总结

Descriptive Statistics

Here is a quick review of some popular functions:

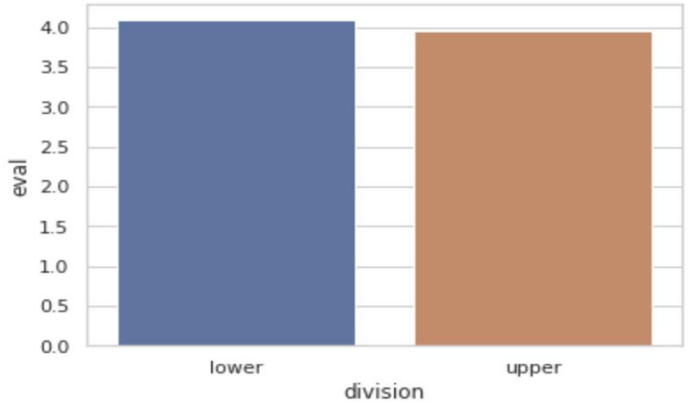
- To find the average of the data, we use the `mean()` function
- To find the median of the data, we use the `median()` function
- To find the mode of the data, we use the `mode()` function
- To find the variance of the data, we use the `variance()` function
- To find the standard deviation of the data, we use the `stdev()` function
- To get the unique values in a dataset, we use the `unique()`. `unique()` prints out the values and `nunique()` prints out the number of unique values.

Data Visualization

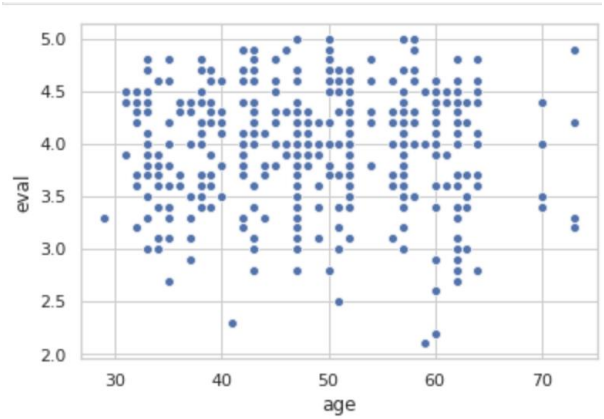
One of the most popular visualization tools is the `seaborn` library. It is a Python Data visualization library that is based on `matplotlib`. You can learn more here. To get access to functions in the `seaborn` library or any library, you must first import the library. To import the `seaborn` library: `import seaborn`.

Here is a quick summary for creating graphs and plots:

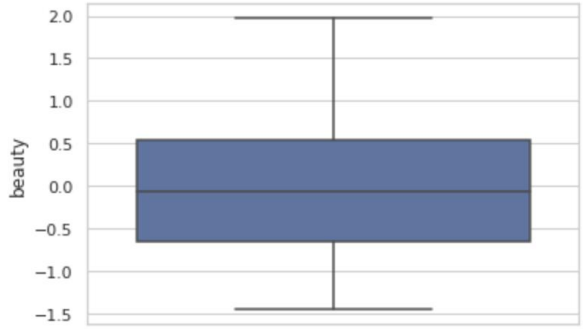
- Barplots: A barplot shows the relationship between a numeric and a categorical variable by plotting the categorical variables as bars in correspondence to the numerical variable. In the seaborn library, barplots are created by using the `barplot()` function. The following code `ax = seaborn.barplot(x="division", y="eval", data=division_eval)` will return a barplot that shows the average evaluation scores for the lower-division and upper-division.



- Scatterplots: This is a two-dimensional plot that displays the relationship between two continuous data. Scatter plots are created by using the `scatterplot()` function in the seaborn library. The following code: `ax = seaborn.scatterplot(x='age', y='eval', hue='gender', data=ratings_df)` will return a plot that shows the relationship between age and evaluation scores:



- Boxplots: A boxplot is a way of displaying the distribution of the data. It returns the minimum, first quartile, median, third quartile, and maximum values of the data. We use the `boxplot()` function in the seaborn library. This code `ax = seaborn.boxplot(y='beauty', data=ratings_df)` will return a boxplot with the data distribution for beauty. We can make the boxplots horizontal by specifying `x='beauty'` in the argument.



Other useful functions include `catplot()` to represent the relationship between a numerical value and one or more categorical variables, `distplot()`, and `histplot` for plotting histograms.

Hypothesis Testing

- Use the `norm.cdf()` function in the `scipy.stats` library to find the standardized (z-score) value. In cases where we are looking for the area to the right of the curve, we will remove the results above from 1. Remember to `import scipy.stats`
- Levene's test for equal variance: Levene's test is a test used to check for equality of variance among groups. We use the `scipy.stats.levene()` from the `scipy.stats` library.
- T-test for two independent samples: This test compares the means of two independent groups to determine whether there is a significant difference in means for both groups. We use the `scipy.stats.ttest_ind()` from the `scipy.stats` library.
- One-way ANOVA: It compares the mean between two or more independent groups to determine whether there is a statistical significance between them. We use the `scipy.stats.f_oneway()` from the `scipy.stats` library or you can use the `anova_lm()` from the `statsmodels` library.
- Chi-square ($\chi 2$) test for association: Chi-square test for association tests the association between two categorical variables. To do this we must first create a crosstab of the counts in each group. We do this by using the `crosstab()` function in the `pandas` library. Then the `scipy.stats.chi2_contingency()` on the contingency table - it returns the $\chi 2$ value, p-value, degree of freedom and expected values.
- Pearson Correlation: Tests the correlation between two continuous variables. we use the `scipy.stats.pearsonr()` to get the correlation coefficient
- To run the tests using Regression analysis, you will need the `OLS()` from the `statsmodels` library. When running these tests using regression analysis, you have `fit()` the model, make predictions using `predict()` and print out the model summary using `model.summary()`