What is the summary statistics of numerical variables?

Summary statistics of numerical variables provide a quick overview of the main characteristics of the data. These statistics include measures of central tendency, dispersion, and shape. Here are the most commonly used summary statistics:

Measures of Central Tendency

1.Mean: The average value of the data.

Mean = $\sum Xi / N$

where Xi are the individual data points and N is the number of data points.

2.Median: The middle value when the data is sorted in ascending order. If the number of data points is even, the median is the average of the two middle numbers.

3.Mode: The value that appears most frequently in the data set.

Measures of Dispersion

Range: The difference between the maximum and minimum values.

$$\text{Range} = \text{Max} - \text{Min}$$

Variance: The average of the squared differences from the mean.

$$\text{Variance}(\sigma^2) = \frac{\sum(X_i - \mu)^2}{N}$$

Standard Deviation: The square root of the variance, indicating how spread out the values are.

$$\text{Standard Deviation}(\sigma) = \sqrt{\text{Variance}} = \sqrt{\frac{\sum(X_i - \mu)^2}{N}}$$

Interquartile Range (IQR): The difference between the 75th percentile (Q3) and the 25th percentile (Q1).

$$\text{IQR}=\text{Q3}-\text{Q1}$$

Measures of Shape

Skewness: A measure of the asymmetry of the data distribution.

Positive skewness indicates a distribution with a longer tail on the right.

Negative skewness indicates a distribution with a longer tail on the left.

Kurtosis: A measure of the "tailedness" of the data distribution.

High kurtosis indicates heavy tails and a sharp peak.

Low kurtosis indicates light tails and a flatter peak.

Example Using Python

Here's how you can calculate these summary statistics using Python's pandas library:

```
import pandas as pd
# Sample data
data = {
   'empid': ['0001', '0002', '0003', '0004', '0005'],
   'salary': [75000, 65000, 80000, 70000, 72000],
   'age': [28, 32, 35, 30, 40]
}

# Create a DataFrame
df = pd.DataFrame(data)

# Summary statistics for salary
summary_stats = df['salary'].describe()
print(summary_stats)

# Additional statistics
mean_salary = df['salary'].mean()
median_salary = df['salary'].median()
mode_salary = df['salary'].mode()[0]  # mode returns a Series
range_salary = df['salary'].max() - df['salary'].min()
variance_salary = df['salary'].var()
std_dev_salary = df['salary'].std()
iqr_salary = df['salary'].quantile(0.75) - df['salary'].quantile(0.25)
skewness_salary = df['salary'].skew()
```

```
kurtosis_salary = df['salary'].kurt()

print(f"Mean: {mean_salary}")
print(f"Median: {median_salary}")
print(f"Mode: {mode_salary}")
print(f"Range: {range_salary}")
print(f"Variance: {variance_salary}")
print(f"Standard Deviation: {std_dev_salary}")
print(f"IQR: {iqr_salary}")
print(f"Skewness: {skewness_salary}")
print(f"Kurtosis: {kurtosis_salary}")
```

Output

The describe() function in pandas will give you a quick summary of the main statistics:

```
count      5.000000
mean    72400.000000
std      5667.524792
min     65000.000000
25%     70000.000000
50%     72000.000000
75%     75000.000000
max     80000.000000
Name: salary, dtype: float64
```

2.What is the distribution of each numerical variables?

To analyze the distribution of each numerical variable in your dataset, we will examine various descriptive statistics and visualizations that provide insights into the data. Here's a detailed approach using Python:

Step-by-Step Guide

Install Necessary Libraries:

If you haven't already, install pandas and seaborn.

```
pip install pandas seaborn matplotlib
```

2.Load the Data:

Create a sample dataset or load your dataset.

```
import pandas as pd
data = {
    'empid': ['0001', '0002', '0003', '0004', '0005'],
    'salary': [75000, 65000, 80000, 70000, 72000],
    'age': [28, 32, 35, 30, 40]
}
df = pd.DataFrame(data)
```

3.Summary Statistics:

Get a quick overview of the summary statistics for each numerical variable.

```
summary_stats = df.describe()
print(summary_stats)
```

This will output something like:

```
       salary       age
count     5.000000   5.000000
mean   72400.000000  33.000000
std     5667.524792   4.949747
min    65000.000000  28.000000
25%    70000.000000  30.000000
50%    72000.000000  32.000000
75%    75000.000000  35.000000
max    80000.000000  40.000000
```

4.Visualize the Distributions:

Use seaborn to visualize the distribution of each numerical variable.

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

# Visualize the distribution of salary
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
sns.histplot(df['salary'], kde=True)
plt.title('Distribution of Salary')

plt.subplot(1, 2, 2)
sns.boxplot(y=df['salary'])
plt.title('Boxplot of Salary')

plt.tight_layout()
plt.show()

# Visualize the distribution of age
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
sns.histplot(df['age'], kde=True)
plt.title('Distribution of Age')

plt.subplot(1, 2, 2)
sns.boxplot(y=df['age'])
plt.title('Boxplot of Age')

plt.tight_layout()
plt.show()
```

Explanation of Visualizations

1.Histogram and KDE:

Shape: Observe the shape of the distribution. It can be normal (bell-shaped), skewed (left or right), or have multiple peaks (bimodal).

Central Tendency: The peak of the KDE plot indicates where the central tendency (mean or median) lies.

Spread: The width of the histogram bins and the KDE plot show the spread of the data.

2.Boxplot:

Central Tendency: The line inside the box represents the median.

Interquartile Range (IQR): The length of the box shows the IQR, which is the range between Q1 and Q3.

Whiskers: The lines extending from the box show the range within 1.5 * IQR from the quartiles.

Outliers: Points outside the whiskers are considered outliers.

Interpretation of the Output

1.Summary Statistics:

Count: Number of observations.

Mean: Average value.

Standard Deviation (std): Measure of spread or dispersion.

Min: Minimum value.

25%: First quartile (Q1), 25th percentile.

50%: Median (second quartile), 50th percentile.

75%: Third quartile (Q3), 75th percentile.

Max: Maximum value.

2.Histogram and KDE:

Salary: The histogram and KDE plot for salary show how the salary values are distributed. Peaks in the KDE plot indicate common salary values.

Age: The histogram and KDE plot for age show the distribution of ages in the dataset.

3.Boxplot:

Salary: The boxplot highlights the median salary, the range of the middle 50% of salaries (IQR), and any outliers.

Age: The boxplot provides a visual summary of the age data, including median, quartiles, and outliers.

3.What are the unique values and their counts for categorical variables?

To find the unique values and their counts for categorical variables in a dataset, you can use Python's pandas library. This process involves identifying which columns in your DataFrame are categorical and then using the value_counts() method to get the counts of unique values in those columns.

Step-by-Step Guide

1.Install pandas:

If you haven’t already, you can install pandas using pip:

```
pip install pandas
```

2.Load the Data:

Create a sample dataset or load your dataset.

```
import pandas as pd


data = {
    'empid': ['0001', '0002', '0003', '0004', '0005'],
    'name': ['John Doe', 'Jane Smith', 'Alice Brown', 'Bob White', 'Charlie Black'],
    'department': ['Data Science', 'Marketing', 'Data Science', 'Sales', 'Marketing']
}


df = pd.DataFrame(data)
```

3.Identify Categorical Variables:

Categorical variables are typically non-numeric columns like name and department.

```
categorical_columns = df.select_dtypes(include=['object']).columns
print("Categorical Columns:", categorical_columns)
```

4.Get Unique Values and Their Counts:

Use the value_counts() method to find the unique values and their counts for each categorical variable.

```
for col in categorical_columns:
    print(f"\nUnique values and counts for column '{col}':")
    print(df[col].value_counts())
```

Example Output

When you run the above code, you will get the unique values and their counts for each categorical variable:

```
Categorical Columns: Index(['empid', 'name', 'department'], dtype='object')


Unique values and counts for column 'empid':
0001    1
0002    1
0003    1
0004    1
0005    1
Name: empid, dtype: int64


Unique values and counts for column 'name':
John Doe        1
Jane Smith      1
Alice Brown     1
Bob White       1
Charlie Black   1
Name: name, dtype: int64


Unique values and counts for column 'department':
Data Science    2
Marketing       2
Sales           1
Name: department, dtype: int64
```

Explanation

empid: Each employee ID is unique, so each value appears once.

name: Each name is unique in this small dataset, so each value appears once.

department: This column has three unique values with the following counts:

Data Science appears 2 times.

Marketing appears 2 times.

Sales appears 1 time.

Using the Result for Analysis

By obtaining the unique values and their counts, you can:

Understand the distribution of categories in your dataset.

Detect any potential issues with data entry (e.g., misspelled category names).

Gain insights into the frequency of different categories, which can be useful for further analysis or reporting.

This process is essential for initial data exploration and helps in understanding the structure and content of your categorical data.