In [5]:

```
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.svm import SVR
import matplotlib.pyplot as plt

# Step 1: Generate Synthetic Data
np.random.seed(0)
n_samples = 600  # Total data points (300 for each fruit)
rainfall = np.random.uniform(0, 100, n_samples)
humidity = np.random.uniform(20, 80, n_samples)
temperature = np.random.uniform(10, 35, n_samples)

# Generate apple and orange yields as dependent variables
apple_yield = 10 + 0.5 * rainfall - 0.2 * humidity + 0.3 * temperature + np.
orange_yield = 8 + 0.3 * rainfall - 0.1 * humidity + 0.4 * temperature + np.

# Step 2: Define Features and Target Variables
features = np.column_stack((rainfall, humidity, temperature))
target_apple = apple_yield
target_orange = orange_yield

# Step 3: Split the Data
X_train_apple, X_test_apple, y_train_apple, y_test_apple = train_test_split(
X_train_orange, X_test_orange, y_train_orange, y_test_orange = train_test_sp

# Step 4: Train Linear Regression Models
lr_apple = LinearRegression()
lr_orange = LinearRegression()

lr_apple.fit(X_train_apple, y_train_apple)
lr_orange.fit(X_train_orange, y_train_orange)

# Step 5: Train Support Vector Regression (SVR) Models
svr_apple = SVR(kernel='linear')
svr_orange = SVR(kernel='linear')

svr_apple.fit(X_train_apple, y_train_apple)
svr_orange.fit(X_train_orange, y_train_orange)

# Evaluate the models (you can use metrics like R-squared, Mean Squared Erro

# Predictions for linear regression
y_pred_lr_apple = lr_apple.predict(X_test_apple)
y_pred_lr_orange = lr_orange.predict(X_test_orange)

# Predictions for SVR
y_pred_svr_apple = svr_apple.predict(X_test_apple)
y_pred_svr_orange = svr_orange.predict(X_test_orange)


# You can then evaluate the performance of these models using appropriate me
from sklearn.metrics import mean_squared_error, r2_score

# Predictions for linear regression
y_pred_lr_apple = lr_apple.predict(X_test_apple)
y_pred_lr_orange = lr_orange.predict(X_test_orange)

# Predictions for SVR
y_pred_svr_apple = svr_apple.predict(X_test_apple)
y_pred_svr_orange = svr_orange.predict(X_test_orange)
```

```python
# Model Evaluation for Linear Regression (Apple)
mse_lr_apple = mean_squared_error(y_test_apple, y_pred_lr_apple)
r2_lr_apple = r2_score(y_test_apple, y_pred_lr_apple)
print("Linear Regression (Apple) - MSE:", mse_lr_apple)
print("Linear Regression (Apple) - R-squared:", r2_lr_apple)

# Model Evaluation for Linear Regression (Orange)
mse_lr_orange = mean_squared_error(y_test_orange, y_pred_lr_orange)
r2_lr_orange = r2_score(y_test_orange, y_pred_lr_orange)
print("Linear Regression (Orange) - MSE:", mse_lr_orange)
print("Linear Regression (Orange) - R-squared:", r2_lr_orange)

# Model Evaluation for SVR (Apple)
mse_svr_apple = mean_squared_error(y_test_apple, y_pred_svr_apple)
r2_svr_apple = r2_score(y_test_apple, y_pred_svr_apple)
print("SVR (Apple) - MSE:", mse_svr_apple)
print("SVR (Apple) - R-squared:", r2_svr_apple)

# Model Evaluation for SVR (Orange)
mse_svr_orange = mean_squared_error(y_test_orange, y_pred_svr_orange)
r2_svr_orange = r2_score(y_test_orange, y_pred_svr_orange)
print("SVR (Orange) - MSE:", mse_svr_orange)
print("SVR (Orange) - R-squared:", r2_svr_orange)
```

```
Linear Regression (Apple) - MSE: 28.40097029111129
Linear Regression (Apple) - R-squared: 0.8864539618761027
Linear Regression (Orange) - MSE: 12.091934810147864
Linear Regression (Orange) - R-squared: 0.8912279252603972
SVR (Apple) - MSE: 28.30753379749695
SVR (Apple) - R-squared: 0.8868275175524528
SVR (Orange) - MSE: 12.19634892110526
SVR (Orange) - R-squared: 0.8902886761113367
```

In [ ]: