Data Visualization Fundamentals

Data Visualization in Python:

As we have covered in earlier weekly modules, data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends and correlations that might not otherwise be detected can be exposed. Python offers multiple great graphing libraries that come packed with lots of different features. In prior tutorials, we have looked at boxplots, heatmaps, and histograms. These are just a few of many charting options available in Python. In this tutorial, we will be using the pandas, seaborn, and matplotlib libraries. The following URLs list (1) data visualization packages we discuss in this tutorial, (2) types of charts available via the Seaborn package library, and (3) eight other data visualization package libraries outside of this tutorial:

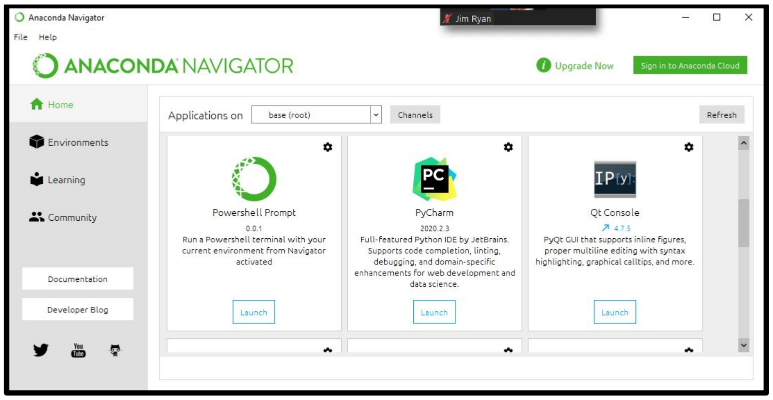
(1) - https://towardsdatascience.com/introduction-to-data-visualization-in-python-89a54c97fbed

(2) - https://python-graph-gallery.com/all-charts/

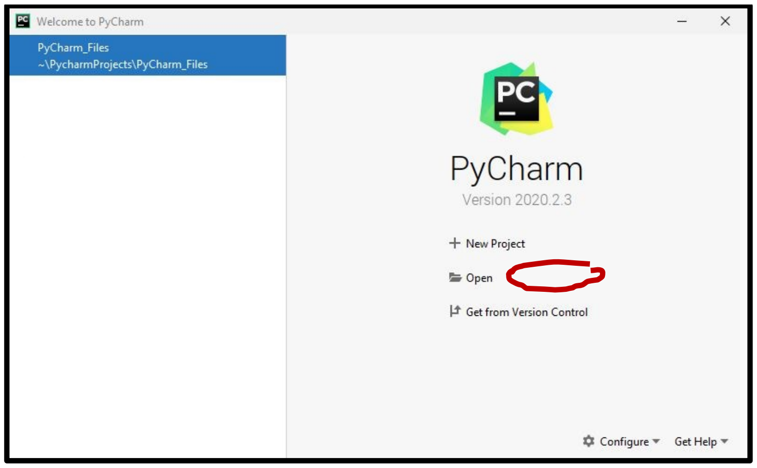
(3) - https://mode.com/blog/python-data-visualization-libraries/

Create a DataVisualization Project in PyCharm

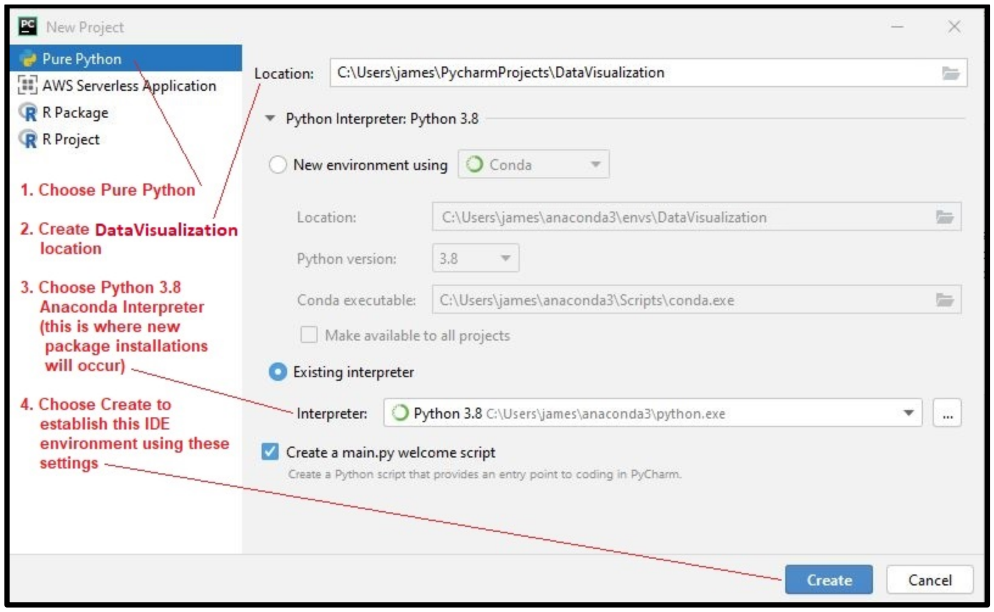
0) To ensure your various Python package libraries are available to import, be sure to use Anaconda Navigator to launch PyCharm.


Anaconda Navigator
Jim Ryan
File Help
ANACONDA NAVIGATOR
Upgrade Now
Sign in to Anaconda Cloud
Home
Environments
Learning
Community
Documentation
Developer Blog
Applications on base (root)
Channels
Refresh
Powershell Prompt
0.0.1
Run a Powershell terminal with your current environment from Navigator activated
Launch
PyCharm
2020.2.3
Full-featured Python IDE by JetBrains. Supports code completion, linting, debugging, and domain-specific enhancements for web development and data science.
Launch
Qt Console
4.7.5
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Launch

1) Launch PyCharm and choose a New Project.



1) Choose a Pure Python project,
2) Change your project location to create a DataVisualization folder
3) Be sure to choose the Python interpreter as Python 3.8 associated with Anaconda.
4) Choose Create to open PyCharm with these settings.



1. Choose Pure Python
2. Create **DataVisualization** location
3. Choose Python 3.8 Anaconda Interpreter (this is where new package installations will occur)
4. Choose Create to establish this IDE environment using these settings

Location: C:\Users\james\PycharmProjects\DataVisualization

Python Interpreter: Python 3.8

New environment using Conda

Location: C:\Users\james\anaconda3\envs\DataVisualization

Python version: 3.8

Conda executable: C:\Users\james\anaconda3\Scripts\conda.exe

Make available to all projects

Existing interpreter

Interpreter: Python 3.8 C:\Users\james\anaconda3\python.exe

Create a main.py welcome script

Create a Python script that provides an entry point to coding in PyCharm.

These images can be viewed at 400% magnification for details of the screen shots.

5) *Download* the single_family_home_values.csv file from the Week 11 Canvas Module to the DataVisualization directory created in the last step.

Reading a CSV file into a Pandas DataFrame:

Pandas is an open-source Python Library used for high-performance data manipulation and data analysis using its powerful data structures. Pandas use Series and DataFrame on top of Numpy array. In this tutorial, we will look at DataFrame or df. Python with pandas is in use in a variety of academic and commercial domains, including Finance, Economics, Statistics, Advertising, Web Analytics, and more. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data — load, organize, manipulate, model, and analyze the data. For more specific details about Pandas not covered in this tutorial, please refer to the following URL:
https://www.tutorialspoint.com/python_data_science/python_pandas.htm

1) Anaconda will load the libraries in PyCharm for you. To access the three libraries we are using for Tutorial Nine (e.g., pandas, seaborn, and matplotlib.pyplot), enter the following code into Main.py in the DataVisualization project directory of PyCharm:

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

2) To format the run window to handle the size of the csv data file we will be wrangling in Tutorial Seven, enter the following Python code:

```
desired_width = 400
pd.set_option('display.width', desired_width)
pd.set_option('display.max_columns', 20)
```

3) To read the single_family_home_values.csv file into a pandas dataframe, type the following Python code into main.py via PyCharm in the line below the previous code above.

```
df = pd.read_csv(r'single_family_home_values.csv')   # reads Zillow file
```

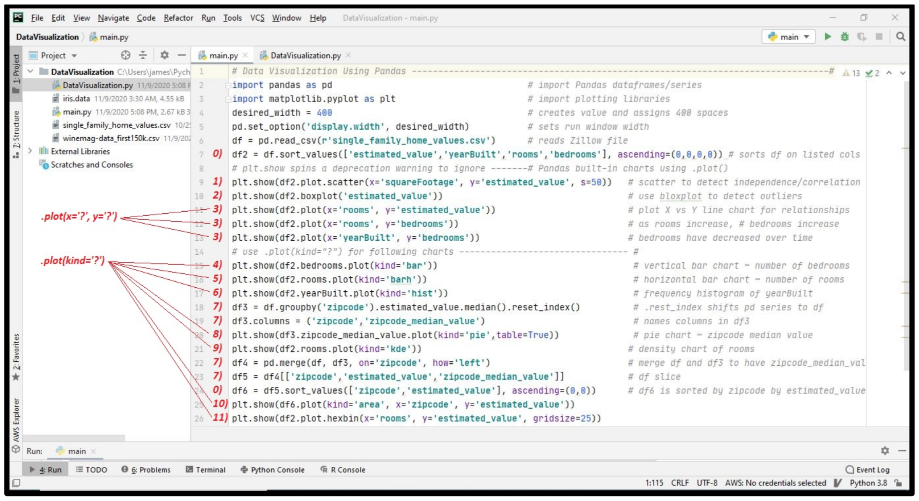
4) Once the csv file is loaded into a dataframe (df), we can begin data wrangling. Any of the following Python functions may be used with a df, once it has been assigned.

```
print(df.head(2))     # returns top two rows
print(df.tail(5))     # returns bottom five rows
print(type(df))       # returns type of df
print(df.shape)       # returns row & column count
print(df.info())      # returns column attributes
print(df.describe()) # returns profiling stats
```

5) In addition to ‘single_family_home_values.csv’, we will also use data files ‘iris.data’ and ‘winemag-data_first150k.csv’. All three of these data files are available for download via the Week 11 Module in Canvas. Feel free to use either of these data sets to practice data visualization tools.

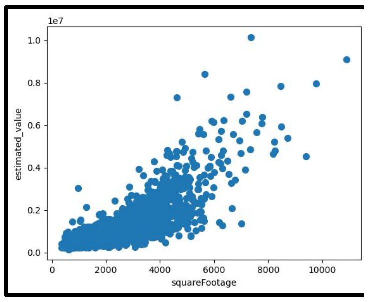
Data Visualization using Pandas:

The Pandas package libraries have built-in data visualization tools for many different types of charts and plots. For more information outside this Tutorial on data visualization using Pandas refer to the URL: https://pandas.pydata.org/pandas-docs/stable/user_guide/visualization.html The following PyCharm screenshot shows the Python code for (0) sorting data frame values, (1) graphing a scatter plot, (2) boxplot, (3) three different x-y plots, (4) vertical bar chart, (5) horizontal bar chart, (6) frequency histogram, (7) group by, merge, or slice of data frames, (8) graphing a pie chart with data table, (9) density distribution chart, (10) area chart, and (11) hexoganol bin chart . All these data visualization examples use the plt.show() function from the matplotlib.pyplot package library to display the graphic.

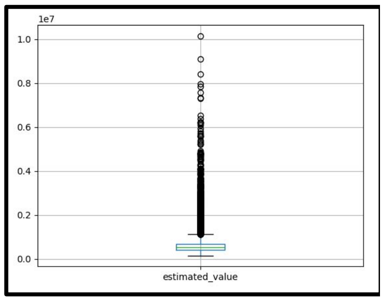


```
    # Data Visualization Using Pandas -------------------------------------------------------------------------#
    import pandas as pd                                  # import Pandas dataframes/series
    import matplotlib.pyplot as plt                      # import plotting libraries
    desired_width = 400                                  # creates value and assigns 400 spaces
    pd.set_option('display.width', desired_width)        # sets run window width
    df = pd.read_csv(r'single_family_home_values.csv')   # reads Zillow file
0)  df2 = df.sort_values(['estimated_value','yearBuilt','rooms','bedrooms'], ascending=(0,0,0,0)) # sorts df on listed cols
    # plt.show spins a deprecation warning to ignore -------# Pandas built-in charts using .plot()
1)  plt.show(df2.plot.scatter(x='squareFootage', y='estimated_value', s=50))  # scatter to detect independence/correlation
2)  plt.show(df2.boxplot('estimated_value'))                                   # use bloxplot to detect outliers
3)  plt.show(df2.plot(x='rooms', y='estimated_value'))                         # plot X vs Y line chart for relationships
3)  plt.show(df2.plot(x='rooms', y='bedrooms'))                                # as rooms increase, # bedrooms increase
3)  plt.show(df2.plot(x='yearBuilt', y='bedrooms'))                            # bedrooms have decreased over time
    # use .plot(kind="?") for following charts -------------------------------- #
4)  plt.show(df2.bedrooms.plot(kind='bar'))                                     # vertical bar chart ~ number of bedrooms
5)  plt.show(df2.rooms.plot(kind='barh'))                                       # horizontal bar chart ~ number of rooms
6)  plt.show(df2.yearBuilt.plot(kind='hist'))                                   # frequency histogram of yearBuilt
7)  df3 = df.groupby('zipcode').estimated_value.median().reset_index()          # .rest_index shifts pd series to df
7)  df3.columns = ('zipcode','zipcode_median_value')                            # names columns in df3
8)  plt.show(df3.zipcode_median_value.plot(kind='pie',table=True))              # pie chart ~ zipcode median value
9)  plt.show(df2.rooms.plot(kind='kde'))                                       # density chart of rooms
7)  df4 = pd.merge(df, df3, on='zipcode', how='left')                          # merge df and df3 to have zipcode_median_val
7)  df5 = df4[['zipcode','estimated_value','zipcode_median_value']]            # df slice
0)  df6 = df5.sort_values(['zipcode','estimated_value'], ascending=(0,0))      # df6 is sorted by zipcode by estimated_value
10) plt.show(df6.plot(kind='area', x='zipcode', y='estimated_value'))
11) plt.show(df2.plot.hexbin(x='rooms', y='estimated_value', gridsize=25))
```

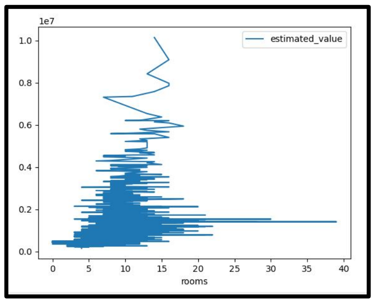
A screen scraping tool like the Windows Snipping Tool can be used to save a plot.show() graphic or as we learned in Tutorial Eight-B under Network Mining, the plt.savefig() function from matplotlib.pyplot will save a copy of the graphic to disk in the Python project directory. From the Python code in the previous PyCharm screenshot, the following graphics are chart screenshots.



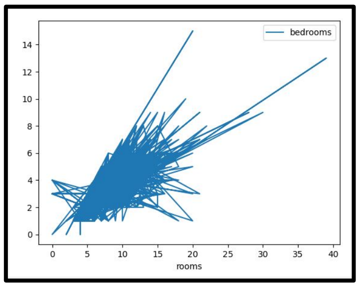
(1) Scatterplot



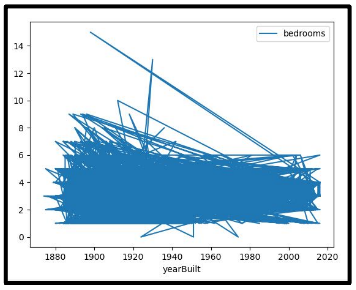
(2) boxplot



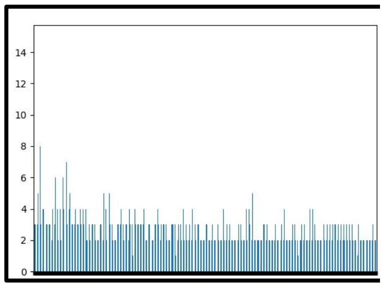
(3) plot(x,y)



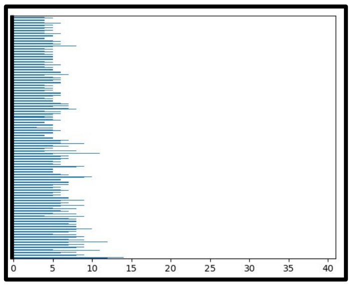
(3) plot(x,y)



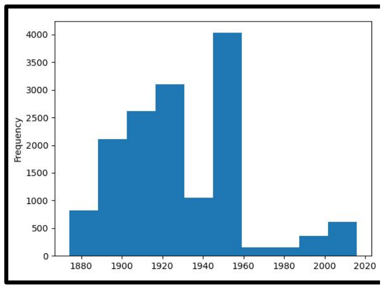
(3) plot(x,y)



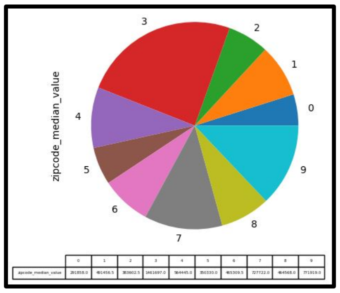
(4) vertical bar



(5) horizontal bar

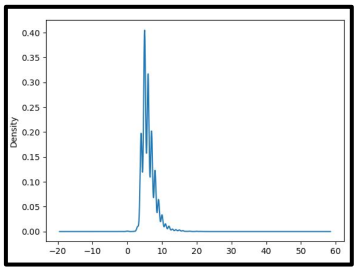


(6) frequency histogram

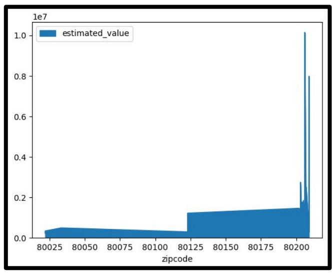


	0	1	2	3	4	5	6	7	8	9
zipcode_median_value	291858.0	491456.5	353602.5	1461697.0	564445.0	350330.0	465309.5	727722.0	464568.0	771919.0

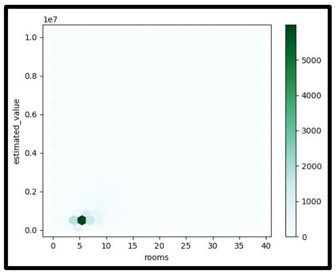
(8) pie chart with data table



(9) density distribution chart



(10) area chart

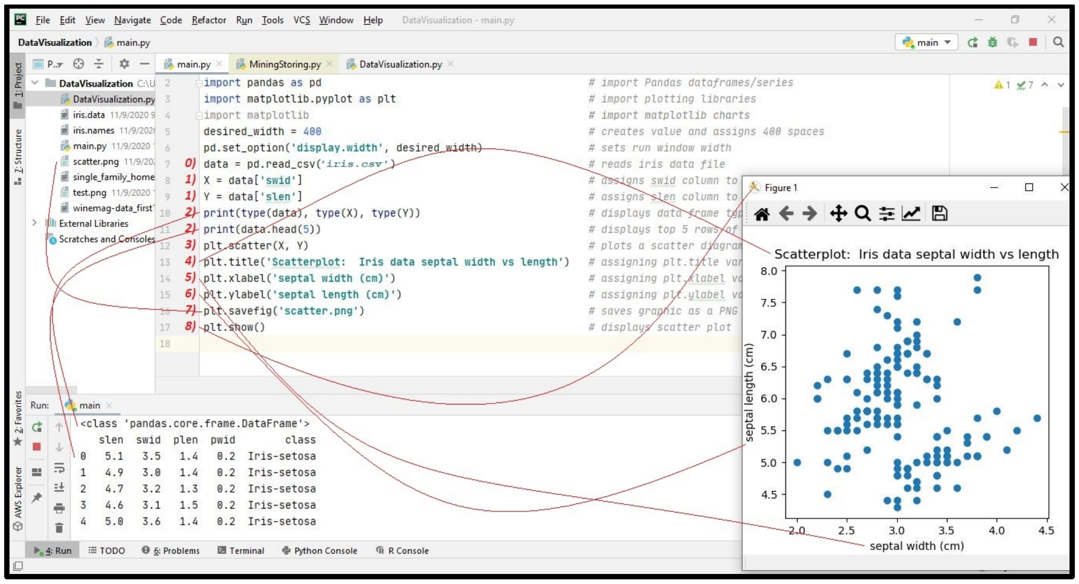


(11) hexagonal bin chart

Exercise deliverable: Construct an example screenshot of using Pandas built-in data visualization functions in Python code in either of the Week 11 data files or your data project csv. If you are using the csv above, then choose different column attributes for the charts. (1) Choose four different charts. (2) Capture a PyCharm screenshot of the Python code creating the four charts. (3) Either use a screen scraper or plt.savfig() function to save charts to disk as PNG files. Save the PyCharm screenshot and chart PNG files for use later in this tutorial.

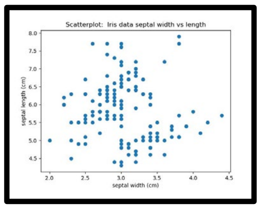
Data Visualization Using Matplotlib:

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible. In this and prior Tutorials, we have used the plt.show() and plt.savefig() functions to display or save charts. Consider all of the charts available via the matplotlib libraries via the URL: https://matplotlib.org/gallery/index.html . In the PyCharm screenshot below, we use matplotlib.pyplot to create a scatterplot from the 'iris.csv' file. The Python code assigns plt.title, plt.Xlabel, and plt.Ylabel variables to appear on the scatterplot diagram. Specifically, the Python code demonstrates (0) reading the CSV file, (1) create a data frame slice for X and Y, (2) print the data frame type and the top five rows of the data frame, (3) create the scatterplot using X and Y data frames, (4) assign the plot title, (5) assign the X label title, (6) assign the Y label title, (7) save the scatterplot to disk as file scatter.png, and (8) display the scatterplot chart. Most matplotlib functions are used with NumPy and this example is an exception.



```
    import pandas as pd                                    # import Pandas dataframes/series
    import matplotlib.pyplot as plt                        # import plotting libraries
    import matplotlib                                      # import matplotlib charts
    desired_width = 400                                    # creates value and assigns 400 spaces
    pd.set_option('display.width', desired_width)          # sets run window width
0)  data = pd.read_csv('iris.csv')                         # reads iris data file
1)  X = data['swid']                                       # assigns swid column to
1)  Y = data['slen']                                       # assigns slen column to
2)  print(type(data), type(X), type(Y))                    # displays data frame typ
2)  print(data.head(5))                                    # displays top 5 rows of
3)  plt.scatter(X, Y)                                      # plots a scatter diagram
4)  plt.title('Scatterplot:  Iris data septal width vs length')  # assigning plt.title var
5)  plt.xlabel('septal width (cm)')                        # assigning plt.xlabel va
6)  plt.ylabel('septal length (cm)')                       # assigning plt.ylabel va
7)  plt.savefig('scatter.png')                             # saves graphic as a PNG
8)  plt.show()                                             # displays scatter plot
```

```
<class 'pandas.core.frame.DataFrame'>
   slen  swid  plen  pwid        class
0   5.1   3.5   1.4   0.2  Iris-setosa
1   4.9   3.0   1.4   0.2  Iris-setosa
2   4.7   3.2   1.3   0.2  Iris-setosa
3   4.6   3.1   1.5   0.2  Iris-setosa
4   5.0   3.6   1.4   0.2  Iris-setosa
```



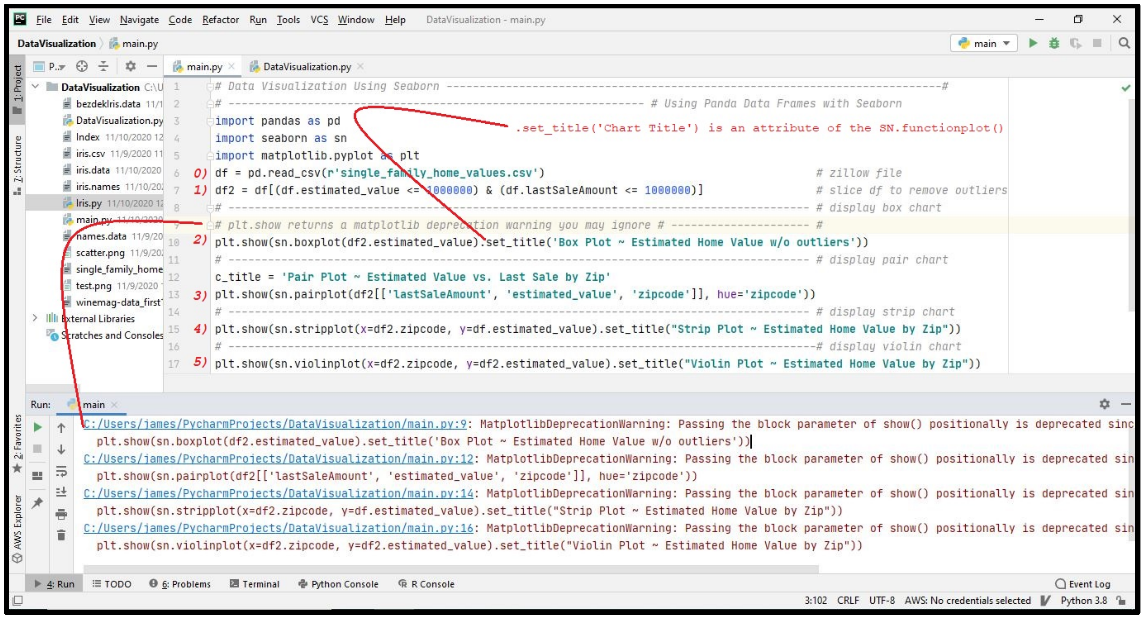
The figure to the left is the scatterplot saved to disk in the PyCharmProjects subdirectory. Saving a chart to disk allows you to access the graphic at a later date, rather than recreating the graphic. Note the plt.title, plt.Xlabel, and plt.Ylabel variables on the scatter.png graphic. All illustrations in this Tutorial may be viewed at 400 times.

Exercise deliverable: Construct an example scatterplot from your project data set. Be sure to assign titles for the chart as well as the X and Y axis. Remember the utility of scatter diagrams is the review the independence between variables X and Y. Variable Y is not independent of variable X if any correlation (positive or negative) can be construed from the scatter diagram. Save the PyCharm screenshot showing the Python code and the scatter diagram PNG file (i.e., use a screen scraper or plt.savefig() for later in this tutorial. (If you have issues using your project data set, then use one of the data sets in Week 11 module).

Using Seaborn for Data Visualization:

Seaborn is a Python data visualization library based on matplotlib and integrates closely with Pandas data structures. It provides a high-level interface for drawing attractive and informative statistical graphics. In prior Tutorials, we have used the seaborn boxplot as well as heat map. More graphics are available to review via the URL: https://seaborn.pydata.org/examples/index.html

In the PyCharm screenshot below, the we combine Pandas data frames and Seaborn plotting functions to create four different statistical charts (e.g., box, pair, strip, and violin). Specifically, the Python code (0) reads the Zillow home sales CSV into a Pandas df, (1) slices the df to remove outliers, (2) displays a box chart with the inter-quartile range, (3) displays a pair chart comparing scatter diagrams and area charts for estimated value and lastSaleAmount, (4) displays a strip chart of estimated home value distributions by zip code, and (5) a violin chart of estimated home value box plots by zip code. All for of these charts help describe distributions and the plt.show() function from matplotlib.pyplot enables the display of the seaborn graphic. All Python code from the PyCharm screenshots in Tutorial Nine may be found in the file DataVisualization.py, which can be downloaded from the Week 11 Module in Canvas.

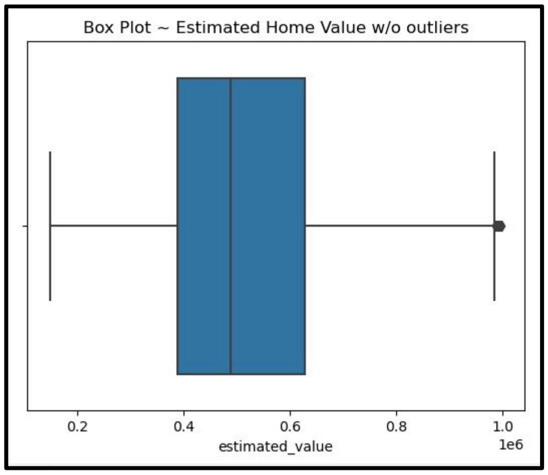

File Edit View Navigate Code Refactor Run Tools VCS Window Help DataVisualization - main.py
DataVisualization main.py
main.py DataVisualization.py
DataVisualization C:\U
bezdekIris.data
DataVisualization.py
Index
iris.csv
iris.data
iris.names
Iris.py
main.py
names.data
scatter.png
single_family_home
test.png
winemag-data_first
External Libraries
Scratches and Consoles
Run: main
4: Run TODO 6: Problems Terminal Python Console R Console Event Log
3:102 CRLF UTF-8 AWS: No credentials selected Python 3.8

.set_title('Chart Title') is an attribute of the SN.functionplot()

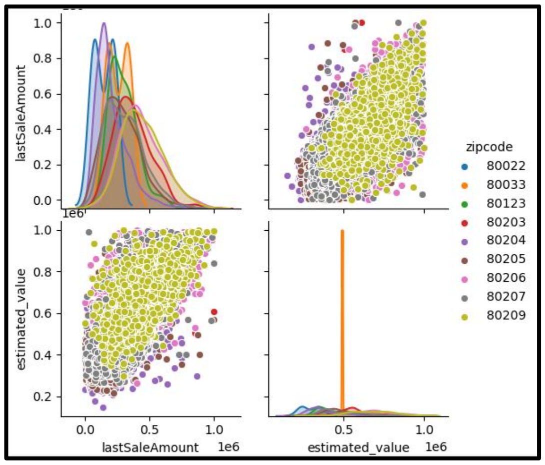
```
   # Data Visualization Using Seaborn ----------------------------------------------------------------#
   # ------------------------------------------------------ # Using Panda Data Frames with Seaborn
   import pandas as pd
   import seaborn as sn
   import matplotlib.pyplot as plt
0) df = pd.read_csv(r'single_family_home_values.csv')                                  # zillow file
1) df2 = df[(df.estimated_value <= 1000000) & (df.lastSaleAmount <= 1000000)]          # slice df to remove outliers
   # ----------------------------------------------------------------------------------- # display box chart
   # plt.show returns a matplotlib deprecation warning you may ignore # -------------------- #
2) plt.show(sn.boxplot(df2.estimated_value).set_title('Box Plot ~ Estimated Home Value w/o outliers'))
   # ----------------------------------------------------------------------------------- # display pair chart
   c_title = 'Pair Plot ~ Estimated Value vs. Last Sale by Zip'
3) plt.show(sn.pairplot(df2[['lastSaleAmount', 'estimated_value', 'zipcode']], hue='zipcode'))
   # ----------------------------------------------------------------------------------- # display strip chart
4) plt.show(sn.stripplot(x=df2.zipcode, y=df.estimated_value).set_title("Strip Plot ~ Estimated Home Value by Zip"))
   # -----------------------------------------------------------------------------------# display violin chart
5) plt.show(sn.violinplot(x=df2.zipcode, y=df2.estimated_value).set_title("Violin Plot ~ Estimated Home Value by Zip"))
```

```
C:/Users/james/PycharmProjects/DataVisualization/main.py:9: MatplotlibDeprecationWarning: Passing the block parameter of show() positionally is deprecated sinc
  plt.show(sn.boxplot(df2.estimated_value).set_title('Box Plot ~ Estimated Home Value w/o outliers'))
C:/Users/james/PycharmProjects/DataVisualization/main.py:12: MatplotlibDeprecationWarning: Passing the block parameter of show() positionally is deprecated sin
  plt.show(sn.pairplot(df2[['lastSaleAmount', 'estimated_value', 'zipcode']], hue='zipcode'))
C:/Users/james/PycharmProjects/DataVisualization/main.py:14: MatplotlibDeprecationWarning: Passing the block parameter of show() positionally is deprecated sin
  plt.show(sn.stripplot(x=df2.zipcode, y=df.estimated_value).set_title("Strip Plot ~ Estimated Home Value by Zip"))
C:/Users/james/PycharmProjects/DataVisualization/main.py:16: MatplotlibDeprecationWarning: Passing the block parameter of show() positionally is deprecated sin
  plt.show(sn.violinplot(x=df2.zipcode, y=df2.estimated_value).set_title("Violin Plot ~ Estimated Home Value by Zip"))
```

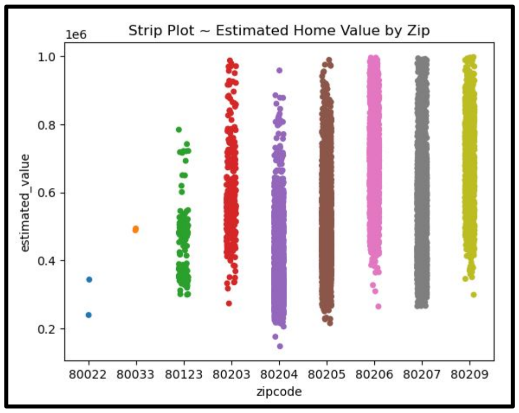
The four charts created by the Python code in the previous PyCharm screenshot are viewable as



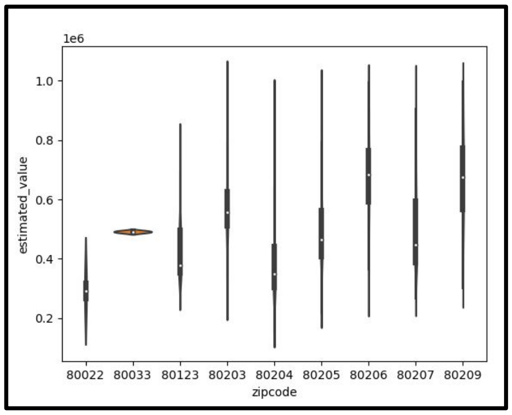
Box Chart



Pair Chart



Strip Chart



Violin Chart

Exercise deliverable: Construct two graphic examples of a box chart, pair chart, strip chart, and/or violin chart for practice in data visualization tools. Capture the chart graphics using the plt.savefig() function or a screen scraper. Use your project data set or either of the CSV files in the Week 11 Module. Save the PyCharm screenshot displaying the Python code and the two chart graphics for use later in this tutorial.

Deliverables from Exercises Above:

Take the noted deliverables (i.e., PyCharm screenshots) from each exercise above (i.e., total of 4 screenshots), insert the screenshots into a MS Word document as you label each screenshot deliverable from the corresponding exercise, save the document as one PDF, and submit the PDF to the Tutorial Seven Canvas Assignment uplink having the following qualities:

a. Your screenshot for each of the two exercises above is included.
b. Each screenshot is labeled appropriately for each exercise.
   i. Pandas Built-in Data Visualization ………….……………. 5 screenshots
   ii. Matplotlib Data Visualization ……………………………… 2 screenshots

iii. Seaborn Data Visualization …..................................... 3 screenshot

Total screenshots to submit ……………………………….. 10 screenshots

c. All screenshots are legible output for each PyCharm exercise deliverable.