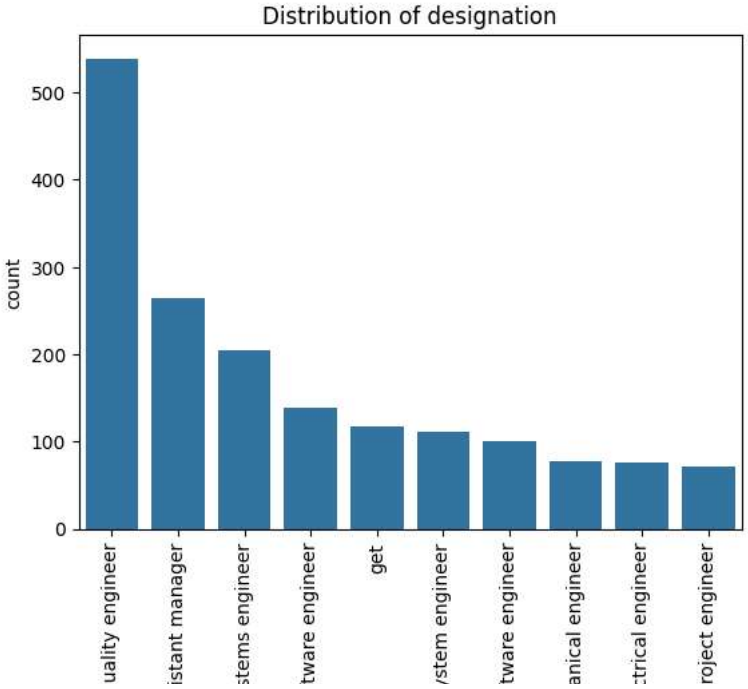

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```




```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be re
```

```
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()
```



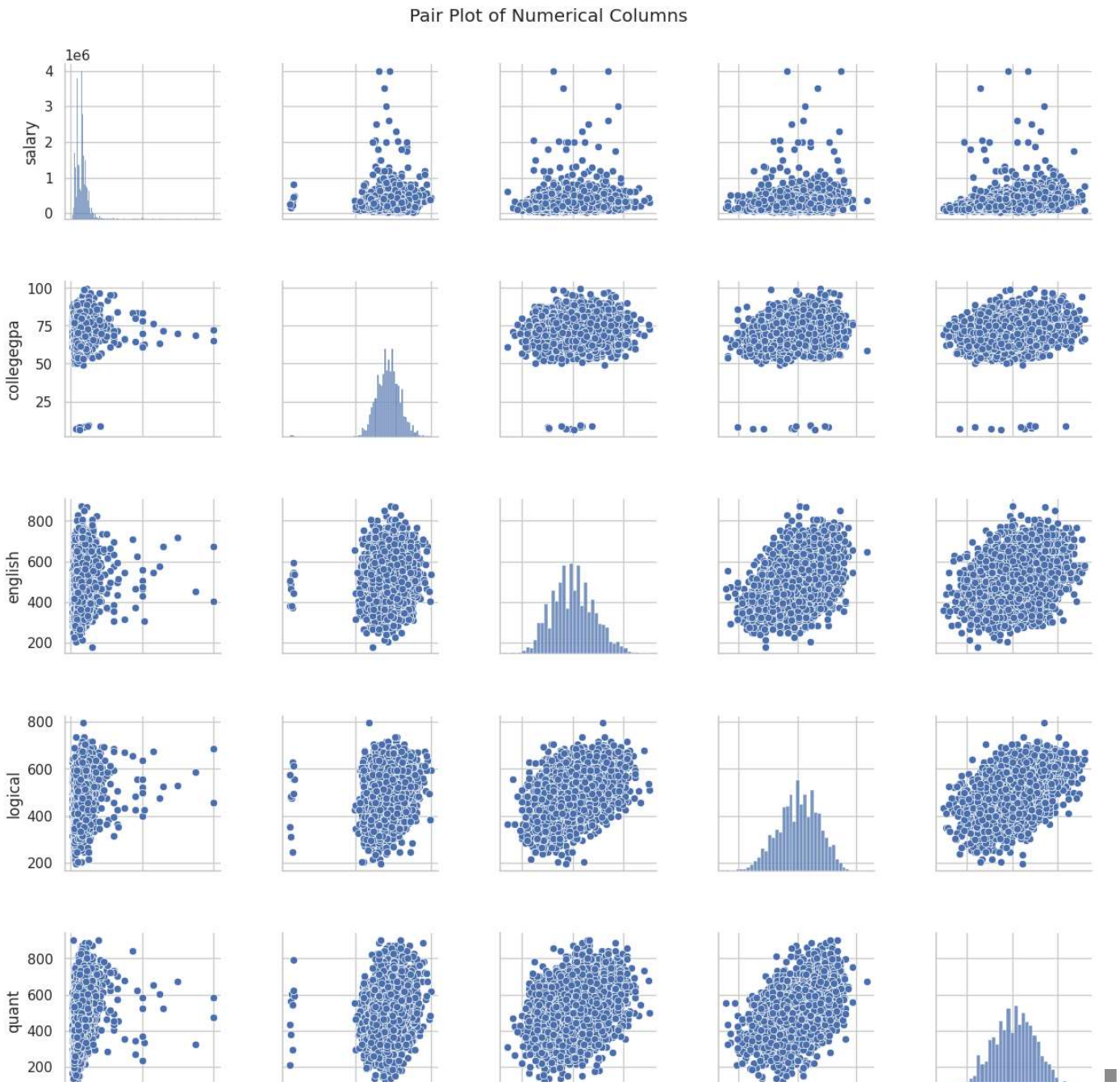


Start coding or generate with AI.

Start coding or generate with AI.

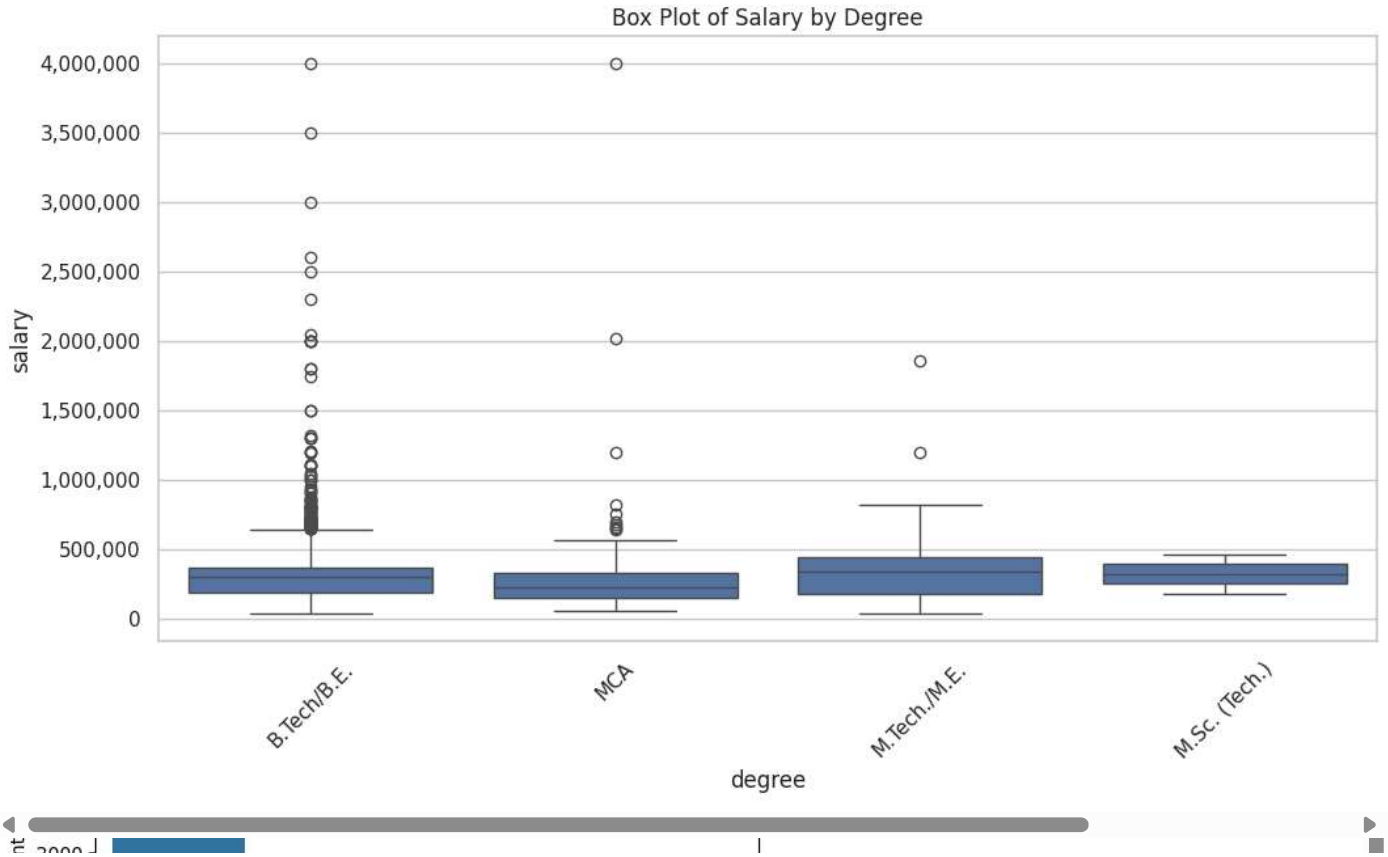
```
# Bivariate Analysis
# Discover the relationships between numerical columns using Scatter plots, hexbin plots, pair plots, etc..
# Identify the patterns between categorical and numerical columns using swarmplot, boxplot, barplot, etc..
# Identify relationships between categorical and categorical columns using stacked bar plots.
# Mention observations after each plot.
```

```
numerical_columns = ['salary', 'collegegpa', 'english', 'logical', 'quant']
sns.set(style="whitegrid")
pair_plot = sns.pairplot(df[numerical_columns])
plt.suptitle('Pair Plot of Numerical Columns', y=1.02)
plt.subplots_adjust(hspace=0.4, wspace=0.4)
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```

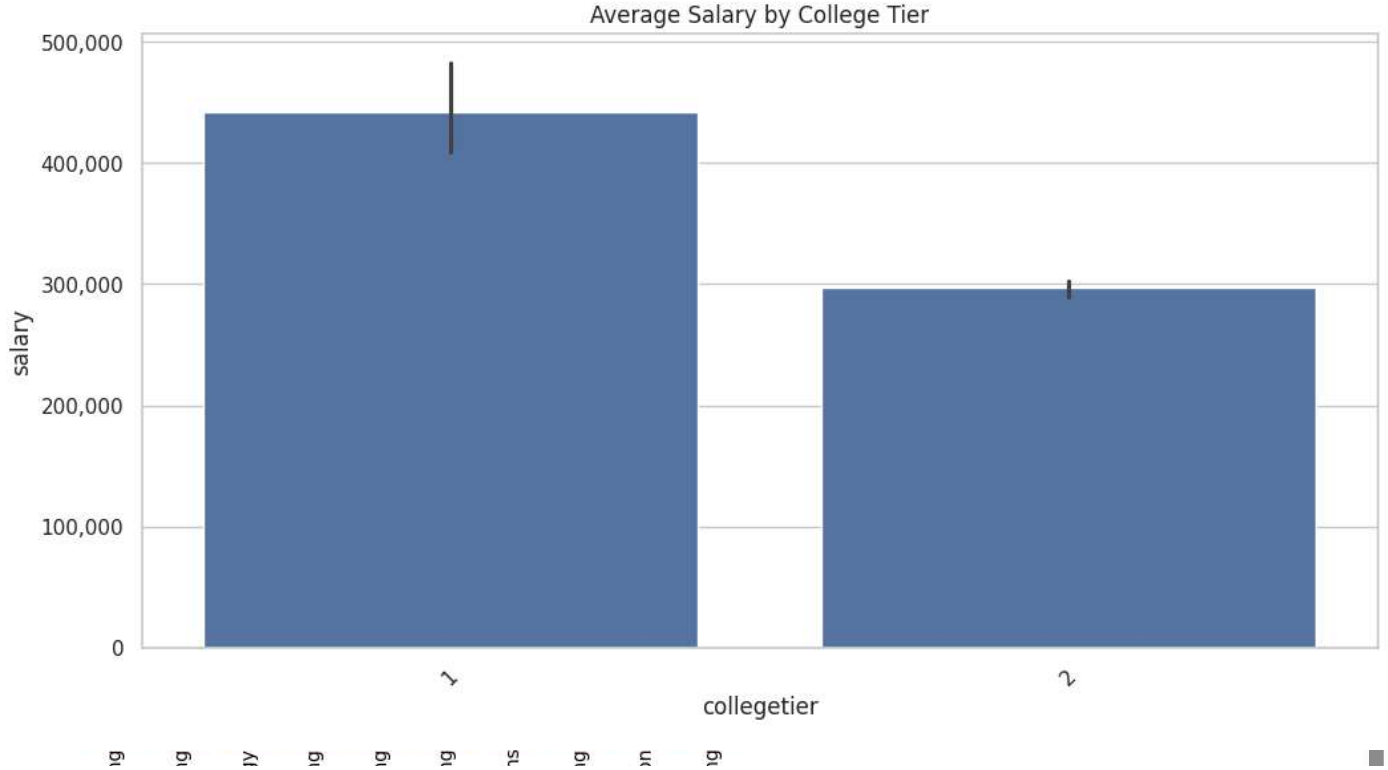



```
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='degree', y='salary')
plt.title('Box Plot of Salary by Degree')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/categorical.py:640: FutureWarning: SeriesGroupBy.grouper is deprecated and will be remov
  positions = grouped.grouper.result_index.to_numpy(dtype=float)
```

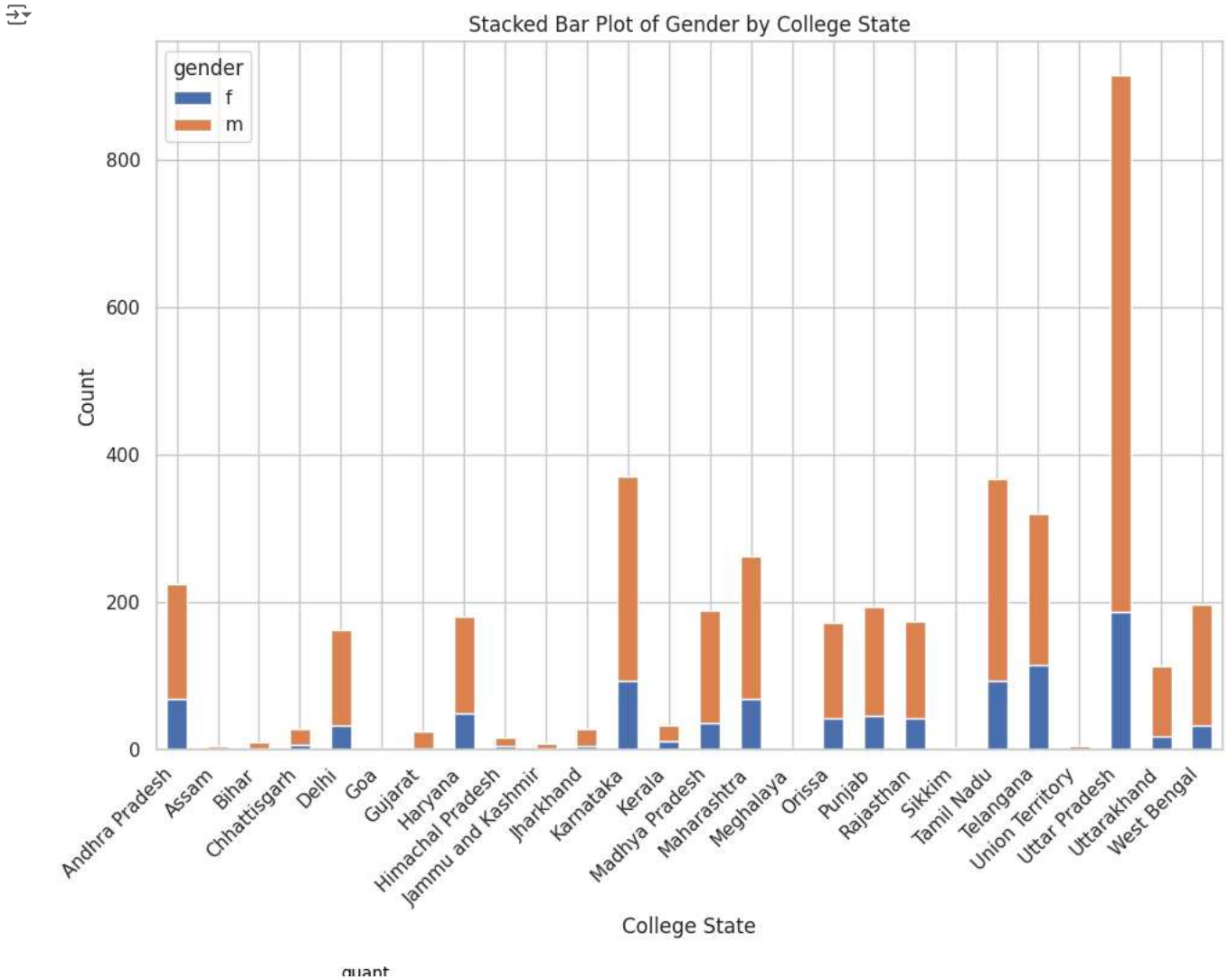



```
plt.figure(figsize=(12, 6))
sns.barplot(data=df, x='collegetier', y='salary', estimator=np.mean)
plt.title('Average Salary by College Tier')
plt.xticks(rotation=45)
plt.gca().yaxis.set_major_formatter(FuncFormatter(currency))
plt.show()
```




```
# Create a pivot table
pivot_table = df.pivot_table(index='collegestate', columns='gender',
```

```
                            values='salary', aggfunc='count').fillna(0)

# Plot the stacked bar plot
pivot_table.plot(kind='bar', stacked=True, figsize=(10, 8))
plt.title('Stacked Bar Plot of Gender by College State')
plt.xlabel('College State')
plt.ylabel('Count')
plt.xticks(rotation=45, ha='right')  # Adjusted alignment to 'right'
plt.tight_layout()  # Adjust layout to prevent clipping
plt.show()
```



quant

```
job_titles = ['Programming Analyst', 'Software Engineer', 'Hardware Engineer','Associate Engineer']
salary_data = df[df['designation'].isin(job_titles)]
 # Calculate the average salary for each job title
average_salaries = salary_data.groupby('designation')['salary'].mean().reset_index()
# Check if average salaries are within the claimed range of 2.5 to 3 Lakhs
average_salaries['within_claimed_range'] = average_salaries['salary'].apply(lambda x: 2.5 <= x <= 3)
print("Average Salaries for Specified Job Titles:")
print(average_salaries)
print("\nAverage Salaries within Claimed Range:")
print(average_salaries[average_salaries['within_claimed_range']])
```

```
Average Salaries for Specified Job Titles:
Empty DataFrame
Columns: [designation, salary, within_claimed_range]
Index: []

Average Salaries within Claimed Range:
Empty DataFrame
Columns: []
Index: []
```

```
import pandas as pd
from scipy import stats

# Create a contingency table
contingency_table = pd.crosstab(df['gender'], df['specialization'])

# Display the contingency table
print("Contingency Table:")
print(contingency_table)

# Perform Chi-Square test
chi2_stat, p_value, dof, expected = stats.chi2_contingency(contingency_table)

# Create a results DataFrame with reset index
results = pd.DataFrame({
    'Metric': ['Chi-Squared Statistic', 'P-value', 'Degrees of Freedom', 'Conclusion'],
    'Value': [
        chi2_stat,
        p_value,
        dof,
        "Reject the null hypothesis" if p_value < 0.05 else "Fail to reject the null hypothesis"
    ]
})

# Reset the index of the results DataFrame
results.reset_index(drop=True, inplace=True)

# Display the results
print("Chi-Square Test Results:")
print(results)
```

```
Contingency Table:
specialization  aeronautical engineering  \
gender
f                                       1
m                                       2

specialization  applied electronics and instrumentation  \
gender
f                                                     2
m                                                     7

specialization  automobile/automotive engineering  biomedical engineering  \
gender
f                                               0                       2
m                                               5                       0

specialization  biotechnology  ceramic engineering  chemical engineering  \
gender
f                            9                    0                     1
m                            6                    1                     8

specialization  civil engineering  computer and communication engineering  \
gender
f                               6                                        0
m                              23                                        1

specialization  computer application  ...  internal combustion engine  \
gender                                  ...
f                                  59  ...                           0
m                                 185  ...                           1

specialization  mechanical & production engineering  \
gender
f                                                 0
m                                                 1

specialization  mechanical and automation  mechanical engineering  \
gender
f                                       0                      10
m                                       5                     191

specialization  mechatronics  metallurgical engineering  other  \
gender
f                           1                          0      0
```