Murach's Python for Data Analysis

Student projects

These projects let you practice the skills that you learn as you progress through *Murach's Python for Data Analysis*. These projects provide a range of difficulty levels, and your instructor will assign selected projects as you progress through this course.

In the project names, the first number specifies the chapter that you should complete before starting the exercise. For example, you should complete chapter 2 before starting project 2-1, 2-2, or 2-3, and you should complete chapter 8 before starting project 8-1 or 8-2.

General guidelines

Data files supplied by your instructor

- Each project gets its data from a file such as a CSV file. These files are identified in the specifications for each project, and your instructor should make these starting files available to you.

Notebooks created by you

- For each project, create one Jupyter Notebook.
- When creating the filenames for your Notebooks, please use the convention specified by your instructor. Otherwise, name the Notebook *first_last_proj_X-X* where *first_last* specifies your first and last names and *X-X* specifies the project number.
- Start each Notebook by importing any modules, classes, or functions required by the project.
- Use one Code cell to store the code that accomplishes each task and question.
- Use comments to describe your thought process and to document any code that's hard to understand.
- Use Markdown cells to create headings that organize your Notebooks.

Dataset summary

Here's a brief summary of each dataset used by the projects.

Exams

- This dataset contains math, reading and writing scores based on students' gender, race, level of parent education, whether they receive a free or reduced-cost lunch, and whether they completed a test preparation course.

Ramen

- This dataset rates various types of ramen based on the brand, variety, style, and country that sells it.

Retail

- This dataset contains sales information, including the customer ID, SKU information that identifies the product, quantity, and the sales amount. It also includes two columns, Unnamed: 0 and Transaction_ID, that contain sequential numbers that are irrelevant to analyzing this data.

Avocados

- This dataset contains historical data on avocado prices and sales volume in multiple U.S. markets. One of the columns in this dataset, Unnamed: 0, contains sequential numbers that are irrelevant to analyzing this data. Three of the other columns contain sales for PLU (price look-up) codes 4046, 4225, and 4770. These columns will also not be used by the analyses done in these projects.
- If you review the data, you'll see that some of the regions overlap. For example, one of the regions is the entire U.S., and all of the other regions are parts of the U.S. Because of that, you would need to review this data carefully before determining the best way to analyze it. For the purposes of these exercises, though, the overlapping regions won't be taken into consideration.

Gold

- This dataset contains information about gold prices by date.

Diamonds

- This dataset contains information about diamond prices based on carat weight, cut quality, color, clarity, depth percent, and table percent. This dataset also includes x, y, and z columns that represent the diamond's length, width, and depth. These columns won't be used by the analyses done in these projects.

MPG

- This dataset contains information about automobile MPG based on the number of cylinders, the displacement, the horsepower, the weight, the acceleration, the model year, the origin, and the car.

Project 2-1: Analyze the ramen data

The data file

`ramen-ratings.csv`

Tasks

1. Read the data from the CSV file into a DataFrame.
2. Display the first five rows of data.
3. Display the last five rows of data.
4. Display statistical information for the numeric columns using the describe() method.
5. Display the number of unique values for each column.
6. Display only rows where the country is Vietnam.
7. Display only the Brand and Style columns.
8. Display only the Country column.
9. Display the data after it has been sorted by the Stars column from high values to low values.
10. In the Country column, replace "USA" with "United States". Make sure this change is saved in the DataFrame, and then display the first five rows to be sure the change was made correctly.

Questions

1. How many countries are represented in the data?
2. Which three countries have the highest average rating?
3. Which three countries have the lowest average rating?
4. Which three countries have the most brands, and how many brands does each of these countries have?

Project 2-2: Analyze the avocado data

The data file

`avocado.csv`

Tasks

1. Read the data from the CSV file into a DataFrame.
2. Display type, memory consumption, and null count information using the info() method.
3. Display the number of unique values in each column.
4. Display all the rows of data that JupyterLab displays by default.
5. Display the first and last five rows of data and the first and last four columns of data.
6. Choose any three columns, access them with bracket notation, and display the first five rows of this data.
7. Select one column and access it with dot notation.
8. Multiply the Total Volume and AveragePrice columns, and store the result in a new column called EstimatedRevenue. Then, display the first five rows of this data to confirm that the column was added and has the correct values.
9. Create a DataFrame that's grouped by region and type and that includes the average price for the grouped columns. Then, reset the index and display the first five rows.
10. Create a bar plot that shows the mean, median, and standard deviation of the Total Volume column by year.

Questions

1. How many unique regions are there?
2. What is the average price for each type of avocado (organic and conventional)? Be sure to include just the type and AveragePrice columns in the results.
3. Which region has the lowest average price for organic avocados? Hint: Create wide data from the grouped data that you created in task 8.
4. Have the Total Bags sold per year of each type of avocado become more or less consistent over time?

Project 2-3: Analyze the exam data

The data file

`exams.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Display the basic information for the DataFrame and its columns using the info() method.
3. Display statistical information for the math score, reading score, and writing score columns using the describe() method.
4. Group the data by the race/ethnicity column and display the mean scores.
5. Display a single column as a DataFrame with bracket notation.
6. Display a single column as a Series with bracket notation.
7. Display a single column as a Series with dot notation.
8. Display only rows for females with a math score greater than or equal to 90.

Questions

1. Does taking a test preparation course improve average scores?
2. Which gender is better on average at math?
3. Which gender is better on average at all three subjects? Hint: Start by adding a column to the DataFrame with the total score.
4. Does the parents' level of education have an effect on the average scores?

Project 3-1: Plot the ramen data with Pandas

The data file

```
ramen-ratings.csv
```

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Create a histogram for the Stars column using the number of bins necessary to display the ratings for each quarter of a star.
3. Create a KDE plot for the Stars column.
4. Create a bar plot that shows the average number of stars by ramen style. Hint: You'll need to group the data before you plot it.
5. Create a bar plot that shows the total number of reviews for each ramen type.
6. Create a bar plot that shows the highest score given to each ramen type.

Question

1. What is the most common style of ramen?

Project 3-2: Plot the avocado data with Pandas

The data file

`avocado.csv`

Tasks

1. Read the data from the .csv file into a DataFrame and display the first five rows.
2. Create a new DataFrame that contains the total of the Small Bags, Large Bags, and XLarge Bags columns grouped by type, and then display the DataFrame.
3. Use the grouped data to create a bar plot that shows the number of small, large, and extra-large bags for both types of avocado.
4. Use the original data to create a scatter plot for the Total Volume and AveragePrice columns.

Questions

1. In what year were the most bags of avocados sold?
2. Do the total bags of avocados appear to have an effect on the average price?

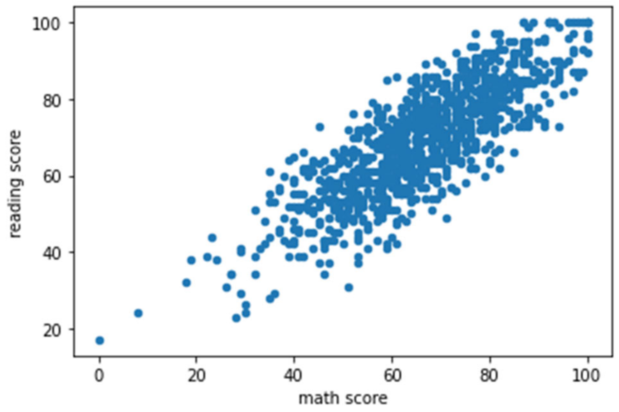
Project 3-3: Plot the exam data with Pandas

The data file

`exams.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Use the pandas plot() method to create a box plot.
3. Use the pandas plot() method to create a plot like the one that follows.



4. Group the data by the gender column and calculate the average scores. Then create a bar plot.

Questions

1. What is the relationship between the reading score and the writing score?
2. Does the type of lunch appear to have an effect on the average test scores?

Project 4-1: Plot the ramen data with Seaborn

The data file

`ramen-ratings.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Create a bar plot to show the Stars by Style with a Seaborn specific method.
3. Create a bar plot to show the Stars by Style with a Seaborn generic method.
4. Modify the plot you just created to use a custom title and x-axis label.
5. Use the query() method to filter the data to contain only data for Japan, India, Taiwan, and the U.S. Plot the stars by style for each country on a subplot with two subplots per row.
6. Modify the plot you just created to have a custom title for each subplot. Hint: Use the enumerate() method.

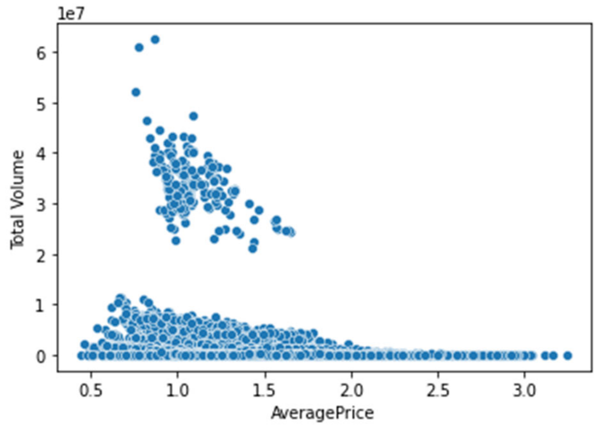
Project 4-2: Plot the avocado data with Seaborn

The data file

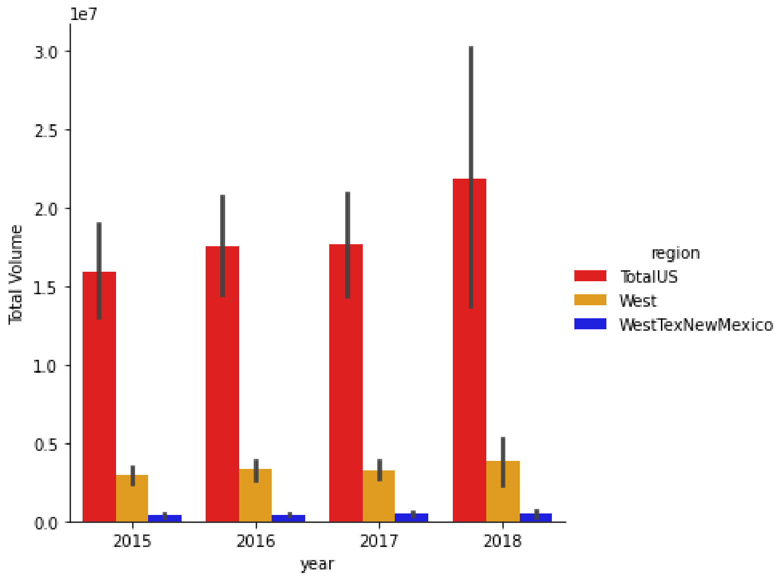
`avocado.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Create the following plot with a Seaborn specific method.



3. Create the same plot with a generic method.
4. Create the same plot but with the hue parameter set to year and the dots for the total volume ranging in size from 10 to 100.
5. Create a plot that looks like the one that follows.



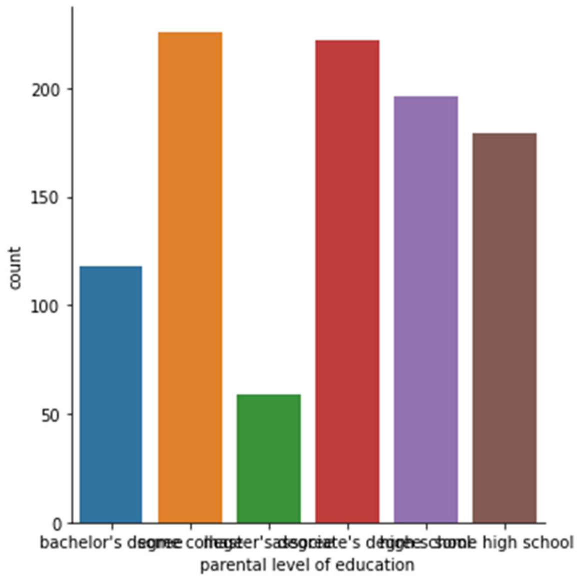
Project 4-3: Plot the exam data with Seaborn

The data file

`exams.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Use the Seaborn catplot() method to create a plot like the one below. To make that easy to do, you can use a count plot.

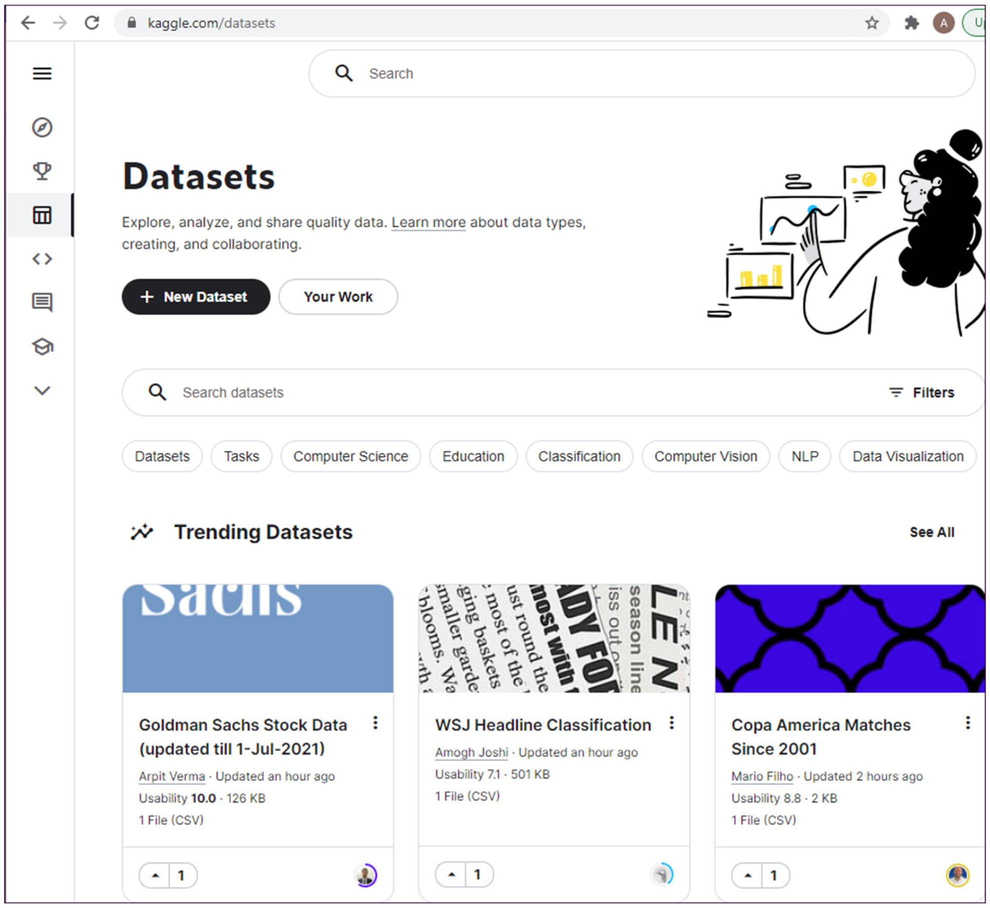


3. Rotate the x labels for the above plot to make them readable.
4. Create a scatter plot with a specific method that compares the writing score with the reading score. Use a different color to show which students took a test prep course.
5. Create the same plot with a general method.
6. Adjust the size of the scatter plot you created with the general method so it's more like the size of the scatter plot with the specific method.

Project 5-1: Get different types of data

Tasks

1. Download the CSV file at this link and read the data into a DataFrame:
   https://raw.githubusercontent.com/fivethirtyeight/data/master/thanksgiving-2015/thanksgiving-2015-poll-data.csv
2. Download the Stata file at this link and read the data into a DataFrame:
   http://www.stata-press.com/data/r8/cancer.dta
3. Download the CSV file at this link and load the data into a DataFrame. You'll need to create a Kaggle account if you do not already have one.
   https://www.kaggle.com/kaggle/sf-salaries
4. Search Kaggle for something that you are interested in to see if there is any data for that interest. If there is, download that dataset and load it into a DataFrame. If not, just download one of the trending datasets in the trending section as shown below:



Datasets

Explore, analyze, and share quality data. Learn more about data types, creating, and collaborating.

Trending Datasets

Goldman Sachs Stock Data (updated till 1-Jul-2021)
Arpit Verma · Updated an hour ago
Usability 10.0 · 126 KB
1 File (CSV)

WSJ Headline Classification
Amogh Joshi · Updated an hour ago
Usability 7.1 · 501 KB
1 File (CSV)

Copa America Matches Since 2001
Mario Filho · Updated 2 hours ago
Usability 8.8 · 2 KB
1 File (CSV)

Project 6-1: Clean the ramen data

The data file

```
ramen-ratings.csv
```

Tasks

1. Read the data from the CSV file into a DataFrame and display the data to get a count of the number of rows.
2. Rename the Stars column to Rating.
3. Convert the Style column to the category type.
4. Drop the country column.
5. Check that the changes were applied to the DataFrame.
6. Rename the Brand column to Company and the Variety column to Product.
7. Use the duplicated() method to display all rows with duplicate Company/Product combinations.
8. Drop all rows with duplicate Company/Product combinations except for the first row in each set. Then, display the data to get a count of the number of rows, and compare that with the original number of rows to see how many were dropped.
9. Drop any rows that have missing values.
10. Check again to make sure that all of your changes were applied.

Project 6-2: Clean the avocado data

The data file

`avocado.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Check to see if any of the columns have missing values.
3. Display the unique values in the region column, and notice that one of the regions is TotalUS.
4. Drop rows that have a region of TotalUS, and then check to see how many rows were dropped.
5. Drop the Unnamed: 0, 4046, 4225, and 4770 columns.
6. Convert the Date column to the DateTime type.
7. Check that these changes have been applied to the DataFrame.

Project 7-1: Prepare the ramen data

The data file

`ramen-ratings.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Add a column to the DataFrame that stores the average rating for each country. Then, display the first five rows and verify that the column was added and that its values appear to be correct.
3. Group the data by the Country and Style columns and take the mean. Then, unstack the Style column and display the data for the Stars column.
4. Write a function that takes a row as input and returns a shortened version of the values in the Variety column. The values should consist of the first two words of the variety name, followed by a space and an ellipsis (…). Hint: You'll need to split the values into a list of words, get the first two words in the list, add the ellipsis to the list, and then join the words in the list.
5. Apply the function you just wrote to the DataFrame, and store the results in a column named ShortVariety. Then, display the first five rows to be sure this worked.
6. Set an index on the Brand column and then display the first five rows of data.
7. Reset the index to its default.

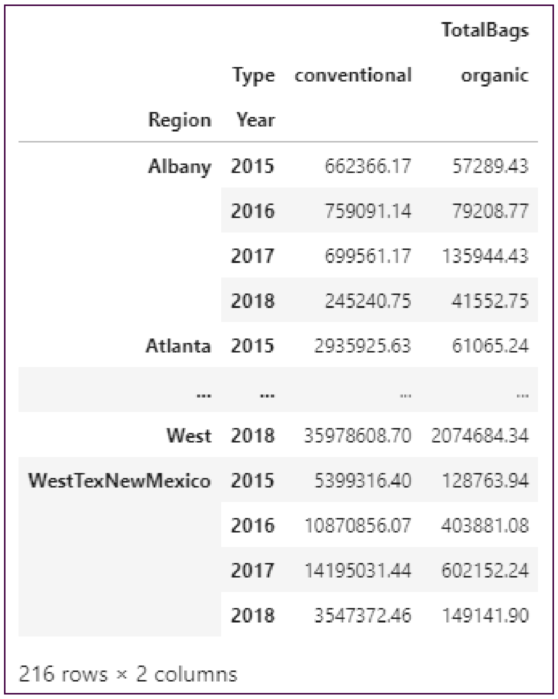
Project 7-2: Prepare the avocado data

The data file

`avocado.csv`

Tasks

1. Read the data from the .csv file into a DataFrame and display the first five rows.
2. Notice that the column names are inconsistent. Change the column names so they all use title case with no spaces.
3. Add a column with the percentage of the bags in the TotalBags column that are extra-large bags.
4. Add a column with the percentage of the bags in the TotalBags column that are large bags.
5. Add a column with the percentage of the bags in the TotalBags column that are small bags.
6. Display the first five rows of data to view the new columns.
7. Assign the Region, Type, Year, and TotalBags columns to a new DataFrame.
8. Group, sum, and unstack this data to produce a DataFrame that looks like this:



		TotalBags	
	Type	conventional	organic
Region	Year		
Albany	2015	662366.17	57289.43
	2016	759091.14	79208.77
	2017	699561.17	135944.43
	2018	245240.75	41552.75
Atlanta	2015	2935925.63	61065.24
...	...	...	...
West	2018	35978608.70	2074684.34
WestTexNewMexico	2015	5399316.40	128763.94
	2016	10870856.07	403881.08
	2017	14195031.44	602152.24
	2018	3547372.46	149141.90

216 rows × 2 columns

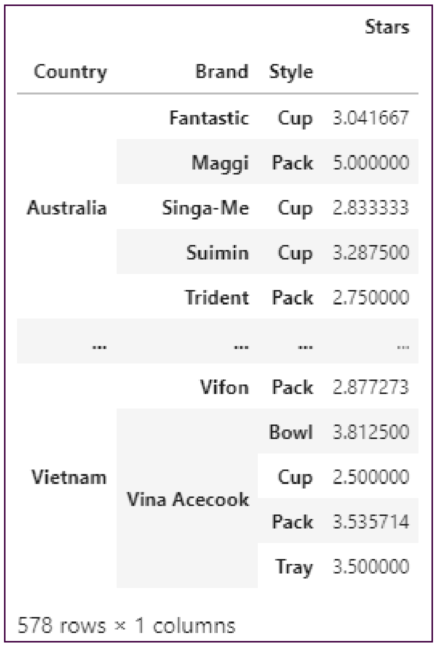
Project 8-1: Analyze the ramen data

The data file

`ramen-ratings.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Group the data and get the average of the groups to produce a DataFrame that looks like this:



			Stars
Country	**Brand**	**Style**	
Australia	**Fantastic**	**Cup**	3.041667
	Maggi	**Pack**	5.000000
	Singa-Me	**Cup**	2.833333
	Suimin	**Cup**	3.287500
	Trident	**Pack**	2.750000
...	**...**	**...**	...
Vietnam	**Vifon**	**Pack**	2.877273
	Vina Acecook	**Bowl**	3.812500
		Cup	2.500000
		Pack	3.535714
		Tray	3.500000

578 rows × 1 columns

3. Group the data in the same way, but this time get both the average and the count.
4. Create a pivot table that shows the average stars by ramen style for each country.
5. Calculate the total number of stars for each country, and then rank the countries in descending order by total stars.
6. Bin the stars data into 11 bins. Label the bins from 0.0 to 5.0 in increments of .5, and then display the count of the stars in each bin.
7. Bin the stars data into five quantiles. Give each bin a descriptive label, and then display the count of the stars in each bin.

Questions

1. Which five countries have the most stars overall? Hint: Use the grouped data that you created in task 5.
2. What is the only country that produces canned ramen?

Project 8-2: Analyze the avocado data

The data file

`avocado.csv`

Tasks

1. Read the data from the CSV file into a DataFrame and display the first five rows.
2. Display basic information about the DataFrame and its columns to see that the Date column has the object type.
3. Convert the Date column to the datetime type.
4. Filter the data so it contains just the rows for 2015, for the conventional type, and for the region named Albany. Store just the Date, Total Bags, and Small Bags columns in a new DataFrame.
5. Melt the data in the Total Bags and Small Bags columns, but not the values in the Date column. Name the column that contains the type of bag Bags, and name the column that contains the number of bags Count. Then, display the resulting DataFrame.
6. Plot the melted data with Seaborn in a line plot, using the hue parameter to distinguish between the bag types.
7. Bin the data in the Total Volume column into four quantiles labeled 'poor', 'modest', 'good', and 'excellent', and store the bin labels in a new column.
8. Plot the binned data by year using a Seaborn count plot.

Questions

1. Of the three bag sizes, which size sells the most? Hint: Melt the three bag size columns and plot the bags by year.
2. Which type of avocado sells the most?

Project 9-1: Analyze gold prices over time

The data file

`gold.csv`

Tasks

1. Read the data from the CSV file into a DataFrame called gold.
2. Run the info() method, and note the data types of the two columns.
3. Convert the Date column to the DateTime data type. Then, display the first five rows of data, and note that the dates are for the first of each month starting in 1950.
4. Plot the data with a Pandas line plot.
5. Index the data on the Date column. Then, generate time periods for the first day of each year from 1950 to 2020.
6. Reindex the data using the new time periods, and display the first five rows again to see that there is just one row for each year.
7. Plot the reindexed data with a Pandas line plot.
8. Resample the data to a quarterly frequency and plot the data again.
9. Resample the data to a yearly frequency and plot the data one more time.
10. Plot the rolling mean for the data using a Pandas line plot. Experiment with different values for the window parameter.

Project 9-2: Analyze retail sales over time

The data file

`scanner_data.csv`

Tasks

1. Read the data from the CSV file into a DataFrame.
2. Run the info() method and display the first five rows of the data.
3. Drop the two index columns as well as the SKU column.
4. Convert the Date column to the datetTime data type.
5. Group the data by the Date, Customer_ID, and SKU_Category columns, and calculate the sum of the Quantity and Sales_Amount columns. Then, display the first five rows of grouped data.
6. Resample the data to a monthly frequency, and save the results in a new DataFrame.
7. Plot the monthly Quantity and Sales_Amount columns by date in a Pandas line plot.
8. Resample the data to a quarterly frequency, and save the results in a new DataFrame.
9. Plot the quarterly Quantity and Sales_Amount columns by date in a Pandas line plot..
10. Add the running total for the Sales_Amount column to the DataFrame that contains the monthly data.
11. Plot the running total and the Sales_Amount column together in a Pandas bar plot.

Project 10-1: Use linear regression with the diamond data

Tasks

1. The data file for this project is available from the Seaborn website. To load this data into a DataFrame, run this code:
```
data = sns.load_dataset('diamonds')
```
2. Display the first five rows of data.
3. Use a condensed heatmap to identify correlations for the price column. Sort the results, include annotations that format the values with three decimal places, and remove the color bar.
4. Use the column with the strongest correlation to split the data into test and training datasets, where the test dataset consists of 30% of the total dataset. Be sure to specify a value for the random_state parameter.
5. Create a LinearRegression object, and then fit the training dataset to the model.
6. Score the model using the test dataset.
7. Predict the y values based on the x values in the test dataset, and store the results in a variable.
8. Put the predicted values in a DataFrame.
9. Join the y_test and predicted data with the x_test data, save the joined data in a new DataFrame, and then display the first five rows of data.
10. Melt the actual and predicted price columns together, assigning appropriate values to the variable and value names.
11. Use a Seaborn line plot without a confidence interval to plot the predicted data.

Project 10-2: Use linear regression with the MPG data

Tasks

1. The data file for this project is available from the Seaborn website. To load this data into a DataFrame, run this code:
   ```
   mpg = sns.load_dataset('mpg')
   ```
2. Display the first five rows of data.
3. Get the correlation data for the mpg column. Sort the results to make it easier to see the columns with the stronger correlations.
4. Use the column with the strongest positive or negative correlation to the mpg column to split the data into test and training datasets, where the test dataset consists of 20% of the total dataset. Be sure to specify a value for the random_state parameter.
5. Create a linear regression model from the training dataset.
6. Score the model using the test dataset.
7. Score the model using the training dataset.
8. Predict the y values based on the x values in the test dataset, and store the results in a variable. Then, put the predicted values in a DataFrame.
9. Join the y_test and predicted data with the x_test data, save the combined data in a new DataFrame, and then display the first five rows of data.
10. Add the residuals to the new DataFrame.
11. Plot the residuals in a Seaborn KDE plot.
12. Melt the actual and predicted MPG columns together, assigning appropriate values to the variable and value names.
13. Use a Seaborn scatter plot to plot the predicted data.

Project 11-1: Use multiple regression with the diamond data

Tasks

1. The data file for this project is available from the Seaborn website. To load this data into a DataFrame, run this code:
   ```
   data = sns.load_dataset('diamonds')
   ```
2. Display the first five rows of data.
3. Display the correlation data for the price column.
4. Create test and training datasets using the carat, table, and depth columns as the independent variables and the price as the dependent variable. (The x, y, and z columns contain information that's related to the table and depth columns, so it's not necessary to use those columns.) The test dataset should consist of 30% of the total dataset, and you should specify a value for the random_state parameter.
5. Create and fit the model.
6. Score the model with the test dataset.
7. Score the model with the training dataset.
8. Use the model to make predictions about the test data, and store the results in a DataFrame.
9. Create a DataFrame that contains the columns used to make predictions, along with the actual price and the predicted price. Then, display the first five rows of data to see how close the predicted prices are.
10. Calculate the residuals for the regression, and store the results in a new column in the DataFrame you created in step 9. Then, display the first five rows of data to see the residual values.
11. Plot the residuals using a Seaborn KDE plot.

Project 11-2: Use categorical variables with the diamond data

Tasks

1. The data file for this project is available from the Seaborn website. To load this data into a DataFrame, run this code:
   ```
   data = sns.load_dataset('diamonds')
   ```
2. Display the first five rows of data.
3. Drop the x, y, and z columns, since they won't be used to make predictions.
4. Use the info() method to display the data types for the columns.
5. Convert the three columns with categorical data into dummy variables and store the results in a new DataFrame.
6. Drop the categorical columns from the original DataFrame and join the DataFrame with the dummy variables to it. Store the result in a DataFrame named dataDummies, and then use the info() method to display the resulting columns.
7. Rescale the data in the numeric columns, and then display the rescaled data.
8. Display the correlation data for the price column.
9. Create test and training datasets using the five columns with the highest correlation. The test dataset should consist of 30% of the total dataset, and you should specify a value for the random_state parameter.
10. Create and fit the model.
11. Score the model with the test dataset.
12. Score the model with the training dataset.
13. Use the model to make predictions about the test data, and store the results in a DataFrame.
14. Create a DataFrame that contains the columns used to make predictions, along with the actual price and the predicted price. Then, display the first five rows of data to see how close the predicted prices are.
15. Calculate the residuals for the regression, and store the results in a new column in the DataFrame you created in step 14. Then, display the first five rows of data to see the residual values.
16. Plot the residuals using a Seaborn KDE plot.
17. Use a for loop with the feature selection model to test different numbers of features. Be sure not to include the price column in the list of independent variables. Note that it may take a few minutes for this code to run.
18. Plot the test and training scores with a line plot based on the number of features.

Project 11-3: Use multiple regression with the MPG data

Tasks

1. The data file for this project is available from the Seaborn website. To load this data into a DataFrame, run this code:
   ```
   data = sns.load_dataset('mpg')
   ```
2. Display the first five rows of data.
3. Use the info() method to display the data types for the columns.
4. Get the brand from the name column and store it in a new column named brand. Hint: To do that, you can use the apply() method with a lambda expression.
5. Drop the name column, and then display the first five rows of data.
6. Display all of the unique values in the brand column.
7. Review the brand data, and then fix the spelling errors in the brand column.
8. Drop all rows from the DataFrame that have missing values.
9. Get the two categorical columns (the one with the object data type), and store them in a list. Then, convert those columns to dummy variables, and store the results in a new DataFrame.
10. Drop the categorical columns from the original DataFrame, and join the DataFrame with the dummy variables to it. Store the result in a DataFrame named dataDummies, and then use the info() method to display the resulting columns.
11. Rescale the data in the numeric columns, and then display the rescaled data.
12. Display the correlation data for the mpg column.
13. Split the data into test and training datasets. The test dataset should consist of 20% of the total dataset, and you should specify a value for the random_state parameter. Note that there aren't any non-numeric columns to drop.
14. Use a for loop with the feature selection model to test different numbers of features. Be sure not to include the price column in the list of independent variables. Note that it may take a few minutes for this code to run.
15. Plot the test and training scores with a line plot based on the number of features.