IDC4251C Module 7 Project

Consider a simple example of using the k-Nearest Neighbors (kNN) algorithm with financial data to predict whether the stock price of a company will increase or decrease based on certain financial indicators.

Given historical data with the following features for various companies:

- Feature 1: Price-to-Earnings (P/E) Ratio
- Feature 2: Debt-to-Equity (D/E) Ratio
- Feature 3: Return on Equity (ROE)
- Target: Stock Price Trend (Increase or Decrease) (1 or 0 in the data, respectively)

For more information on these features please refer to the following web page:

https://groww.in/blog/financial-ratios-every-investor-should-know

Use these features to form a feature vector for each company in our dataset. The kNN algorithm can then be used to classify a new company's stock price trend based on the trends of its 'k' nearest neighbors in this feature space.

For example, to predict the stock price trend for Company X with the following data:

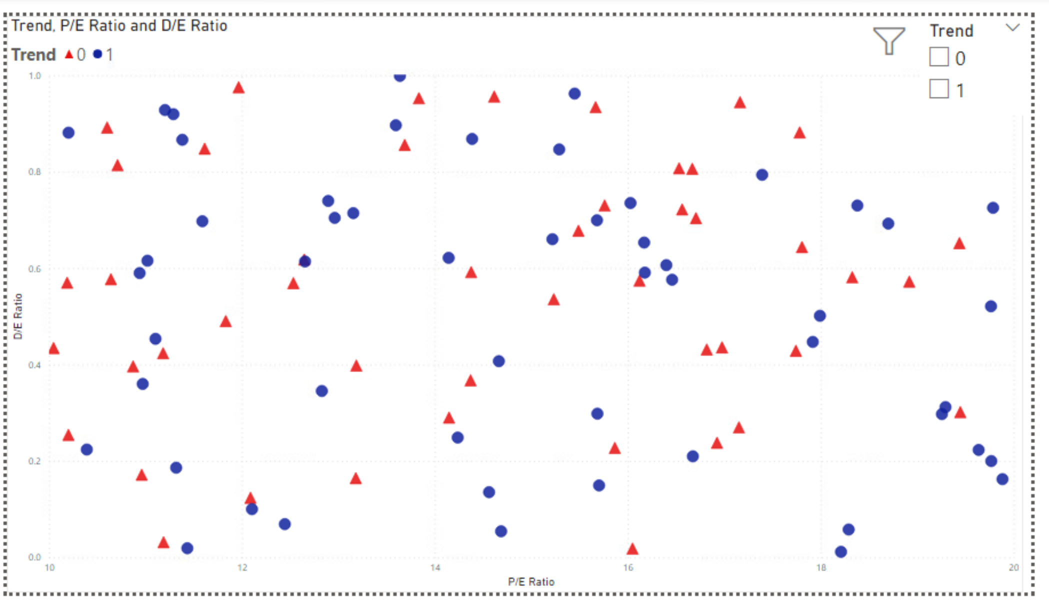
P/E Ratio: 15
D/E Ratio: 0.5
ROE: 10%

- calculate the distance (Euclidean or rectilinear) from Company X to all other companies in the dataset.
- select the **k** closest companies and look at the majority trend among these neighbors.

If k is 5, and three out of the five nearest companies have a stock price increase trend, the algorithm would predict that Company X's stock price will likely increase.

This is a very simplified example; in real-world financial modeling, there more features, a lot more data, and noise and non-linearity that would need to be accounted for. Financial data is typically noisy and non-stationary, meaning that the relationships between variables can change over time, which can make kNN less effective without careful feature selection and preprocessing.

To plot these features in a 2D space, only two at a time can be selected. For instance, the P/E Ratio could be plotted on the x-axis and the D/E Ratio on the y-axis, with different markers or colors to represent the stock price trend (increase or decrease) as shown in the following image.



The blue dots indicate companies with an increasing stock price trend (1), while the red triangles indicate companies with a decreasing trend (0).

For this assignment we will visualize a dataset used for KNN classification and run an embedded Python script in the PowerBI workbook. You will need to complete two tasks using a new workbook:

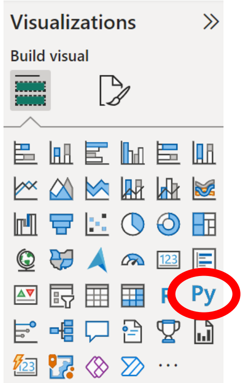
Step 1. Duplicate the scatter plot shown above using the dataset provided in the GitHub Classroom repo. (financial_data_with_tickers_knn.csv) . Be sure to include the slicer which allows the two trend values to be filtered. This plot should be displayed on the Page 1 report.

*** The following step will not be available on Horizon until the software is updated, you can submit your workbook to the repo now without this step or, if you would like to try it on your personal system, be sure that Python is installed along with the following packages: numpy, pandas, matplotlib, and sklearn. If you aren't sure how to do this you should probably just skip this step for now.

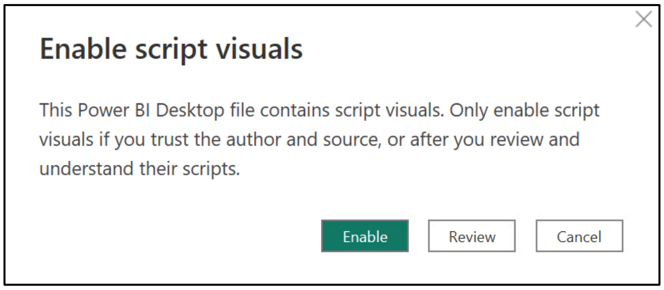
Step 2. Create a Page 2 report and insert a Python visual which runs a Python script (provided in the GitHub Classroom repo) which uses KNN to predict company stock trends using the dataset from step 1. Follow the instructions as shown below.

Adding the Python Visual (New Report Page)

The Python visual is the “Py” icon found in the visualization tool palette as shown here:



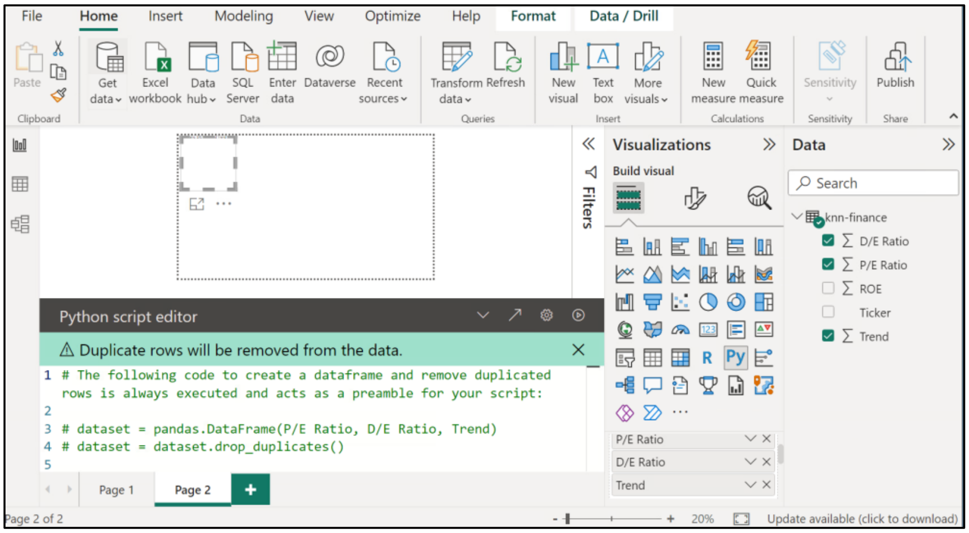
After clicking on the icon you may be prompted to enable script visuals, be sure to click Enable:



Enable script visuals

This Power BI Desktop file contains script visuals. Only enable script visuals if you trust the author and source, or after you review and understand their scripts.

You will then be presented with a script editor. Click on the following fields in the data pane: P/E Ratio, D/E Ratio, and Trend. This will add the preamble to the top of the script editor as shown here:

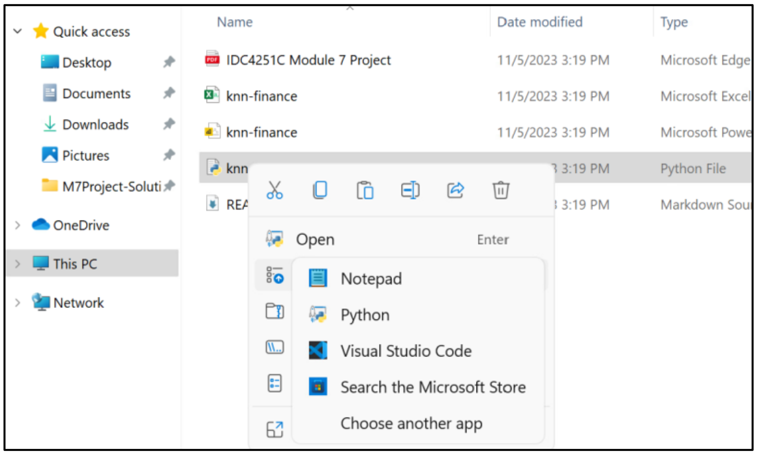


⚠ Duplicate rows will be removed from the data.

```
# The following code to create a dataframe and remove duplicated
rows is always executed and acts as a preamble for your script:

# dataset = pandas.DataFrame(P/E Ratio, D/E Ratio, Trend)
# dataset = dataset.drop_duplicates()
```

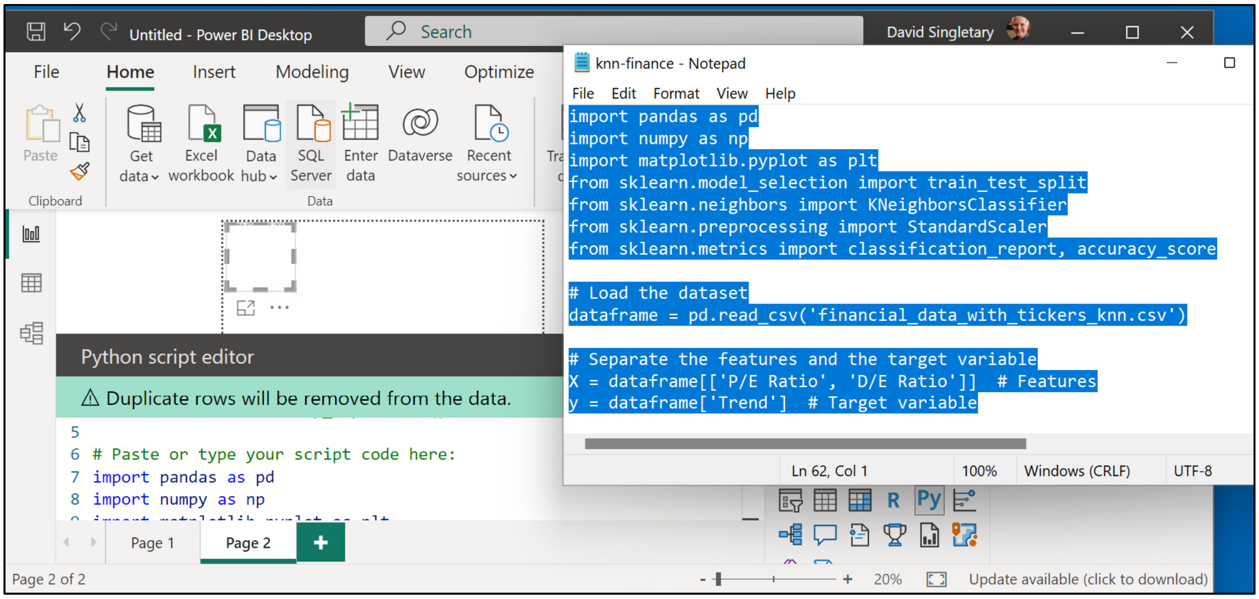
The preamble will create a Pandas dataframe (Pandas is a data manipulation library in Python, a dataframe is a type of table) which is used to run the algorithm.

Locate the Python script in the cloned GitHub Classroom repo, right-click on it and select “Open With” then scroll down and open it with the Notepad application (any text editor will work here):



Name	Date modified	Type
IDC4251C Module 7 Project	11/5/2023 3:19 PM	Microsoft Edge
knn-finance	11/5/2023 3:19 PM	Microsoft Excel
knn-finance	11/5/2023 3:19 PM	Microsoft Powe
knn	3:19 PM	Python File
REA	3:19 PM	Markdown Sou

In the Notepad application, Select All (Ctrl-A), then copy the Python code into the clipboard (Ctrl-C) and paste it into the Python script editor in Power BI starting at line 7, after the line which says “# Paste or type your script code here:”

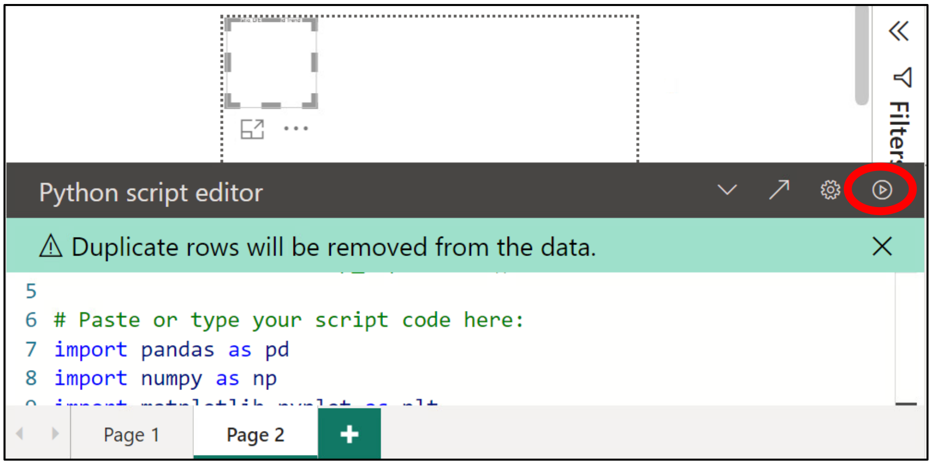


```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, accuracy_score

# Load the dataset
dataframe = pd.read_csv('financial_data_with_tickers_knn.csv')

# Separate the features and the target variable
X = dataframe[['P/E Ratio', 'D/E Ratio']]  # Features
y = dataframe['Trend']  # Target variable
```

⚠ Duplicate rows will be removed from the data.

```
5
6 # Paste or type your script code here:
7 import pandas as pd
8 import numpy as np
```

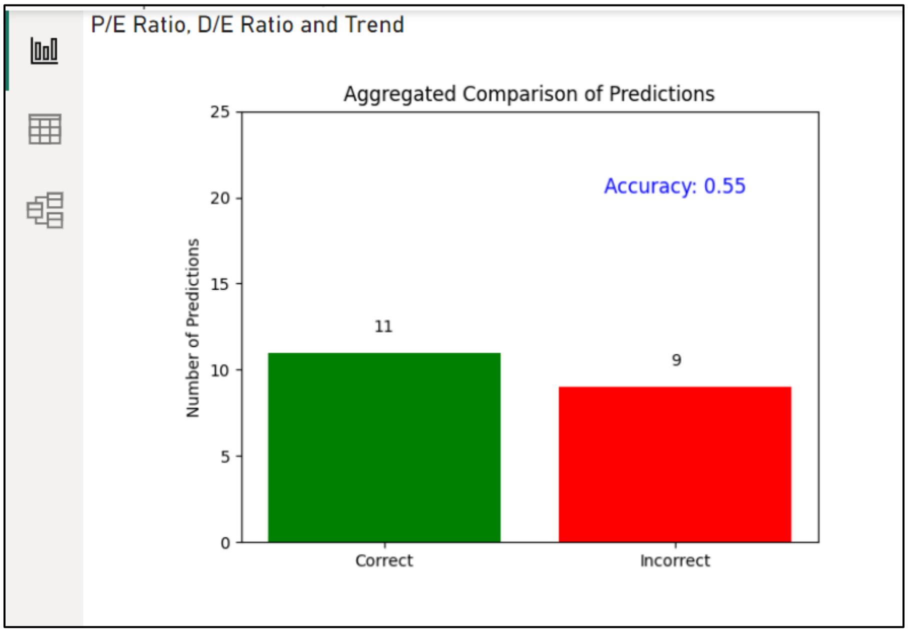
Click the “Run” button to execute the script:



⚠ Duplicate rows will be removed from the data.

```
# Paste or type your script code here:
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

This script will perform the KNN analysis and then create a plot which shows the accuracy of the predictions:



Read through the script to see how it works (we work directly with Python in the machine learning class, IDC4022C). You can experiment with different values of k on this line (defaults to 5)

```
# Create a kNN classifier
knn = KNeighborsClassifier(n_neighbors=5)
```

You can also experiment with different train/test data allocations here by changing the test_size parameter (a percentage of the total data; defaults to 20% test data and 80% training data):

```
# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```