In [1]:

```
import numpy as np
np.random.seed(0)
num_samples = 300
rainfall = np.random.uniform(50, 150, num_samples)
humidity = np.random.uniform(30, 80, num_samples)
temperature = np.random.uniform(10, 35, num_samples)
apple_yield = 10 * rainfall + 5 * humidity - 2 * temperature + np.random.normal(0, 10,
orange_yield = 8 * rainfall + 3 * humidity + 4 * temperature + np.random.normal(0, 8,
```

In [2]:

```
import pandas as pd
data = pd.DataFrame({
    'Rainfall (mm)': rainfall,
    'Humidity (%)': humidity,
    'Temperature (C)': temperature,
    'Apple Yield': apple_yield,
    'Orange Yield': orange_yield})
```

In [3]:

```
from sklearn.model_selection import train_test_split
X = data[['Rainfall (mm)', 'Humidity (%)', 'Temperature (C)']]
y_apple = data['Apple Yield']
y_orange = data['Orange Yield']
X_train, X_test, y_apple_train, y_apple_test, y_orange_train, y_orange_test = train_te
    X, y_apple, y_orange, test_size=0.2, random_state=42)
```

In [4]:

```
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
apple_regressor = LinearRegression()
apple_regressor.fit(X_train, y_apple_train)
orange_regressor = LinearRegression()
orange_regressor.fit(X_train, y_orange_train)
apple_svr = SVR(kernel='linear')
apple_svr.fit(X_train, y_apple_train)
orange_svr = SVR(kernel='linear')
orange_svr.fit(X_train, y_orange_train)
```

Out[4]:

```
SVR(kernel='linear')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [6]:

```
from sklearn.metrics import mean_squared_error, r2_score
apple_pred_lr = apple_regressor.predict(X_test)
orange_pred_lr = orange_regressor.predict(X_test)
apple_pred_svr = apple_svr.predict(X_test)
orange_pred_svr = orange_svr.predict(X_test)
apple_mse_lr = mean_squared_error(y_apple_test, apple_pred_lr)
orange_mse_lr = mean_squared_error(y_orange_test, orange_pred_lr)
apple_mse_svr = mean_squared_error(y_apple_test, apple_pred_svr)
orange_mse_svr = mean_squared_error(y_orange_test, orange_pred_svr)
apple_r2_lr = r2_score(y_apple_test, apple_pred_lr)
orange_r2_lr = r2_score(y_orange_test, orange_pred_lr)
apple_r2_svr = r2_score(y_apple_test, apple_pred_svr)
orange_r2_svr = r2_score(y_orange_test, orange_pred_svr)

print("Linear Regression - Apple MSE:", apple_mse_lr)
print("Linear Regression - Orange MSE:", orange_mse_lr)
print("Linear Regression - Apple R^2:", apple_r2_lr)
print("Linear Regression - Orange R^2:", orange_r2_lr)
print("SVR - Apple MSE:", apple_mse_svr)
print("SVR - Orange MSE:", orange_mse_svr)
print("SVR - Apple R^2:", apple_r2_svr)
print("SVR - Orange R^2:", orange_r2_svr)
```

```
Linear Regression - Apple MSE: 81.56490758427329
Linear Regression - Orange MSE: 66.31270232358295
Linear Regression - Apple R^2: 0.9989976759898246
Linear Regression - Orange R^2: 0.9987963601844135
SVR - Apple MSE: 85.62915633108307
SVR - Orange MSE: 66.32848649982377
SVR - Apple R^2: 0.9989477317892743
SVR - Orange R^2: 0.9987960736863172
```

In [11]:

```
import numpy as np
A=np.array([[6,1,1,2],
            [4,-2,5,3],
            [2,8,7,1],
            [2,4,5,7]])
print("Rank of A:",np.linalg.matrix_rank(A))
print("\nTrace of A:",np.trace(A))
print("\nDeterminant of A:",np.linalg.det(A))
print("\nInverse of A:\n",np.linalg.inv(A))
print("\nMatrix A raised to power3:\n",np.linalg.matrix_power(A,3))
```

```
Rank of A: 4

Trace of A: 18

Determinant of A: -1767.9999999999998

Inverse of A:
 [[ 1.76470588e-01  9.04977376e-03  7.91855204e-03 -5.54298643e-02]
 [ 5.88235294e-02 -1.38009050e-01  6.67420814e-02  3.28054299e-02]
 [-1.17647059e-01  1.60633484e-01  7.80542986e-02 -4.63800905e-02]
 [ 7.47265497e-18 -3.84615385e-02 -9.61538462e-02  1.73076923e-01]]

Matrix A raised to power3:
 [[ 472  350  492  390]
 [ 576  376  780  484]
 [ 820  904 1172  648]
 [ 884  944 1356  872]]
```

In [ ]:

In [ ]:

In [ ]: