Activity #5

Support Vector Regression (SVR)

Activity #5

5.1

Data Preparation

5.2

Model Training and Cross Validation

5.3

Model Evaluation

TOPICS

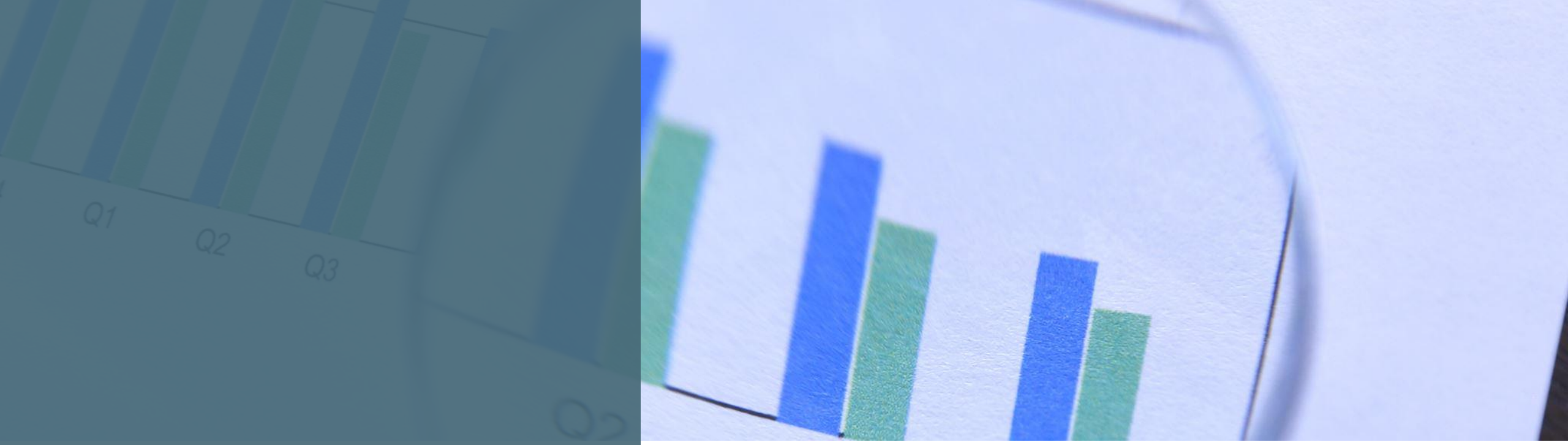
5.1 Data Exploration / Transform / Feature Selection / Time Series windows

5.2 Model Training and Cross Validation

5.3 Model Evaluation

LIBRARIES

1. import numpy as np
2. import pandas as pd
3. import matplotlib.pyplot as plt
4. import seaborn as sns
5. from sklearn.linear_model import LinearRegression
6. from sklearn.svm import SVR
7. from sklearn import preprocessing
8. from sklearn import metrics
9. import pandas_datareader.data as web



5.1 Data Preparation

5.1 (a) Get Data

1

- #Read stock data use pandas_datareader.data from web
  - # Get Stock Data
  - stk_tickers = ['MSFT', 'IBM', 'GOOGL']
  - ccy_tickers = ['DEXJPUS', 'DEXUSUK']
  - idx_tickers = ['SP500', 'DJIA', 'VIXCLS']
  - stk_data = web.DataReader(stk_tickers, 'yahoo')
  - ccy_data = web.DataReader(ccy_tickers, 'fred')
  - idx_data = web.DataReader(idx_tickers, 'fred')

2

- Select columns
  - Base = stk_data.loc[:, ('Adj Close', 'MSFT')]
  - X1 = stk_data.loc[:, ('Adj Close', ('GOOGL', 'IBM'))]
  - X2 = ccy_data
  - X3 = idx_data

5.1 (b) Standardized Data

1

- #Standardized data (X1, X2, X3) with kept index (date)
  - X1 = standard_scaler.fit_transform(X1.values),index = X1.index, columns=X1.columns)

2

- Calculate ความแตกต่างของค่า ราคา 'Adj Close', 'MSFT') ย้อนหลัง return_period วัน
  - Y = base. shift(-return_period)
  - X4_3DT = base.diff(3*return_period).shift(-3*return_period)
  - X4_6DT = base.diff(6*return_period).shift(-6*return_period)
  - X4_12DT = base.diff(12*return_period).shift(-12*return_period)
  - X4 = pd.concat([X4_3DT, X4_6DT, X4_12DT], axis=1)
  - X4.columns = ['MSFT_3DT', 'MSFT_6DT', 'MSFT_12DT']
  - X4 = pd.DataFrame(standard_scaler.fit_transform(X4.values), index = X4.index,columns=X4.columns)

3

- # Forming Dataset
  - X = pd.concat([X1, X2, X3, X4], axis=1)
  - dataset = pd.concat([Y, X], axis=1)

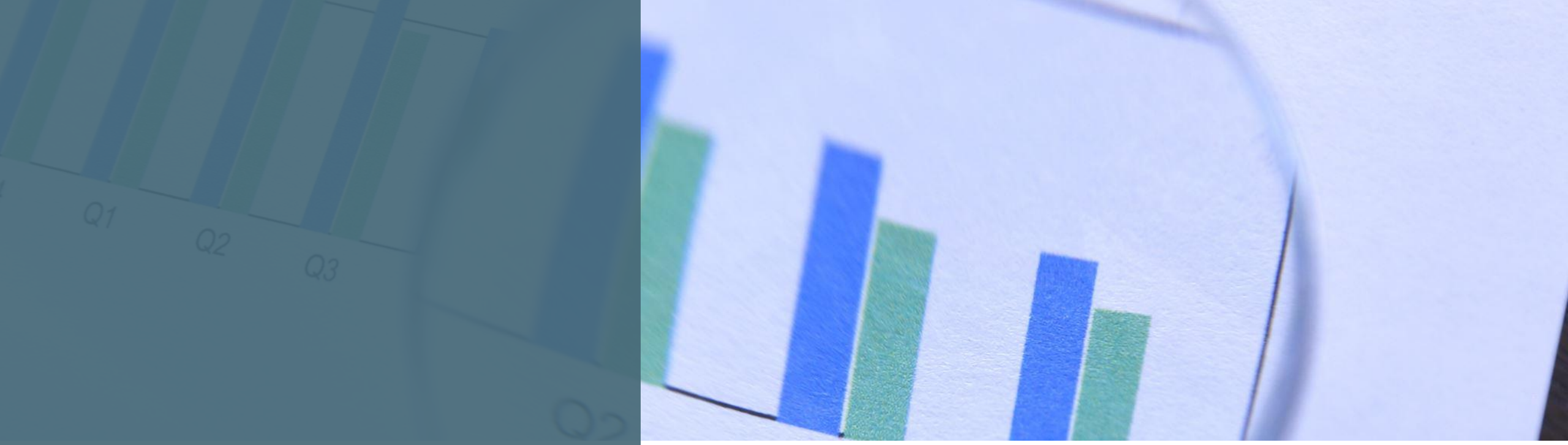
5.1 (c) Data Preparation

1
- # Drop NA
  - Dropna()
- # View Statistics
  - Describe()

2
- # Assign X, Y (drop datetime index)
  - Y = dataset (1st column)
  - X = dataset (2nd :last column)

3
- # feature selection (correlation)
  - Calculate correlation between variables for only continuous data columns
    - corr()
  - Reduce Corr() to Lower Matrix
  - Drop columns if |correlation value| > 0.9

4
- # Train / Test Preparation
- Test_size = 0.3 * len(X)
- Train_size = 0.7 * len(X)
- X_train, X_test = X[0:train_size], X[train_size:len(X)]
- Y_train, Y_test = Y[0:train_size], Y[train_size:len(X)]



5.2

Model Training and Cross Validation

5.2 (a) Model Training and Cross Validation

1

- #Set number of fold / Seed value
  - Num_fold
  - Seed

2

- # Cross Validation Model
- # set k-fold crossvalidation with shuffle
- kfold = model_selection.KFold(n_splits=mum_fold, shuffle = True, random_state=seed)

3

- # Model selection
  - Model_LM = LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
  - c_val ลองอย่างน้อย 3 ค่า [0.1, 1, 10, 100]
  - svr_lin = SVR(kernel='linear', C=c_val)
  - svr_rbf = SVR(kernel='rbf', C=c_val, gamma=0.01)
  - svr_poly = SVR(kernel='poly', C=c_val, degree=2)

5.2 (b) Model Training and Cross Validation

1

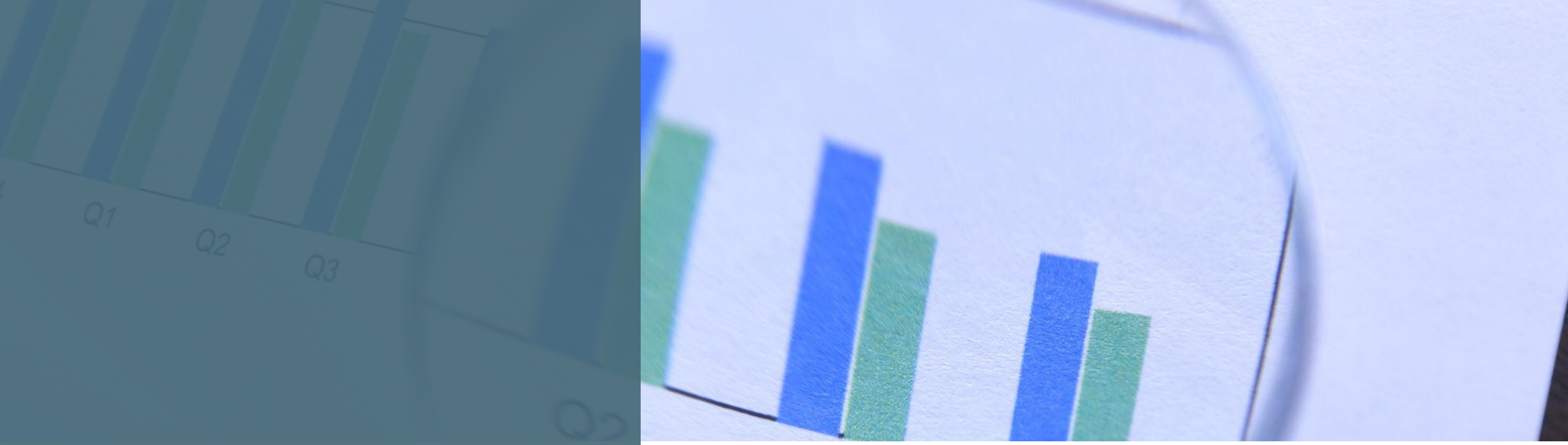
- # Calculate accuracy score for each model
  - score_LM = model_selection.cross_val_score(model_LM, X_train, y_train, cv=kfold)
  - score_lin = model_selection.cross_val_score(svr_lin, X_train, y_train, cv=kfold)
  - score_rbf = model_selection.cross_val_score(svr_rbf, X_train, y_train, cv=kfold)
  - score_poly = model_selection.cross_val_score(svr_poly, X_train, y_train, cv=kfold)

2

- # View score k-fold
  - # Valication score comparison
  - score = pd.DataFrame({'Linear Model':score_LM,'SVR_linear':score_lin, 'SVR_rbf': score_rbf, 'SVR_poly': score_poly})
  - score_mean = pd.DataFrame({'AVG Linear Model':[score_LM.mean()],'AVG SVR_linear':[score_lin.mean()], 'AVG SVR_rbf': [score_rbf.mean()], 'AVG SVC_poly': [score_poly.mean()]})

3

- Print(score)
- Print(score_mean)
- Display(plot (score))



5.3 Model Evaluation

5.3 Prediction and Evaluation

1

- # Predict all models (LM, SVR_linear, SVR_rbf, SVR Poly)
  - LM_pred = model_LM.fit(X_train, y_train).predict(X_test)

2

- # Scatter Plot (X_test, Predict) for all model (LM, SVR_linear, SVR_rbf, SVR Poly)
  - plt.scatter(X_test, LM_pred,c='magenta')

3

- # Model prediction performance evaluation for all model (LM, SVR_linear, SVR_rbf, SVR Poly)
  - MSE
    - LM_MSE = metrics.mean_squared_error(y_test, LM_pred)
  - R2
    - LM_r2 = metrics.r2_score(y_test, LM_pred)

4

- # Display Prediction MSE, R2 for all models
  - Bar()