Tools used/download

(a) Web scraping
- Jupyter Notebook
- Python

(b) Database (PostgreSQL)
- To download Postgres (Version: 6.15)
    1. Go to the postgreSQL Website https://www.postgresql.org/
    2. Click on the 'Download' button located in the top menu
    3. Select the version of PostgreSQL you want to download
    4. Choose your operating system
    5. Follow the instructions to download and install PostgreSQL
- Extra reference: https://youtu.be/0n41UTkOBb0

(c) Dashboard (Power BI)
- To download Power BI (Version: 2.112.1161.0 64-bit (December, 2022))
    1. Go to the Microsoft Power BI website https://powerbi.microsoft.com/en-us/
    2. Click on "Get started for free' button
    3. Create a Microsoft account , if you are new user
    4. After logging in to your Microsoft account , click on the 'Download Power BI Desktop'
    5. Follow the on-screen instructions to complete the installation
- Extra reference :https://youtu.be/GT2NcTE6UEo

- To download ODBC driver
    1. Go to Connecting Power BI to PostgreSQL via ODBC Driver Website https://docs.devart.com/odbc/postgresql/?powerbi.htm
    2. Click on the "Using ODBC Driver" then "Installation", then select the operating system that is compatible with your laptop.

Source of data

(a) Historical and daily update of stock price
- Yahoo finance api (Python library)

(b) Financial metric (Dividend, Dividend yield, ROE , DPR,EPS)
- KLSE screener
- Url: https://www.klsescreener.com/v2/

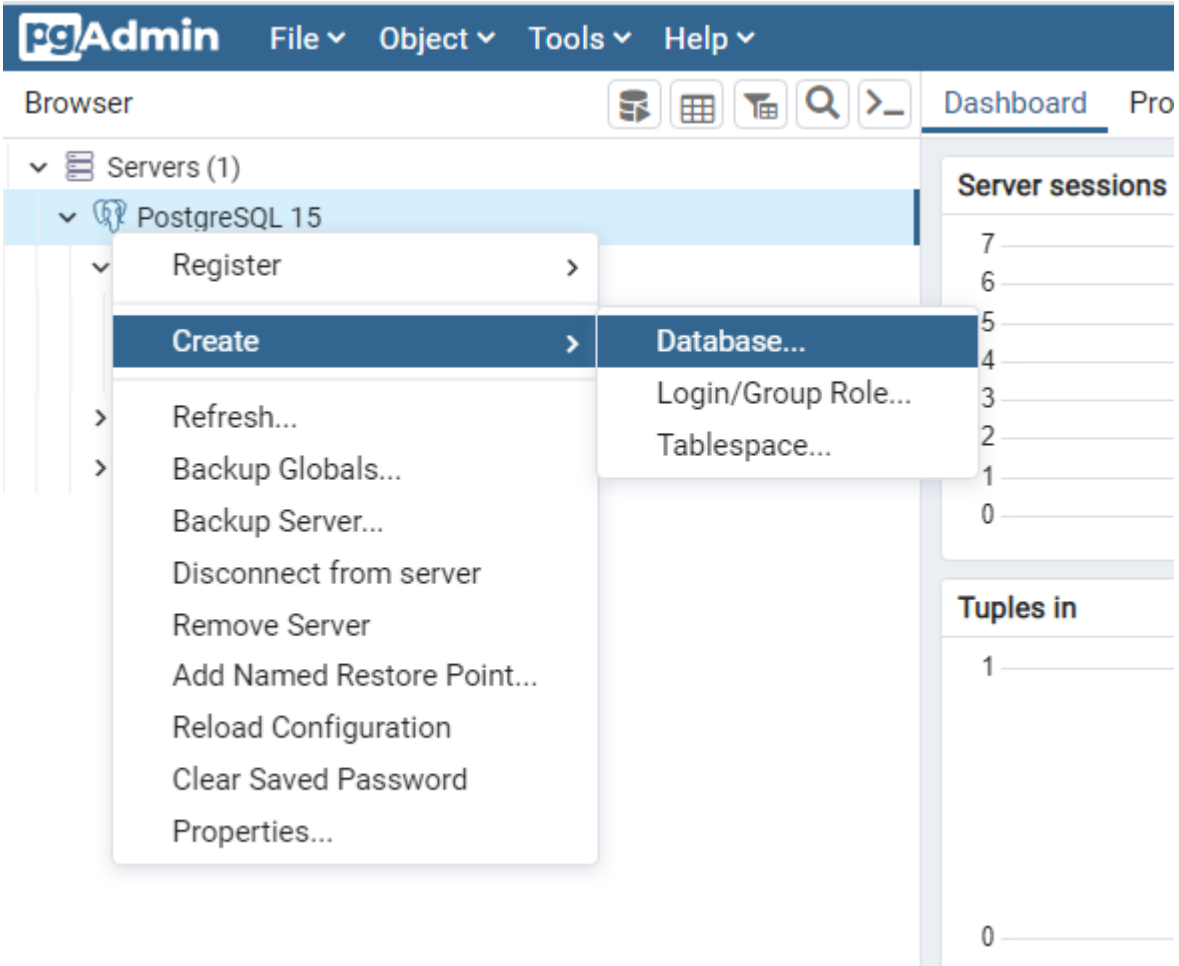
(c) Financial metric (P/CF, P/B, P/E, Dividend 5 year growth rate)
- Bursa Malaysia website
- Url : https://www.bursamarketplace.com/

Web scraping
1. Historical data
2. Latest data

3. Financial metrics

Create database using PostgreSQL

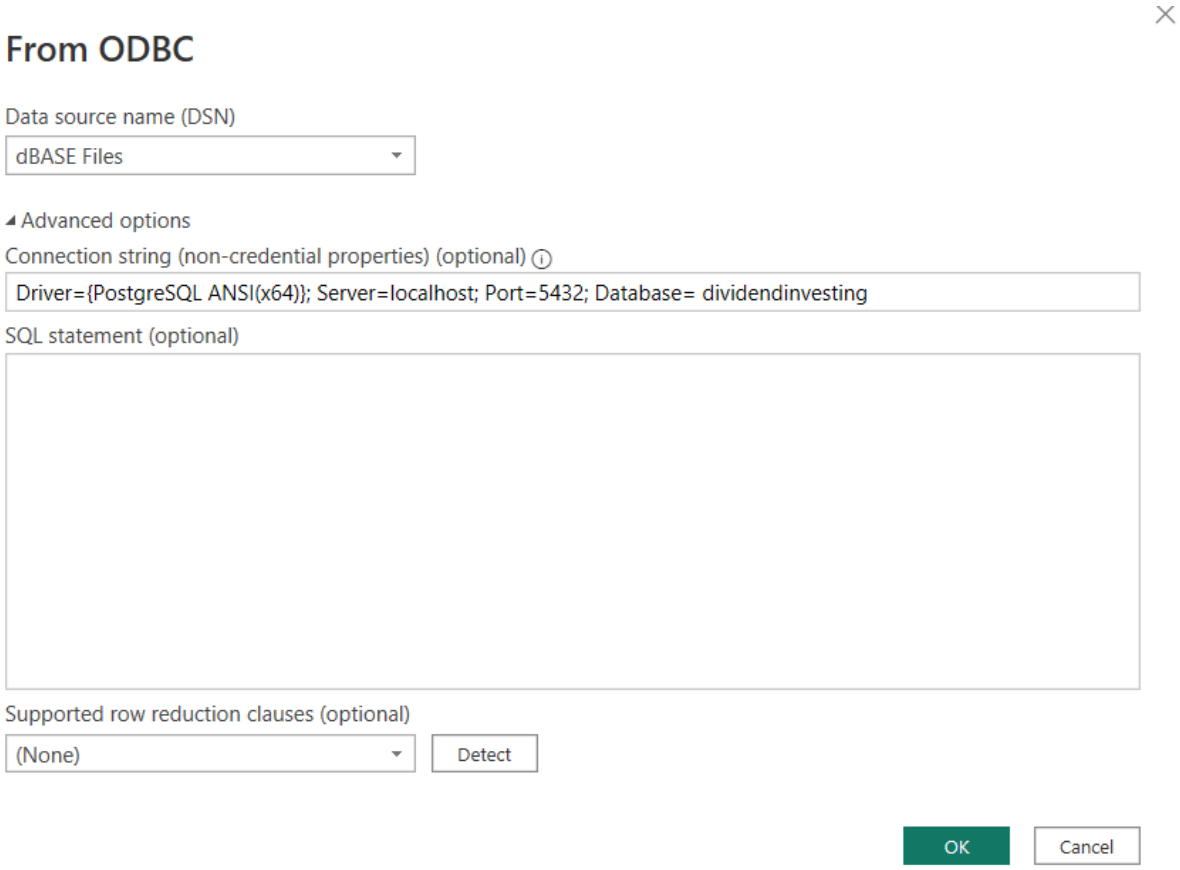
1. Create a database: You can do it either by code (CREATE DATABASE database_name;) or right click *"PostgreSQL 15"*, *"Create"*, then *"Database"*.



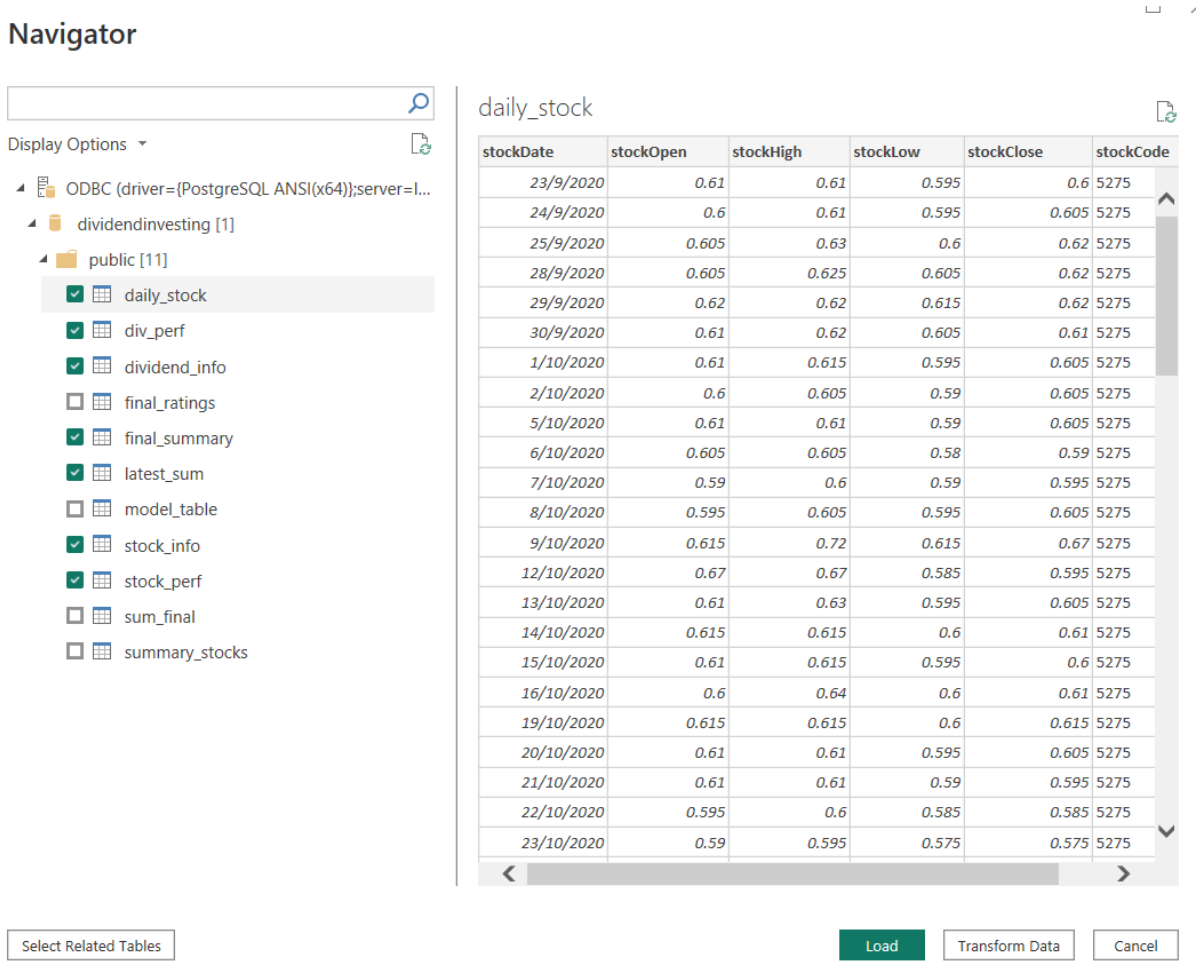
2. Create tables: Since we are developing this project using python, all our tables are imported from Jupyter Notebook into our database.

Create dashboard using PowerBI

Connect to data: Before you can create a dashboard, you need to get the data you want to work with. To do this, you need to select "Get Data" from the Home tab of PowerBi Desktop, and then select the appropriate data source (such as Excel, SQL Server, SharePoint or other data source). In this project, you will need to connect to ODBC, which is a C programming language interface that allows applications to connect to various database management systems (DBMSs).



Expand the advanced options and put in the connection string, Driver={PostgreSQL ANSI(x64)}; Server=localhost; Port=5433; Database= my_database. You will need to change to your own port number and the Database name.



stockDate	stockOpen	stockHigh	stockLow	stockClose	stockCode
23/9/2020	0.61	0.61	0.595	0.6	5275
24/9/2020	0.6	0.61	0.595	0.605	5275
25/9/2020	0.605	0.63	0.6	0.62	5275
28/9/2020	0.605	0.625	0.605	0.62	5275
29/9/2020	0.62	0.62	0.615	0.62	5275
30/9/2020	0.61	0.62	0.605	0.61	5275
1/10/2020	0.61	0.615	0.595	0.605	5275
2/10/2020	0.6	0.605	0.59	0.605	5275
5/10/2020	0.61	0.61	0.59	0.605	5275
6/10/2020	0.605	0.605	0.58	0.59	5275
7/10/2020	0.59	0.6	0.59	0.595	5275
8/10/2020	0.595	0.605	0.595	0.605	5275
9/10/2020	0.615	0.72	0.615	0.67	5275
12/10/2020	0.67	0.67	0.585	0.595	5275
13/10/2020	0.61	0.63	0.595	0.605	5275
14/10/2020	0.615	0.615	0.6	0.61	5275
15/10/2020	0.61	0.615	0.595	0.6	5275
16/10/2020	0.6	0.64	0.6	0.61	5275
19/10/2020	0.615	0.615	0.6	0.615	5275
20/10/2020	0.61	0.61	0.595	0.605	5275
21/10/2020	0.61	0.61	0.59	0.595	5275
22/10/2020	0.595	0.6	0.585	0.585	5275
23/10/2020	0.59	0.595	0.575	0.575	5275

In the Navigator window, you need to load all the relevant datasets to Power BI. With all datasets loaded, you can now create reports using the visualisations provided by Power BI.

Project Information

Name: Dividend Investing

Purpose:

1. Build a dashboard to present and analyse the financial metrics to identify stocks that would be good investments for dividends
2. Create models to forecast the future dividend performance of a stock

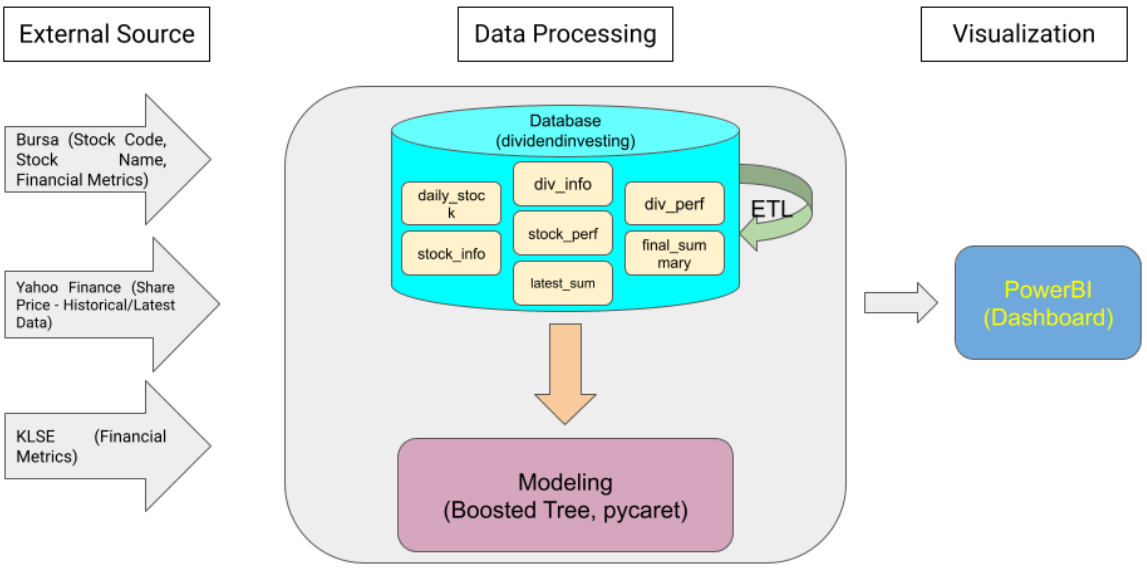
Schedule:

Project Schedule

Step 1	Get stock code and stock name - (2 days)
Step 2	Get historical stock price - (5 days)
Step 3	Obtain latest stock price - (3 days)
Step 4	Get financial metrics (KLSE and Bursa Malaysia) - (6 days)
Step 5	Perform data cleaning/formatting and joining (ETL) - (10 days)
Step 6	Stored it into database - (3 days)
Step 7	Build a dashboard in Power BI - (3 days)
Step 8	Build predictive model - (4 days)

Architecture of project:

Architecture of the project



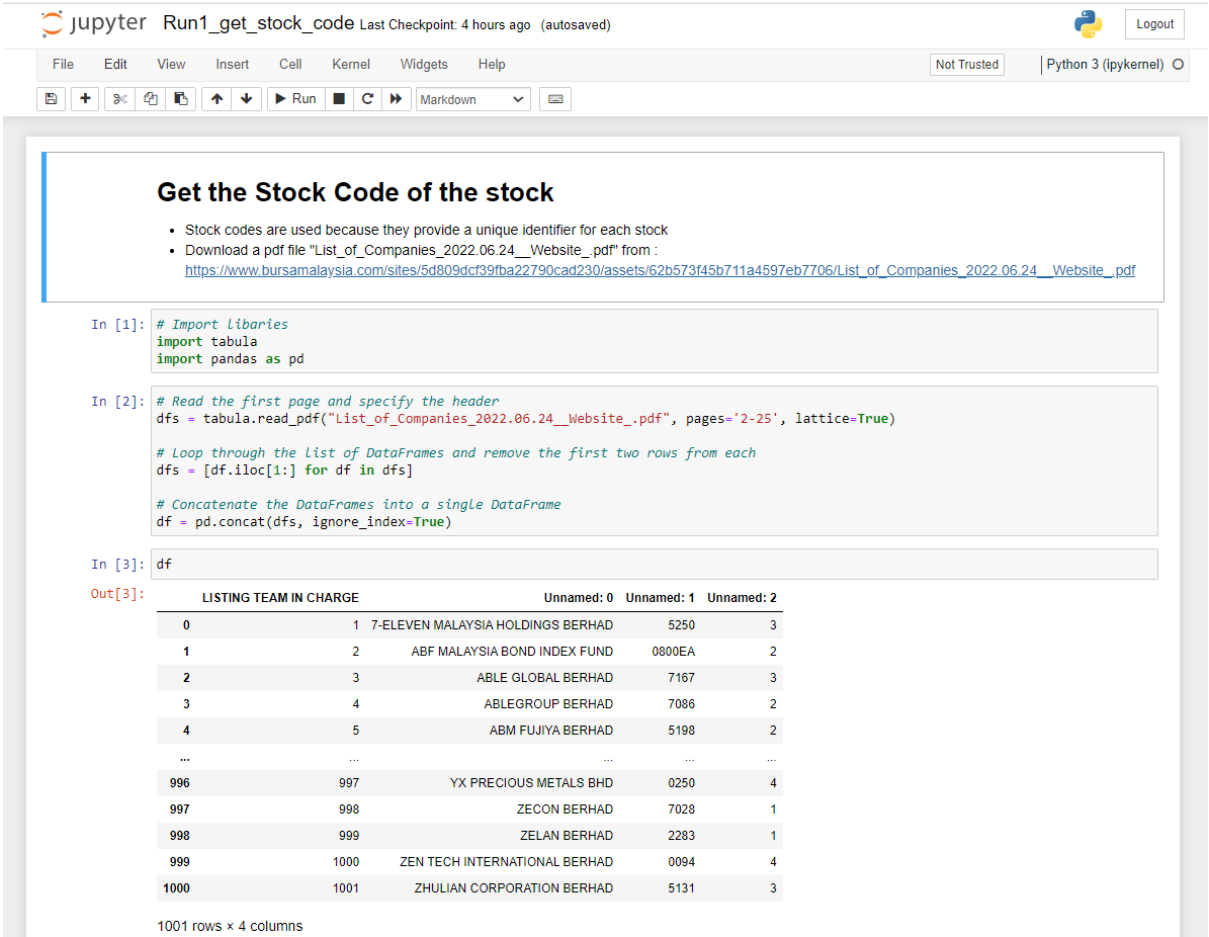
Complete steps of developing the project (Dividend Investing)

Step 1: Get stock code and stock name

First, download "List_of_Companies_2022.06.24__Website_.pdf" from Bursa Malaysia, where the PDF contains a complete list of stock codes and stock names. Next, read the PDF and grab the tables using a python library called "tabula". After grabbing all the information, combine the stock code and stock name into one dataframe.

Problem: Some stock codes are duplicated or delisted.

Solution: Use the unique function to detect duplicates and manually remove duplicates. For delisted stocks, you may encounter delisted stock errors when running historical stock price data, and you need to run another loop to update the latest stock code.



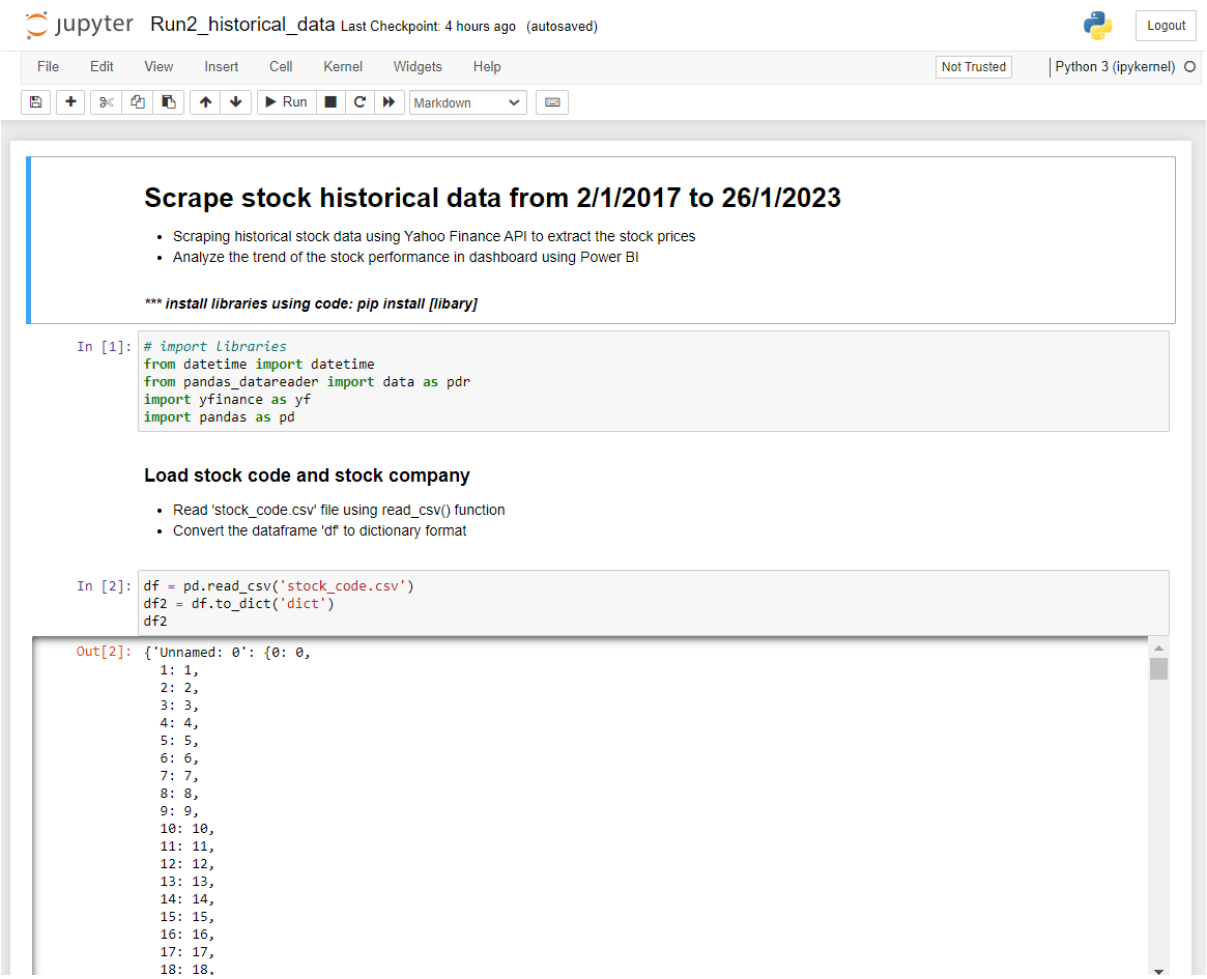
Run1_get_stock_code.ipynb

Step 2: Get historical stock price

The purpose of collecting historical stock prices is to understand the trend of a company's stock price over the years. For this project, we will use the "yfinance" API to scrape each company's share price from 2017 to 2023. Scrapping historical stock prices is a one-time event.

Problem: Not all historical stock price data can be scraped at one time

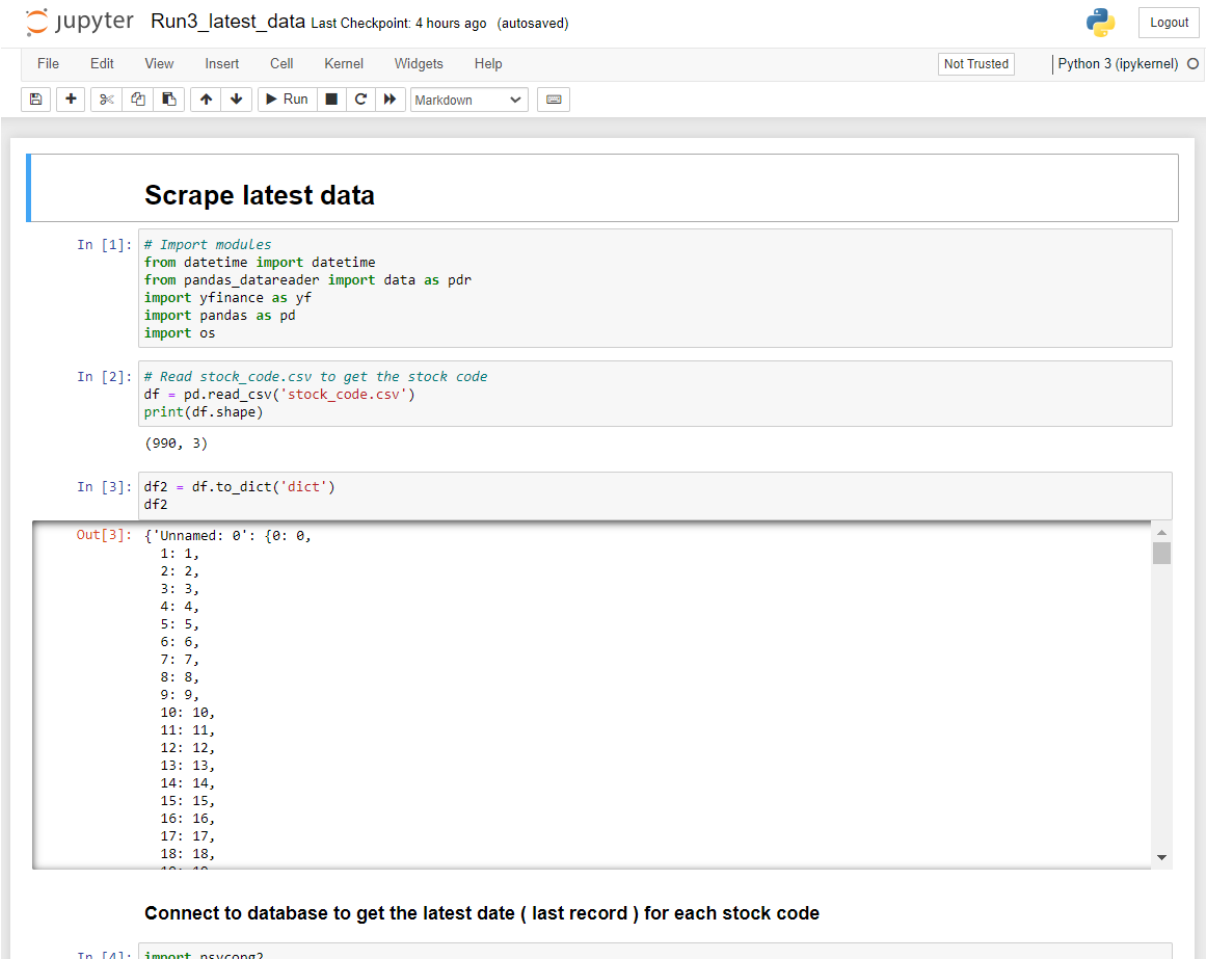
Solution: Save the missing stock price data in a list and run a second loop



Run2_historical_data.ipynb

Step 3: Get latest stock price

The stock price for each stock must be scraped daily so that the data is up to date. With the latest stock price, we can know the performance of the stock for the day.



Run3_latest_data.ipynb

Step 4: Obtain dividend information

In this step, we will scrape dividend information from KLSE. Dividend information can be found separately in the annually and quarterly tables, where it raises potential issues when crawling the information.

Question: The tables are numbered and different financial indicators can be found in different tables.

Solution: Identify unique columns for each table

After grabbing all the dividend information, we use this information to calculate the dividend yield for each stock.

jupyter Run4_WebScraping(div_info and div_yield) Last Checkpoint: 4 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Markdown

Scrape dividend info from KLSE and calculate dividend yield

- Scrape dividend information (Financial Year, Ex.Date, Announced Date, Amount) from KLSE website
- https://www.klsescreener.com/v2/stocks/view/7152/jaycorp-bhd

In [2]:
```
# Import libraries
from selenium import webdriver
from bs4 import BeautifulSoup as bs
from datetime import datetime
import pandas as pd
import time
import random
import warnings
warnings.filterwarnings("ignore")
```

Get stock code

In [2]:
```
# Get the stock code
df = pd.read_csv('stock_code.csv')
df
```

Out[2]:

	Unnamed: 0	stock_name	stock_code
0	0	7-ELEVEN MALAYSIA HOLDINGS BERHAD	5250.KL
1	1	ABF MALAYSIA BOND INDEX FUND	0800EA.KL
2	2	ABLE GLOBAL BERHAD	7167.KL
3	3	ABLEGROUP BERHAD	7086.KL
4	4	ABM FUJIYA BERHAD	5198.KL
...	...	...	...
985	985	YX PRECIOUS METALS BHD	0250.KL
986	986	ZECON BERHAD	7028.KL
987	987	ZELAN BERHAD	2283.KL
988	988	ZEN TECH INTERNATIONAL BERHAD	0094.KL
989	989	ZHULIAN CORPORATION BERHAD	5131.KL

990 rows × 3 columns

Run4_WebScraping(div_info and div_yield).ipynb

Step 5: Scrape quarter report from KLSE

On the KLSE website, there is a quarterly table that reports the financial year, EPS, DPS, and ROE for each quarter of the stock. The information will be used to determine the financial performance of each company for each quarter throughout the years. We performed a loop for all the stock codes and used the “beautiful soup” library to get the tables. The ROE metric can only be found in the quarterly table, hence we will use that as our guide to find the quarterly table and extract the information we need. If quarterly tables cannot be found, the list of stock code will be appended into the no_quarter_list, so that we can perform a second loop to obtain all the information.

jupyter Run5_WebScraping(quarter_report) Last Checkpoint: 3 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Scrape quarter report from KLSE

- Scrape quarter report information (EPS,DPS,Financial Year,ROE) from KLSE website
- https://www.klsescreener.com/v2/stocks/view/7152/jaycorp-bhd

In [1]:
```
# Import Libraries
from selenium import webdriver
from bs4 import BeautifulSoup as bs
from datetime import datetime
import pandas as pd
import time
import random
import warnings
warnings.filterwarnings("ignore")
```

Get stock code

In [2]:
```
df = pd.read_csv('stock_code.csv')
df
```

Out[2]:

	Unnamed: 0	stock_name	stock_code
0	0	7-ELEVEN MALAYSIA HOLDINGS BERHAD	5250.KL
1	1	ABF MALAYSIA BOND INDEX FUND	0800EA.KL
2	2	ABLE GLOBAL BERHAD	7167.KL
3	3	ABLEGROUP BERHAD	7086.KL
4	4	ABM FUJIYA BERHAD	5198.KL
...	...	...	...
985	985	YX PRECIOUS METALS BHD	0250.KL
986	986	ZECON BERHAD	7028.KL
987	987	ZELAN BERHAD	2283.KL
988	988	ZEN TECH INTERNATIONAL BERHAD	0094.KL
989	989	ZHULIAN CORPORATION BERHAD	5131.KL

990 rows × 3 columns

Run5_WebScraping(quarter_report).ipynb

Step 6: Scrape the annual report from KLSE

In this step, we will perform a similar process to the previous step to extract information from the annual table on the KLSE website. The financial metrics we will extract from the annual table are the financial year, EPS, and DP%. Similar to step 5 of finding quarterly statements, where DP% is used as a guide to find the annual table. If no annual table is found, the list of stock code will be appended into the no_annual_list and a second loop will be performed to extract all the information for each stock code.

jupyter Run6_WebScraping(annual_report) Last Checkpoint: 8 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Scrape annual report from KLSE

-scrape quarter report information (EPS,Financial Year,DP%) from KLSE website

- https://www.klsescreener.com/v2/stocks/view/7152/jaycorp-bhd

Import libraries

In [1]:
```
from selenium import webdriver
from bs4 import BeautifulSoup as bs
from datetime import datetime
import pandas as pd
import time
import random
import warnings
warnings.filterwarnings("ignore")
```

Get stock code

In [2]:
```
df = pd.read_csv('stock_code.csv')
df
```

Out[2]:

	Unnamed: 0	stock_name	stock_code
0	0	7-ELEVEN MALAYSIA HOLDINGS BERHAD	5250.KL
1	1	ABF MALAYSIA BOND INDEX FUND	0800EA.KL
2	2	ABLE GLOBAL BERHAD	7167.KL
3	3	ABLEGROUP BERHAD	7086.KL
4	4	ABM FUJIYA BERHAD	5198.KL
...	...	...	...
985	985	YX PRECIOUS METALS BHD	0250.KL
986	986	ZECON BERHAD	7028.KL
987	987	ZELAN BERHAD	2283.KL
988	988	ZEN TECH INTERNATIONAL BERHAD	0094.KL
989	989	ZHULIAN CORPORATION BERHAD	5131.KL

Run6_WebScraping(annual_report).ipynb

Step 7&8: Scrape financial metrics from Bursa

Both of these steps scraped the financial metrics such as PE, PB, PCF and 5-year dividend growth rate from the Bursa website. The reason we are grabbing the two financial metrics separately in two notebooks is because we want to reduce errors in extracting information, which can be time consuming. If the financial metrics/urls of the stock code cannot be scraped/found, it will print an error message with which financial metrics/urls of the stock code could not be scraped. The missing urls of the stock code will be appended into the list_url.

jupyter Run7_WebScraping(pepb) Last Checkpoint: 02/13/2023 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Scrape financial metrics from Bursa

-scrape Price per earnings(pe) and Price to book value(pb) from Bursa Malaysia

- https://www.bursamarketplace.com/mkt/themarket/stock/JAYC

Import libraries

In [1]:
```
from selenium import webdriver
from bs4 import BeautifulSoup as bs
from datetime import datetime
import pandas as pd
import time
import random
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.common.exceptions import NoSuchElementException
```

In [2]:
```
df = pd.read_csv('stock_code.csv')
```

In [3]:
```
# Remove the last three character
df['stock_code']=df['stock_code'].str.slice(0,-3)
```

In [4]:
```
# Put it into a list called stock_code
stock_code=list(df['stock_code'])
stock_code
```

Out[4]:
```
['5250',
 '0800EA',
 '7167',
 '7086',
 '5198',
 '03028',
 '7131',
 '0218',
 '0122',
 '1481',
 '5281',
 '9148',
 '7191',
 '7146',
```

Run7_WebScraping(pepb).ipynb

jupyter Run8_WebScraping(pcf_div_growth) Last Checkpoint: Last Monday at 10:44 AM (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Scrape financial metrics from Bursa

-scrape Price to cash flow(div_pcf) and 5 Year dividend growth rate(div_5year_growth) from Bursa Malaysia

- https://www.bursamarketplace.com/mkt/themarket/stock/JAYC/financials

Import libraries

In [2]:
```
from selenium import webdriver
from bs4 import BeautifulSoup as bs
from datetime import datetime
import pandas as pd
import time
import random
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.common.exceptions import NoSuchElementException
```

In [2]:
```
df = pd.read_csv('stock_code.csv')
```

In [3]:
```
df['stock_code']=df['stock_code'].str.slice(0,-3)
```

In [4]:
```
df
```

Out[4]:

	stock_company	stock_code
0	7-ELEVEN MALAYSIA HOLDINGS BERHAD	5250
1	ABF MALAYSIA BOND INDEX FUND	0800EA
2	ABLE GLOBAL BERHAD	7167
3	ABLEGROUP BERHAD	7086
4	ABM FUJIYA BERHAD	5198
...	...	...
985	YX PRECIOUS METALS BHD	0250
986	ZECON BERHAD	7028
987	ZELAN BERHAD	2283
988	ZEN TECH INTERNATIONAL BERHAD	0094
989	ZHULIAN CORPORATION BERHAD	5131

Run8_WebScraping(pcf_div_growth).ipynb

Step 9: Define the definitions and scoring all the financial metrics

We define definitions and scores for each financial metric we use. This is to give the reader an understanding of the purpose of each financial metric and how each stock is performing according to the score.

Definition of financial metrics

Financial Metrics	Definitions
Dividend Yield (DY)	Annual dividend per share/Current Share price
Dividend Payout Ratio (DPR)	Annual dividend/Net income attributed to common shareholders
Earning per share (EPS)	Net income of the company/Average outstanding share of the company
Return on equity (ROE)	Annual net income/Shareholders equity
Price to book value (PB)	Market price per share/Book value per share
Price to earning ratio (PE)	Stock's current price/Latest EPS
Price to cash flow (P/CF)	Market capitalization/Cash flow from operations
Growth rate (GR)	(Latest Dividend/Initial Dividend)^1/n - 1, where n is the period

Scoring for all the financial metrics

Financial Metrics	Scoring				
	1	2	3	4	5
DY	< 1	1 ≤ x ≤ 3	3 ≤ x ≤ 6	6 ≤ x ≤ 10	> 10
DPR	< 10 or > 90	70 < x ≤ 90	10 ≤ x < 20	50 < x ≤ 70	20 ≤ x ≤ 50
EPS	< 20	20 ≤ x < 40	40 ≤ x < 60	60 ≤ x < 90	≥ 80
ROE	< 20	20 ≤ x < 50	50 ≤ x < 70	70 ≤ x < 100	≥ 100
PB	> 2.5	1.5 < x ≤ 2.5	1 < x ≤ 1.5	0.5 < x ≤ 1	0 ≤ x ≤ 0.5
PE	> 70	50 < x ≤ 70	25 < x ≤ 50	20 ≤ x ≤ 25	< 20
P/CF	>100	20 < x ≤ 100	10 < x ≤ 20 or > 0	0.75 < x ≤ 10	0 ≤ X ≤ 0.75
GR	≤ -25	-25 < x ≤ 0	0 < x ≤ 25	25 < x ≤ 50	> 50

Step 10: Create models to predict the top 10 stock code for dividend investing

In this step, we will create a model such as boosted tree regression and use pycaret, a python library that automates the machine learning workflow and helps detect the best model for the data. First, we will use the data from 2018 to 2021 as training data and test data, while the data from 2022 to the latest will be used to predict the top 10 stock codes that are most suitable for dividend investing. DY, DPR, EPS, and ROE will be used as predictor variables, and from here they will predict the overall score for each stock.

For the Boosted Tree regression model, we split the data into 70% as training data and 30% as testing data. Next, we fit the training data to the GradientBoostingRegressor function with a random state of 42. Finally, we tested the model accuracy using the test data and got a pretty good result with MSE = 0.07 and an R-squared score close to 1, 0.99. We use the model to predict the top 10 stock codes due to its good accuracy.

For another model, we split the data into 90% as training data and 10% as testing data. After setting up the environment using the setup function by defining all the parameters, it will compare which models will produce the lowest error score and the highest R2 score. It can be seen that the catboost model produced the best results, so it was used to predict the top 10 stocks.

jupyter Run9_Modeling(boosted_tree_regression) Last Checkpoint: a day ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Code

Create a model that predicts the top 10 stock code for dividend investing

- Using boosted tree regression to test the accuracy and compare it with another model identified by pycaret

In [1]:
```
# Import libraries
import numpy as np
import pandas as pd
import psycopg2
from sqlalchemy import create_engine
from urllib.parse import quote
from datetime import timedelta
```

In [2]:
```
# Connect to database
conn_string = 'postgresql://postgres:Jiawei1105@localhost/dividendinvesting'

#connect
db = create_engine(conn_string)
conn = db.connect()
conn = psycopg2.connect(conn_string)
conn.autocommit = True
cursor = conn.cursor()
```

Create a table, 'train_test_tab' in database where it will contains data starting from Year 2018 to 2021 for each stock code and it will be used to split it into training and test data

In [3]:
```
# Using sql query to get recent date
sql2 ='''SELECT "sum_stockCode", sum_year, sum_status, "sum_dividend(RM)", "sum_averageDividend_y_%", "sum_averageAnnual_dp(%)",
FROM train_test_tab;;'''
cursor.execute(sql2)
train_test_table = cursor.fetchall()
df = pd.DataFrame(train_test_table, columns =["sum_stockCode", 'sum_year', 'sum_status', "sum_dividend(RM)", "sum_averageDividenc
conn.commit()
conn.close()
```

In [4]:
```
df
```

Out[4]:

	sum_stockCode	sum_year	sum_status	sum_dividend(RM)	sum_averageDividend_y_%	sum_averageAnnual_dp(%)	sum_averageQuar_eps	sum_averageQu
0	0001	2018	Y	0.0150	1.650	0.0	0.021	

Run9_Modeling(boosted_tree_regression).ipynb

jupyter Run10_Modeling(pycaret_catboost) Last Checkpoint: Last Monday at 14:57 (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Create a model that predicts the top 10 stock code for dividend investing

- Use pycaret to identify which algorithm has the lowest errors and also the highest R2 score
- pycaret is a python library that helps to identify which algorithms will create a model that best fits the data

In [1]:
```
# Import libraries
import pycaret as pc
import psycopg2
import pandas as pd
from sqlalchemy import create_engine
import numpy as np
from pycaret.regression import *
from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

In [2]:
```
# Connect to database
conn_string = 'postgresql://postgres:Jiawei1105@localhost/dividendinvesting'

# Connect
db = create_engine(conn_string)
conn = db.connect()
conn = psycopg2.connect(conn_string)
conn.autocommit = True
cursor = conn.cursor()
```

Using the data in the 'train_test_tab' table to split the data into training and test data

In [3]:
```
# Using sql query
sql ='''SELECT "sum_stockCode", sum_year, sum_status, "sum_dividend(RM)", "sum_averageDividend_y_%", "sum_averageAnnual_dp(%)", '
FROM train_test_tab;;'''
cursor.execute(sql)
train_test_tab = cursor.fetchall()
data = pd.DataFrame(train_test_tab, columns =["sum_stockCode", 'sum_year', 'sum_status', "sum_dividend(RM)", "sum_averageDividenc
conn.commit()
conn.close()
```

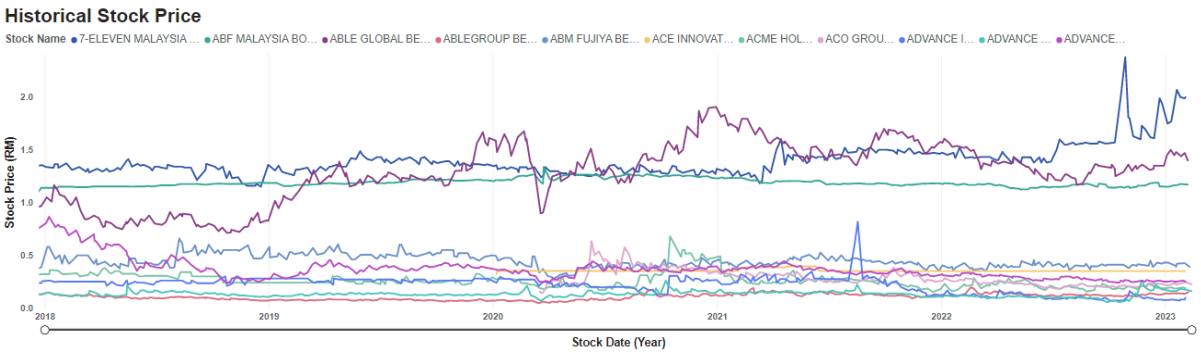
In [4]:
```
data
```

Out[4]:

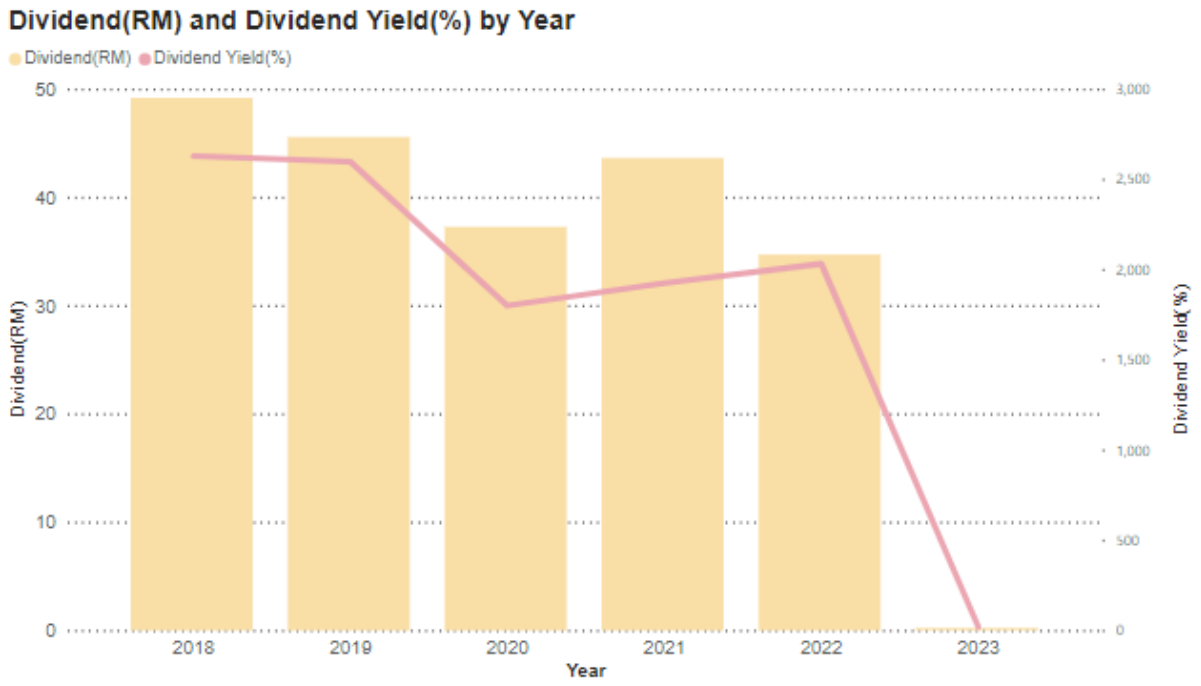
Run10_Modeling(pycaret_catboost).ipynb

Step 11: Create a dashboard to show the result

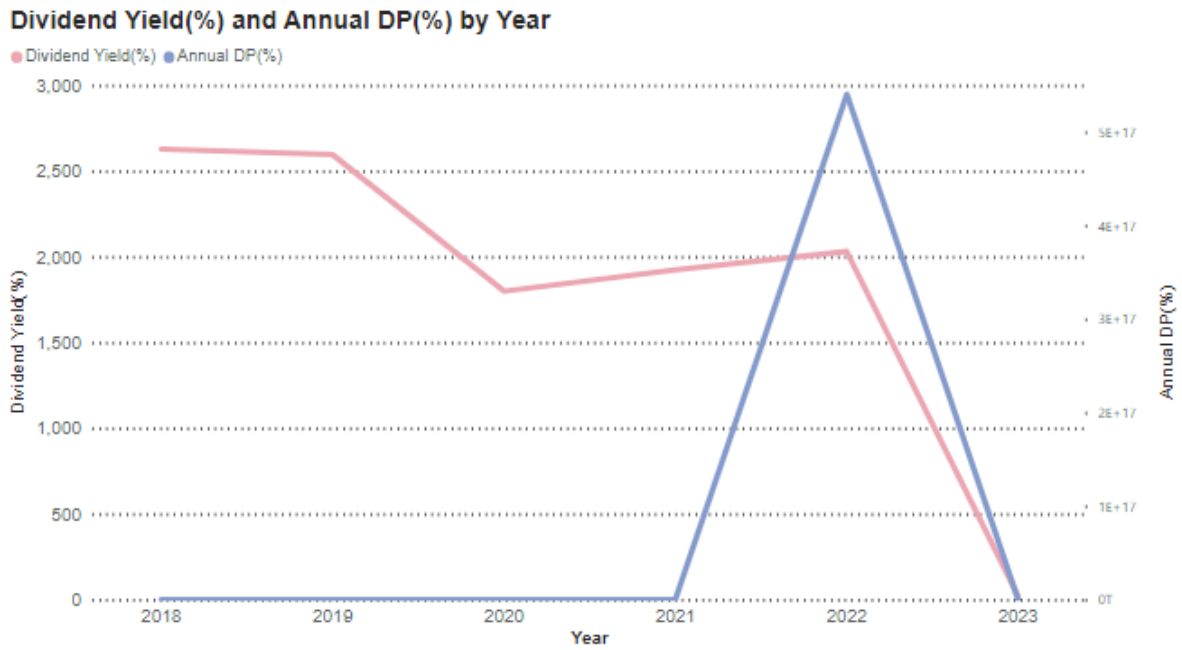
With everything done, we will now create a dashboard with the data we got. The first is to create a line chart to show the stock price trend from 2018 to the latest data we have available.



Next, create a line column chart where the bars represent the dividends paid by the company and the line represents the dividend yield in percentage.



Create a line chart where the first y-axis is the dividend yield and the second y-axis is the dividend payout ratio. The chart shows how much dividends a company pays each year compared to its share price, and how much a company is willing to pay investors after reporting its net income.



A stock summary table was created to show financial metrics for each stock, including whether the company paid dividends to its investors in that particular year.

Summary of Stocks

Stock Code	Dividend Status	Dividend(RM)	Dividend Yield(%)	DPR(%)	Earning per Share(EPS)	ROE(%)
⊟ 0001						
2018	Y	0.02	1.65	0.00	0.02	10.90
2019	Y	0.02	1.16	51.00	0.03	9.50
2020	Y	0.02	0.97	44.00	0.03	10.10
2021	Y	0.02	1.34	44.00	0.03	8.70
2022	Y	0.01	0.28	57.00	0.04	8.30
⊟ 0002						
2018	Y	0.05	3.40	0.00	0.12	10.50
2019	Y	0.07	4.31	0.00	0.16	13.70
2020	Y	0.09	3.85	0.00	0.21	16.50
2021	Y	0.09	3.17	0.00	0.17	12.20
2022	Y	0.26	5.42	23.00	0.42	27.00
⊞ 0005	N	0.00	0.00	0.00		
⊞ 0006	N	0.00	0.00	0.00		
⊞ 0007	N	0.00	0.00	0.00		
⊟ 0008						
2018	Y	0.01	2.45	37.00	0.03	7.40
2019	Y	0.02	3.24	49.00	0.03	9.30
2020	Y	0.02	3.02	41.00	0.04	10.50
2021	Y	0.02	3.85	48.00	0.03	8.50
⊞ 0010	N	0.00	0.00	0.00		
⊟ 0011						
2018	Y	0.02	7.45	97.00	0.02	7.40
2019	Y	0.02	6.61	69.00	0.02	9.70
2020	Y	0.02	3.53	86.00	0.02	7.40
2021	Y	0.01	2.62	24.00	0.03	11.50
⊟ 0012						
2018	Y	0.02	2.74	34.00	0.06	9.00
2019	Y	0.02	2.79	33.00	0.06	8.50
2020	Y	0.02	2.73	36.00	0.06	8.30
2021	Y	0.03	3.02	32.00	0.10	11.90

From the summary of the stocks table, we convert the values into the scores we defined earlier and insert an overall score column to see the scores for each stock code for each year.

Final Ratings for all the stocks

Stock Code	Dividend Status	DY Score	DPR Score	EPS Score	ROE Score	Overall Score
⊟ 0001						
2018	Y	2.00	1.00	1.00	1.00	5.00
2019	Y	2.00	4.00	1.00	1.00	8.00
2020	Y	1.00	5.00	1.00	1.00	8.00
2021	Y	2.00	5.00	1.00	1.00	9.00
2022	Y	1.00	4.00	1.00	1.00	7.00
⊟ 0002						
2018	Y	3.00	1.00	1.00	1.00	6.00
2019	Y	3.00	1.00	1.00	1.00	6.00
2020	Y	3.00	1.00	2.00	1.00	7.00
2021	Y	3.00	1.00	1.00	1.00	6.00
2022	Y	3.00	5.00	3.00	2.00	13.00
⊞ 0005	N	1.00	1.00	0.00	0.00	2.00
⊞ 0006	N	1.00	1.00	0.00	0.00	2.00
⊞ 0007	N	1.00	1.00	0.00	0.00	2.00
⊟ 0008						
2018	Y	2.00	5.00	1.00	1.00	9.00
2019	Y	3.00	5.00	1.00	1.00	10.00
2020	Y	3.00	5.00	1.00	1.00	10.00
2021	Y	3.00	5.00	1.00	1.00	10.00
⊞ 0010	N	1.00	1.00	0.00	0.00	2.00
⊞ 0011	Y	3.25	3.00	1.00	1.00	8.25
⊞ 0012	Y	2.20	5.00	1.00	1.00	9.20
⊟ 0017						
2018	N	1.00	1.00	0.00	0.00	2.00
2019	N	1.00	1.00	0.00	0.00	2.00
2020	N	1.00	1.00	0.00	0.00	2.00
2021	N	1.00	1.00	0.00	0.00	2.00
2022	N	1.00	1.00	0.00	0.00	2.00
⊞ 0018	N	1.00	1.00	0.00	0.00	2.00
⊞ 0020	N	1.00	1.00	0.00	0.00	2.00
⊞ 0021	Y	2.00	1.00	1.00	1.00	5.00

Since we scraped financial metrics from Bursa Malaysia separately in steps 7 and 8, we will now concatenate them together to create subsequent tables so that users can see the latest financial metrics immediately.

jupyter Run11_(Latest_Summary) Last Checkpoint: 18 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Create latest summary table which contain all the financial metrics for stock from 2022 to current

In [1]:
```
import pandas as pd
```

In [2]:
```
df= pd.read_csv('Share.csv')
df
```

Out[2]:

	Unnamed: 0	stock_code	pcf	div_5year_growth
0	0	5250	9.620	3.211
1	1	0800EA	36.661	0.000
2	2	7167	8.926	27.226
3	3	7086	122.935	0.000
4	4	5198	7.515	0.000
...	...	...	...	...
983	983	2283	13.774	0.000
984	984	0094	3.792	0.000
985	985	5131	18.297	14.870
986	986	7016	5.352	-0.448
987	987	7139	36.661	0.000

988 rows × 4 columns

In [3]:
```
del df['Unnamed: 0']
```

In [4]:
```
df[df['stock_code']=='03006']
```

Out[4]:

	stock_code	pcf	div_5year_growth
600	03006	50.013	0.000

In [5]:
```
df2 = pd.read_csv('pepb.csv')
```

In [6]:
```
df2
```

Run11_(Latest_Summary).ipynb

Finally, a table with the latest annual metrics (including add-on metrics, DPS, P/B, P/CF, P/E, Dividend 5-year growth rate) are converted to a score to show their total score.

Latest Year Metrics

Stock Code	Year	Dividend Status	Dividend Yield(%)	DPR (%)	DPS(RM)	EPS	P/B	P/CF	P/E	Dividend 5 Year Growth Rate	ROE (%)
7247	2022	Y	463.70	40.00	1.91	0.16	0.63	2.87	0.41	1.36	15.40
5255	2022	Y	78.82	0.00	0.07	0.01	0.83	3.60	13.69	0.00	4.60
6645	2022	Y	74.77	82.00	0.25	0.31	0.23	1.08	1.23	-4.37	13.20
5673	2022	Y	56.60	1.00	0.30	0.47	3.71	10.56	2.97	0.00	34.40
7229	2022	Y	40.67	9.00	0.85	0.10	0.57	4.84	11.91	-11.81	3.70
5077	2022	Y	27.78	79.00	0.10	0.08	0.78	1.51	3.14	0.00	14.70
8044	2022	Y	25.97	45.00	0.20	0.44	11.14	11.73	0.00	0.00	139.10
3336	2022	Y	13.85	95.00	0.21	0.22	0.53	12.43	34.76	-4.37	7.60
5254	2022	Y	13.52	49.00	0.11	0.23	0.56	4.03	2.57	-10.45	16.70
5168	2022	Y	13.08	77.00	0.57	0.95	1.04	1.54	0.00	68.07	48.70
3514	2022	Y	12.12	142.00	0.02	0.01	0.84	9.73	9.80	0.00	6.30
5121	2022	Y	11.89	38.00	0.08	0.07	0.61	24.78	0.00	-24.44	6.10
2542	2022	Y	11.72	88.00	0.40	0.23	0.60	8.67	9.85	-7.79	3.90
9288	2022	Y	10.90	85.00	0.19	0.22	1.40	6.84	8.69	11.56	11.70
7106	2022	Y	10.66	39.00	0.11	0.28	0.45	2.64	25.10	53.26	14.90
6262	2022	Y	10.61	102.00	0.16	0.16	2.35	7.02	7.06	55.19	23.50
5185	2022	Y	10.52	0.00	0.23	0.54	0.48	6.93	3.24	-15.22	11.10

Latest Year Ratings

Stock Code	Year	Dividend Status	DY Score	DPR Score	EPS Score	P/B Score	P/CF Score	P/E Score	GR Score	ROE Score	Overall Score
5168	2022	Y	5	2	5	3	4	5	5	2	31
1163	2022	Y	4	3	5	4	4	5	4	1	30
5012	2022	Y	4	4	4	4	4	5	4	1	30
7106	2022	Y	5	5	2	5	4	3	5	1	30
1082	2022	Y	2	5	5	4	4	5	3	1	29
1929	2022	Y	3	5	5	4	4	5	2	1	29
4006	2022	Y	2	5	5	4	4	5	3	1	29
5139	2022	Y	3	5	5	3	4	5	3	1	29
6009	2022	Y	5	5	2	4	4	5	3	1	29
8044	2022	Y	5	5	3	1	3	5	2	5	29
9059	2022	Y	4	5	2	4	4	5	3	2	29
1066	2022	Y	2	5	3	4	4	5	4	1	28
2127	2022	Y	3	3	3	5	5	5	2	2	28
2488	2022	Y	4	5	2	4	4	5	3	1	28
4383	2022	Y	3	5	1	5	4	5	4	1	28
5007	2022	Y	4	5	2	4	4	5	3	1	28
5135	2022	Y	4	5	2	4	4	5	3	1	28
5197	2022	Y	4	5	2	4	4	5	3	1	28