Experiment No.1

Title: Linear Regression Analysis: Curve Fitting, Generalization, and Model Evaluation.

Aim: Generate a proper 2-D data set of N points. Split the data set into Training Data set and Test Data set. i) Perform linear regression analysis with Least Squares Method. ii) Plot the graphs for Training MSE and Test MSE and comment on Curve Fitting and Generalization Error. iii) Verify the Effect of Data Set Size and Bias-Variance Tradeoff. iv) Apply Cross Validation and plot the graphs for errors. v) Apply Subset Selection Method and plot the graphs for errors. vi) Describe your findings in each case

Objectives:

1. To generate a 2D dataset and split it into training and test sets.
2. To perform linear regression using the least squares method.
3. To evaluate training and test errors and analyze curve fitting and generalization error.
4. To investigate the effect of dataset size on bias-variance tradeoff.
5. To apply cross-validation and subset selection techniques and analyze errors.

Problem Statement:

Given a dataset of N points in a 2D space, the objective is to fit a linear regression model using the least squares method and evaluate its performance. The analysis includes training and test error comparisons, bias-variance tradeoff examination, cross-validation implementation, and subset selection impact on generalization error.

Outcomes:

1. A well-fitted linear regression model with evaluated MSE for training and test data.
2. Understanding of curve fitting and generalization error through error plots.
3. Insights into bias-variance tradeoff as dataset size varies.
4. Visualization of cross-validation errors and their implications.
5. Effect of subset selection on model performance and generalization.

Tools Required: 4GB RAM, Anaconda, Notebook

Theory:

Linear regression is a fundamental statistical method used for modeling the relationship between a dependent variable (response) and one or more independent variables (predictors). The objective is to find the best-fitting line that minimizes the error between predicted and actual values.

Least Squared Method:

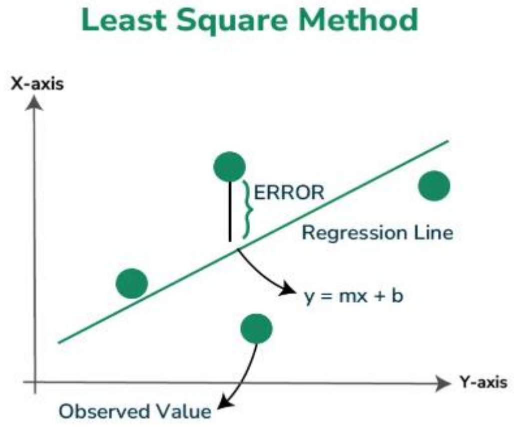
The least squares method is used to minimize the sum of squared residuals (differences between actual and predicted values). The linear regression model follows the equation:

$y = wX + b$

where:

- y is the predicted output,

- X is the input feature,
- w is the weight (slope), and
- b is the bias (intercept).



Curve Fitting and Generalization Error:

Curve fitting determines how well a model captures patterns in the data. An underfitted model has high bias, while an overfitted model has high variance. The generalization error measures how well a model performs on unseen data. A model should balance bias and variance to minimize test error.

Bias-Variance Tradeoff:

- High Bias (Underfitting): The model is too simple and fails to capture patterns in the data.
- High Variance (Overfitting): The model learns noise from the training data and fails to generalize well.
- Optimal Model: Achieves a balance between bias and variance, leading to low test error.

Cross-Validation:

Cross-validation (e.g., k-fold cross-validation) helps assess model performance by dividing the dataset into multiple subsets. The model is trained on different training subsets and validated on different test subsets, ensuring that performance is consistent across different data splits.

Subset Selection:

Subset selection involves choosing a subset of the available features or training samples to improve model simplicity and interpretability. By selecting the most relevant data points or features, the model can avoid overfitting and improve generalization.

Algorithm:

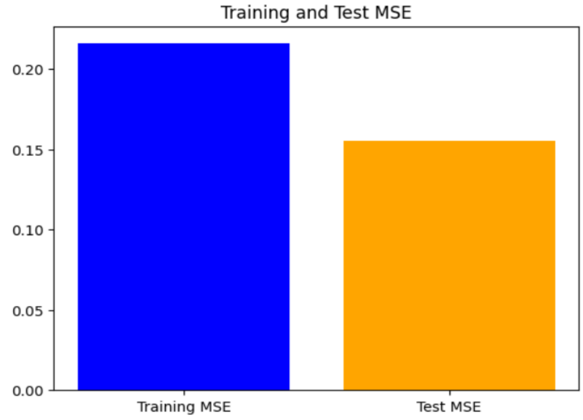
Step 1: Data Generation: Create a 2D dataset with N points and add noise.
Step 2: Data Splitting: Divide the dataset into training (80%) and test (20%) sets.
Step 3: Linear Regression Model: Train the model using the least squares method.
Step 4: Compute MSE: Calculate training and test mean squared errors.
Step 5: Plot Errors: Visualize training vs test MSE to analyze generalization error.
Step 6: Bias-Variance Analysis: Evaluate the effect of training set size on model performance.
Step 7: Cross-Validation: Perform k-fold cross-validation and plot errors.
Step 8: Subset Selection: Train models on different subset sizes and analyze errors.
Step 9: Findings: Summarize the results and their implications.

Source Code:

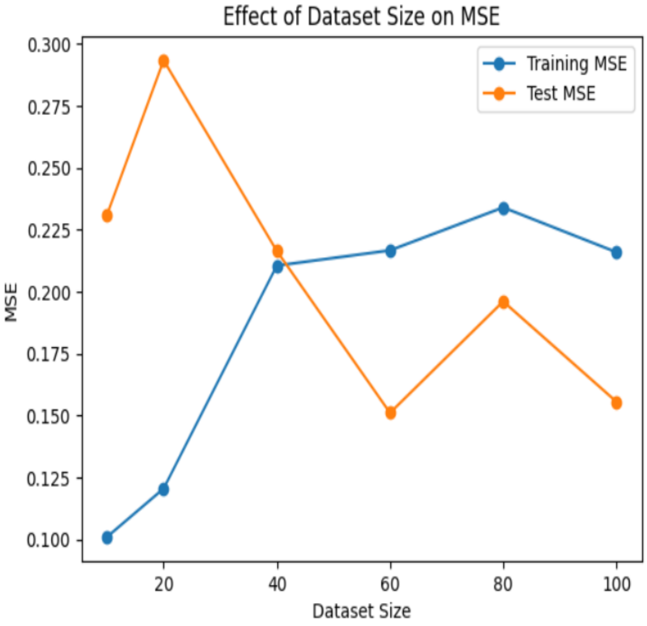
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, KFold
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
def generate_data(N=100, noise=0.5):
    np.random.seed(42)
    X = np.linspace(0, 10, N).reshape(-1, 1)
    y = 2.5 * X.squeeze() + np.random.normal(0, noise, N)
    return X, y
def split_data(X, y, test_size=0.2):
    return train_test_split(X, y, test_size=test_size, random_state=42)
def linear_regression(X_train, y_train, X_test, y_test):
    model = LinearRegression()
    model.fit(X_train, y_train)
    y_train_pred = model.predict(X_train)
    y_test_pred = model.predict(X_test)
    return y_train_pred, y_test_pred, model
def plot_mse(X_train, y_train, X_test, y_test, y_train_pred, y_test_pred):
    train_mse = mean_squared_error(y_train, y_train_pred)
    test_mse = mean_squared_error(y_test, y_test_pred)
    print(f"Training MSE: {train_mse:.4f}, Test MSE: {test_mse:.4f}")
    plt.bar(['Training MSE', 'Test MSE'], [train_mse, test_mse], color=['blue', 'orange'])
    plt.title('Training and Test MSE')
    plt.show()
def bias_variance_tradeoff(X, y):
    train_sizes = [10, 20, 40, 60, 80, 100]
    train_errors = []
    test_errors = []
    for size in train_sizes:
        X_train, X_test, y_train, y_test = split_data(X[:size], y[:size])
        y_train_pred, y_test_pred, _ = linear_regression(X_train, y_train, X_test, y_test)
        train_errors.append(mean_squared_error(y_train, y_train_pred))
        test_errors.append(mean_squared_error(y_test, y_test_pred))
    plt.plot(train_sizes, train_errors, label='Training MSE', marker='o')
    plt.plot(train_sizes, test_errors, label='Test MSE', marker='o')
    plt.xlabel('Dataset Size')
    plt.ylabel('MSE')
    plt.title('Effect of Dataset Size on MSE')
    plt.legend()
    plt.show()
def cross_validation(X, y, folds=5):
    kf = KFold(n_splits=folds, shuffle=True, random_state=42)
    train_errors = []
    test_errors = []
    for train_index, test_index in kf.split(X):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]
        y_train_pred, y_test_pred, _ = linear_regression(X_train, y_train, X_test, y_test)
        train_errors.append(mean_squared_error(y_train, y_train_pred))
        test_errors.append(mean_squared_error(y_test, y_test_pred))
    plt.bar(['Train Error', 'Test Error'], [np.mean(train_errors), np.mean(test_errors)],
   color=['blue', 'orange'])
    plt.title(f'{folds}-Fold Cross Validation Errors')
    plt.show()
def subset_selection(X, y):
    subsets = [10, 20, 50, 100]
    errors = []
    for subset in subsets:
        X_train, X_test, y_train, y_test = split_data(X[:subset], y[:subset])
        _, y_test_pred, _ = linear_regression(X_train, y_train, X_test, y_test)
        errors.append(mean_squared_error(y_test, y_test_pred))
    plt.plot(subsets, errors, marker='o', color='purple')
    plt.xlabel('Subset Size')
    plt.ylabel('Test MSE')
    plt.title('Effect of Subset Size on Test MSE')
    plt.show()
def main():
    X, y = generate_data()
    X_train, X_test, y_train, y_test = split_data(X, y)
    y_train_pred, y_test_pred, _ = linear_regression(X_train, y_train, X_test, y_test)
    plot_mse(X_train, y_train, X_test, y_test, y_train_pred, y_test_pred)
    print("Effect of Dataset Size on Bias-Variance Tradeoff:")
    bias_variance_tradeoff(X, y)
    print("Cross Validation:")
    cross_validation(X, y)
    print("Subset Selection Method:")
    subset_selection(X, y)
if __name__ == "__main__":
    main()
```

Output:

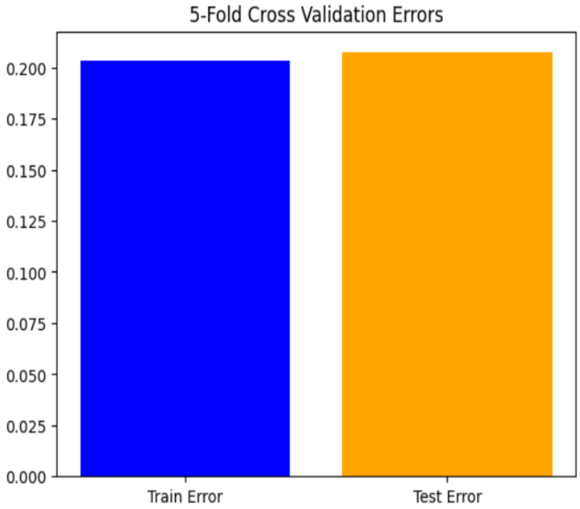
```
Training MSE: 0.2159, Test MSE: 0.1555
```



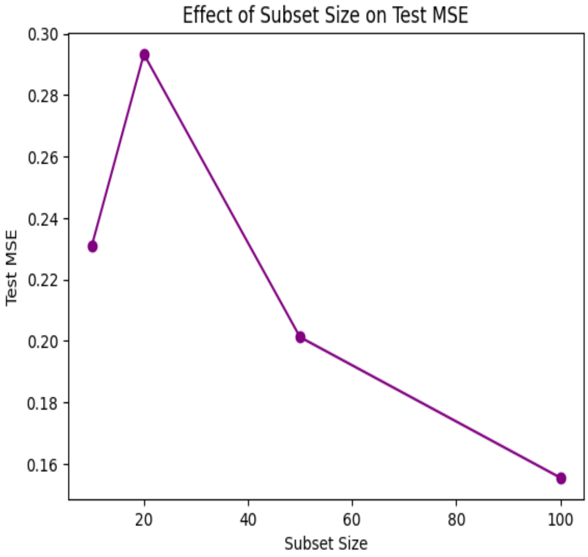
```
Effect of Dataset Size on Bias-Variance Tradeoff:
```



Cross Validation:



Subset Selection Method:



Conclusion:

____

____

____

____