Summarizing a Single Feature

Pandas .describe() method

The pandas method, `.describe()` provides summary statistics for all features in a dataset. Setting `include = 'all'` includes summary statistics for both quantitative and categorical features.

```
df.describe(include = 'all')
```

Central tendency statistics

To summarize the central tendency, or typical value, of a quantitative variable, we can use statistics such as the mean, median, and mode. These can be calculated using the pandas methods `.mean()`, `.median()`, and `.mode()`, respectively.

```
#calculate mean of a column
df.column_name.mean()

#calculate median of a column
df.column_name.median()

#calculate mode of a column
df.column_name.mode()
```

Spread statistics

To summarize the spread, or variation, of a quantitative variable, we can use statistics such as the range, interquartile range, variance, standard deviation, and mean absolute deviation. These can be calculated as shown.

```
#calculate range of a column
df.column_name.max() - df.column_name.min()

#calculate IQR of a column
df.column_name.quantile(0.75) - df.column_name

#calculate variance of a column
df.column_name.var()

#calculate standard deviation of a column
df.column_name.std()

#calculate MAD of a column
df.column_name.mad()
```

Visualize the distribution of a quantitative/continuous feature

To inspect the distribution of a quantitative variable, we can use visualizations such as histograms and box plots. We can create these plots using the seaborn functions `histplot()` and `boxplot()`, respectively.

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

#create histogram
sns.histplot(x = 'column_name', data = data_na
plt.show()

#create boxplot
```

```
sns.boxplot(x = 'column_name', data = data_nam
plt.show()
```

Summary statistics for categorical data

To summarize the distribution of a categorical/discrete feature, we can calculate the number or proportion of observations in each category using the pandas method `.value_counts`.

```
#calculate the number in each category
df.column_name.value_counts()

#calculate the proportion in each category
df.column_name.value_counts(normalize = True)
```

Visualizing categorical data

To inspect and explore categorical features, we can use visualizations such as bar charts or pie charts. The provided code demonstrates how to create these plots.

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

#create bar chart
sns.countplot(x = 'column_name', data = data_n
plt.show()

#create pie chart
df.column_name.value_counts().plot.pie()
plt.show()
```

Next →

Learn More on Codecademy