INNOVATION. AUTOMATION. ANALYTICS

PROJECT ON

"Exploratory Data Analysis on AMEO Dataset"

Sakshi Yogendra Yadav

22nd February 2024

1 About Me

A passionate student of Data Science & Artificial Intelligence. My interest in this profession stemmed from my deep fascination with data's ability to clarify difficult issues and provide insightful new information. I have a background in statistics and am now working for a master's degree in data science and artificial intelligence. I am excited in delving into the complexity of data and using cutting-edge analytical methods to draw insightful conclusions. I am continuously motivated by the challenge of turning raw data into usable intelligence, which is why I work in statistical modeling, machine learning techniques, and data visualization.

Additionally, to being a career, data science is a deeply held passion of mine that drives me to constantly research, educate myself, and come up with new ideas in this dynamic and always changing field.

In addition to, I am driven to constantly learn and adjust to new technologies and processes by the field's ever-evolving nature and the wide range of applications it has across industries. In the end, what motivates me is the possibility of having a real influence through the use of data-driven methods to spur creativity, guide judgment, and solve.

LinkedIn Profile: **linkedin**

Github Profile: **github**

Link to Project Repo: **project**

I. INTRODUCTION

Aspiring Minds produced the **Aspiring Minds Employment Outcome** (AMEO) dataset, which provides a thorough compilation of data about engineering graduates' employment results. This dataset, which was created especially to examine the job environment for engineering specialties, is an invaluable tool for learning about the variables affecting graduates' prospects for employment and other career outcomes.

II. OBJECTIVE OF THE PROJECT

To obtain a thorough understanding of the factors impacting pay offers for engineering graduates, **the Aspiring Minds Employment Outcome (AMEO)** dataset with **Salary** as the goal variable is being analyzed. This analysis aims to gain insights and understanding from the provided dataset, particularly focusing on the relationship between various features and the target variable, which is **Salary**.

Specifically, the goals of this analysis include:

- **Describing** the dataset and its features comprehensively.
- **Identifying** any **patterns** or **trends** present in the data.
- Exploring the **relationships** between independent and target variables (Salary).
- Identifying any **outliers** or anomalies in the data.

1.1 III. OVERVIEW OF DATA

Origin: Aspiring Minds performed the Aspiring Minds Employment Outcome 2015 (AMEO) study, which is where the dataset originated.

Range: The dataset, which mostly focuses on students with engineering credentials, records many aspects of their job search, such as compensation offers, job titles, locations, and standardized test results for cognitive, technical, and personality qualities.

Size: The dataset provides a broad and diversified set of data for research, with about 4000 data points and about 40 independent variables.

A. Dependent Variables:

Salary: Reflecting the annual Cost to Company (CTC) offered to candidates.

Job Titles: Detailing the 'Designations' offered in job placements.

Job Locations: Identifying the geographical locations (cities & states) of job placements.

B. Independent Variables:

Educational Background: Including grades, board curriculums, college GPA, graduation year, college tier, degree pursued, and specialization.

Demographic Information: Gender and date of birth.

Standardized Test Scores: Covering areas such as English, logical reasoning,

quantitative aptitude, computer programming, and various engineering disciplines.

Personality Traits: Assessing traits such as conscientiousness, agreeableness, extraversion, neuroticism, and openness to experience.

1.2 IV. DATA CLEANING AND PREPROCESSING

1.2.1 A. Data Conversion

Several transformations were used to improve the Aspiring Minds Employment Outcome (AMEO) dataset's usability and analytical potential during the data conversion and pretreatment stages of the analysis process. First, datetime format was applied to the Date of Birth (DOB) and Date of Joining (DOJ) variables. Time-based analytics and trend identification are made easier by this conversion, which allows the dataset to accurately capture temporal information. The dataset is made more structured and suitable for time series analysis by converting these variables into a standardized datetime format. This allows researchers to investigate connections between employment outcomes and temporal aspects like tenure and age at joining. Furthermore, the Date of Leaving (DOL) variable was adjusted to include a present value, indicating that individuals who have not left their current positions have a DOL value replaced by the current date (today_date()).

1.2.2 B. Case Transformation

Executed a transformation on all categorical feature variables by converting them to lowercase. This comprehensive adjustment aimed to standardize the representation of categorical data throughout the dataset, thereby enhancing its consistency and facilitating more robust analyses.

1.2.3 C. Collapsing Categories

Categorical variables that had values of 0 or-1, which mean they were incomplete or missing data, were combined into one category called "others." These numbers frequently indicate situations in which respondents omitted information or for which data was unavailable, which, if ignored, could distort studies.

1.3 FEATURE ENGINEERING

Age Calculation:

By deducting the year of birth (DOB) from 2024, an extra column denoting age has been added to the dataset, indicating the individual's age as of that year.

1.4 V. EXPLORATORY DATA ANALYSIS

1.4.1 A. Univariate analysis

1. Categorical Features:

1.4.1.1 Designation

The AMEO dataset's horizontal bar plot of the "Designation" variable, which has 373 distinct values, offered useful information regarding the distribution of career designations among engineering graduates.

The title that is most frequently seen is "Software Engineer," closely followed by "System Engineer" and "Software Developer."

1.4.1.2 JobCity

We conclude that there is a significant variation in the job locations of engineering graduates in the AMEO dataset, with 216 distinct values in the "JobCity" variable. But after some study, we determined which cities were best for finding jobs; Bangalore came out on top, followed by Hyderabad and Noida. Note: Since the other category received no responses, we disregarded it.

1.4.1.3 Gender

The gender distribution of candidates in the AMEO dataset reveals a considerable class imbalance, with a significantly higher number of male applicants than female candidates. In particular, there are 744 female candidates and 2358 male candidates in the sample. Determining that the male category is actually bigger than the female one.

1.4.1.4 Degree

The distribution of educational backgrounds in the AMEO dataset indicates that B.Tech/B.E. degrees are overwhelmingly common, accounting for 2868 of the total. Students from MCA, M.Tech/M.E, and M.Sc. (Tech.) degrees, on the other hand, are significantly underrepresented—1961, 41, and only 2 people, respectively.

1.4.1.5 Specialization

"Electronics and Communication Engineering" is the most common engineering specialization in the AMEO dataset, with 717 persons belonging to this category. This specialization covers topics including wireless communication, signal processing, telecommunications, analog and digital electronics, and the design, development, and application of electronic devices and communication systems. "Computer Science & Engineering," with 587 people, is just behind. The study of computer systems, software development, algorithms, and programming languages are at the center of this specialization. With 504 people, "Information Technology" is yet another noteworthy specialization in the dataset. Information technology pertains to the administration, setup, and upkeep of IT infrastructure and systems in businesses.

1.4.1.6 CollegeState

The AMEO dataset contains 26 distinct values for the "CollegeState" variable, which suggests a heterogeneous distribution of colleges among various states. With 690 people having finished their schooling there, Uttar Pradesh is the most common state among them. Karnataka, Tamil Nadu, and Telangana are noteworthy states with 278, 276, and 259 people, respectively, after Uttar Pradesh. This state-by-state breakdown of colleges offers information about the geographic distribution of academic establishments that support the engineering labor force.

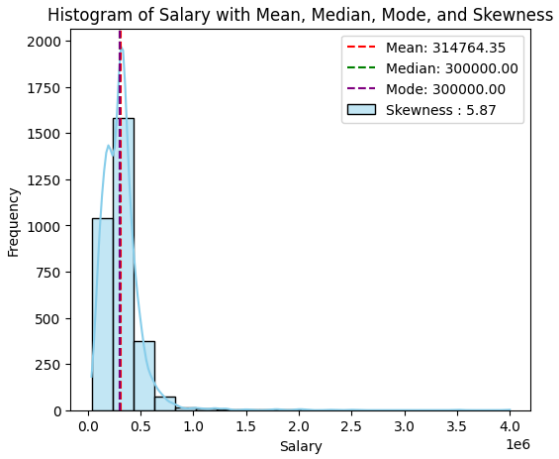
1.4.1.7 10^{th}Board & 12^{th}Board

In the AMEO dataset, there are 221 unique values for the "10board" variable and 266 unique values for the "12board" variable. Of these, state board and CBSE are the most common boards for both 10th and 12th grades; the other boards make up a lower percentage of the dataset.

2. Numerical Features:

2.1. Salary

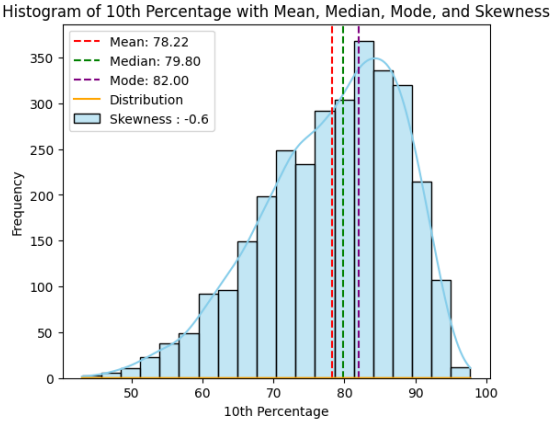
The AMEO dataset's salary summary statistics offer important information about the range of compensation packages offered to engineering graduates. The dataset includes 3,102 records that show variations in compensation offers across the sample. The mean income is around 314,764.30 INR, and the standard deviation is roughly 199,058.90 INR. The Salary data illustrates the wide range of compensation packages available to engineering graduates, with minimum income of 35,000 INR and maximum salary of 4,000,000 INR. Furthermore, the median 50% of wage offers fall within the interquartile range (IQR), which extends from the 25th percentile value of 200,000 INR to the 75th percentile value of 380,000 INR. The 50th percentile, or median pay, is 300,000 INR. This means that half of the salary offers in the dataset are below this figure and the other half are beyond it. This statistic offers a reliable way to quantify central tendency, particularly for datasets with skewed distributions.



With a skewness value of roughly 6, the data shows strong positive skewness, which deviates from a normal distribution. The mean, median, and mode—the three central tendency measurements—are about equal.

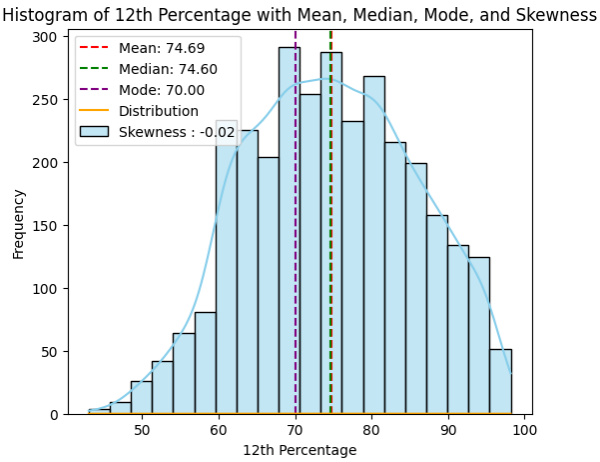
2.2. 10percentage

The average percentage obtained in grade 10 exams stands at approximately 78.22%, with a standard deviation of around 9.73%. The minimum score recorded is 43%, while the maximum score is 97.76%. The 25th percentile falls at 72%, and the 75th percentile at 86%.



2.3. 12percentage

Exam results for grade 12 typically yield an average percentage of 74.69%, with a standard deviation of roughly 11.0 percent. 43% is the lowest recorded score, and 98.2% is the highest. At 66.4% and 83%, respectively, are the 25th and 75th percentiles.

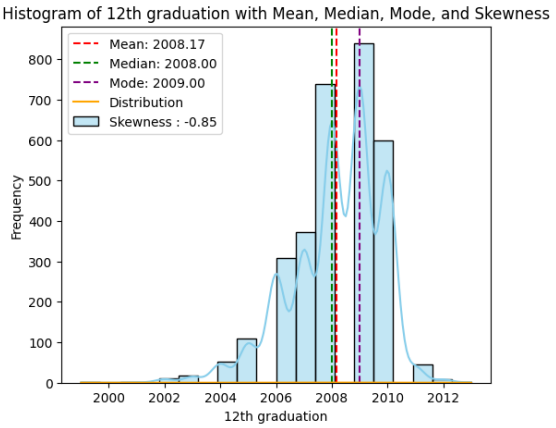


1.1. CollegeTier

The CollegeTier variable, representing the tier of the college attended by graduates, has an average value of approximately 1.93, indicating that a majority of graduates attended colleges classified as Tier 2. The minimum tier recorded is 1, representing Tier 1 colleges, while the maximum tier recorded is 2, indicating Tier 2 colleges.

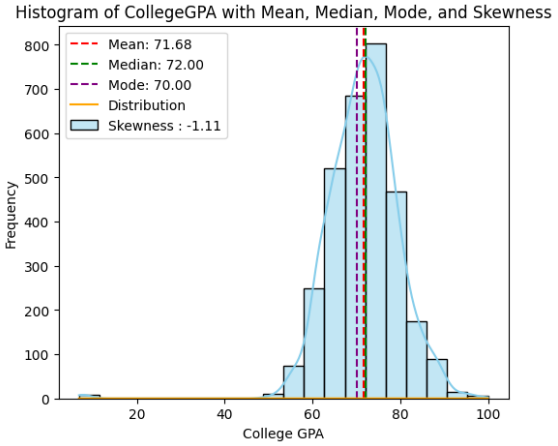
1.2. 12graduation

The average year of graduation from grade 12 is approximately 2008.17, with a standard deviation of approximately 1.61. The earliest graduation year recorded is 1999, while the most recent graduation year is 2013. The 25th percentile graduation year is 2007, and the 75th percentile is 2009. It shows a negative skew.



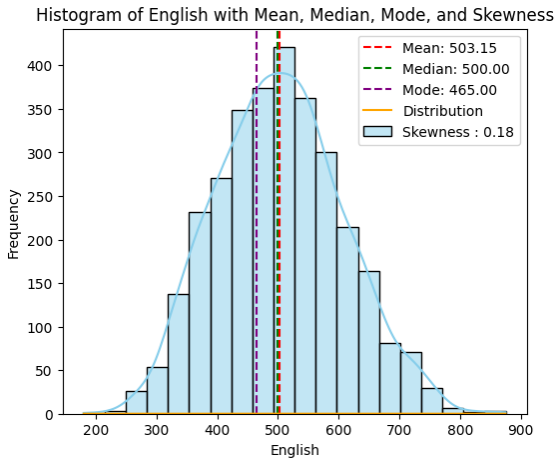
1.3. CollegeGPA

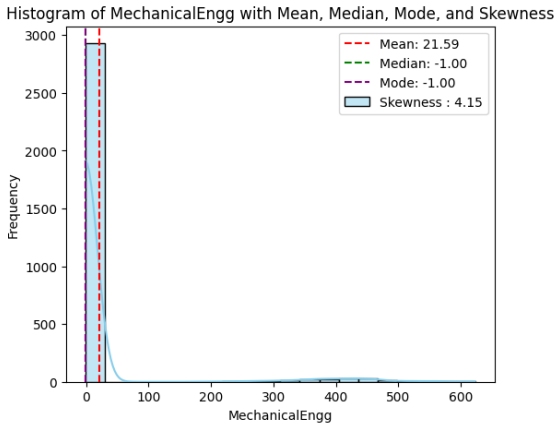
The average GPA stands at approximately 71.68, with a standard deviation of around 8.04. The minimum GPA recorded is 6.80, while the maximum GPA is 99.93. The 25th percentile falls at 66.69, and the 75th percentile at 76.50.

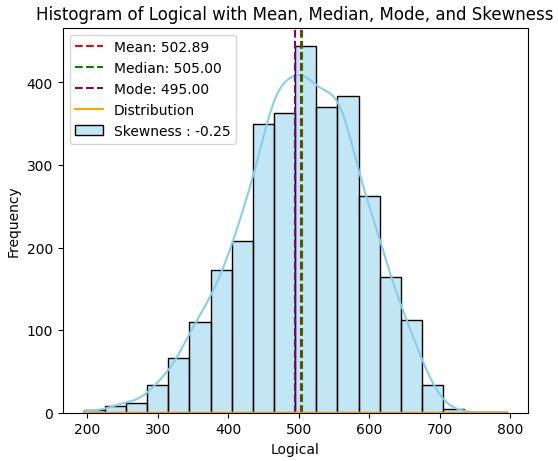


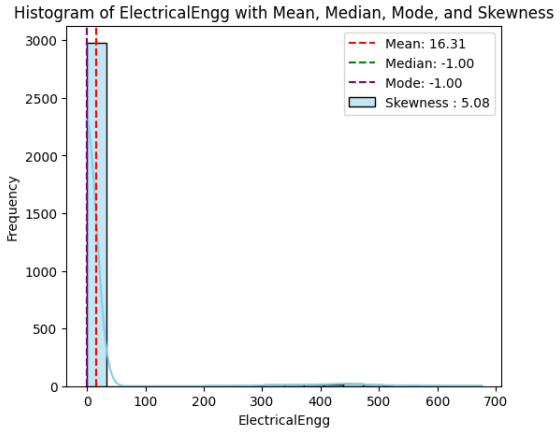
1.4. English & Logical Scores

The average scores for the English and Logical sections of the AMCAT test stand at approximately 503.15 and 502.89, respectively. The standard deviations are approximately 104.63 and 86.64, respectively. The minimum scores recorded for both sections are 180 and 195, respectively, while the maximum scores are 875 and 795, respectively.



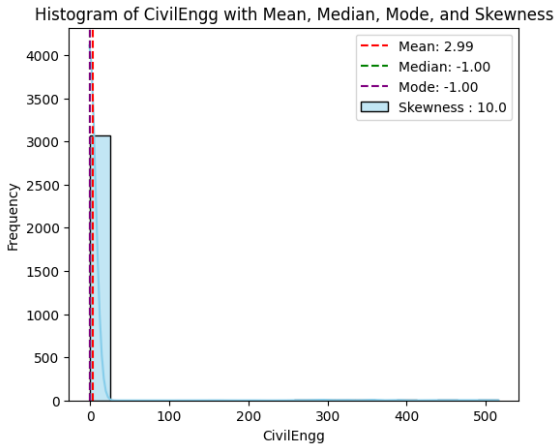




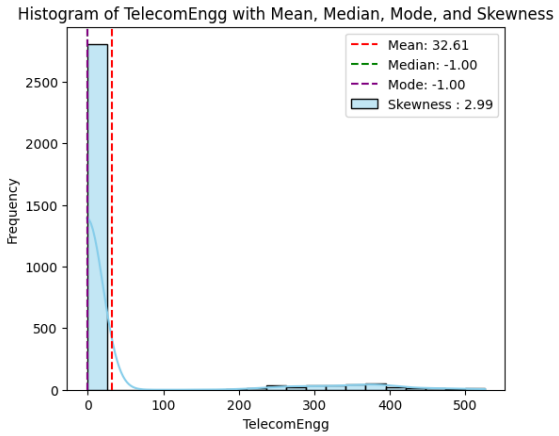


1.5. MechanicalEngg, ElectricalEngg, TelecomEngg, CivilEngg

These variables represent scores in different engineering disciplines within the AMCAT test. The mean scores are approximately 21.59, 16.31, 32.61, and 2.99, respectively. However, the data suggests that the majority of individuals did not take these specific sections, as indicated by the prevalence of -1 values.

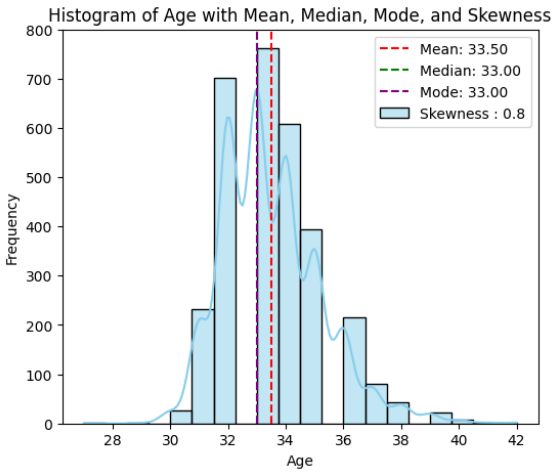


INNOMATICS RESEARCH LABS



1.6. Age

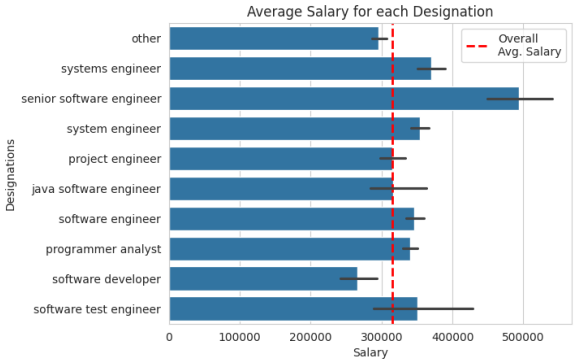
The "Age" column in the dataset offers insights into the ages of individuals, likely representing their ages at the time of data collection or assessment. With a total of 3102 entries, the average age of individuals stands at approximately 33.50 years, showcasing the central tendency of the age distribution. The standard deviation of approximately 1.72 years indicates the degree of spread or dispersion of ages around the mean, with ages ranging from a minimum of 27 years to a maximum of 42 years. The quartile values further delineate the age distribution: 25% of individuals are aged 32 years or younger (25th percentile), while 75% are aged 34 years or younger (75th percentile). The median age, representing the middle value of the dataset, is 33 years, indicating that half of the individuals are aged 33 years or younger.



A skewness value of 5.87 indicates that the distribution of the data in the "Age" column is highly positively skewed. Positive skewness means that the tail of the distribution is longer on the right side

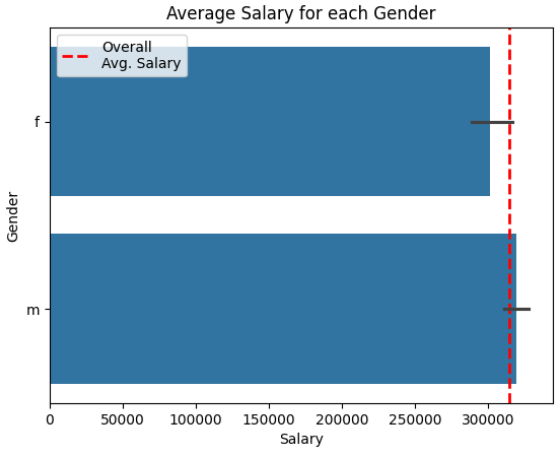
B. Bivariate analysis

1.4.2 1. Salary & Designation



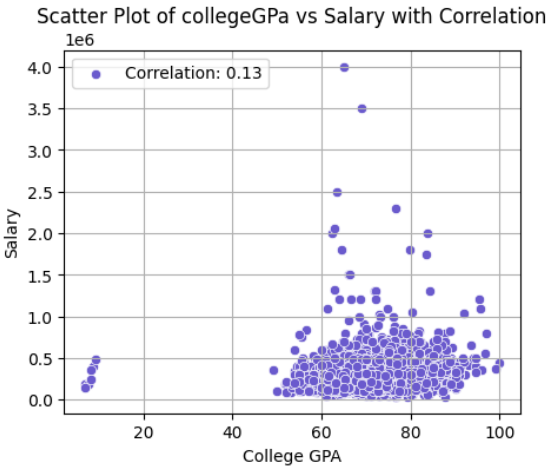
Senior Software Engineers have the highest salary, but also the highest standard deviation. Software Developers and Technical Support Engineers have salaries below the average.

1.4.3 2. Gender & Salary



Both male and female salaries are approximately equal on average, suggesting no gender bias overall, though females tend to receive salaries below the overall average.

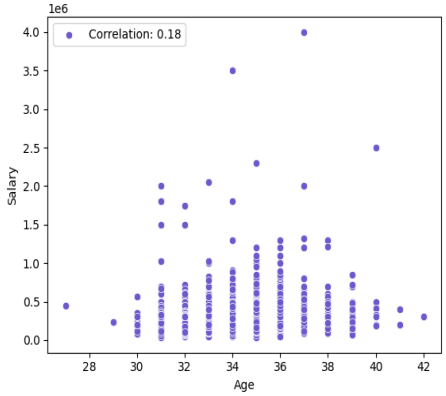
1.4.4 3. College GPA & Salary



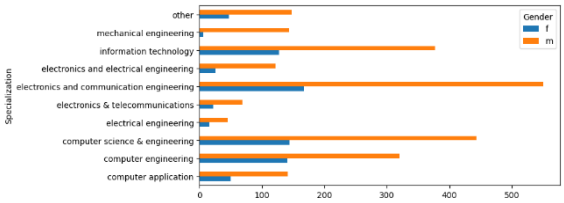
There is no significant correlation between salary and score in College GPA.

1.4.5 4. Age & Salary

There's no apparent relationship between age and salary after removing outliers.



1.4.6 5. Gender & Specialization



Male participation is approximately double that of females across all specializations, with fewer females opting for mechanical and electronics.

1.4.7 VI. RESEARCH OUTCOMES

1. ***"Times of India article dated Jan 18, 2019 states that "After doing your Computer Science Engineering if you take up jobs as a Programming Analyst, Software Engineer, Hardware Engineer and Associate Engineer you can earn up to 2.5-3 lakhs as a fresh graduate."***

Designation	t-value	p -value	Result
Programmer Analyst	9.75	2.65118e-10	Reject Null Hypothesis
Software Engineer	7.58	6.09466e-12	Reject Null Hypothesis
Hardware Engineer	NaN	NaN	Not Enough Evidence
Associate Engineer	NaN	NaN	Not Enough Evidence

The analysis begins by grouping the dataset by job designation, calculating the mean and standard deviation of salaries for each job role. This provides insights into salary distribution across different designations. Notably, Software Engineers have the highest mean salary and standard deviation, indicating both higher earnings and variability in pay compared to Programmer Analysts and Associate Engineers.

Following this, a one-sample t-test is conducted for each job designation to compare their average salary against an expected range. For Programmer Analysts and Software Engineers, the test results show sufficient evidence to reject the null hypothesis, suggesting that their salaries significantly differ from the expected range. However, for Hardware Engineers and Associate Engineers, there is not enough evidence to reject the null hypothesis, indicating that their salaries may not significantly deviate from the expected range.

Overall, these analyses provide valuable insights into salary distributions among different job roles and help in understanding the significance of salary differences within the dataset.

2. ***Is there a relationship between gender and specialization? (i.e. Does the preference of Specialization depend on the Gender?)***

Test	Value
chi2_critical	16.918977604620448
chi2_statistic	47.625462024510526
chi2_p_value	3.000409236234154e-07

The analysis conducted using a Chi-Square test examined the relationship between gender and specialization preferences. The test revealed a statistically significant relationship between the two variables, indicating that specialization preferences are dependent on gender.

The calculated chi2 statistic exceeded the critical value, and the p-value was significantly less than the chosen significance level, leading to the rejection of the null hypothesis. Therefore, there is sufficient evidence to conclude that gender and specialization are related, suggesting that certain fields may be more preferred or accessible to individuals of particular genders. This finding underscores the importance of considering gender diversity and inclusivity in various fields and highlights potential barriers or biases that may exist in certain specializations.

1.4.7.1 VII. CONCLUSION

The extensive data analysis yields several notable discoveries about the factors impacting pay levels in the dataset. While certain criteria, such college level, have a strong link with compensation, others, such as gender and academic performance, have no relationship.

Senior Software Engineers demand the greatest incomes, but with greater unpredictability, while Software Developers and Technical Support Engineers make less than the average. Gender does not appear to play a large impact in income determination on average, yet females do receive less than the total average salary. Academic performance, as measured by 10th, 12th, and college GPA scores, does not clearly correlate with pay levels.

In [ ]:

EDA of AMEO Data

2 1.1 Introduction

2.1 1.1.1 Dataset Description

The Aspiring Mind Employment Outcome 2015 (AMEO) dataset, released by Aspiring Minds, focuses on employment outcomes for engineering graduates. It includes dependent variables such as Salary, Job Titles, and Job Locations, along with standardized scores in cognitive skills, technical skills, and personality skills. With around 40 independent variables and 4000 data points, these variables encompass both continuous and categorical data. The dataset also includes demographic features and unique identifiers for each candidate.

2.2 1.1.2 Objective

The goal of this Exploratory Data Analysis (EDA) is to extensively investigate the provided dataset, with a particular emphasis on understanding the link between various variables and the target variable, Salary. The key aims of this analysis include:

1. Providing a detailed explanation of the dataset's features.
2. Find any observable patterns or trends in the data.
3. Investigating the relationships between the independent factors and the targetvariable(salary).
4. Identify any outliers or abnormalities in the dataset.
5. Offering practical insights and recommendations based on the analysis.

2.2.1 *AMCAT-Aspiring Minds Computer Adaptive Test*

In [ ]:

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

2.2.2 Importing data and viewing the data its shape and description of the data

```
data = pd.read_csv('data.csv')
data
```

Out[2]:

	Unnamed: 0	ID	Salary	DOJ	DOL	Designation	JobCity	Gender	DO

INNOMATICS

In [ ]:

0	train	203097	420000.0	6/1/12 0:00	present	senior quality engineer	Bangalore	f	2/19/ 0:
1	train	579905	500000.0	9/1/13 0:00	present	assistant manager	Indore	m	10/4/ 0:
2	train	810601	325000.0	6/1/14 0:00	present	systems engineer	Chennai	f	8/3/ 0:
3	train	267447	1100000.0	7/1/11 0:00	present	senior software engineer	Gurgaon	m	12/5/ 0:
4	train	343523	200000.0	3/1/14 0:00	3/1/15 0:00	get	Manesar	m	2/27/ 0:
...	...	...	...	...	...	...	...	...	
3993	train	47916	280000.0	10/1/11 0:00	10/1/12 0:00	software engineer	New Delhi	m	4/15/ 0:
3994	train	752781	100000.0	7/1/13 0:00	7/1/13 0:00	technical writer	Hyderabad	f	8/27/ 0:
3995	train	355888	320000.0	7/1/13 0:00	present	associate software engineer	Bangalore	m	7/3/ 0:
3996	train	947111	200000.0	7/1/14 0:00	1/1/15 0:00	software developer	Asifabadbanglore	f	3/20/ 0:
3997	train	324966	400000.0	2/1/13 0:00	present	senior systems engineer	Chennai	f	2/26/ 0:

3998 rows × 39 columns

In [ ]:
```
1 data . columns
```

Out[3]: 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')

In [ ]:

```
1 data . info  ()
```

```
<class 'pandas.core.frame.DataFrame'>

RangeIndex: 3998 entries, 0 to 3997 Data
columns (total 39 columns):

 #   Column              Non-Null Count  Dtype

                                 ---  ------              --------------  -----
  0  Unnamed: 0          3998 non-null   object
  1  ID                  3998 non-null   int64
  2  Salary              3998 non-null   float64
  3  DOJ                 3998 non-null   object
  4  DOL                 3998 non-null   object
  5  Designation         3998 non-null   object
  6  JobCity             3998 non-null   object
  7  Gender              3998 non-null   object
  8  DOB                 3998 non-null   object
  9  10percentage        3998 non-null   float64
 10  10board             3998 non-null   object
 11  12graduation        3998 non-null   int64
 12  12percentage        3998 non-null   float64
 13  12board             3998 non-null   object
 14  CollegeID           3998 non-null   int64
 15  CollegeTier         3998 non-null   int64
 16  Degree              3998 non-null   object
 17  Specialization      3998 non-null   object
 18  collegeGPA          3998 non-null   float64
 19  CollegeCityID       3998 non-null   int64
 20  CollegeCityTier     3998 non-null   int64
 21  CollegeState        3998 non-null   object
 22  GraduationYear      3998 non-null   int64
 23  English             3998 non-null   int64
 24  Logical             3998 non-null   int64
 25  Quant               3998 non-null   int64
 26  Domain              3998 non-null   float64
 27  ComputerProgramming 3998 non-null   int64
 28  ElectronicsAndSemicon 3998 non-null   int64
 29  ComputerScience     3998 non-null   int64
 30  MechanicalEngg      3998 non-null   int64
 31  ElectricalEngg      3998 non-null   int64
 32  TelecomEngg         3998 non-null   int64
 33  CivilEngg           3998 non-null   int64
 34  conscientiousness   3998 non-null   float64
 35  agreeableness       3998 non-null   float64
 36  extraversion        3998 non-null   float64
 37  nueroticism         3998 non-null   float64
```


INNOMATICS

In [ ]:
```
1 data . nunique ()
```

38openess_to_experience 3998 non-null float64 dtypes: float64(10), int64(17), object(12)
memory usage: 1.2+ MB

In [ ]:
```
1 data . duplicated (). sum()
```

Out[5]: 0

Out[6]: Unnamed: 0 1 ID
3998

Salary 177

DOJ 81

DOL 67

Designation 419

JobCity 339

Gender 2

DOB 1872

10percentage 851

10board 275

12graduation 16

12percentage 801

12board 340

CollegeID 1350

CollegeTier 2

Degree 4

Specialization 46 collegeGPA
1282 CollegeCityID 1350

CollegeCityTier 2

CollegeState 26

GraduationYear 11

English 111

Logical 107

Quant 138

INNOMATICS

In [ ]:

```
Domain                243
ComputerProgramming       79
ElectronicsAndSemicon     29
ComputerScience           20
MechanicalEngg            42
ElectricalEngg            31
TelecomEngg               26 CivilEngg
23 conscientiousness      141
agreeableness             149 extraversion
154 nueroticism           217
openess_to_experience     142 dtype:
int64
```

In [ ]:
```
df = data.drop(columns =['Unnamed: 0', 'ID', 'CollegeID', 'CollegeCityID'] 2 #df.head()
```

2.2.3 Dropping the rows where DOL is less then DOJ

```
# DataFrame containing the removed rows
removed_rows = df [ df [ 'DOL' ] < df [ 'DOJ' ]]
print ( removed_rows . shape )

# Remove rows where DateOfLeaving < DateOfJoining
df = df [ df [ 'DOL' ] >= df [ 'DOJ' ]]
print ( df . shape )
```

```
(896 , 35)
(3102 , 35)
```

3 Data Conversion

In [ ]:
```
from datetime import datetime

today_date = datetime . today (). strftime ( '%Y-%m-%d' )
df [ 'DOL' ] = df [ 'DOL' ]. replace ( 'present' , today_date )

df [ 'DOJ' ] = pd. to_datetime ( df [ 'DOJ' ])
df [ 'DOB' ] = pd. to_datetime ( df [ 'DOB' ])
#print(df.head())
```

Replacing 'present' value to today_date of 'DOL' variable

INNOMATICS

In [ ]:

In [ ]:
```
from datetime import datetime
today_date = datetime.today().strftime('%Y-%m-%d')
df['DOL'] = df['DOL'].replace('present', today_date)
#print(df.head())
```

4 Feature Engineering

In [ ]:
```
df [ 'Age'  ]  =  2024  -  df [ 'DOB' ].  dt . year
#print(df.head())
```

EDA (Exploratory Data Analysis)

INNOMATICS

4.1 Categorical Data

In [ ]:
```
# Select categorical columns excluding date columns
categorical_columns = df.select_dtypes(include=['object', 'category'], exc

# Create a new DataFrame with categorical columns
categorical_df = pd.DataFrame(categorical_columns)
categorical_df = categorical_df.drop('DOL', axis=1)
print(categorical_df.head())

```

```
                Designation    JobCity Gender                    10board \
0                            senior quality engineer  Bangalore      f  board ofsecondary
                             education,ap
1                            assistant manager     Indore      m                    cbse
2                            systems engineer    Chennai      f                   cbse
3                            senior software engineer    Gurgaon      m                    cbse
4                            get    Manesar      m                    cbse

                    12board      Degree  \ 0  board of intermediate
education,ap  B.Tech/B.E.

1                                        cbse  B.Tech/B.E.
2                                        cbse  B.Tech/B.E.
3                                        cbse  B.Tech/B.E.
4                                        cbse  B.Tech/B.E.

                    Specialization    CollegeState  0                computer
engineering  Andhra Pradesh

1                            electronics and communication engineering  Madhya Pradesh
2                            information technology   Uttar Pradesh
3                            computer engineering          Delhi
4                            electronics and communication engineering   Uttar Pradesh
```

4.1.1 'Others' are the values where '0' or '-1' was entered

In [ ]:
```
import numpy as np
for column in categorical_df:
if categorical_df[column].dtype == 'object':  # Check if the column co 4
categorical_df[column] = categorical_df[column].apply(lambda x: 'O
5
```

In [ ]:
```
for column in categorical_df:
```

In [ ]:

```
    if categorical_df[column].dtype == 'object':  # Check if the column co
categorical_df[column] = categorical_df[column].str.lower()
```

4.1.2 Non-visual Analysis

4.1.2.1 count, unique, unique & value_counts

```
    def unique_nunique(df):
    for column in categorical_df:
    unique_values = df[column].unique()
    #value_counts = df[column].value_counts()
    column_unique_values = df[column].nunique()
    print('*No. of unique values in', column, ':-', column_unique_values)
    print('\n*Unique values in', column, ':-', unique_values, '\n','=='*50

categorical_n_unique = unique_nunique(categorical_df)
print(categorical_n_unique)
```

```
*No. of unique values in Designation :- 373

*Unique values in Designation :- ['senior quality engineer' 'assistant mana ger' 'systems engineer'
 'senior software engineer' 'get' 'system engineer' 'mechanical engineer'
 'electrical engineer' 'project engineer' 'senior php developer'
 'quality assurance engineer' 'qa analyst' 'java software engineer'
 'network engineer' 'product development engineer'
 'associate software developer' 'data entry operator' 'software engineer'
 'developer' 'programmer analyst' 'systems analyst'
 'telecommunication engineer' 'application developer' 'ios developer'
 'executive assistant' 'online marketing manager'
 'documentation specialist' 'associate software engineer'
 'management trainee' 'software developer' '.net developer' 'ui developer'
 'assistant system engineer' 'android developer' 'customer service'
 'test engineer' 'java developer' 'engineer' 'data analyst'
 'assistant software engineer' 'faculty' 'entry level management trainee'
 'customer service representative' 'software test engineer'
 'firmware engineer' 'php developer' 'research associate'
 '          h          l t' '          lit          i          ' '                    '
```

In [ ]:

```
    def value_counts(df):
    for column in categorical_df:
    value_counts = categorical_df[column].value_counts()
    print('*Value counts in', column, ':-\n')
    print(value_counts[:12])
    print('='*100,'\n')

categorical_value_counts = value_counts(categorical_df)
print(categorical_value_counts) *Value counts in Designation :-
```

```
software engineer          435 software
developer          200 system engineer
179 programmer analyst          118 systems
engineer            99 java software engineer
88 software test engineer        83 project
engineer            64 senior software engineer
58 java developer              57 technical support
engineer    52 test engineer              50 Name:
Designation, dtype: int64

====================================================================================

========================= *Value

counts in JobCity :-
```


INNOMATICS

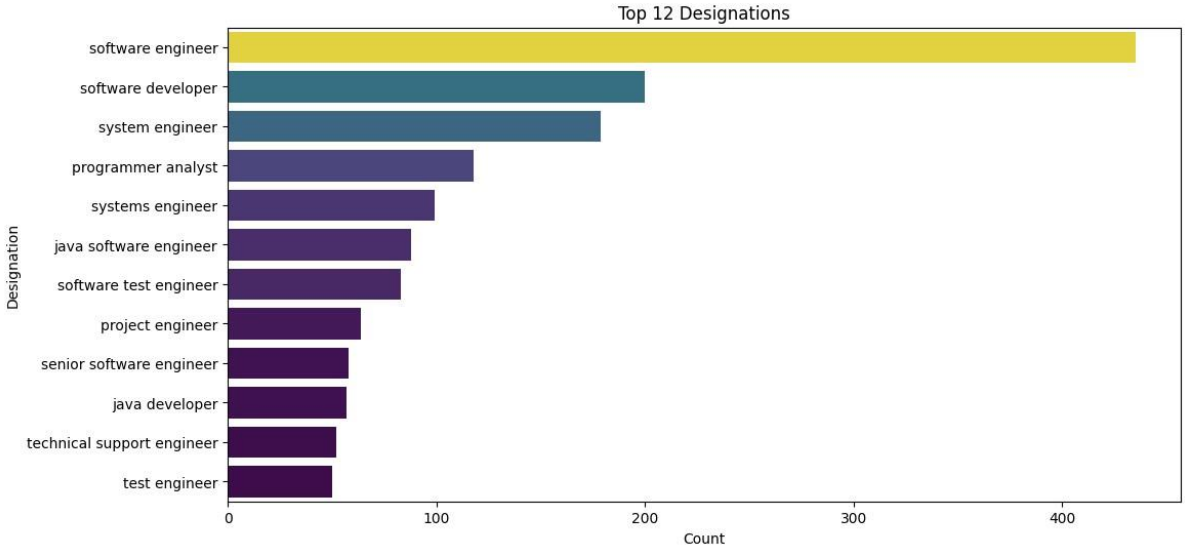
In [ ]:

4.2 Univariate

4.2.1 Visual Analysis

4.2.1.1 Designation

```
import matplotlib.pyplot as plt

# Assuming 'df' is your DataFrame containing the data
top_n = 12

top_designations = categorical_df['Designation'].value_counts().head(top_n

plt.figure(figsize=(12, 6))
sns.barplot(x=top_designations, y=top_designations.index, hue=top_designat
plt.title(f'Top {top_n} Designations')
plt.xlabel('Count')
plt.ylabel('Designation')
plt.show()

```



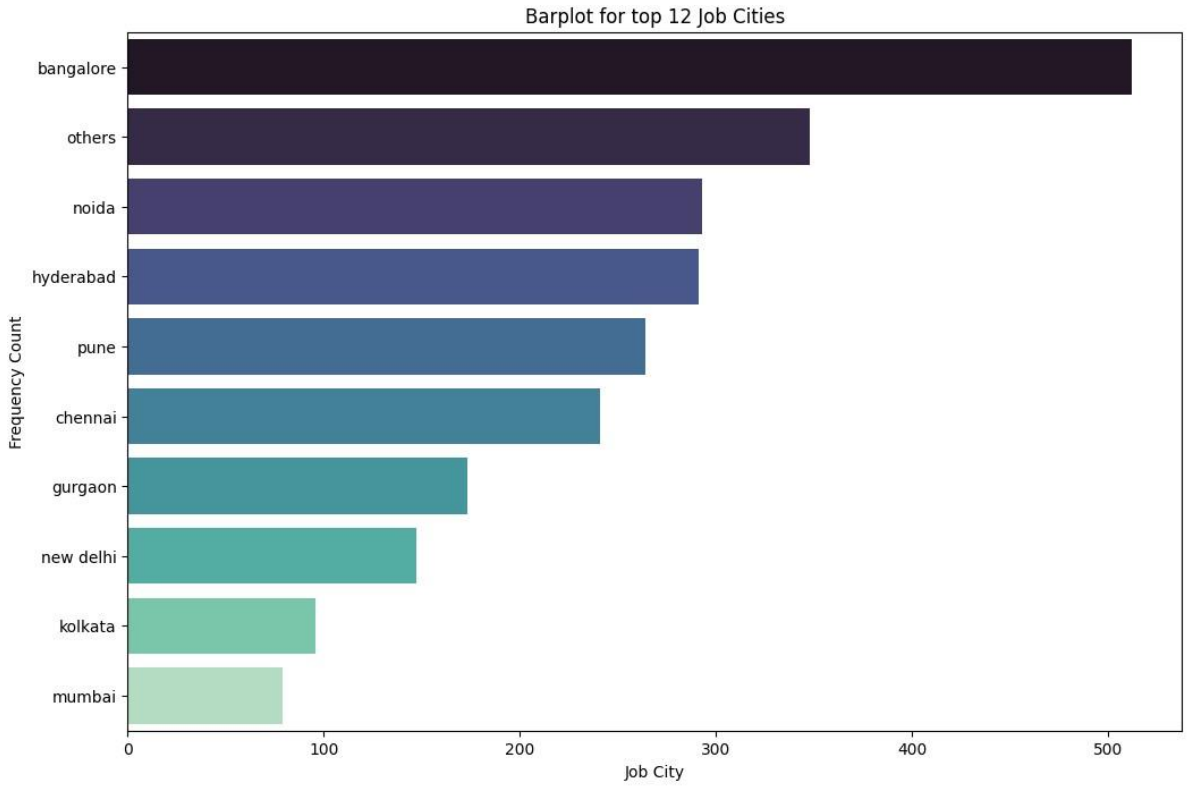
Observation:

Software Engineer is the topmost designation depennding upon the AMCAT data.

In [ ]:

4.2.1.2 JobCity

```
top_cities = categorical_df['JobCity'].value_counts().head(10)


plt.figure(figsize=(12, 8))
sns.barplot(x=top_cities.values, y=top_cities.index, hue=top_cities.index,
plt.title('Barplot for top 12 Job Cities')
plt.xlabel('Job City')
plt.ylabel('Frequency Count')
plt.show()
```

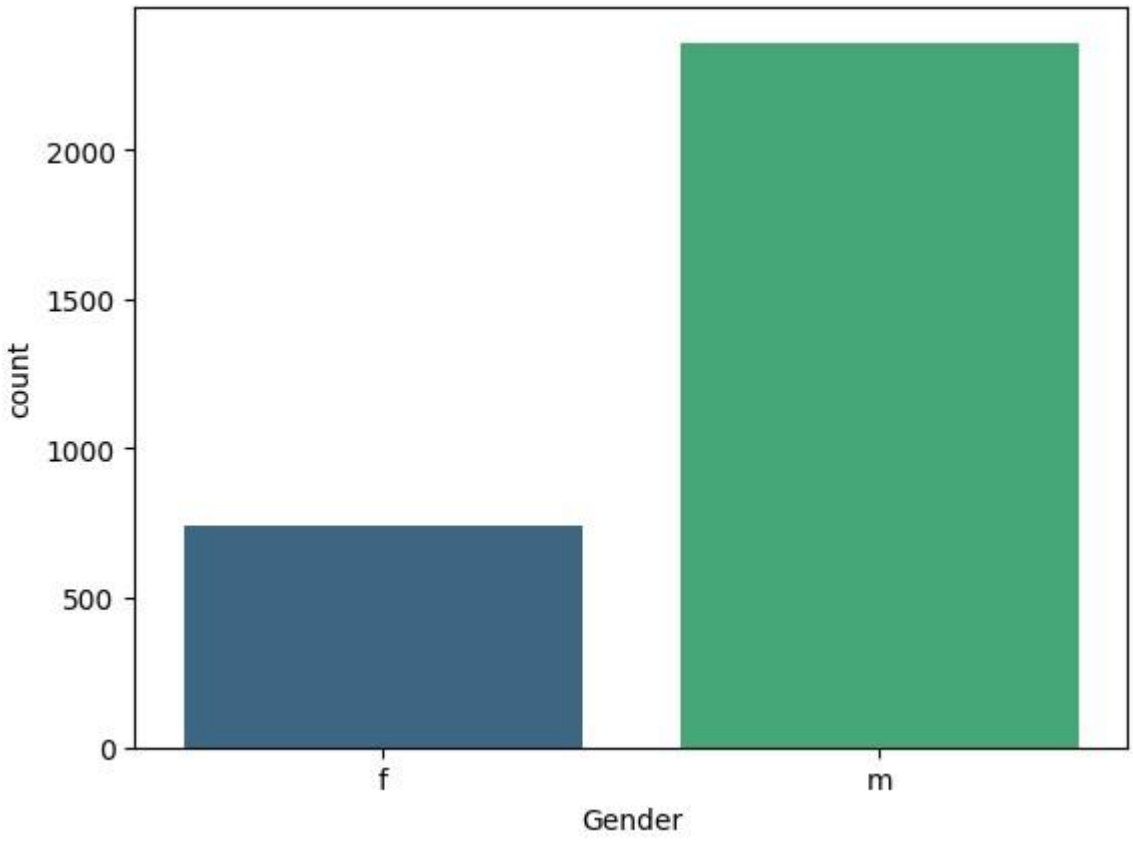


Observations: Bangalore is the top most city

In [ ]:

4.2.1.3 Gender

```
colors = sns . color_palette ( 'mako' , n_colors =2)
sns . countplot ( x='Gender' , data =categorical_df , hue='Gender' , palette = 'vi

plt . show()

```

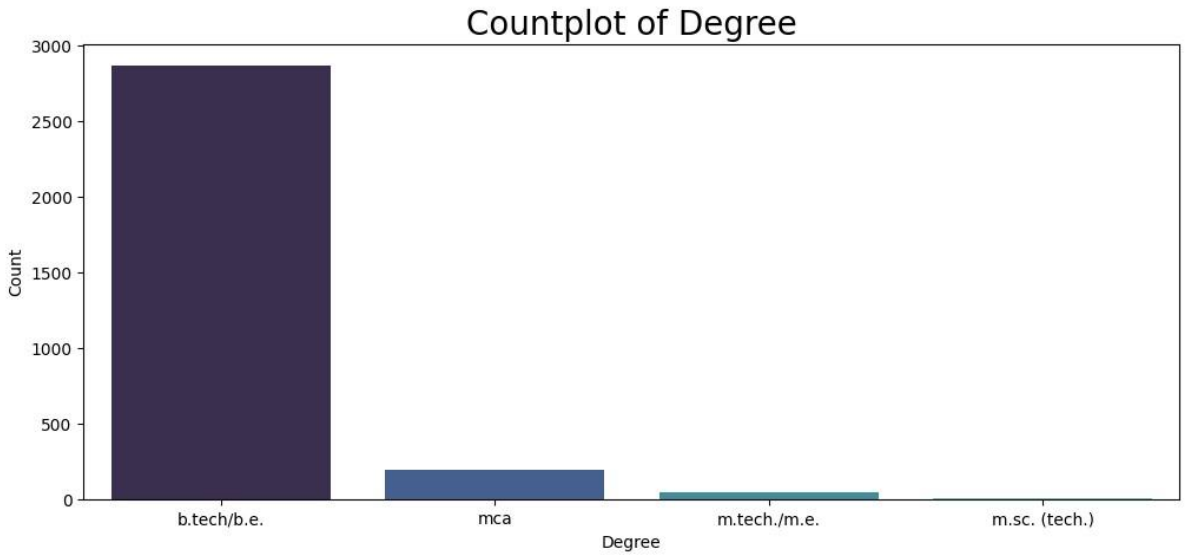


Observations: Male data is dominating female data i.e. we are working on imbalanced data.

In [ ]:

4.2.1.4 Degree

```
colors = sns.color_palette('mako', n_colors=4)
plt.figure(figsize=(12, 5))
sns.countplot(x='Degree', data=categorical_df, hue='Degree', palette='mako
plt.xlabel('Degree', size=10)
plt.ylabel('Count', size=10)
plt.title('Countplot of Degree', size=20)
plt.show()
```

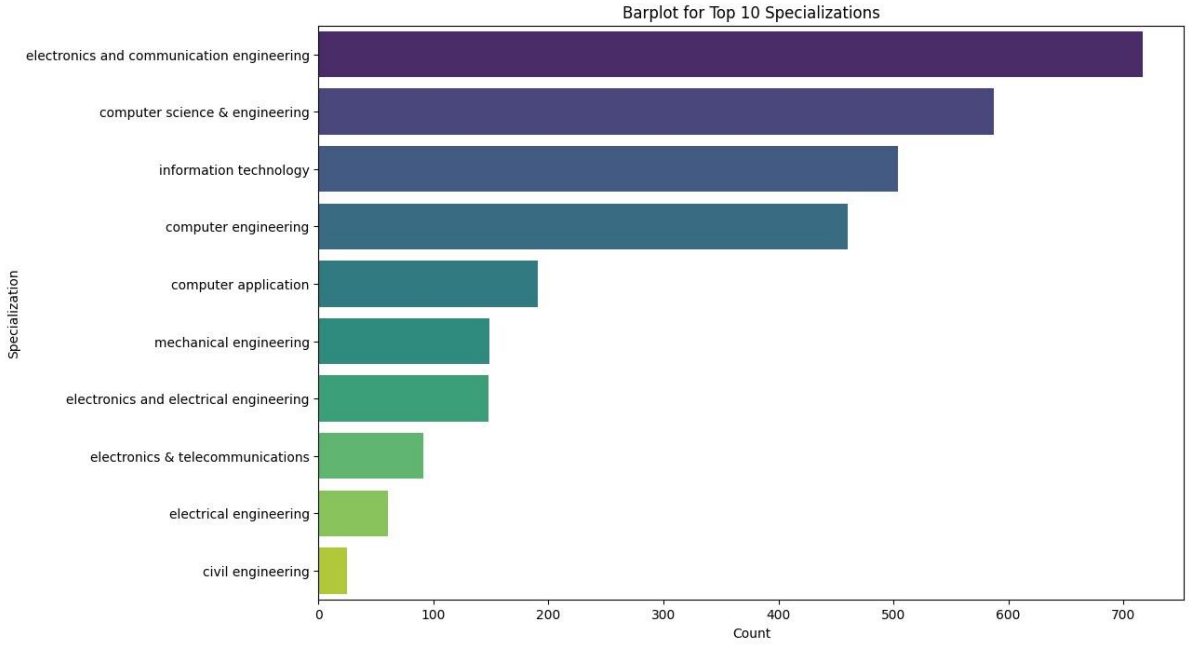


Observations: Most of the students have done their graduation in B.Tech and there are very less students from M.Sc(Tech)

In [ ]:

4.2.1.5 Specialization

```
# Assuming 'df' is your DataFrame containing the data
top_specialisations = categorical_df['Specialization'].value_counts().head

plt.figure(figsize=(12, 8))
sns.barplot(x=top_specialisations.values, y=top_specialisations.index, hue
plt.title('Barplot for Top 10 Specializations')
plt.xlabel('Count')
plt.ylabel('Specialization')
plt.show()
```



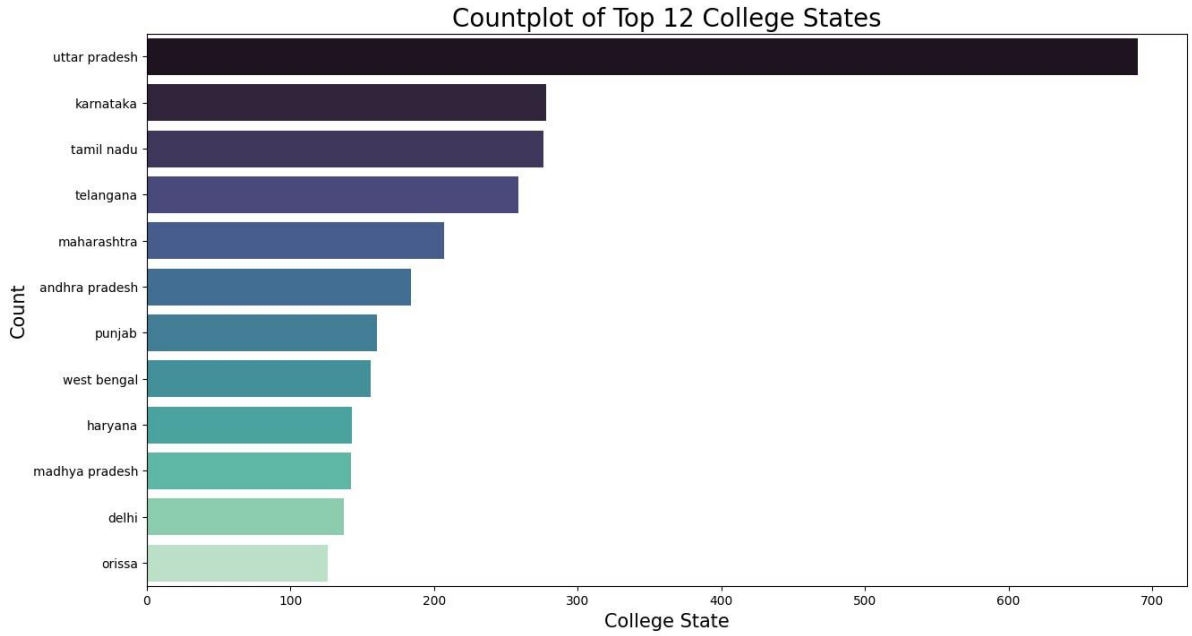
Observations:

Electronics & telecommunication is the most common specialization.

In [ ]:

4.2.1.6 CollegeState

```
top_college_states = categorical_df['CollegeState'].value_counts().head(12
plt.figure(figsize=(15, 8))
sns.barplot(x=top_college_states.values, y=top_college_states.index, hue=t
plt.xlabel('College State', size=15)
plt.ylabel('Count', size=15)
plt.title('Countplot of Top 12 College States', size=20)
plt.show()
```



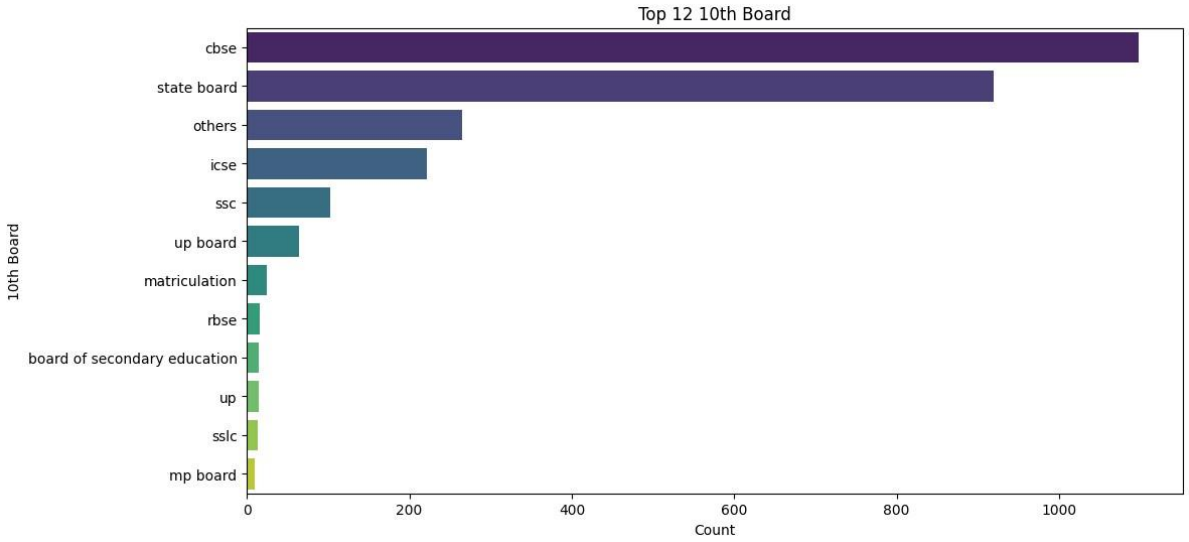
Observation: Uttar Pradesh is the top among all the college state.

Observations:

In [ ]:

4.2.1.7 10th Board

```
top_10_board = categorical_df['10board'].value_counts().head(12)

plt.figure(figsize=(12, 6))
sns.barplot(x=top_10_board.values, y=top_10_board.index, hue=top_10_board.
plt.title(f'Top {top_n} 10th Board')
plt.xlabel('Count')
plt.ylabel('10th Board')
plt.show()
```

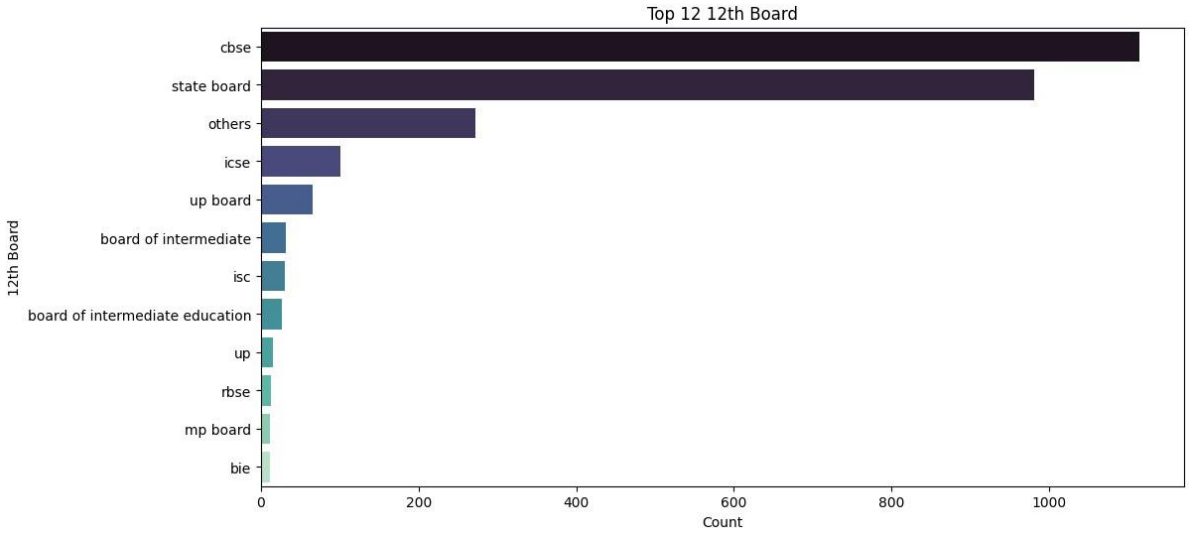


Observations: CBSE is the most common school board for 10th CLass

In [ ]:

4.2.1.8 12th Board

```
top_12_board = categorical_df['12board'].value_counts().head(12)

plt.figure(figsize=(12, 6))
sns.barplot(x=top_12_board.values, y=top_12_board.index, hue=top_12_board.
plt.title(f'Top {top_n} 12th Board')
plt.xlabel('Count')
plt.ylabel('12th Board')
plt.show()
```



Observations: CBSE is the most common school board for 12th CLass

4.3 Numerical Data

In [ ]:
```
numerical_df = df.select_dtypes(include=['float64', 'int64'])
#print(numerical_df.head())
```

4.3.1 Non-Visual Analysis

4.3.1.1 Describe

```
print(numerical_df.describe())
```

Salary 10percentage 12graduation 12percentage CollegeTier \ count 3.102000e+03 3102.000000 3102.000000 3102.000000 3102.000000 mean 3.147643e+05 78.216444 2008.167311 74.694384 1.930045 std 1.990589e+05 9.725405 1.613208 11.004027 0.255112 min 3.500000e+04 43.000000 1999.000000 43.000000 1.000000 25% 2.000000e+05 72.000000 2007.000000 66.400000 2.000000

INNOMATICS

In [ ]:

```
50%   3.000000e+05    79.800000  2008.000000    74.600000    2.000000  75%   3.800000e+05
86.000000  2009.000000    83.000000    2.000000  max   4.000000e+06    97.760000
2013.000000    98.200000    2.000000

      collegeGPA  CollegeCityTier  GraduationYear     English     Logical

\

count  3102.000000    3102.000000    3102.000000  3102.000000  3102.000000 mean    71.675822
0.302708    2012.027079   503.149903   502.889749 std      8.036582       0.459504      36.160450
104.625499    86.639401 min      6.800000       0.000000       0.000000   180.000000   195.000000
25%    66.685000       0.000000    2012.000000   430.000000   445.000000
50%    72.000000       0.000000    2013.000000   500.000000   505.000000 75%    76.500000
1.000000    2014.000000   570.000000   565.000000 max    99.930000       1.000000
2017.000000   875.000000   795.000000

     ... MechanicalEngg  ElectricalEngg  TelecomEngg   CivilEngg  \ count  ...    3102.000000
3102.000000  3102.000000  3102.000000   mean  ...     21.587041     16.313991    32.609929
2.992585  std   ...     95.266479     86.890710   105.736937   38.413125   min   ...     -
1.000000    -1.000000   -1.000000   -1.000000  25%   ...     -1.000000     -1.000000   -
1.000000   -1.000000

50%   ...     -1.000000     -1.000000   -1.000000   -1.000000  75%   ...     -1.000000     -
1.000000   -1.000000   -1.000000  max   ...     623.000000     676.000000   526.000000
516.000000

     conscientiousness  agreeableness  extraversion  nueroticism  \ count       3102.000000
3102.000000  3102.000000  3102.000000   mean       -0.021747     0.150981     0.019222
-0.180846   std         1.027251     0.934672     0.938735     1.007959   min         -4.126700
-5.781600    -4.600900   -2.643000   25%        -0.589900     -0.287100    -0.604800   -
0.868200

50%         0.046400     0.212400     0.091400   -0.234400   75%         0.702700
0.812800     0.672000     0.526200   max         1.995300     1.904800     2.535400
3.352500

     openess_to_experience       Age  count
3102.000000  3102.000000  mean          -0.144115
33.499678  std           1.010308    1.715636  min
-7.375700    27.000000  25%           -0.669200
32.000000

50%                   -0.094300      33.000000   75%
0.502400     34.000000   max                1.822400
42.000000

[8 rows x 25 columns]
```


INNOMATICS

4.3.2 Visual Analysis

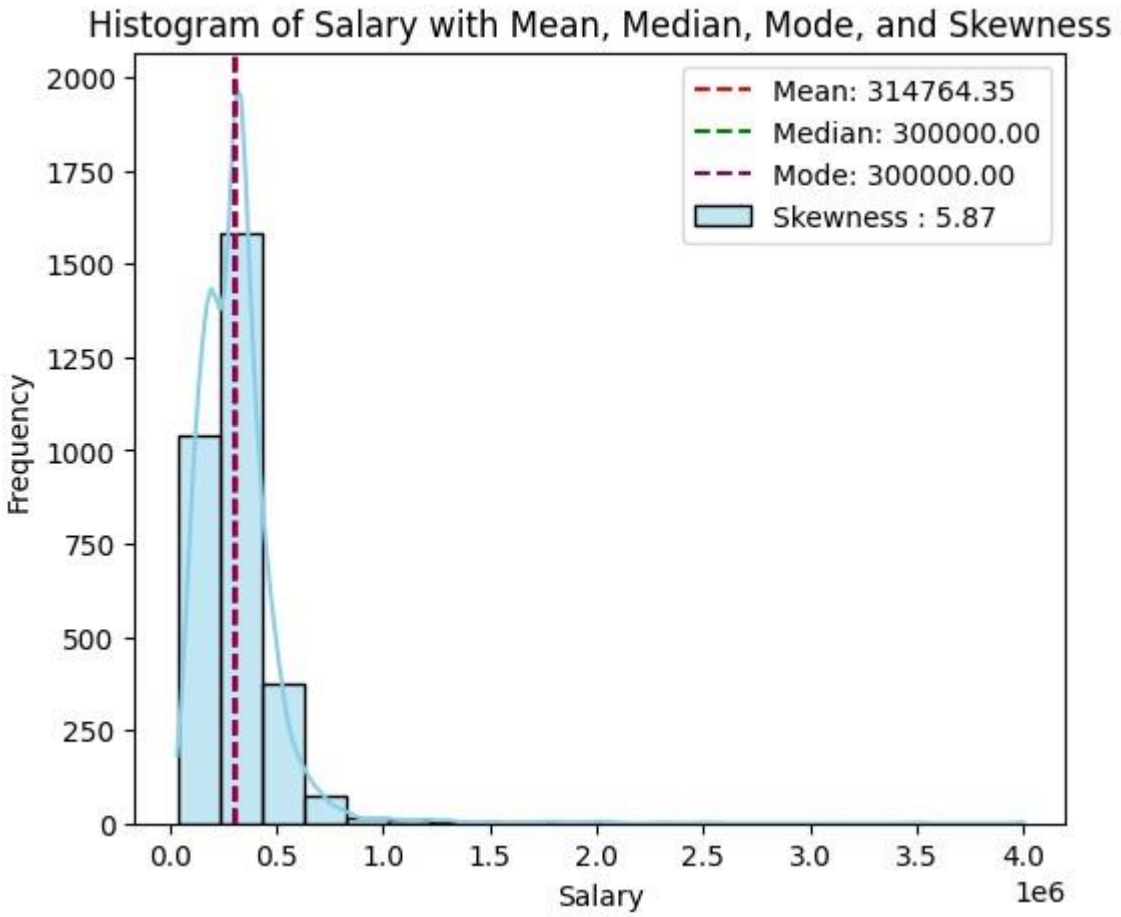
4.3.2.1 Salary

In [ ]:
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='Salary' , color ='skyblue' )
plt . title ( 'Box Plot of Salary' )
plt . ylabel ( 'Salary' )
plt . tight_layout ()
plt . show()
```

In [41]:
```
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats import norm, skew
```


INNOMATICS

```
4
5 salary_values = numerical_df['Salary']
6
7   # Calculate mean, median, mode, and skewness
8   mean_salary = salary_values.mean()
9   median_salary = salary_values.median()
10  mode_salary = salary_values.mode()[0]
11  skewness_salary = skew(salary_values)
12
13 # Plot histogram
14 plt.figure(figsize=(6, 5))
15 sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round
16
17 # Add mean, median, mode, and skewness information
18 plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_ 19
   plt.axvline(median_salary, color='green', linestyle='--', label=f'Median: 20
   plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo
21
22 # Add distribution line
23 x = np.linspace(salary_values.min(), salary_values.max(), 100)
24 #plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange'
25
26 plt.title('Histogram of Salary with Mean, Median, Mode, and Skewness')
27 plt.xlabel('Salary')
28 plt.ylabel('Frequency')
29 plt.legend()
30 plt.show()
31
```

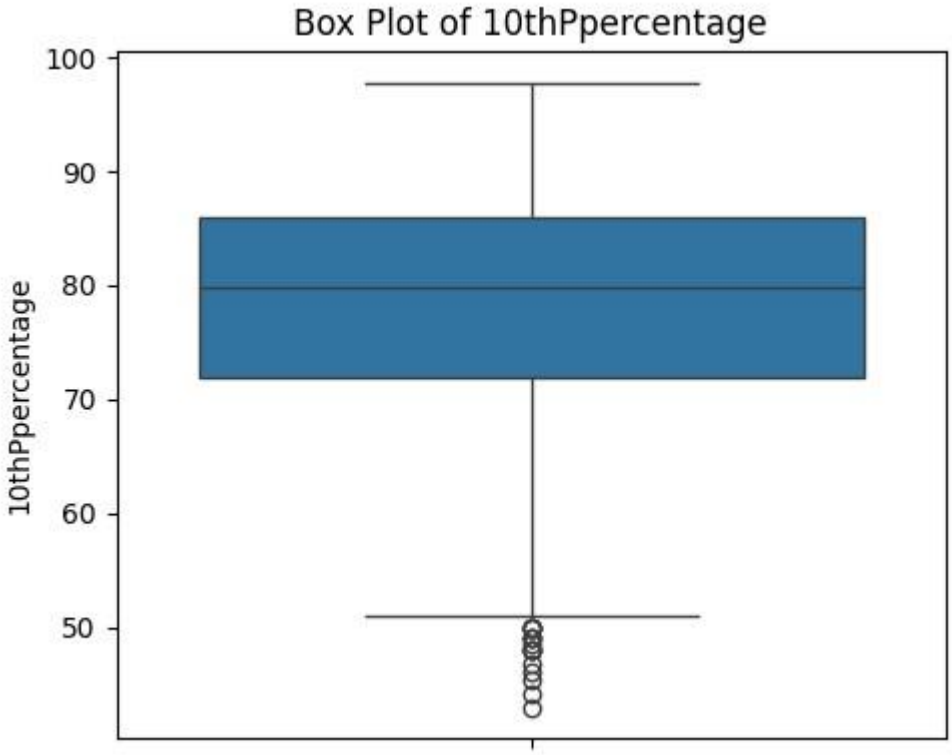


Observation:

1. Salary variations are significant throughout the sample.
2. With a skewness value of roughly 6, the data shows strong positive skewness, which deviates from a normal distribution.
3. The mean, median, and mode—three measures of central tendency—are about equal.

4.3.2.2 10th Percentage

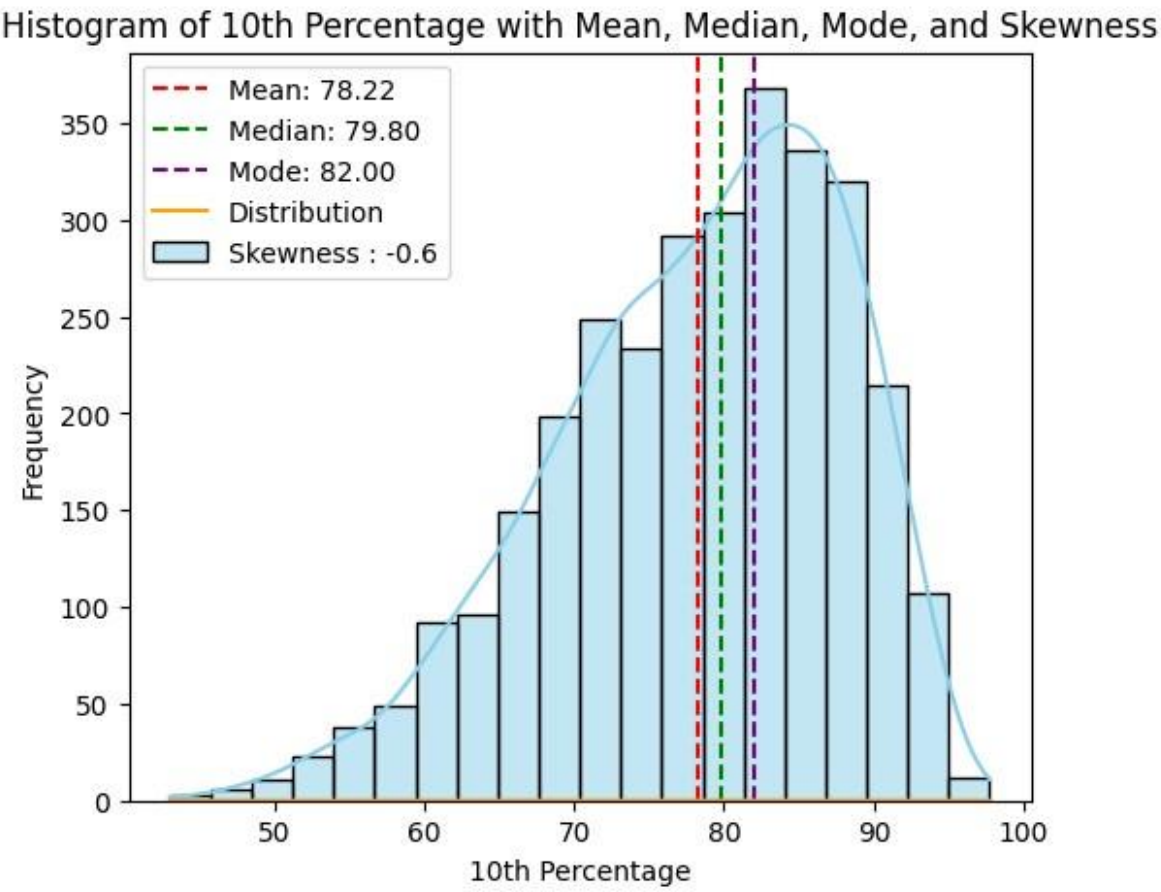
In [ ]:
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='10percentage' )
plt . title ( 'Box Plot of 10thPpercentage' )
plt . ylabel ( '10thPpercentage' )
plt . tight_layout ()
plt . show()
```



In [42]:
```
salary_values = numerical_df['10percentage']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
```


INNOMATICS

```
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of 10th Percentage with Mean, Median, Mode, and Skewn
plt.xlabel('10th Percentage')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

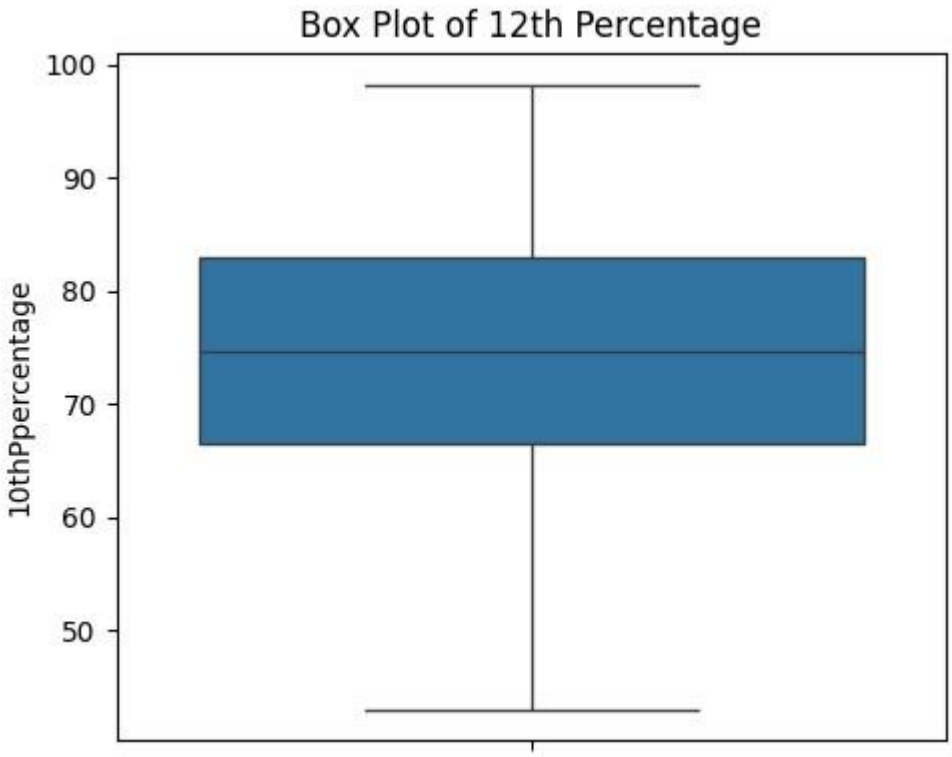


Observations::

1. It shows that thedistribution is negatively skewed with a skewed value around -0.6.
2. It doesn't follow normal distribution.
3. Around 50% of students achieved scores of approximately 80% or less.

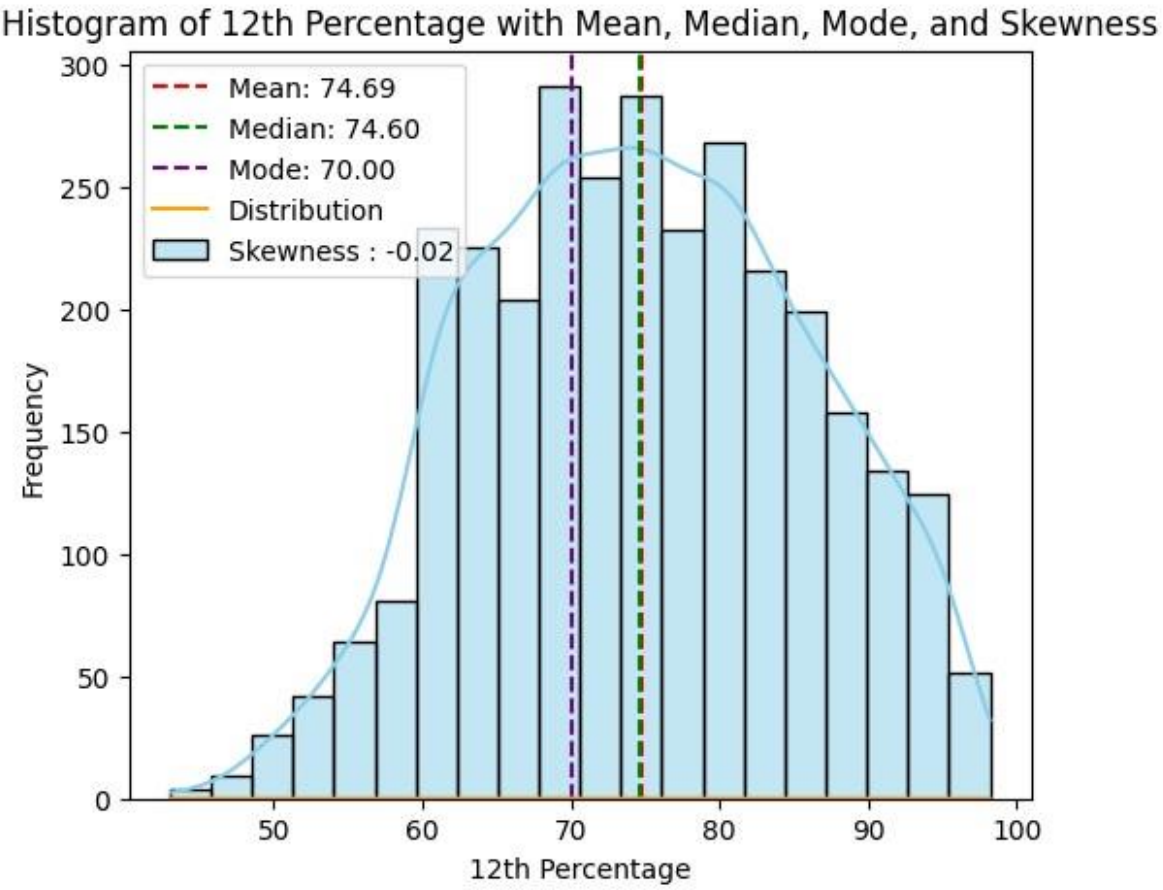
4.3.2.3 12percentage

In [ ]:
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='12percentage' )
plt . title ( 'Box Plot of 12th Percentage' )
plt . ylabel ( '12th Percentage' )
plt . tight_layout ()
plt . show()
```



In [43]:
```
salary_values = numerical_df['12percentage']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
```

```
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of 12th Percentage with Mean, Median, Mode, and Skewn
plt.xlabel('12th Percentage')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

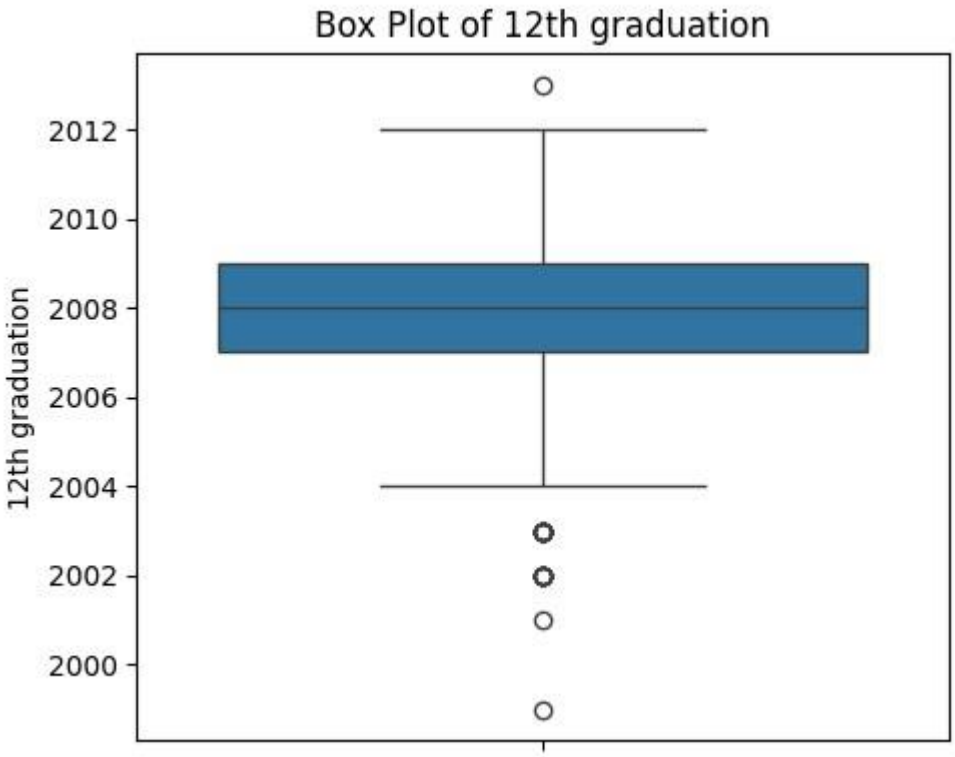


Observations:

1. The box plot indicates only one data point with an extremely low score.
2. Roughly half of the students achieved scores of approximately 78% or lower.
3. It has a skewed value of -0.02 which says the distribution is negatively skewed.

4.3.2.4 12graduation

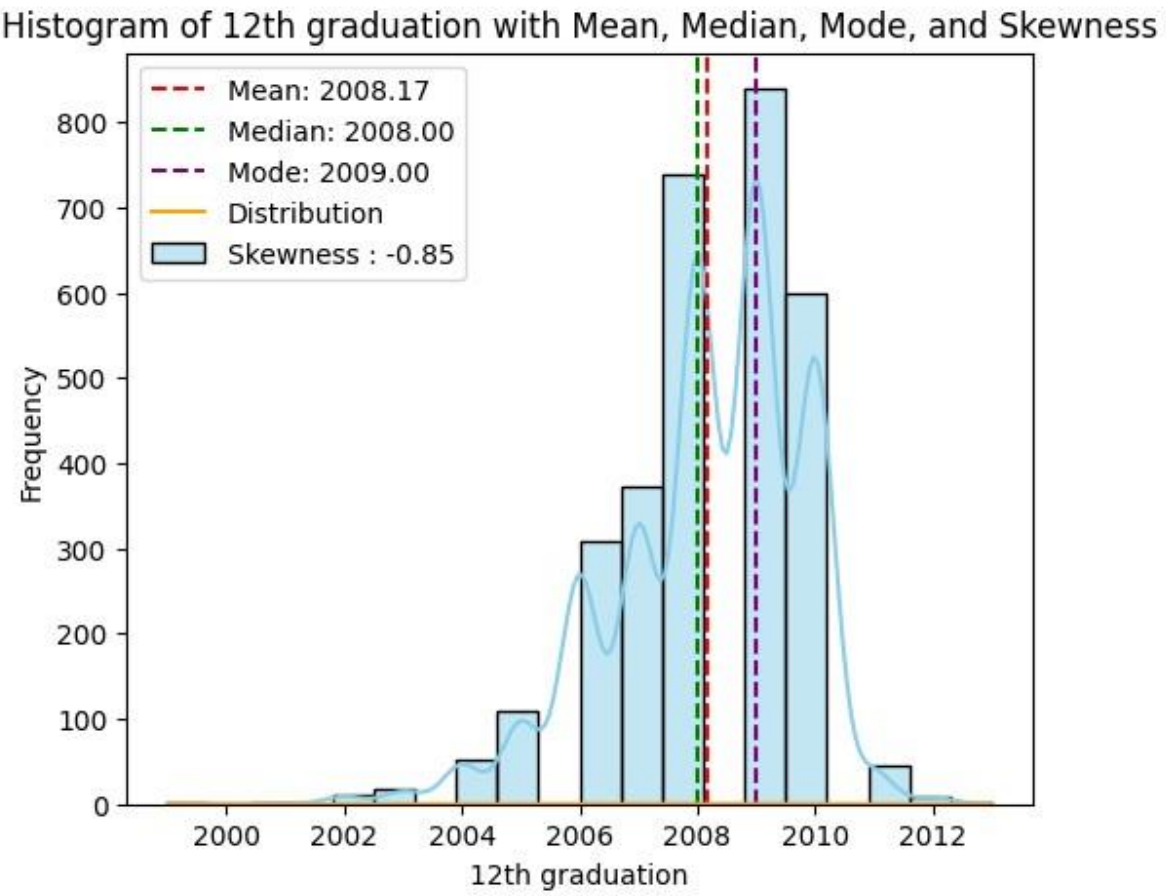
In [ ]:
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='12graduation' )
plt . title ( 'Box Plot of 12th graduation' )
plt . ylabel ( '12th graduation' )
plt . tight_layout ()
plt . show()
```



In [44]:
```
salary_values = numerical_df['12graduation']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
```

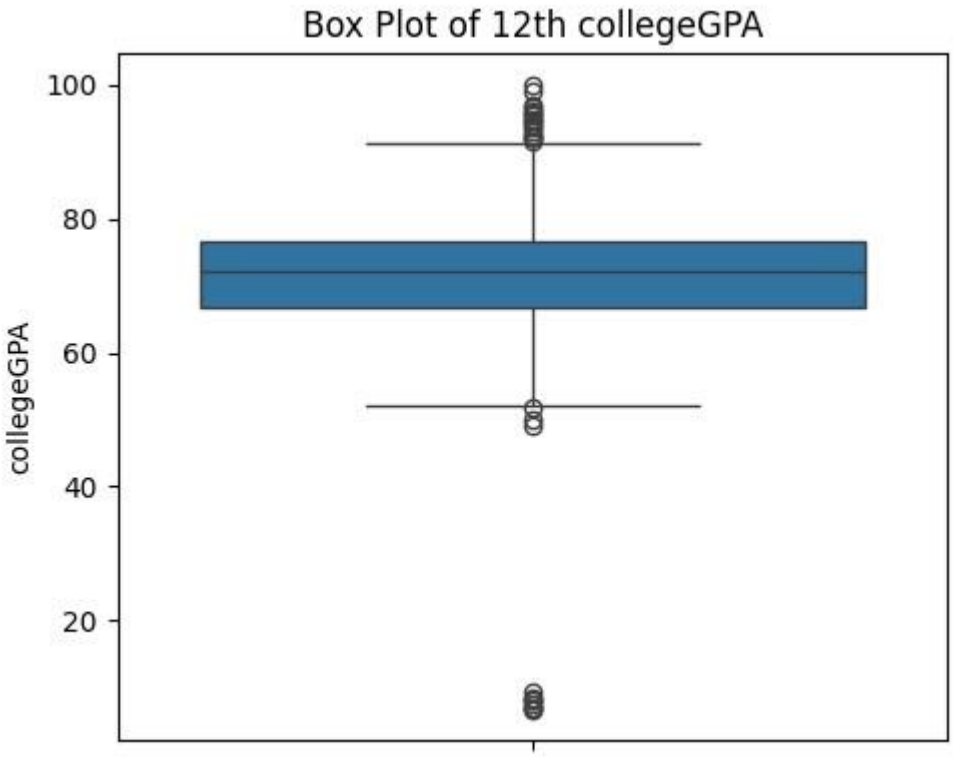

INNOMATICS

```
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of 12th graduation with Mean, Median, Mode, and Skewn
plt.xlabel('12th graduation')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```


Histogram of 12th graduation with Mean, Median, Mode, and Skewness
Mean: 2008.17
Median: 2008.00
Mode: 2009.00
Distribution
Skewness : -0.85
Frequency
0
100
200
300
400
500
600
700
800
2000
2002
2004
2006
2008
2010
2012
12th graduation

4.3.2.5 Observations: collegeGPA

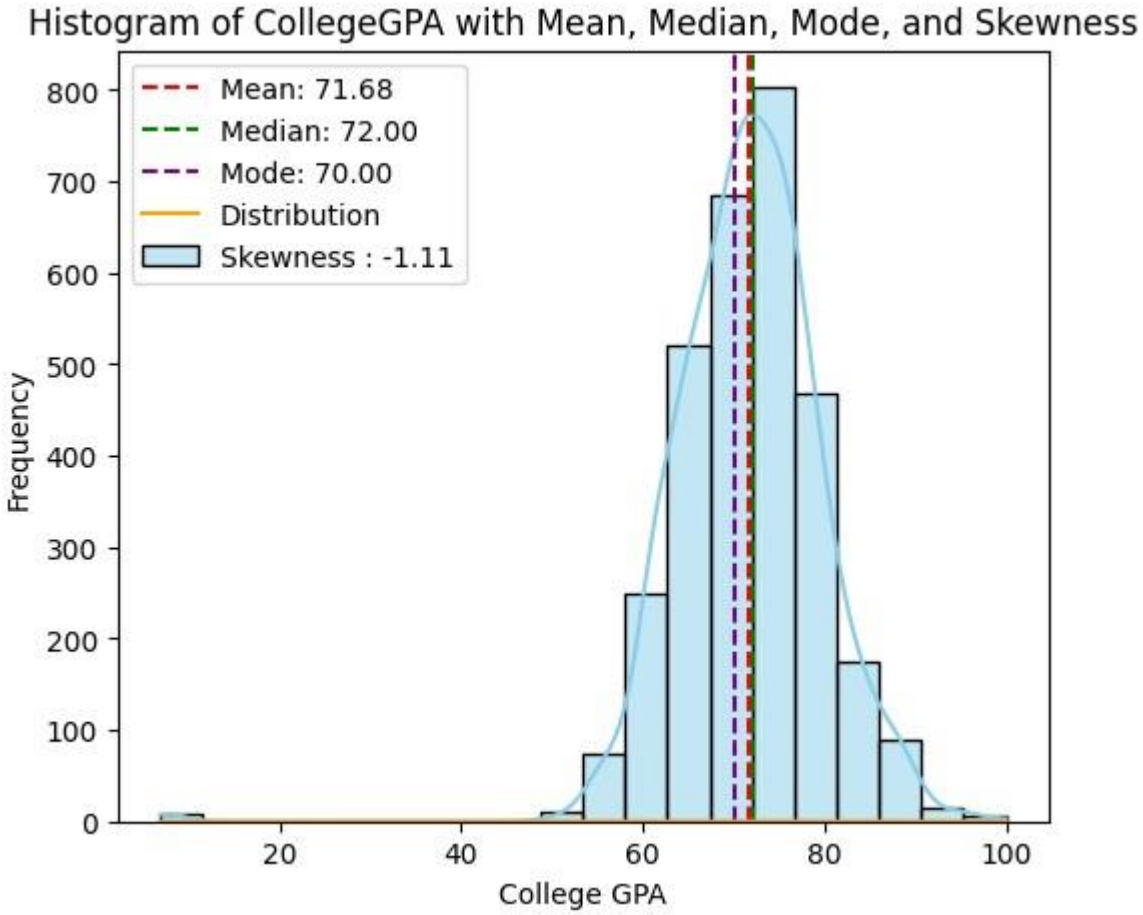
In [ ]:
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='collegeGPA' )
plt . title ( 'Box Plot of 12th collegeGPA' )
plt . ylabel ( 'collegeGPA' )
plt . tight_layout ()
plt . show()
```



In [45]:
```
salary_values = numerical_df['collegeGPA']

# Calculate mean, median, mode, and skewness
```


INNOMATICS

```
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of CollegeGPA with Mean, Median, Mode, and Skewness')
plt.xlabel('College GPA')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

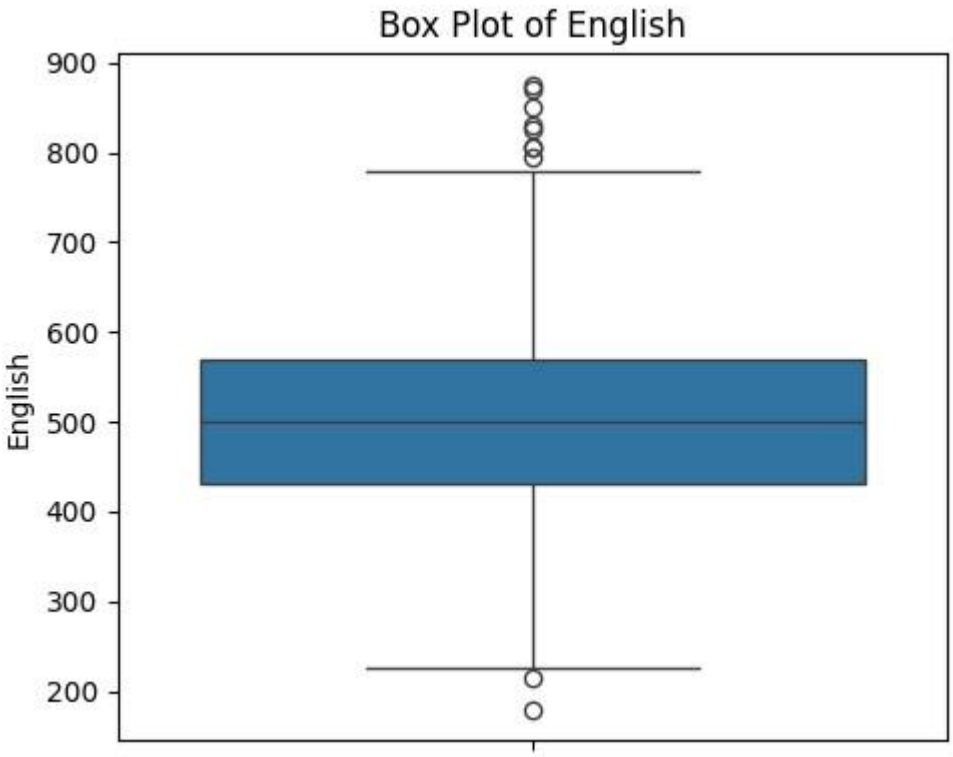


Observations:

1. 75% of students had a GPA of approximately 80% or lower.
2. The box plot reveals the presence of both low.
3. The distribution is negatively skewed.

4.3.2.6 English

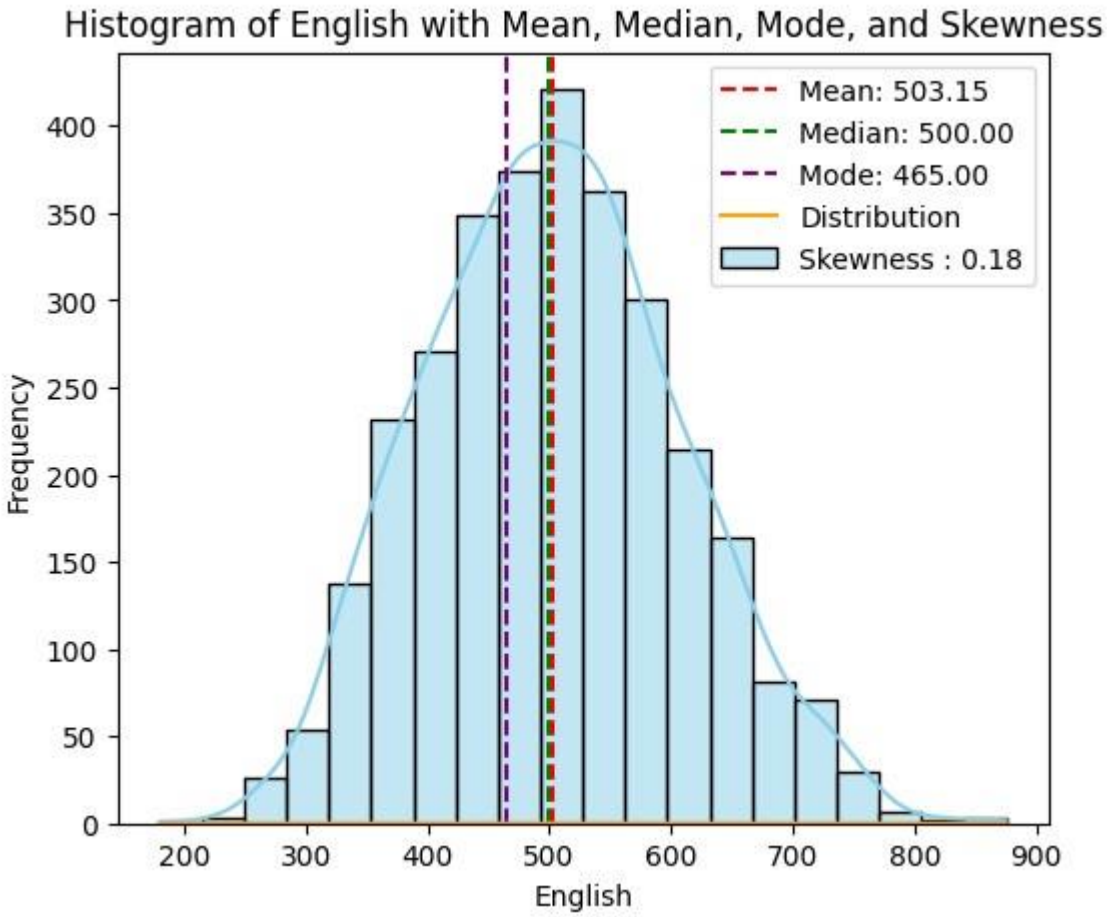
In [75]:
```
plt . figure  ( figsize   =( 5,  4))
sns . boxplot  ( data =numerical_df   ,  y='English'    )
plt . title   ( 'Box Plot of English'          )
plt . ylabel  ( 'English'    )
plt . tight_layout    ()
plt . show()
```



In [76]:
```
salary_values = numerical_df['English']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
```


INNOMATICS

```
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of English with Mean, Median, Mode, and Skewness')
plt.xlabel('English')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

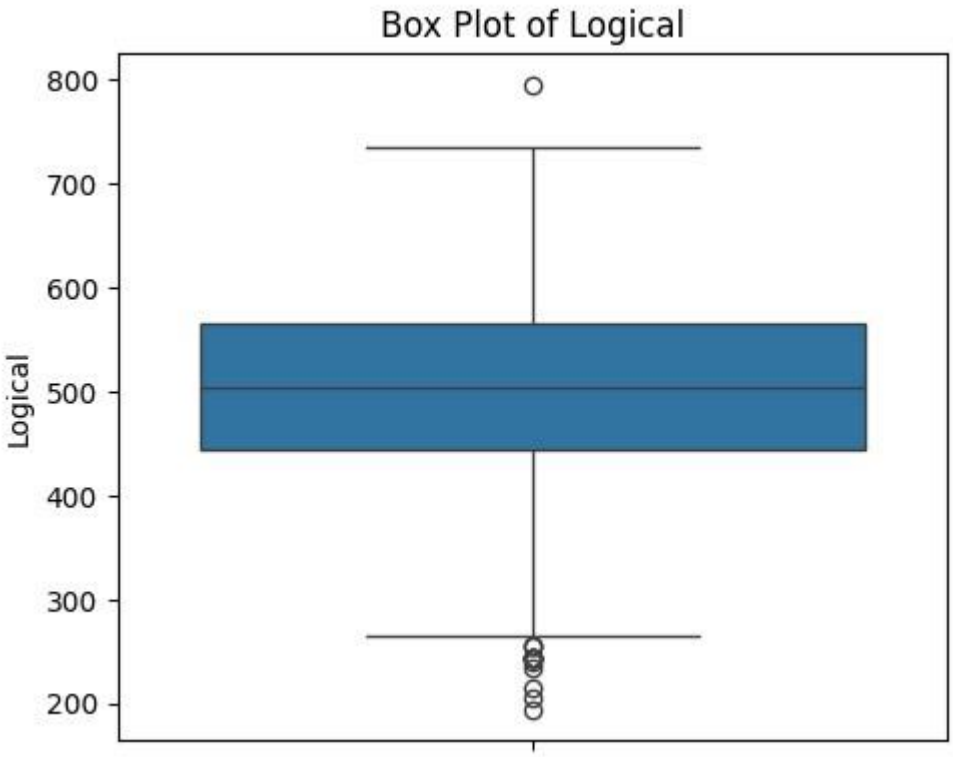


Observations:

1. Half of the students scored below 500 in their English exams.
2. The data follows a nearly normal distribution pattern by observing the distribution (bellshaped).

4.3.2.7 Logical

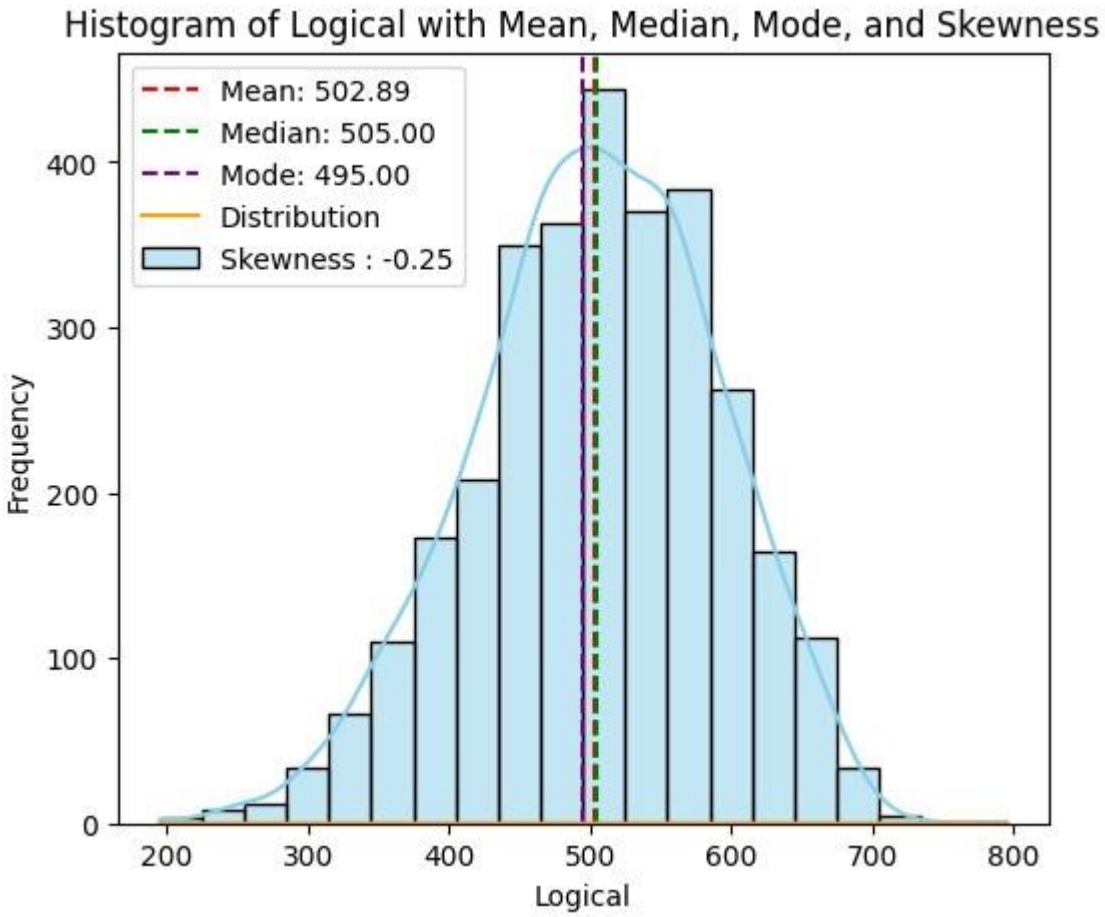
In [ ]:
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='Logical' )
plt . title ( 'Box Plot of Logical' )
plt . ylabel ( 'Logical' )
plt . tight_layout ()
plt . show()
```



In [47]:
```
salary_values = numerical_df['Logical']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
```


INNOMATICS

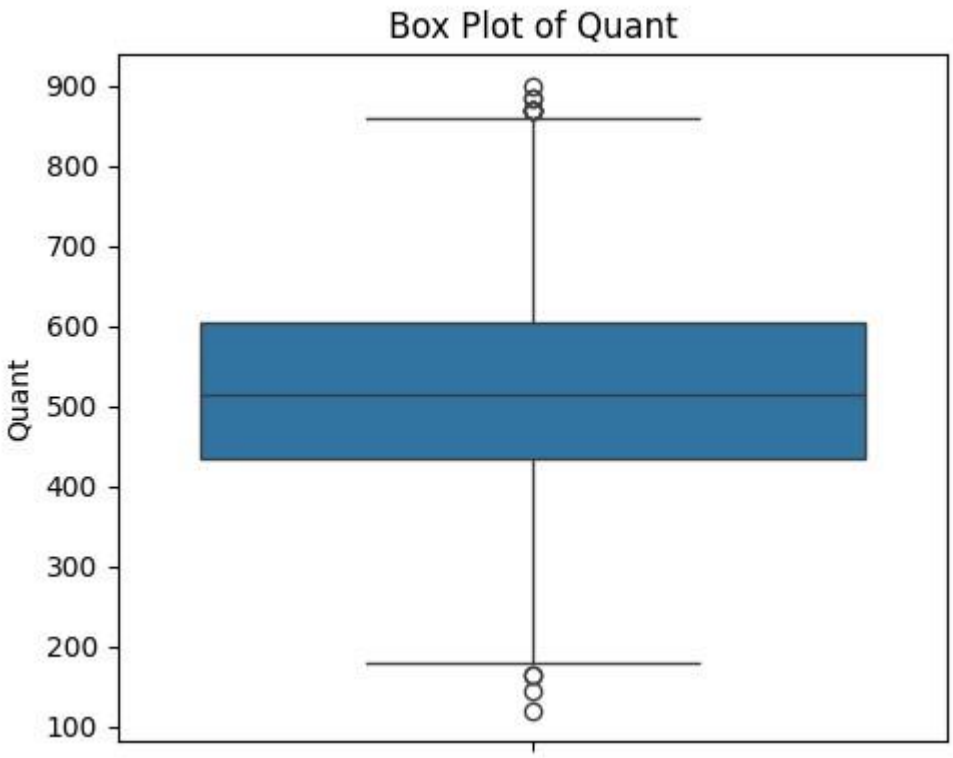
```
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of Logical with Mean, Median, Mode, and Skewness')
plt.xlabel('Logical')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```


Histogram of Logical with Mean, Median, Mode, and Skewness
Mean: 502.89
Median: 505.00
Mode: 495.00
Distribution
Skewness : -0.25
Frequency
0
100
200
300
400
200
300
400
500
600
700
800
Logical

4.3.2.8 Quant

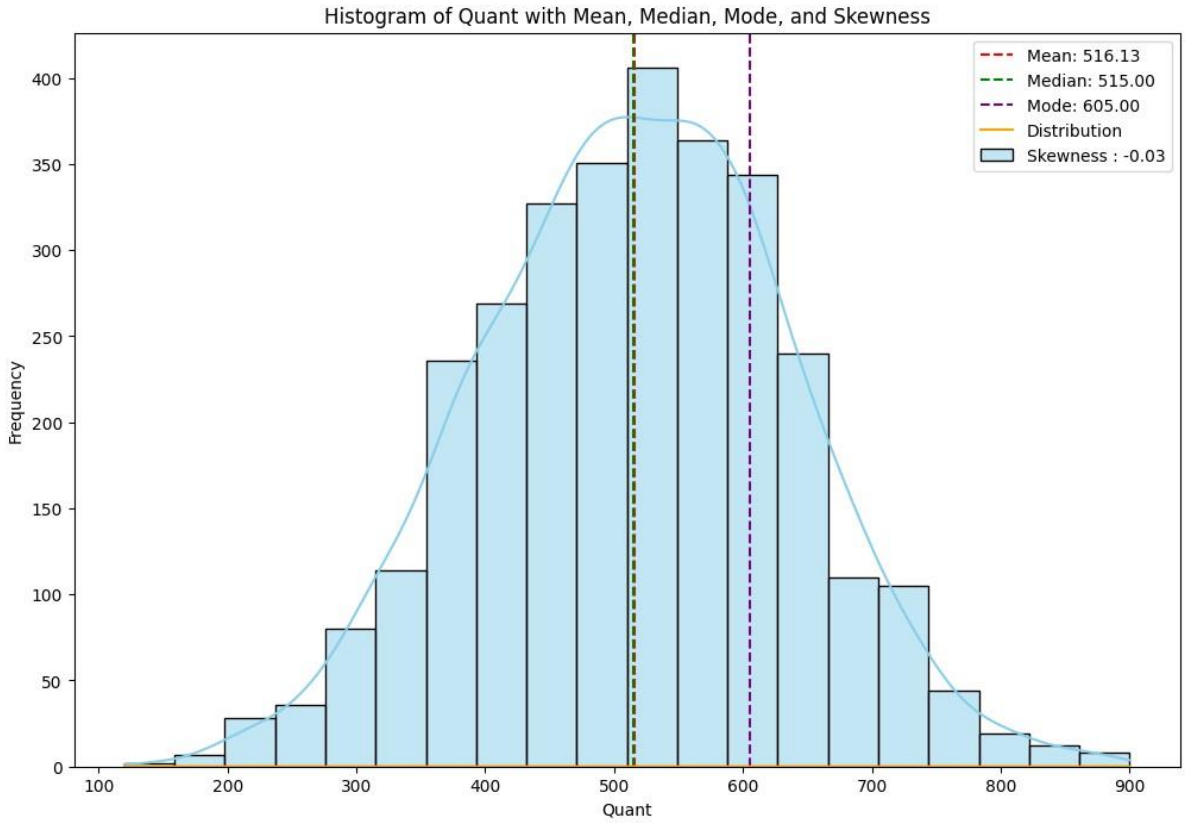
In [ ]:

```
plt . figure  ( figsize   =( 5,  4))
sns . boxplot  ( data =numerical_df  ,  y='Quant'  )
plt . title  ( 'Box Plot of Quant'       )
plt . ylabel  ( 'Quant'  )
plt . tight_layout    ()
plt . show()
```



In [ ]:

```
salary_values = numerical_df['Quant']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(12, 8))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of Quant with Mean, Median, Mode, and Skewness')
plt.xlabel('Quant')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

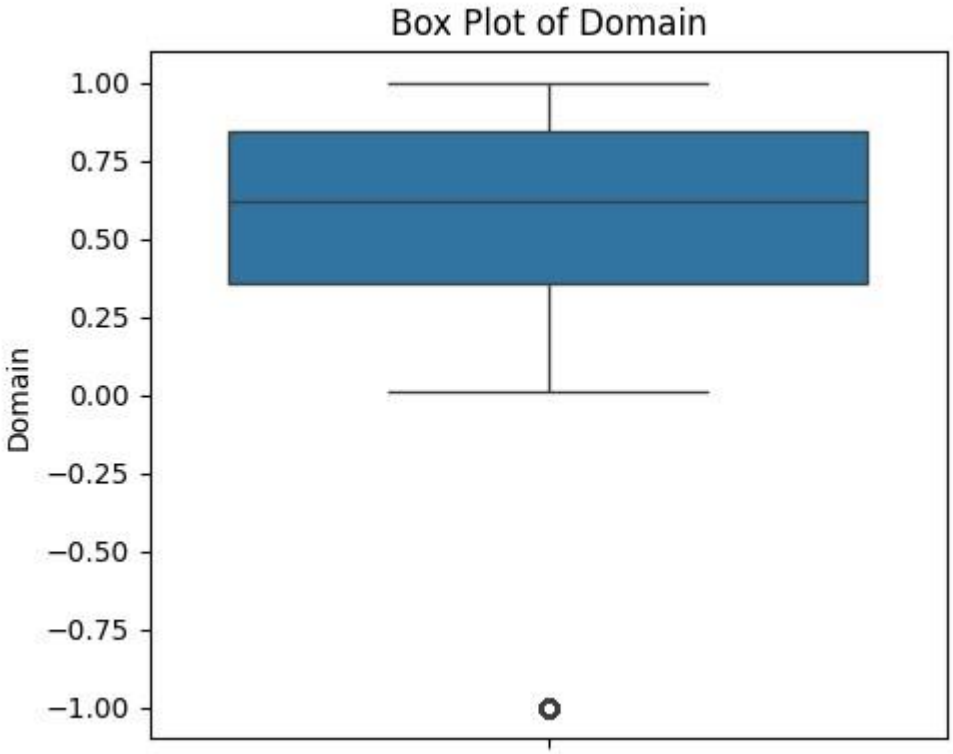


INNOMATICS

In [ ]:

4.3.2.9 Domain

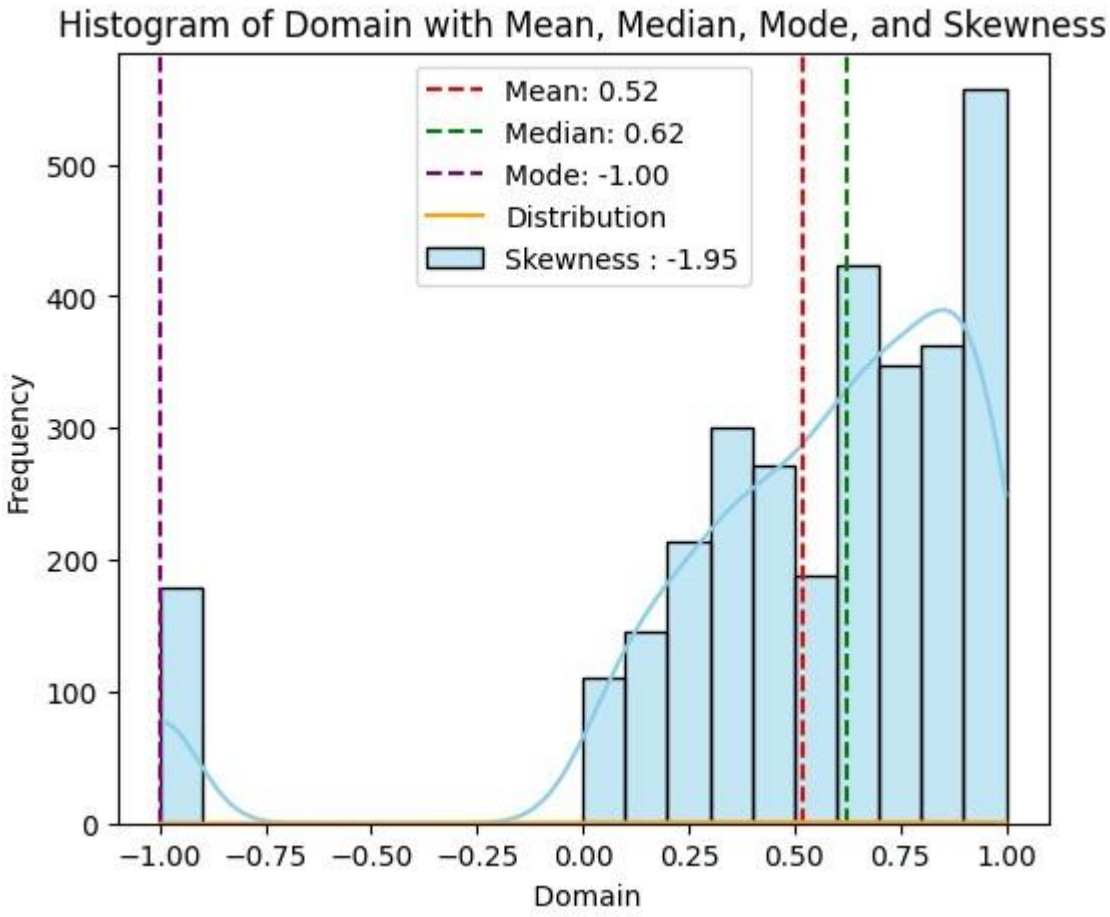
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='Domain' )
plt . title ( 'Box Plot of Domain' )
plt . ylabel ( 'Domain' )
plt . tight_layout ()
plt . show()
```



In [48]:
```
salary_values = numerical_df['Domain']
```

```

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of Domain with Mean, Median, Mode, and Skewness')
plt.xlabel(' Domain')
plt.ylabel('Frequency')
```

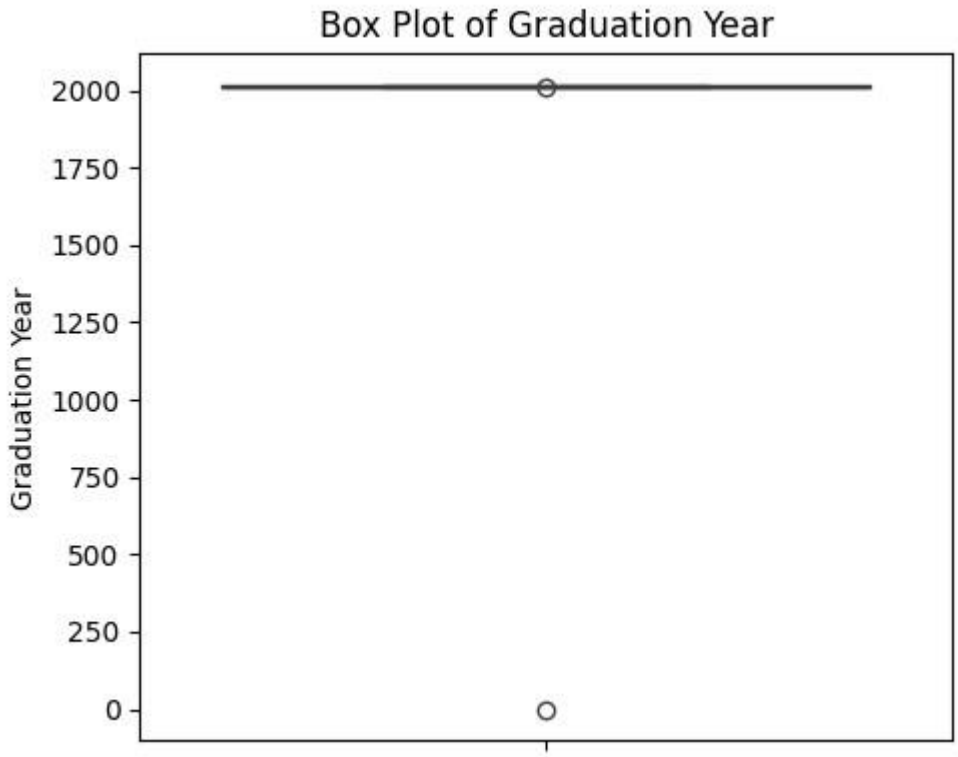
```
plt.legend()
plt.show()
```


Histogram of Domain with Mean, Median, Mode, and Skewness
Mean: 0.52
Median: 0.62
Mode: -1.00
Distribution
Skewness : -1.95
Frequency
0
100
200
300
400
500
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
Domain

4.3.2.10 GraduationYear

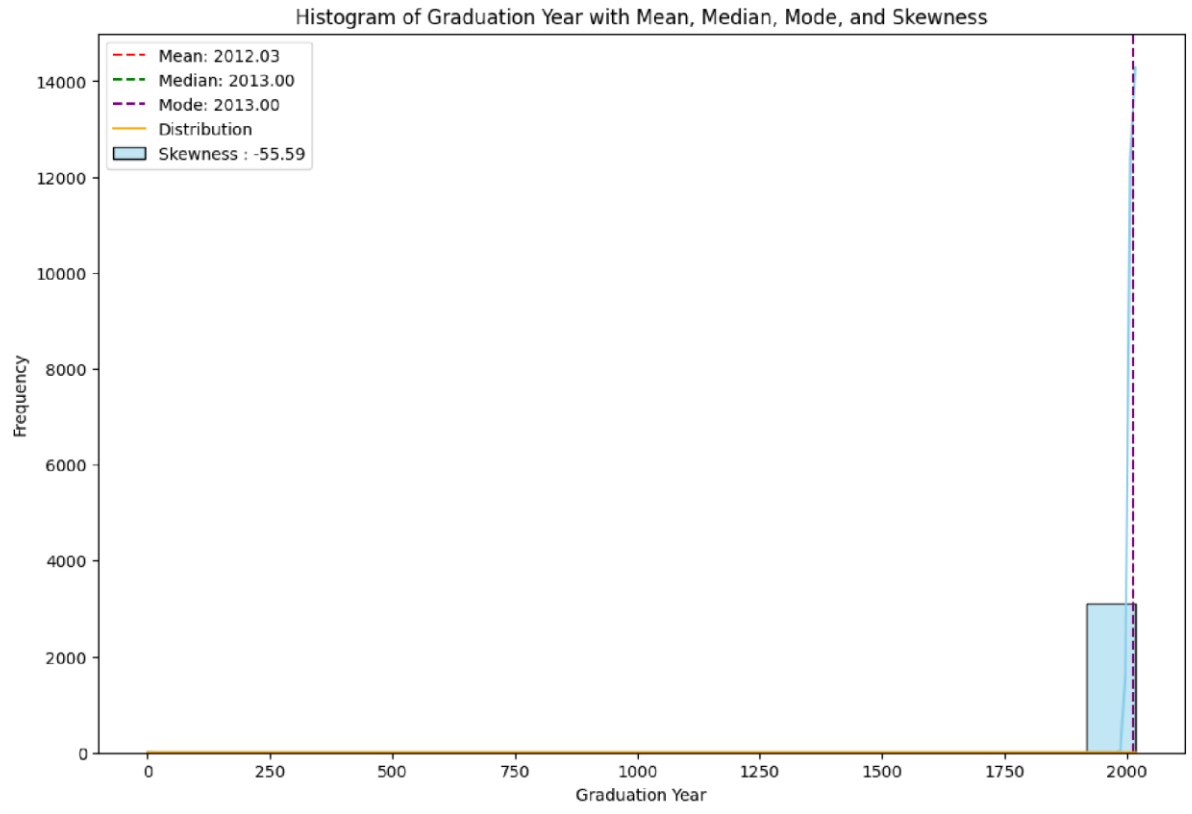
In [ ]:

```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='GraduationYear' )
plt . title ( 'Box Plot of Graduation Year' )
plt . ylabel ( 'Graduation Year' )
plt . tight_layout ()
plt . show()
```



In [ ]:

```
salary_values = numerical_df['GraduationYear']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(12, 8))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

# Add distribution line
x = np.linspace(salary_values.min(), salary_values.max(), 100)
plt.plot(x, norm.pdf(x, mean_salary, salary_values.std()), color='orange',

plt.title('Histogram of Graduation Year with Mean, Median, Mode, and Skewn
plt.xlabel('Graduation Year')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

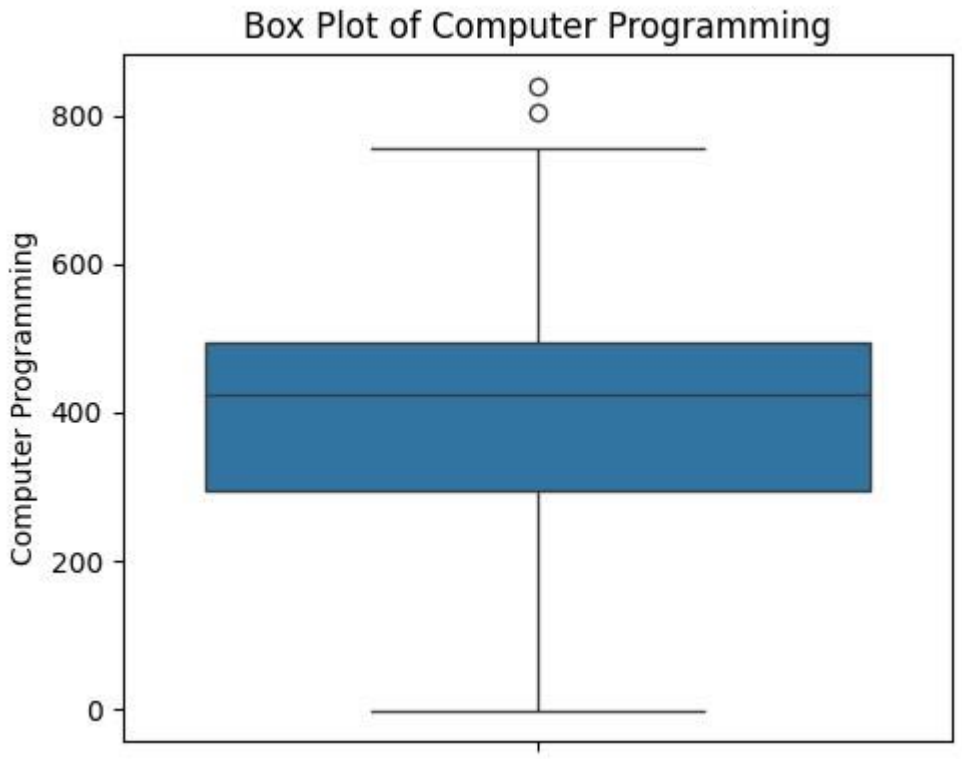


In [ ]:

4.3.2.11 ComputerProgramming

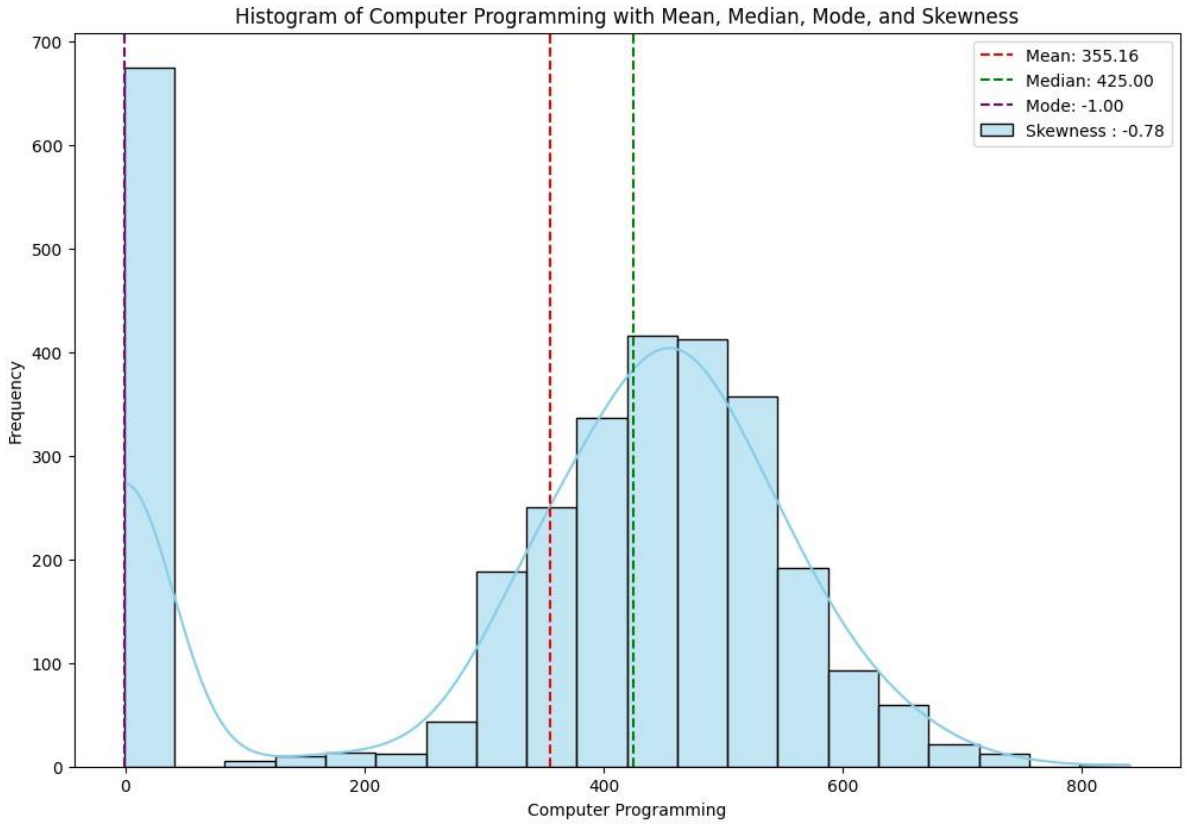
In [ ]:

```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='ComputerProgramming' )
plt . title ( 'Box Plot of Computer Programming' )
plt . ylabel ( 'Computer Programming' )
plt . tight_layout ()
plt . show()
```



In [ ]:

```
salary_values = numerical_df['ComputerProgramming']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(12, 8))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of Computer Programming with Mean, Median, Mode, and
plt.xlabel('Computer Programming')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



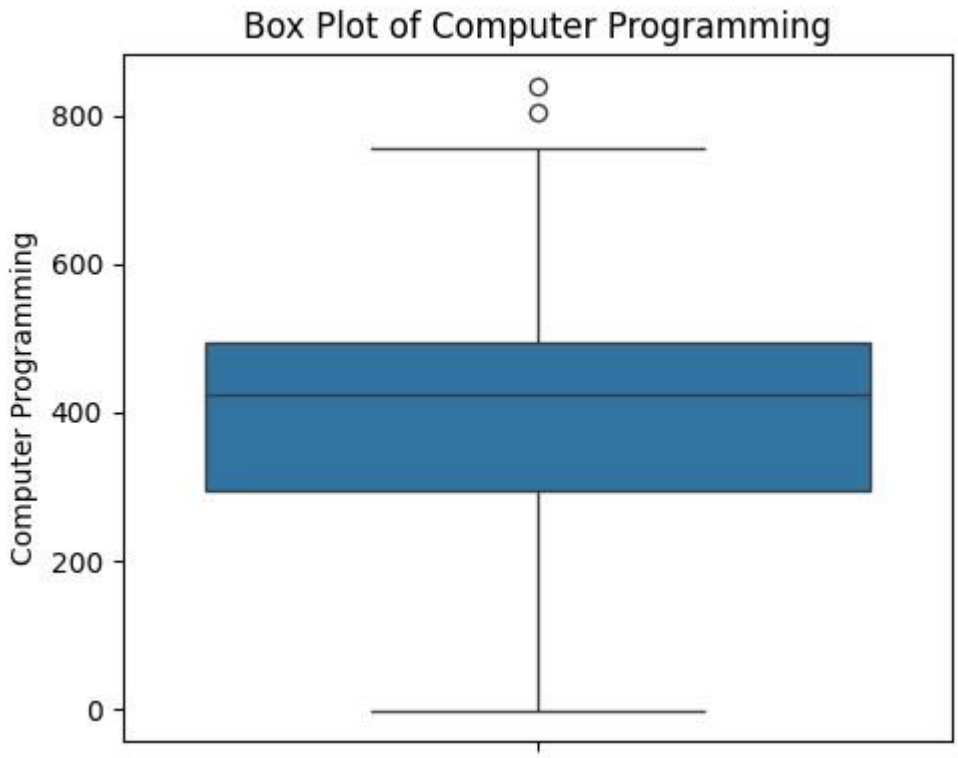
INNOMATICS

In [ ]:

4.3.2.12 ElectronicsAndSemicon

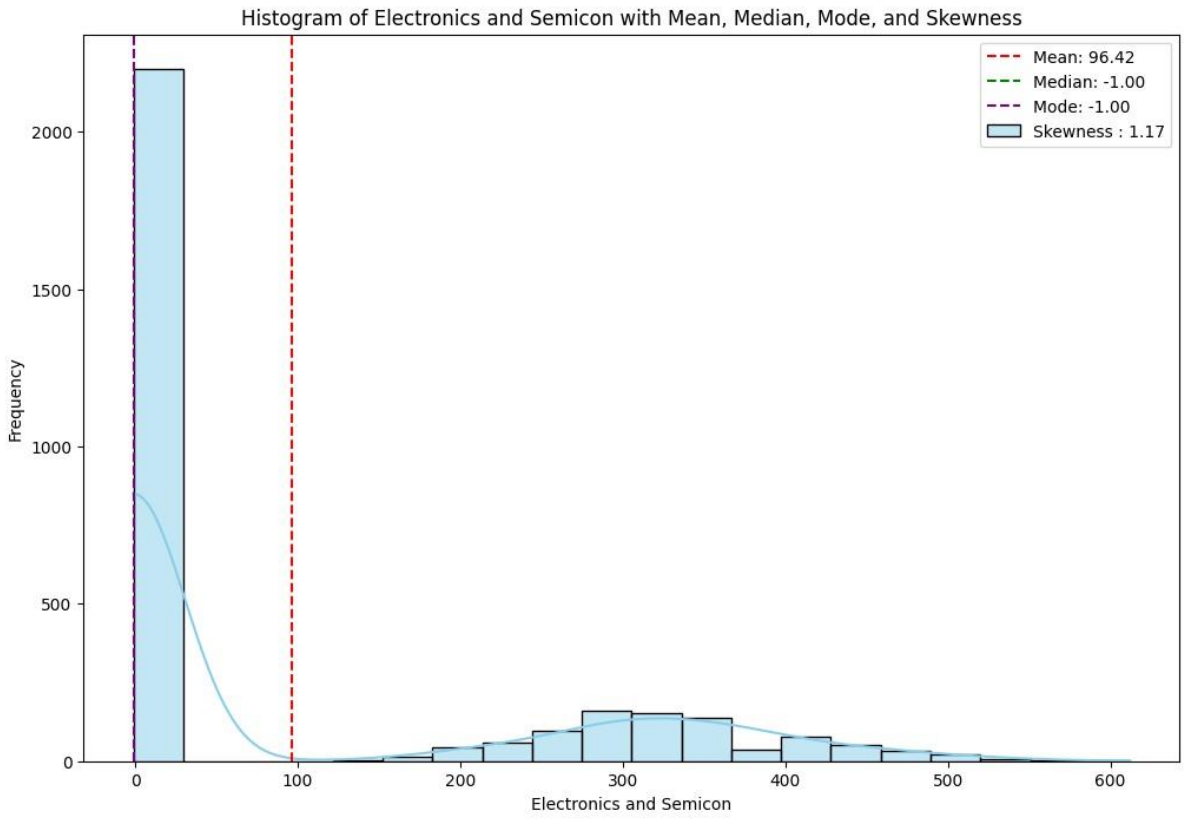
In [ ]:

```
plt . figure  ( figsize   =( 5,  4))
sns . boxplot  ( data =numerical_df  ,  y='ComputerProgramming'  )
plt . title  ( 'Box Plot of Computer Programming'          )
plt . ylabel  ( 'Computer Programming'  )
plt . tight_layout  ()
plt . show()
```



In [ ]:

```
salary_values = numerical_df['ElectronicsAndSemicon']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(12, 8))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of Electronics and Semicon with Mean, Median, Mode, a
plt.xlabel('Electronics and Semicon')
plt.ylabel('Frequency')
plt.legend()
plt.show()


```

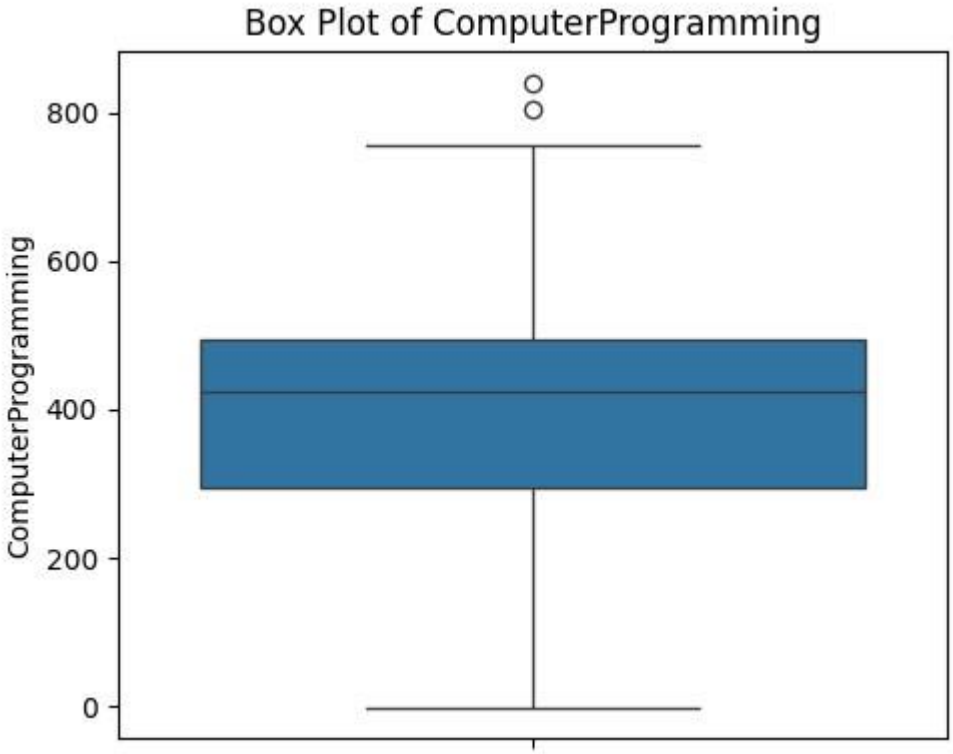


INNOMATICS

In [ ]:

4.3.2.13 ComputerScience

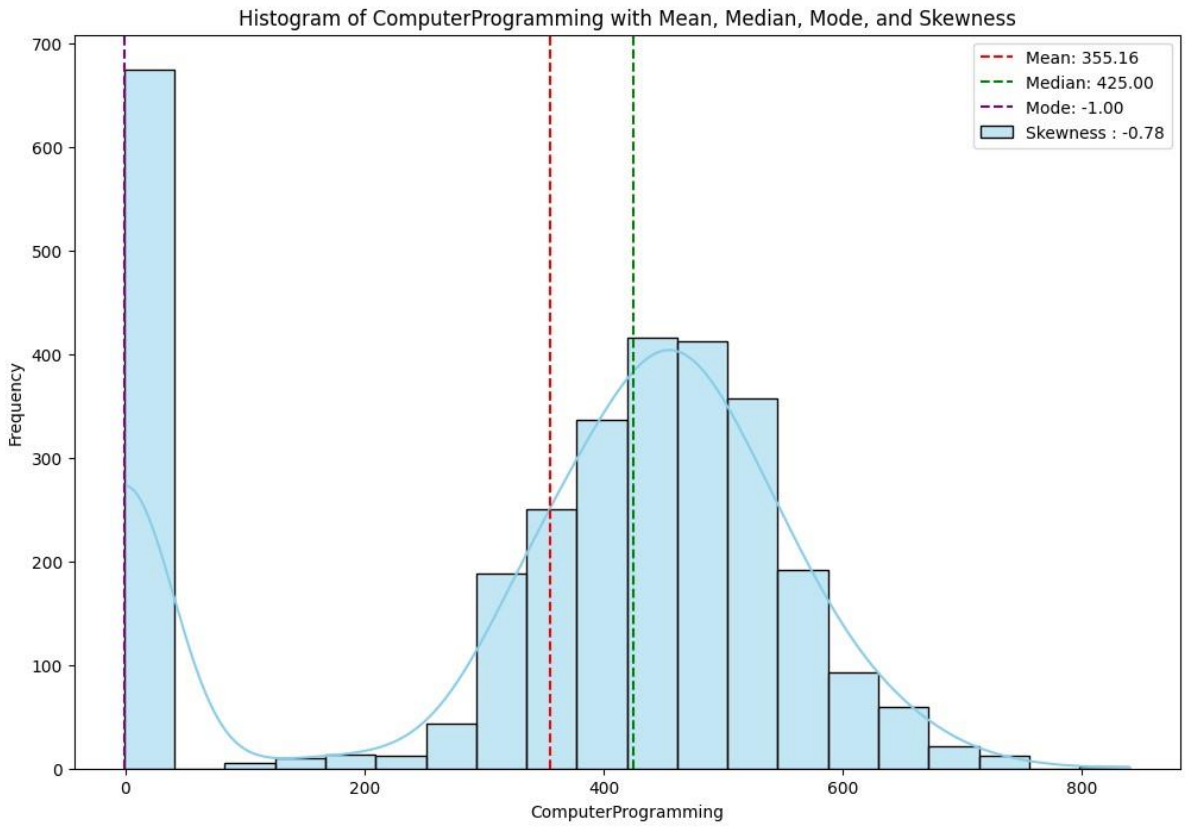
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='ComputerProgramming' )
plt . title ( 'Box Plot of ComputerProgramming' )
plt . ylabel ( 'ComputerProgramming' )
plt . tight_layout ()
plt . show()
```



INNOMATICS

In [ ]:

```
salary_values = numerical_df['ComputerProgramming']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(12, 8))
sns.histplot(salary_values, kde=True, bins=20, label = f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of ComputerProgramming with Mean, Median, Mode, and S
plt.xlabel('ComputerProgramming')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

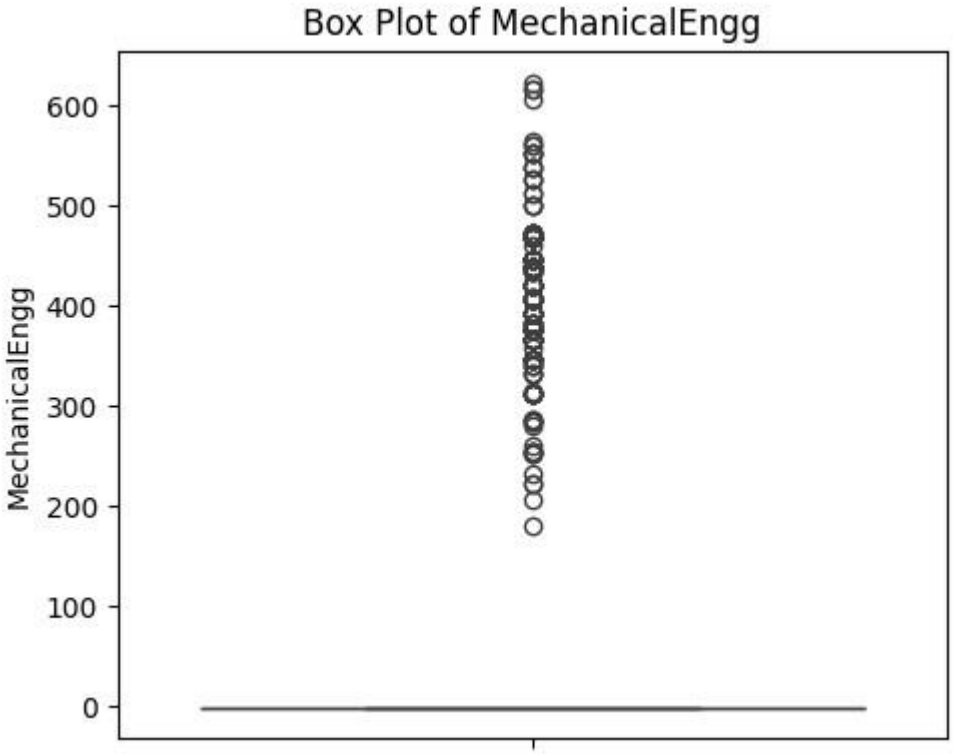


INNOMATICS

In [ ]:

4.3.2.14 MechanicalEngg

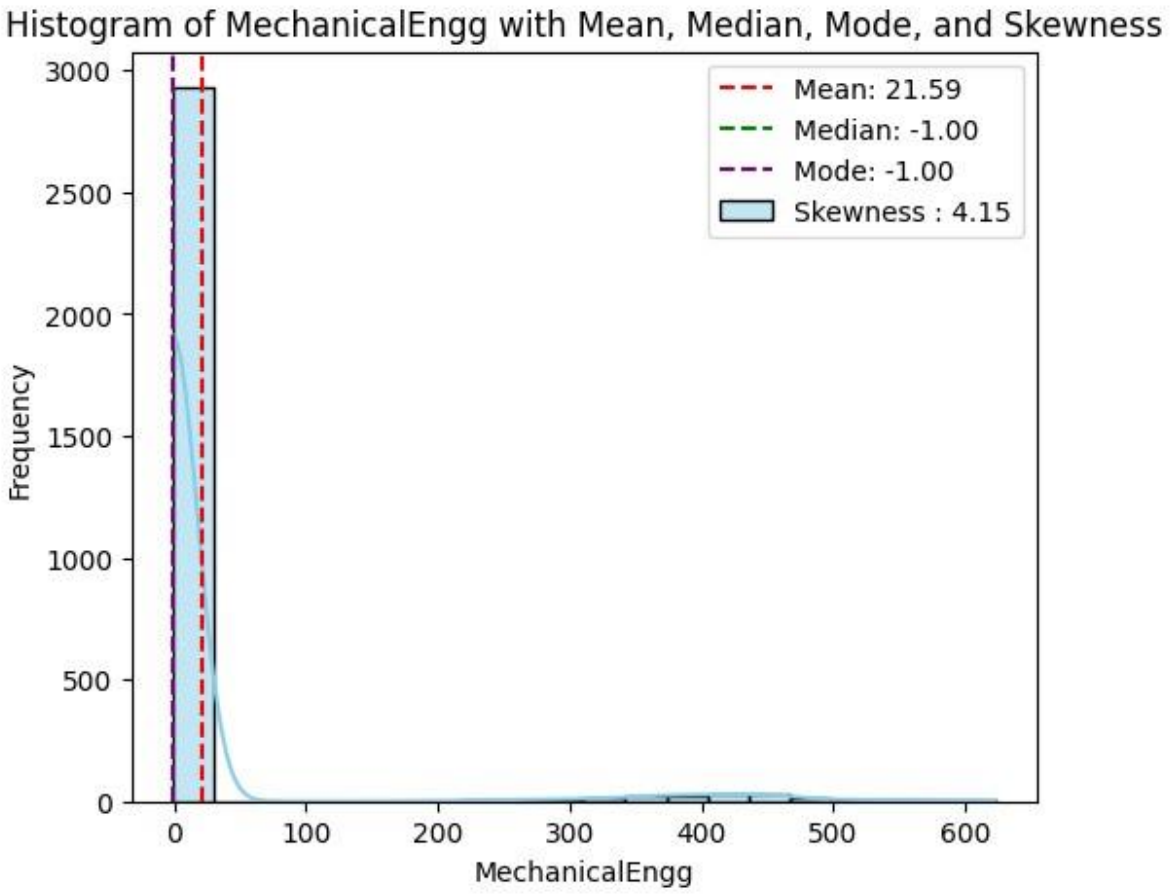
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='MechanicalEngg' )
plt . title ( 'Box Plot of MechanicalEngg' )
plt . ylabel ( 'MechanicalEngg' )
plt . tight_layout ()
plt . show()
```



INNOMATICS

In [49]:

```
salary_values = numerical_df['MechanicalEngg']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6, 5))
sns.histplot(salary_values, kde=True, bins=20, label=f"Skewness: {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of MechanicalEngg with Mean, Median, Mode, and Skewne
plt.xlabel('MechanicalEngg')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

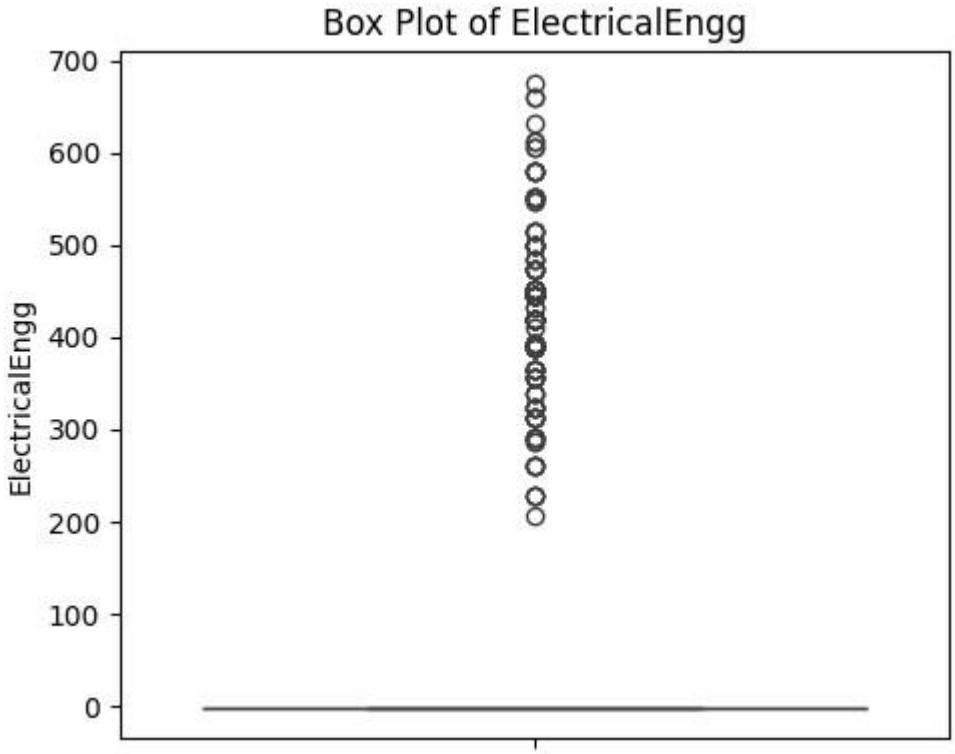


INNOMATICS

In

4.3.2.15 ElectricalEngg

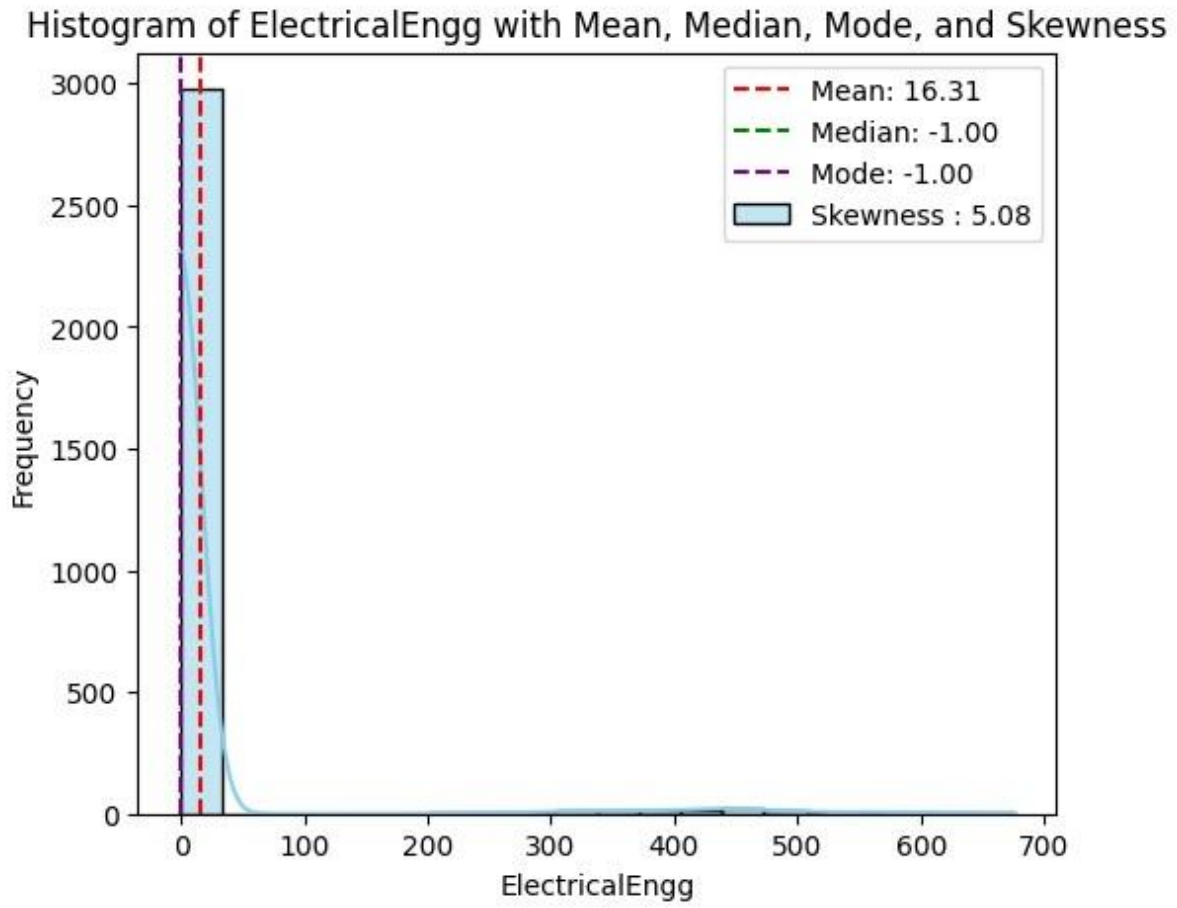
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='ElectricalEngg' )
plt . title ( 'Box Plot of ElectricalEngg' )
plt . ylabel ( 'ElectricalEngg' )
plt . tight_layout ()
plt . show()
```



INNOMATICS

In [50]:

```
salary_values = numerical_df['ElectricalEngg']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6, 5))
sns.histplot(salary_values, kde=True, bins=20, label=f"Skewness: {round
# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of ElectricalEngg with Mean, Median, Mode, and Skewne
plt.xlabel('ElectricalEngg')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```

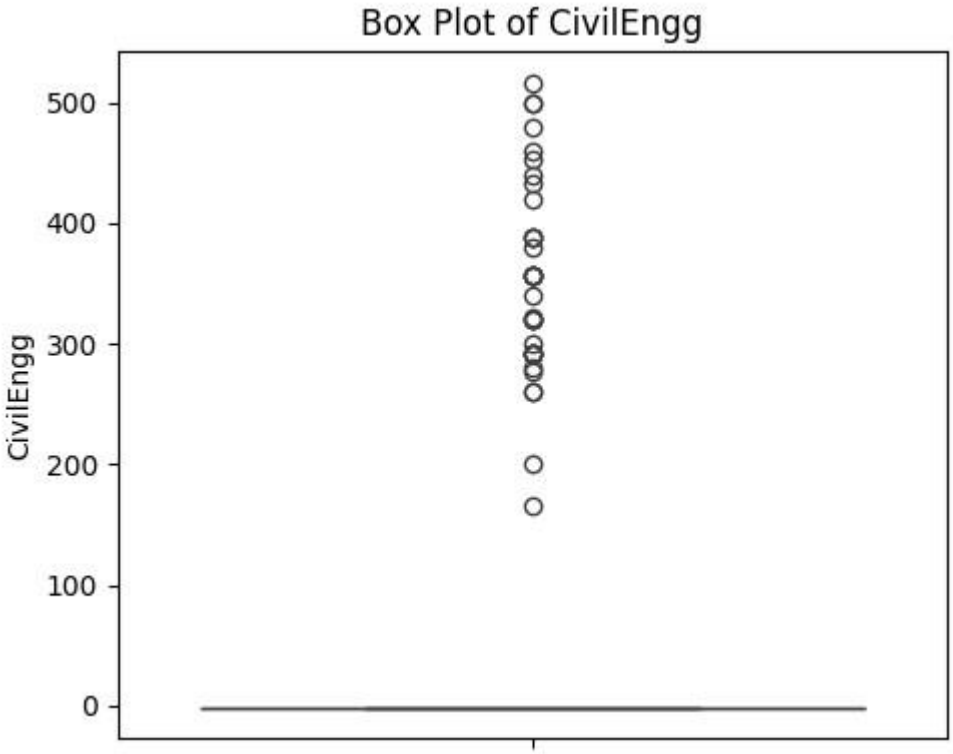


INNOMATICS

In

4.3.2.16 CivilEngg

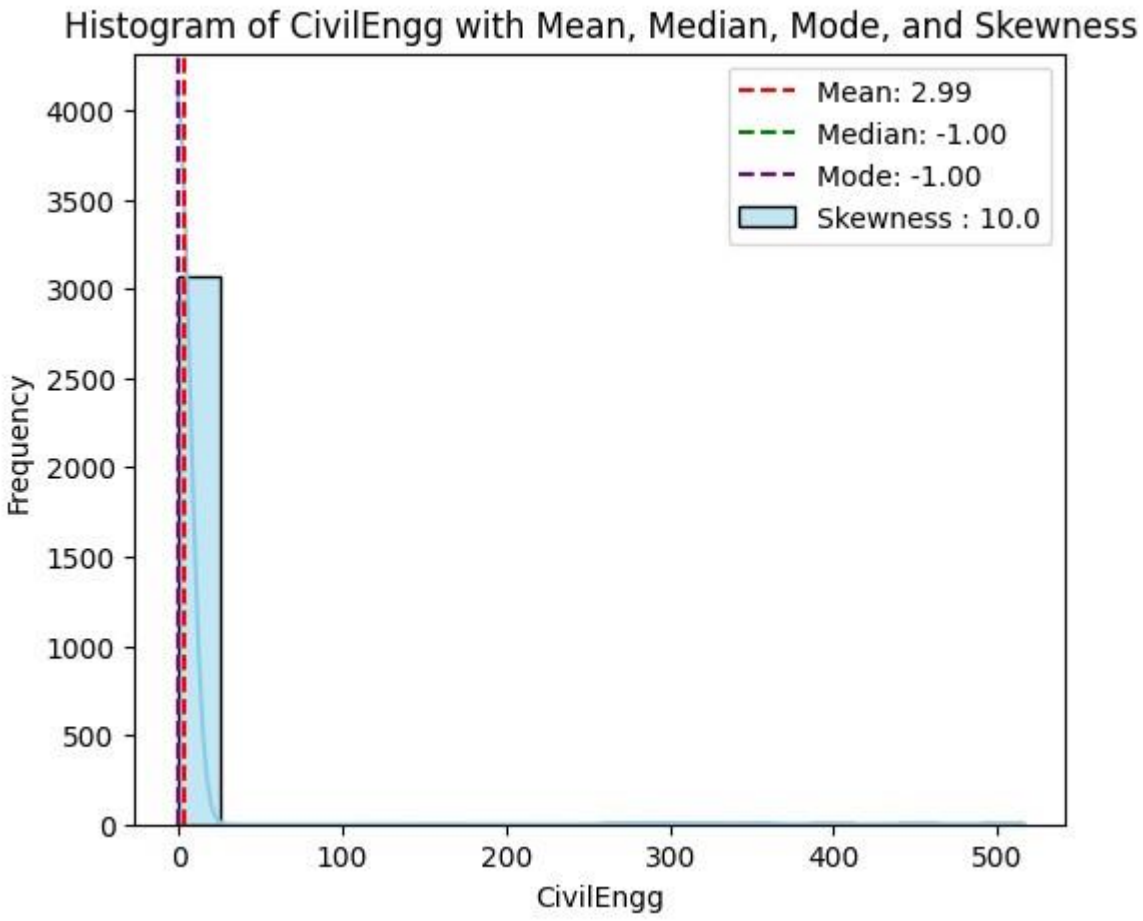
```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='CivilEngg' )
plt . title ( 'Box Plot of CivilEngg' )
plt . ylabel ( 'CivilEngg' )
plt . tight_layout ()
plt . show()
```



INNOMATICS

In [51]:

```
salary_values = numerical_df['CivilEngg']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6, 5))
sns.histplot(salary_values, kde=True, bins=20, label=f"Skewness : {round

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of CivilEngg with Mean, Median, Mode, and Skewness')
plt.xlabel('CivilEngg')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



INNOMATICS

In

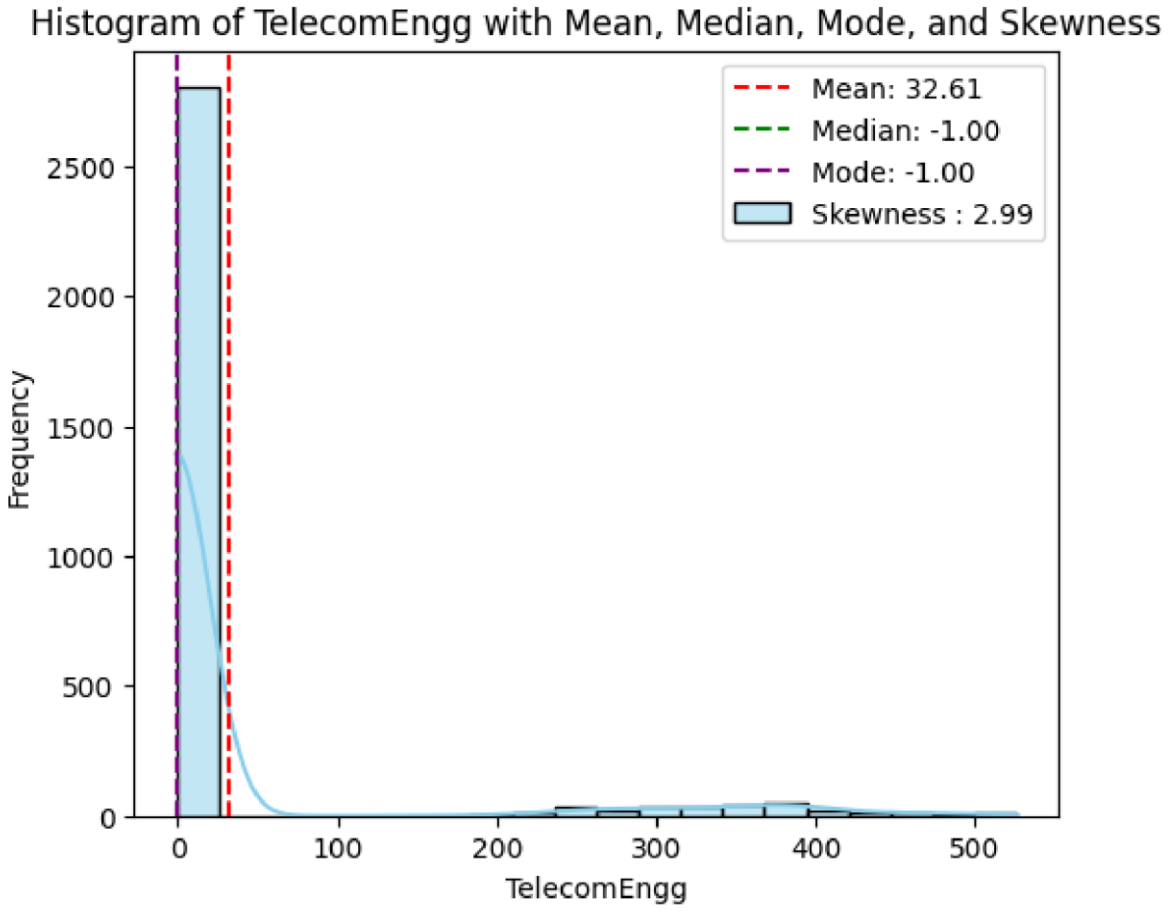
4.3.2.17 TelecomEngg

In [ ]:

```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='TelecomEngg' )
plt . title ( 'Box Plot of Age' )
plt . ylabel ( 'Age' )
plt . tight_layout ()
plt . show()
```

In [52]:

```
salary_values = numerical_df['TelecomEngg']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values,
             kde=True, bins=20,
             label = f"Skewness : {round(numerical_df['TelecomEngg'].skew(
             color='skyblue')

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of TelecomEngg with Mean, Median, Mode, and Skewness'
plt.xlabel('TelecomEngg')
plt.ylabel('Frequency')
plt.legend()
plt.show()


```



INNOMATICS

In

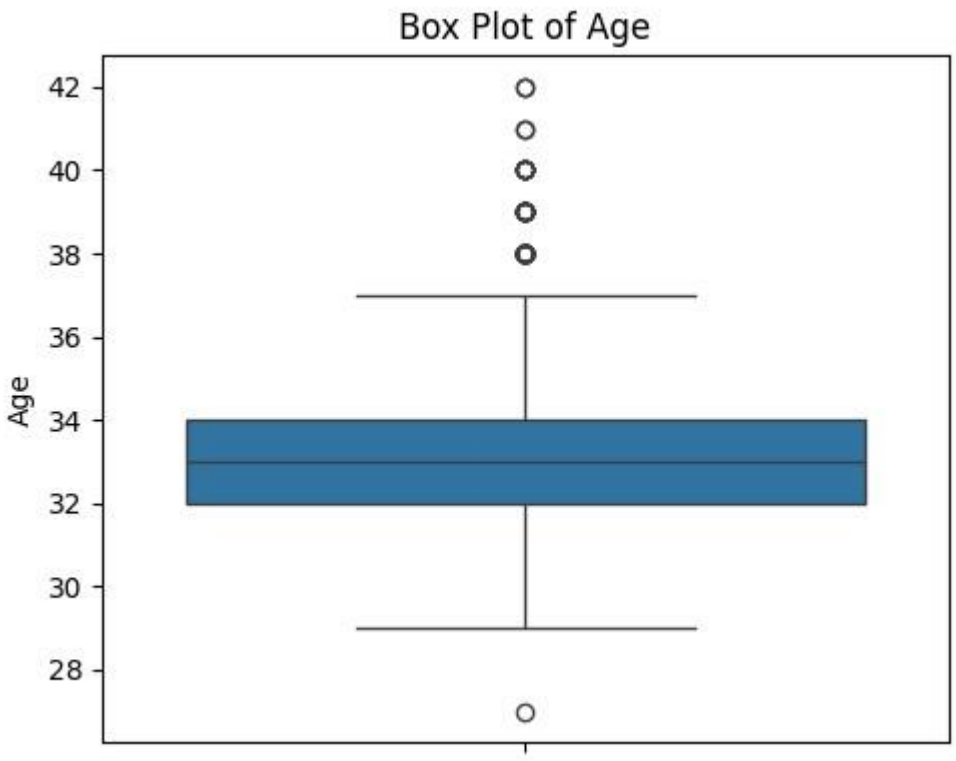
4.3.2.18 Age

In [53]:
```
numerical_df . Age. describe ()
```

Out[53]: count 3102.000000 mean
33.499678 std 1.715636
min 27.000000 25%
32.000000

50% 33.000000 75%
34.000000 max
42.000000
Name: Age, dtype: float64

```
plt . figure ( figsize =( 5, 4))
sns . boxplot ( data =numerical_df , y='Age' )
plt . title ( 'Box Plot of Age' )
plt . ylabel ( 'Age' )
plt . tight_layout ()
plt . show()
```

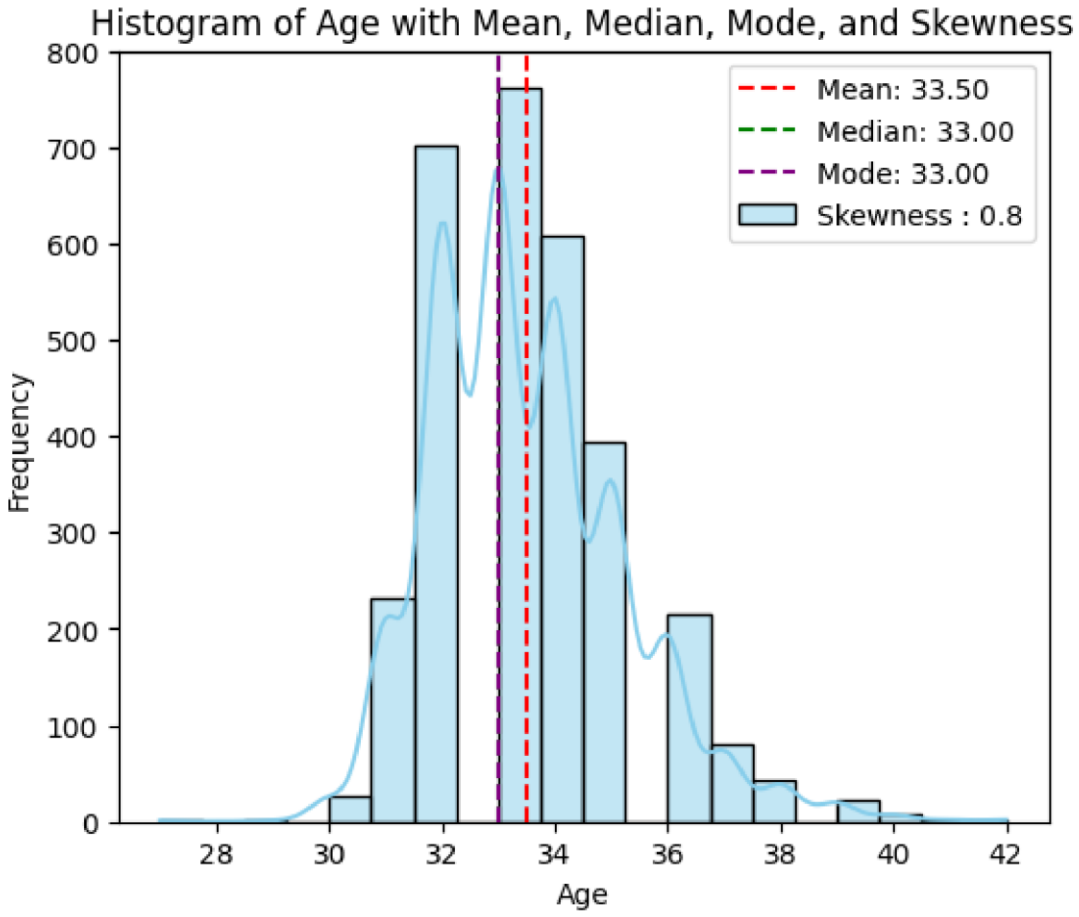


In [ ]:

INNOMATICS

In [54]:

```
salary_values = numerical_df['Age']

# Calculate mean, median, mode, and skewness
mean_salary = salary_values.mean()
median_salary = salary_values.median()
mode_salary = salary_values.mode()[0]
skewness_salary = skew(salary_values)

# Plot histogram
plt.figure(figsize=(6,5))
sns.histplot(salary_values,
             kde=True, bins=20,
             label = f"Skewness : {round(numerical_df['Age'].skew(),2)}",
             color='skyblue')

# Add mean, median, mode, and skewness information
plt.axvline(mean_salary, color='red', linestyle='--', label=f'Mean: {mean_
plt.axvline(median_salary, color='green', linestyle='--', label=f'Median:
plt.axvline(mode_salary, color='purple', linestyle='--', label=f'Mode: {mo

plt.title('Histogram of Age with Mean, Median, Mode, and Skewness')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.legend()
plt.show()


```



INNOMATICS

In

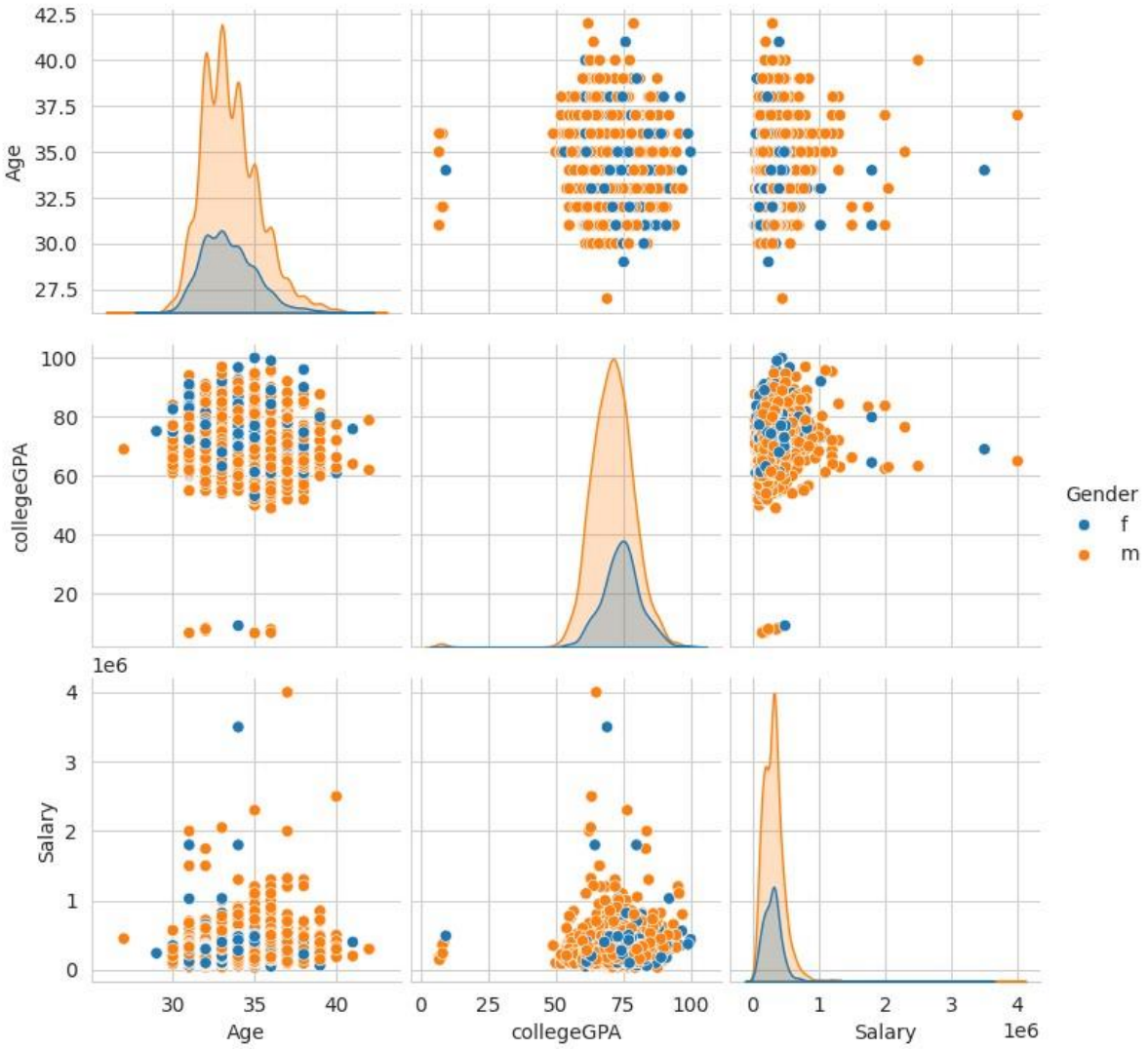
4.3.2.19 Observation: For remaining plots

1. **English & Logical:** Half of the students scored below 500 in the logical exams. The average scores for the English and Logical sections of the AMCAT test stand at approximately 503.15 and 502.89, respectively. The standard deviations are approximately 104.63 and 86.64, respectively. The minimum scores recorded for both sections are 180 and 195, respectively, while the maximum scores are 875 and 795, respectively.
2. **Engg:** These variables represent scores in different engineering disciplines within the AMCAT test. The mean scores are approximately 21.59, 16.31, 32.61, and 2.99, respectively. However, the data suggests that the majority of individuals did not take these specific sections, as indicated by the prevalence of -1 values.
3. **Age:** The average age of individuals stands at approximately 33.50 years, showcasing the central tendency of the age distribution. The standard deviation of approximately 1.72 years.

5 Bivariate Analysis

In [ ]:

```
columns_of_interest = [ 'Age' , 'collegeGPA' , 'Salary' ]

sns . set_style ( "whitegrid" )
sns . pairplot ( df , hue='Gender' , vars =columns_of_interest )
plt . show()
```



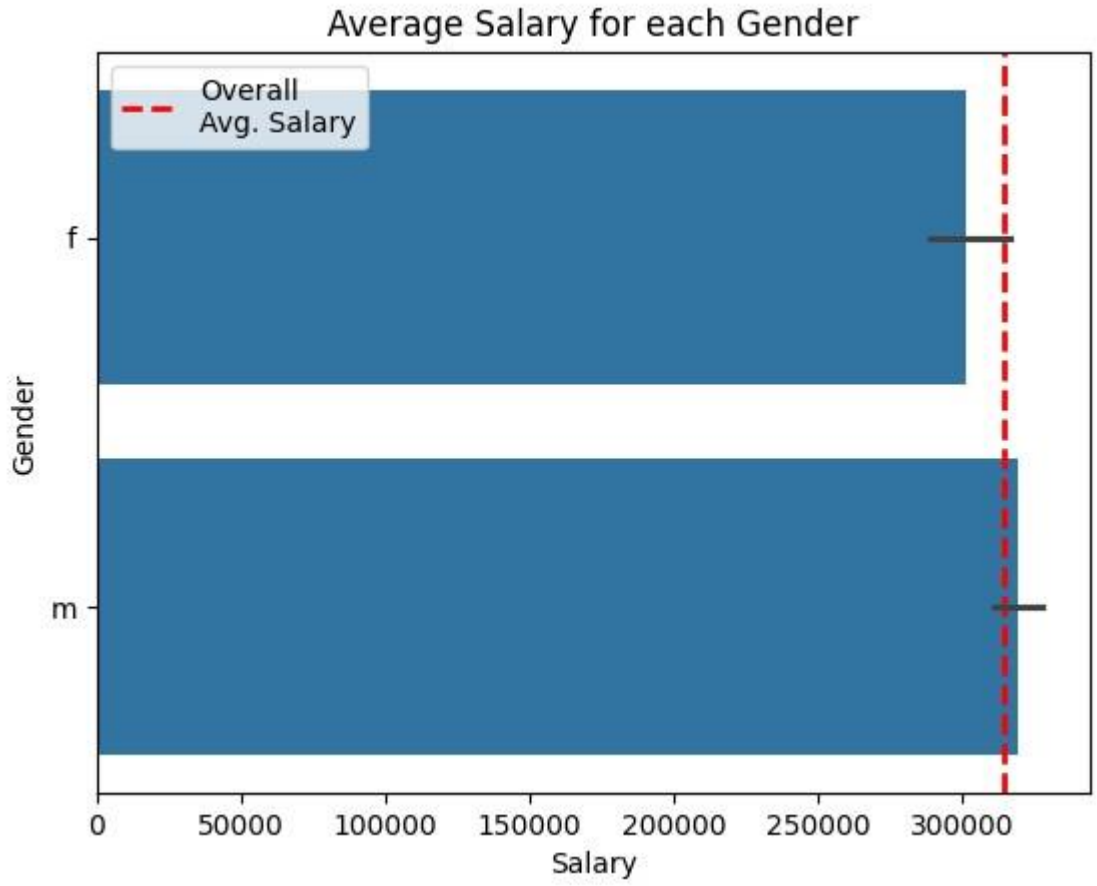
INNOMATICS

5.1 Barplot (CAT vs NUM)

5.1.1 *Gender vs Salary*

[55]:

```
plt . axvline ( df [ 'Salary' ]. mean() , color ='red' , linewidth = 2, linestyle ='--'

sns . barplot ( data =df , x='Salary' , y='Gender' )

plt . title ( 'Average Salary for each Gender' )
plt . xlabel ( 'Salary' )
plt . ylabel ( 'Gender' )
plt . show()
```

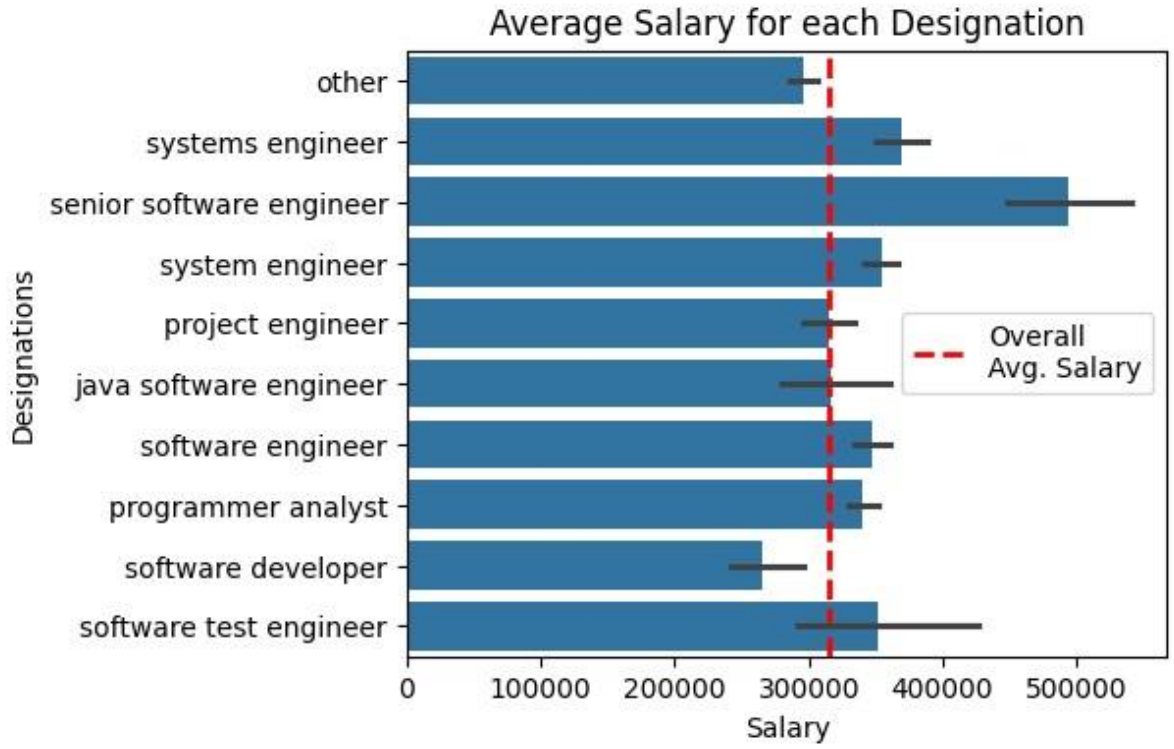


Observation: Both male and female salaries are approximately equal on average, suggesting no gender bias overall, though females tend to receive salaries below the overall average.

5.1.2 *Salary vs Designation*

In [67]:

```
plt . figure ( figsize =( 5, 4))
plt . axvline ( df [ 'Salary' ]. mean() , color ='red' , linewidth = 2, linestyle ='--'

sns . barplot ( data =df , x='Salary' , y='Designation' )

plt . title ( 'Average Salary for each Designation' )
plt . xlabel ( 'Salary' )
plt . ylabel ( 'Designations' )
plt . show()
```



Type *Markdown* and LaTeX: $\square^2$

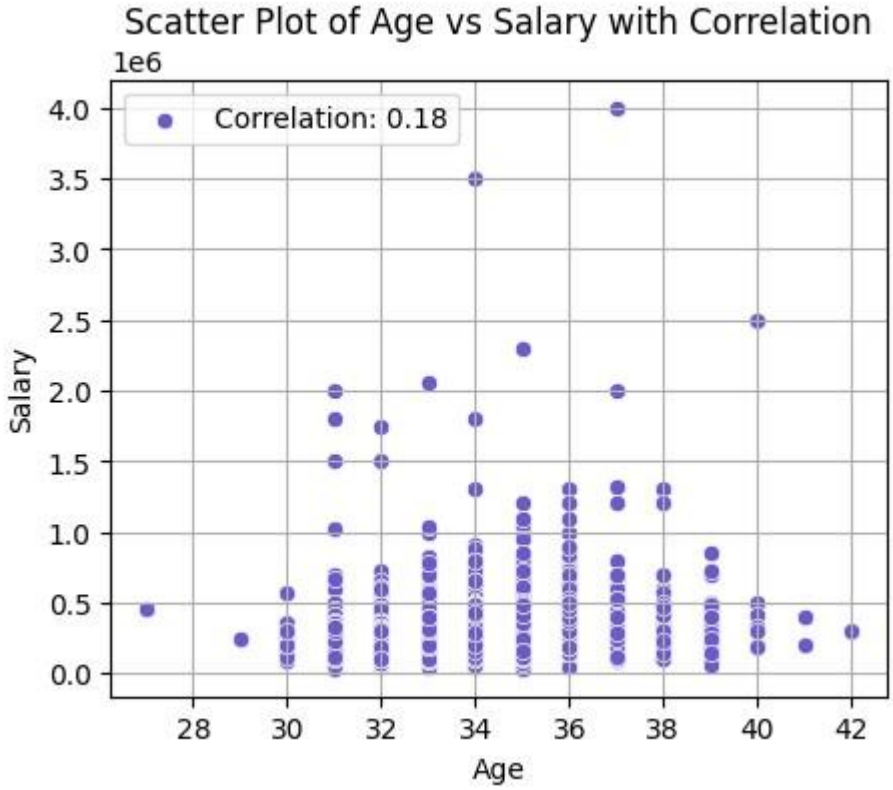
INNOMATICS

5.2 Scatterplot (NUM vs NUM)

5.2.1 *Age vs Salary*

[77]:

```
correlation = df['Age'].corr(df['Salary'])
plt.figure(figsize=(5, 4))
sns.scatterplot(x='Age', y='Salary', data=df,
                label=f'Correlation: {correlation:.2f}',
                color='slateblue')

plt.title('Scatter Plot of Age vs Salary with Correlation')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.grid(True)
plt.show()
```



Observation: Senior Software Engineers have the highest salary, but also the highest standard deviation. Software Developers and Technical Support Engineers have salaries below the average.

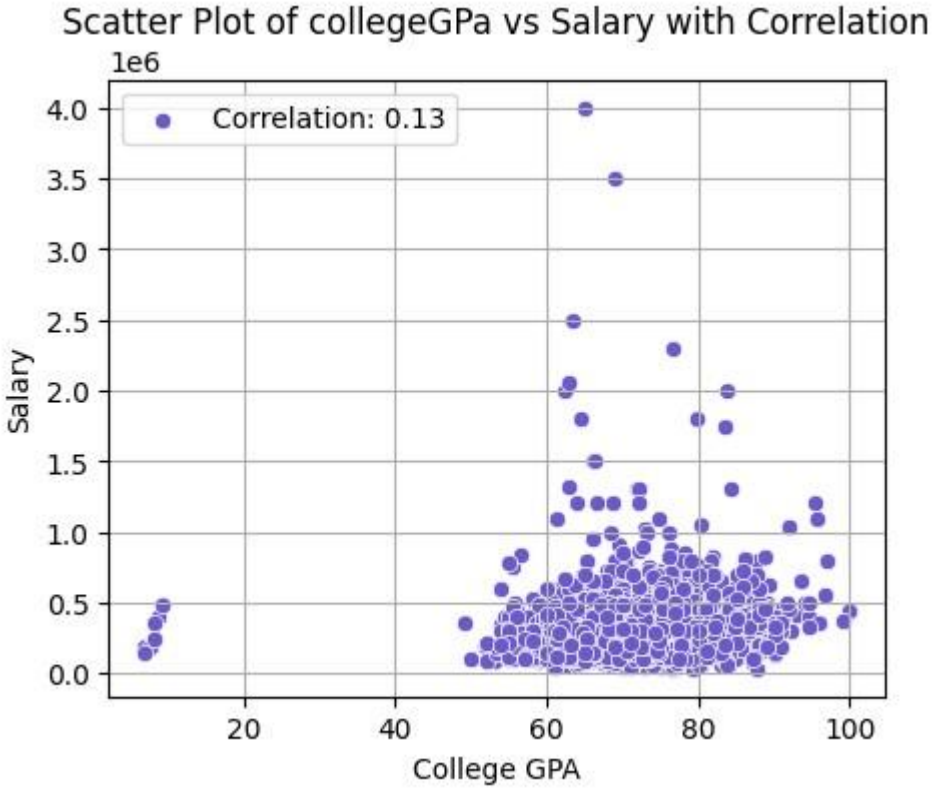
In [ ]:

```
# Calculate correlation
correlation = df['collegeGPA'].corr(df['Salary'])

# Create scatter plot
sns.scatterplot(x='collegeGPA', y='Salary', data=df,
                label=f'Correlation: {correlation:.2f}',
                color='slateblue')

plt.title('Scatter Plot of collegeGPa vs Salary with Correlation')
plt.xlabel('College GPA')
plt.ylabel('Salary')
plt.grid(True)
plt.show()
```

5.2.2 *collgeGPA vs Salary*

INNOMATICS

```
# Calculate correlation
correlation = df['collegeGPA'].corr(df['Salary'])
plt.figure(figsize=(5, 4))
# Create scatter plot
sns.scatterplot(x='collegeGPA', y='Salary', data=df,
                label=f'Correlation: {correlation:.2f}',
                color='slateblue')


plt.title('Scatter Plot of collegeGPa vs Salary with Correlation')
plt.xlabel('College GPA')
plt.ylabel('Salary')
plt.grid(True)
plt.show()
```



[60]:

Type *Markdown* and LaTeX: $\square^2$

In [62]:

```
textual_columns = ['Designation','JobCity','10board','12board','Specializa
```

In [63]:

```
def collapsing_categories(df1, data):
    for Designation in df1[data].unique():
        min_count = df1[data].value_counts()[:10].min()
        if df1[df1[data] == Designation][data].value_counts()[0] < min_count:
            df1.loc[df1[data] == Designation, data] = 'other'
```


INNOMATICS

5.3 In [64]:

```
for cols in textual_columns :
    collapsing_categories ( df , cols )

```

Crosstab (CAT vs CAT)

INNOMATICS

[65]:

```
for cols in textual_columns :
    print ('')
    print ('Top 10 categories in:', cols)
    print ('')
    print (df[cols].value_counts())
    print ('')
    print ('*'*100)


```


INNOMATICS

Top 10 categories in: Designation

other 1778 software engineer
435 software developer 200 system
engineer 179 programmer analyst
118 systems engineer 99 java software
engineer 88 software test engineer 83
project engineer 64 senior software
engineer 58 Name: Designation, dtype:
int64

****

********** Top 10

categories in: JobCity

other 867 Bangalore 495

-1 348

Noida 282

Hyderabad 274

Pune 236

Chennai 215

Gurgaon 163

New Delhi 143

Kolkata 79

Name: JobCity, dtype: int64

****

********** Top 10

categories in: 10board

cbse 1098 state board
920 other 362 0
265 icse 221 ssc
103 up board 64 matriculation
25 rbse 16 up
14 board of secondary education 14 Name:
10board, dtype: int64

INNOMATICS

****

********** Top 10

categories in: 12board

cbse	1115	state board	981

INNOMATICS

other 462 0 272
icse 101 up board
66 board of intermediate 32 isc
31 board of intermediate education 27 up
15 Name: 12board, dtype: int64

****

********** Top 10 categories

in: Specialization

electronics and communication engineering 717 computer science
& engineering 587 information technology 504
computer engineering 460 other
195 computer application 191 mechanical engineering
149 electronics and electrical engineering 148 electronics &
telecommunications 91 electrical engineering
60 Name: Specialization, dtype: int64

****

********** Top 10

categories in: CollegeState

Uttar Pradesh 690 other
607 Karnataka 278

Tamil Nadu 276

Telangana 259

Maharashtra 207

Andhra Pradesh 184

Punjab 160

West Bengal 156

Haryana 143

Madhya Pradesh 142

Name: CollegeState, dtype: int64

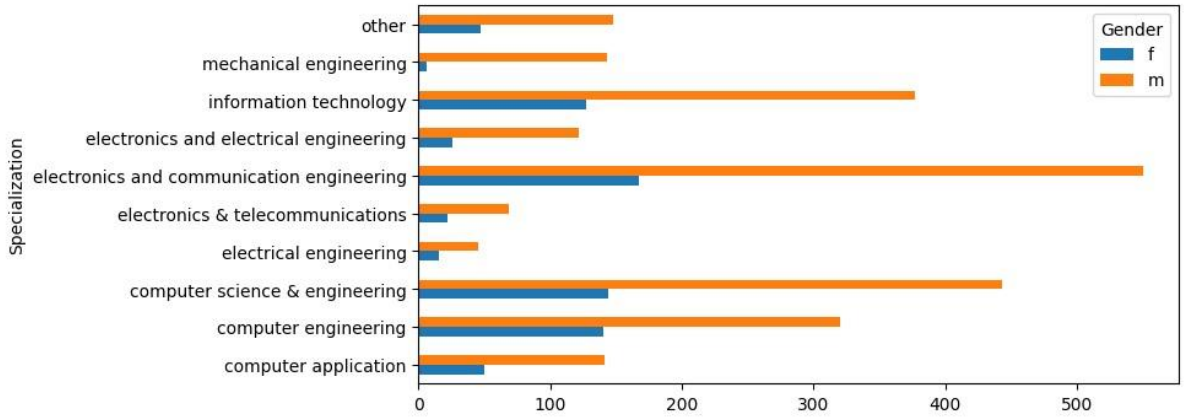
****

5.3.1 **********

Gender vs Specialization

In [66]:
```
1 pd. crosstab ( df [ 'Gender' ] , df [ 'Specialization' ]). T. plot ( kind = 'barh' , figs
```

Out[66]: <Axes: ylabel='Specialization' >



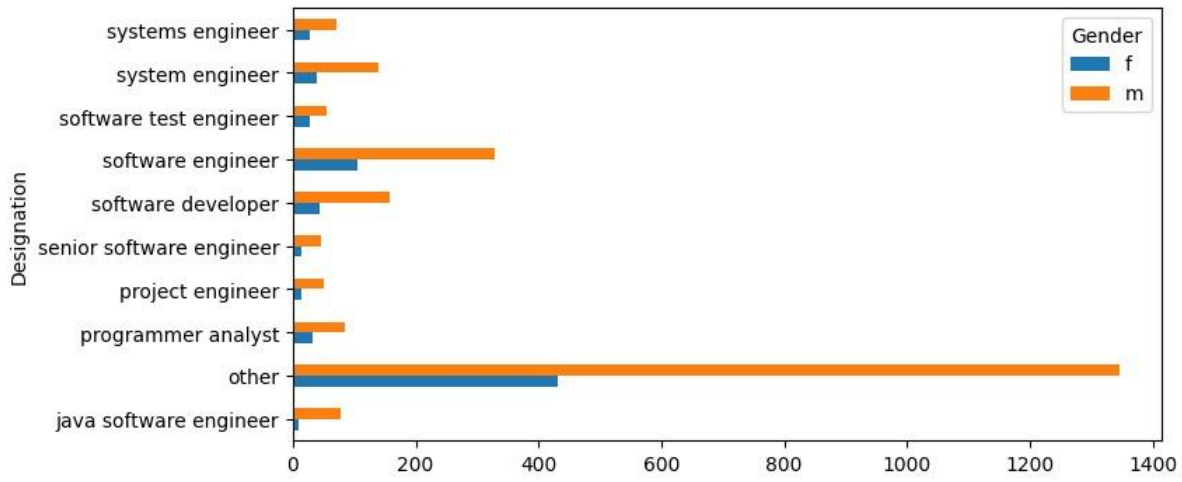
5.3.2 *Gender vs Designation*

In [68]:
```
1 plt.figure(figsize=(5, 4))
2 pd.crosstab(df['Gender'], df['Designation']).T.plot(kind = 'barh', figsize
```

Out[68]: <Axes: ylabel='Designation'>

<Figure size 500x400 with 0 Axes>

INNOMATICS



5.4 Research Questions

5.4.1 *1. Times of India article dated Jan 18, 2019 states that "After doing your Computer Science Engineering if you take up jobs as a Programming Analyst, Software Engineer, Hardware Engineer and Associate Engineer you can earn up to 2.5-3 lakhs as a fresh graduate."*

```
designations = data['Designation'].value_counts().sort_index()
pd.set_option('display.max_rows', None)
#print(designations)
```

In [ ]:
```
data['Designation'] = data['Designation'].replace(['programmer analyst tra
data['Designation'] = data['Designation'].replace(['software eng', 'softwa
```

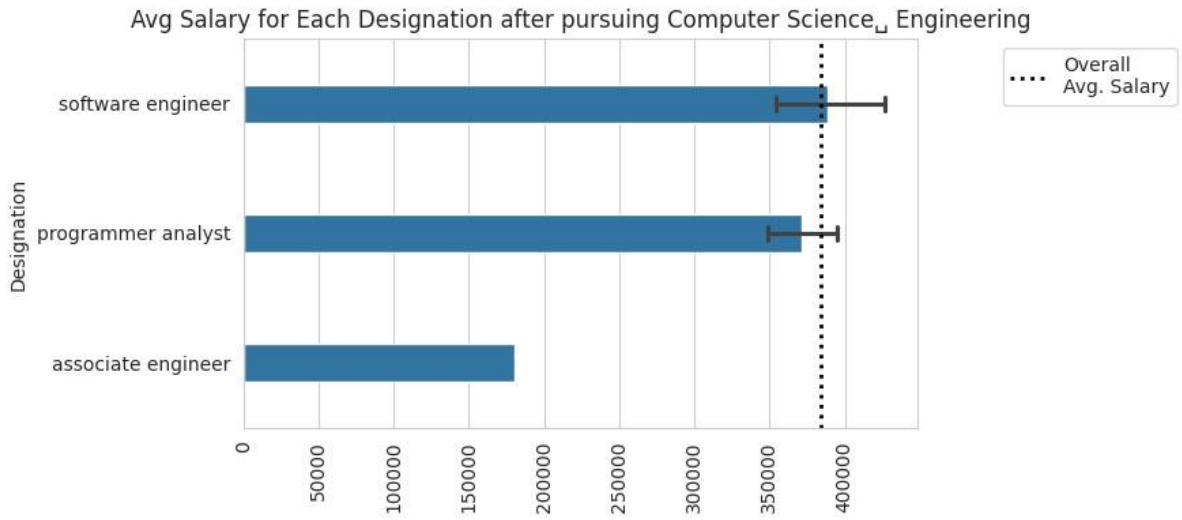
In [ ]:
```
df1 = data[(data["Designation"].isin(["programmer analyst", "software engi
```

In [ ]:
```
fig, ax = plt.subplots(figsize=(10, 4))

    sns.barplot(x='Salary', y='Designation',
    data=df1, capsize=0.1,width=0.3,ax=ax)

ax.axvline(df1['Salary'].mean(), color='k',linestyle=':',linewidth=2, labe ax.set_title('Avg Salary
    for Each Designation after pursuing Computer Scie
ax.legend(loc='upper right', bbox_to_anchor=(1.4, 1))
ax.set_xlabel('')
```

```
9   ax.set_xticklabels(ax.get_xticklabels(), rotation=90)
10  plt.tight_layout()
11  plt.show()
12
```

<ipython-input-164-f6fc85598de9>:9: UserWarning: FixedFormatter should only b e used together with FixedLocator

ax.set_xticklabels(ax.get_xticklabels(), rotation=90)



INNOMATICS

In [ ]:

```
import random
n = 30
salary_random = random.sample(df1['Salary'].tolist(), n)
print(salary_random)

```

[335000.0, 400000.0, 335000.0, 305000.0, 530000.0, 350000.0, 265000.0, 300000.0, 325000.0, 335000.0, 120000.0, 500000.0, 700000.0, 390000.0, 500000.0, 225000.0, 560000.0, 475000.0, 600000.0, 400000.0, 120000.0, 450000.0, 120000.0, 190000.0, 410000.0, 120000.0, 390000.0, 520000.0, 450000.0, 350000.0]

5.4.2 Sample values calculations

In [ ]:

```
from scipy.stats import t,norm

import statistics
print('Sample Mean: ', statistics.mean(salary_random))
print('Sample Standard Deviation: ', statistics.stdev(salary_random))
```

Sample Mean: 369000.0

Sample Standard Deviation: 148308.85759582705

In [ ]:

```
sample_size = n
sample_mean = statistics.mean(salary_random)
pop_mean = data.Salary.mean()
sample_std = statistics.stdev(salary_random)
pop_mean
```

Out[176]: 307699.8499249625

In [ ]:

```
def t_score(sample_size, sample_mean, pop_mean, sample_std):
    numerator = sample_mean - pop_mean
    denomenator = sample_std / sample_size **0.5
    return numerator / denomenator

```

In [ ]:

```

```

5.4.3 Calculating t-value

In [ ]:

```
t_value = t_score(sample_size, sample_mean, pop_mean, sample_std)
print("t-value :", t_value)

```


INNOMATICS

In [ ]:

t-value : 2.2638887197182984

5.4.4 Calculating t-statistic

```
confidence_level = 0.95
alpha = 1 - confidence_level
t_critical = t.ppf(1 - alpha/2, df = 99)
print("t-critical:\t", t_critical)
```

t-critical: 1.9842169515086827

In [ ]:

```
if (t_value < t_critical):
print("There is not enough evidence to reject the Null Hypothesis") else:
print("There is sufficent evidence to reject the Null Hypothesis")
```

There is sufficent evidence to reject the Null Hypothesis

In [ ]:

```
p_value = 2 * (1.0 - norm.cdf(np.abs(t_value))) 
print("p_value :\t ", p_value) if(p_value > alpha):
print("There is not enough evidence to reject the Null Hypothesis") else:
print("There is sufficent evidence to reject the Null Hypothesis")
```

p_value : 0.023580959916433608

There is sufficent evidence to reject the Null Hypothesis

In [ ]:

```
alpha = 0.05
job_group = df1.groupby('Designation')
job_salary_mean = job_group['Salary'].mean()
job_salary_std = job_group['Salary'].std()
```

In [ ]:

```
print("Mean salaries for different job roles:")
print(job_salary_mean)
print("\nStandard deviation of salaries for different job roles:")
print(job_salary_std)
```

Mean salaries for different job roles:
Designation
associate engineer 180000.000000 programmer
analyst 371346.153846 software engineer
388409.090909 Name: Salary, dtype: float64

Standard deviation of salaries for different job roles:

INNOMATICS

In [ ]:

```
Designation

associate engineer           NaN programmer analyst
63443.797054 software engineer    191522.068360

Name: Salary, dtype: float64
```

```
from scipy.stats import ttest_1samp
prog_analyst_salaries = df1.loc[df1['Designation'] == 'programmer analyst'
software_eng_salaries = df1.loc[df1['Designation'] == 'software engineer',
hardware_eng_salaries = df1.loc[df1['Designation'] == 'hardware engineer', 5
assoc_eng_salaries = df1.loc[df1['Designation'] == 'associate engineer', '
6
7
```

In [ ]:

```
1 expected_range = (250000, 300000)

2 for job, salaries in [("programmer analyst", prog_analyst_salaries),("soft 3    t_stat, p_val =
                        ttest_1samp(salaries, expected_range[0], alternative='
      print(f"One-sample t-test for {job}:")
      print(f" t_critical: {t_stat:.2f}") 6    print(f" p_value: {p_val:.5e}") 7
      if p_val < 0.05:
  print(" Result: There is sufficent evidence to reject the Null Hyp 9    else:
                                10       print(" Result: There is not enough evidence to reject the Null Hy
11
```

```
One-sample t-test for programmer analyst: t_critical: 9.75 p_value: 2.65118e-10
 Result: There is sufficent evidence to reject the Null Hypothesis

One-sample t-test for software engineer: t_critical: 7.58 p_value: 6.09466e-12
 Result: There is sufficent evidence to reject the Null Hypothesis

One-sample t-test for hardware engineer: t_critical: nan p_value: nan Result: There is
 not enough evidence to reject the Null Hypothesis

One-sample t-test for associate engineer: t_critical: nan p_value: nan Result: There is
 not enough evidence to reject the Null Hypothesis
```

In [ ]:

```
1
```

Observation:

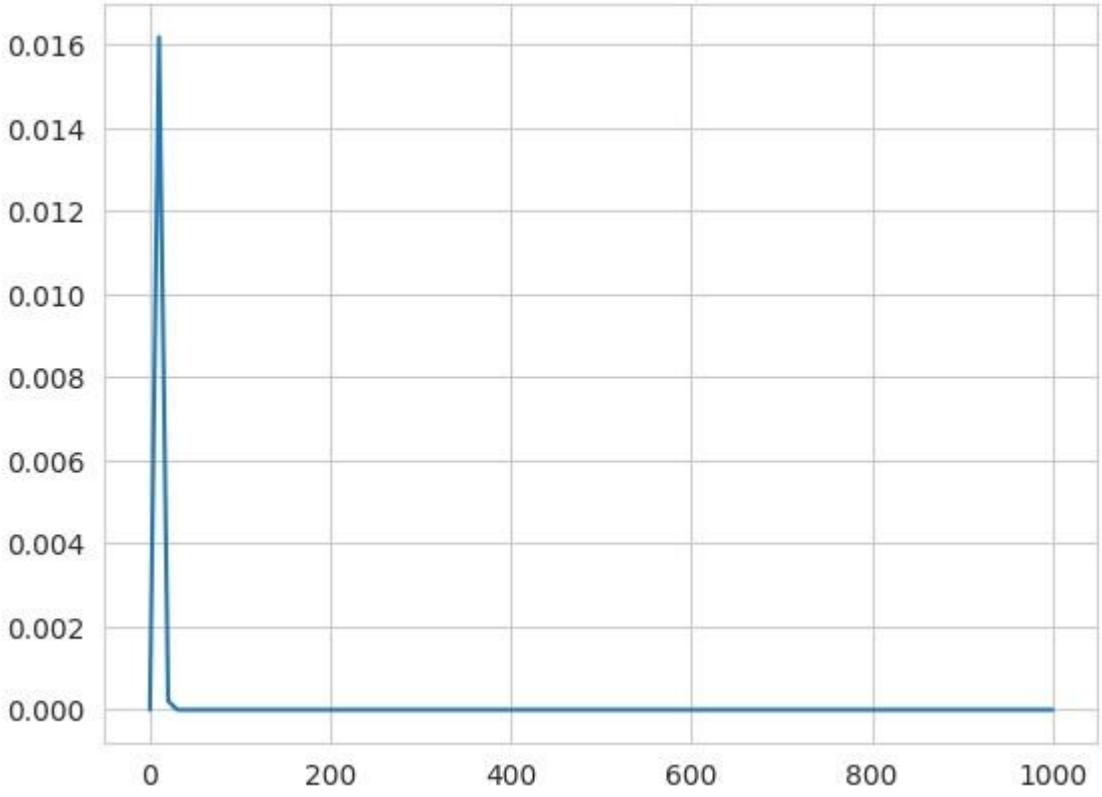
INNOMATICS

In [ ]:

```
from scipy . stats import chi2
from scipy . stats import chi2_contingency

```

```
x = np. linspace ( 0, 1000, 100)
y = chi2 . pdf ( x, df = 4)
plt . plot ( x, y)

```

Out[185]: [< matplotlib.lines.Line2D at 0x7bd4cf7a9ab 0>]




```
observation = pd. crosstab ( df . Specialization , df . Gender )
print ( observation )

```

5.4.4.1 2. Is there a relationship between gender and specialization? (i.e. Does the preference of Specialisation depend on the Gender?)

In [ ]:

INNOMATICS

In [ ]:

In [ ]:

```
Gender                                          f     m Specialization
computer application                           50  141 computer engineering
140  320 computer science & engineering        144  443 electrical
engineering                    15   45 electronics & telecommunications
22    69 electronics and communication engineering   167   550
electronics and electrical engineering         26   122 information
technology                127  377 mechanical engineering                6
143 other                                      47   148 1 chi_2_statistic,
chi_2_p_value, chi_2_dof, chi_2_expected = chi2_contingen
```

```
print("Statistic         :\t", chi_2_statistic, "\n")

print("p-value           :\t", chi_2_p_value, "\n")

print("Degrees of freedom  :\t", chi_2_dof, "\n")

print("Expected frequencies array:\n", chi_2_expected)
```

```
Statistic         : 47.625462024510526 p-value           :
3.000409236234154e-07

   Degrees of freedom  :          9

Expected frequencies array:
 [[ 45.81044487 145.18955513]
 [110.32882012 349.67117988]
 [140.78916828 446.21083172]
 [ 14.39071567  45.60928433]
 [ 21.82591876  69.17408124]
 [171.96905222 545.03094778]
 [ 35.49709865 112.50290135]
 [120.88201161 383.11798839]
 [ 35.73694391 113.26305609]
 [ 46.76982592 148.23017408]]
```

Calculating chi-square critical value

INNOMATICS

In [ ]:

```
confidence_level = 0.95
alpha = 1 - confidence_level
chi_2_critical = chi2 . ppf ( 1 - alpha , chi2_dof )
chi_2_critical
```

Out[207]: 16.918977604620448

In [ ]:

In [ ]:

```
if(chi_2_statistic > chi_2_critical):
print("There is not enough evidence to reject the Null Hypothesis") else:
print("There is sufficent evidence to reject the Null Hypothesis")
```

There is not enough evidence to reject the Null Hypothesis

Observation:

- **Salary:** With a mean pay of roughly 314, 764.30 INR, the data shows that engineering graduates offer a wide range of salary options. The distribution is positively skewed, which implies that there is a concentration of higher income values. This suggests that there may be differences in the salary offers made to engineers.

- **Designation:** Among engineering graduates, software engineer, system engineer, and software developer are the most popular titles. This distribution illustrates areas of specialization and the need for particular skill sets, reflecting the common work roles in the engineering business.
- **JobCity:** Hyderabad, Noida, and Bangalore are the top three cities in which engineering graduates find employment. The geographic concentration of employment prospects in technology hubs is highlighted by this distribution, highlighting the significance of location for engineering professionals when making career selections.
- **Gender:** The dataset exhibits an imbalance in gender distribution, with a higher representation of males compared to females among engineering graduates. This finding underscores the gender gap prevalent in the engineering workforce and highlights the need for initiatives to promote gender diversity and inclusion in the field.
- **Specialization:** Engineering specializations such as Electronics and Communication Engineering, Computer Science & Engineering, and Information Technology are prominent among graduates. This distribution reflects the prevalence of these disciplines in the engineering education landscape and their relevance in the industry.
- **CollegeTier:** The majority of engineering graduates in the dataset attended Tier 2 colleges, indicating a distribution skewed towards colleges with moderate rankings. This observation suggests a diverse educational background among graduates, with implications for recruiting strategies and industry-academia collaborations.
- **Degree:** The dataset primarily comprises individuals with B.Tech/B.E. degrees, highlighting the dominance of undergraduate engineering programs among graduates. This distribution underscores the importance of undergraduate education in shaping career pathways for engineering professionals.
- **English, Logical, Quant Scores:** Standardized scores in English, Logical, and Quantitative sections of the AMCAT test exhibit varying distributions, reflecting differences in cognitive abilities among graduates. These scores serve as indicators of analytical and problem-solving skills, which are critical for success in technical roles.
- **Personality Traits:** Standardized scores for personality traits such as conscientiousness, agreeableness, extraversion, neuroticism, and openness to experience vary among graduates. These traits play a significant role in shaping individual behaviors, attitudes, and interpersonal dynamics in the workplace.

Overall, the analysis of each variable provides valuable insights into the characteristics of engineering graduates facilitating a deeper understanding of the factors influencing career

5.5 Additional Research Question

In a comparative study of recruitment practices among leading companies, does AMEO's hiring policy of recruiting candidates with a minimum percentage of 70% and maintaining an average percentage of 80% hold true?

In [69]:
```
average_percentage = df['12percentage'].mean()
average_percentage
```

Out[69]: 74.69438426821405

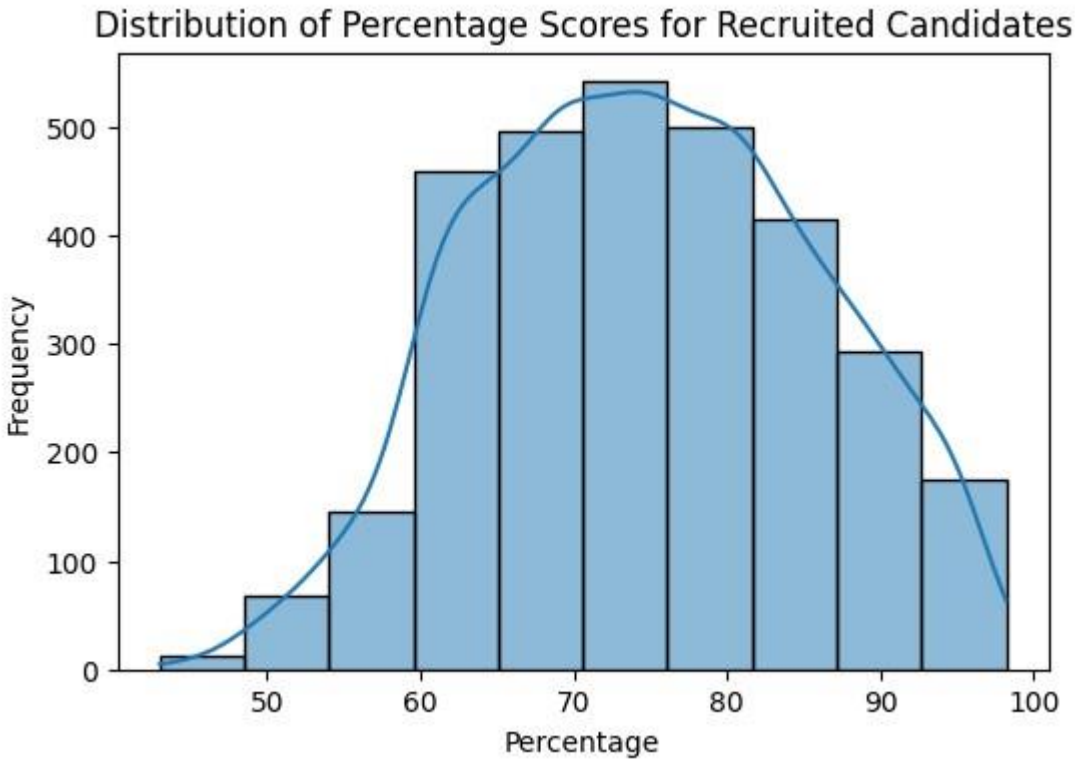
In [70]:
```
Difference = 80 - average_percentage
Difference
```

Out[70]: 5.305615731785949

Since the difference is positive (5.3), it indicates that the average percentage obtained (74.7%) is below the stated goal of 80%.

In [73]:

```
plt . figure ( figsize =( 6, 4))
sns . histplot ( df [ '12percentage' ] , bins =10, kde=True )
plt . title ( 'Distribution of Percentage Scores for Recruited Candidates' )
plt . xlabel ( 'Percentage' )
plt . ylabel ( 'Frequency' )
plt . show()

```



In [74]:

```
if average_percentage < 80:
    print ( "Recommendation: AMEO should consider revising its minimum perce
else :
    print ( "No specific recommendation." )

```

Recommendation: AMEO should consider revising its minimum percentage requireme nt or implementing additional screening processes to improve the quality of re cruited candidates.

Observation:

5.5.1 To raise the caliber of applicants it recruits, MEO ought to think about changing its