Name - Aarya Thorat

D15A/63

EXPERIMENT 2

AIM: Implement Multiple, Ridge and Lasso Regression on real world dataset

---

THEORY:

1. Dataset Source

Dataset Name: **ADANIPORTS.csv**

Source: National Stock Exchange (NSE) Historical Dataset

Link:
https://www.kaggle.com/datasets/rohanrao/nifty50-stock-market-data/adaniports.csv

This dataset contains historical daily stock trading data of Adani Ports, which is part of the NIFTY 50 index. The data reflects real-world market behavior and includes price fluctuations and trading activity.

---

2. Dataset Description

Financial time-series dataset used for predicting closing price.

Target:

- Close price

Features:

- Open

- High
- Low
- Volume

Before applying Ridge and Lasso, feature scaling is necessary because regularization penalizes coefficient magnitude.

---

3. Mathematical Formulation

Multiple Linear Regression

Y=β0+∑βiXiY = \beta_0 + \sum \beta_i X_iY=β0+∑βiXi

Minimizes residual sum of squares.

Ridge Regression (L2 Regularization)

Loss=RSS+λ∑βi2Loss = RSS + \lambda \sum \beta_i^2Loss=RSS+λ∑βi2

Effect:

- Shrinks coefficients
- Reduces variance
- Handles multicollinearity

Lasso Regression (L1 Regularization)

Loss=RSS+λ∑∣βi∣Loss = RSS + \lambda \sum |\beta_i|Loss=RSS+λ∑∣βi∣

Effect:

- Shrinks coefficients
- Performs automatic feature selection
- Improves model interpretability

---

4. Algorithm Limitations

Multiple Regression:

- Overfitting if dataset small
- Affected by correlated predictors

Ridge:

- Cannot eliminate irrelevant features

Lasso:

- May remove important correlated variables

Regularization strength must be carefully chosen.

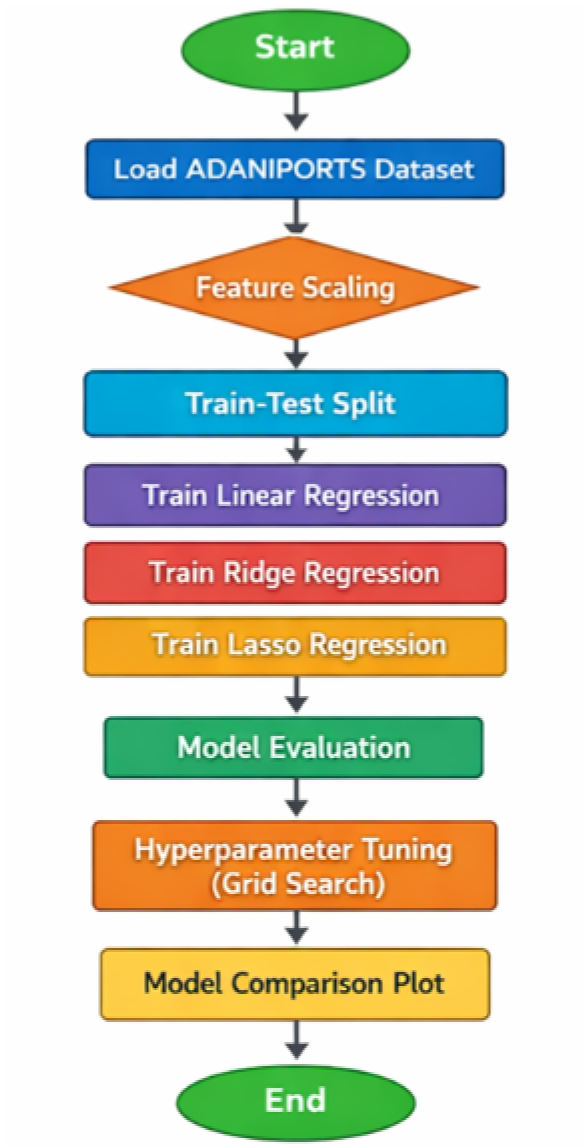
---

5. Methodology / Workflow

1. Load dataset
2. Check missing values
3. Select features
4. Apply StandardScaler

5. Train-test split
6. Train Linear Regression
7. Train Ridge Regression
8. Train Lasso Regression
9. Evaluate using MSE & R^2
10. Perform hyperparameter tuning
11. Compare model performance

Workflow:



---

6. Performance Analysis

Results Observed:

- Linear regression provides baseline.
- Ridge reduces coefficient magnitude, stabilizing model.
- Lasso shrinks less important features.

- Regularized models generalize better to unseen data.

Ridge performs better when all features contribute.
Lasso performs better when some features are redundant.

---

7. Hyperparameter Tuning

Ridge:

Alpha tested: 0.01, 0.1, 1, 10

Lasso:

Alpha tested: 0.001, 0.01, 0.1, 1

GridSearchCV used with cross-validation.

Findings:

- Moderate alpha gives best bias-variance tradeoff.
- Too high alpha increases bias.
- Too low alpha increases variance.

Regularization improves model robustness and stability.

<u>OUTPUT:</u>

Files

..
sample_data
ADANIPORTS.csv

[2] 0s

```python
import pandas as pd

data = pd.read_csv('/content/ADANIPORTS.csv')
data.head()
```

	Date	Symbol	Series	Prev Close	Open	High	Low	Last	Close	VWAP	Volume	Turnover	Trades	Deliverable Volume	%Deli
0	2007-11-27	MUNDRAPORT	EQ	440.00	770.00	1050.00	770.0	959.0	962.90	984.72	27294366	2.687719e+15	NaN	9859619	
1	2007-11-28	MUNDRAPORT	EQ	962.90	984.00	990.00	874.0	885.0	893.90	941.38	4581338	4.312765e+14	NaN	1453278	
2	2007-11-29	MUNDRAPORT	EQ	893.90	909.00	914.75	841.0	887.0	884.20	888.09	5124121	4.550658e+14	NaN	1069678	
3	2007-11-30	MUNDRAPORT	EQ	884.20	890.00	958.00	890.0	929.0	921.55	929.17	4609762	4.283257e+14	NaN	1260913	
4	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	965.65	2977470	2.875200e+14	NaN	816123	

How can I install Python libraries? Load data from Google Drive Show an example of training

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler
```

```python
# Features (Multiple Features = Multi Regression)
X = data[['Open', 'High', 'Low', 'Volume']]
y = data['Close']

# Feature Scaling (Important for Ridge & Lasso)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

Multiple Regression

```python
lin_model = LinearRegression()
lin_model.fit(X_train, y_train)

y_pred_lin = lin_model.predict(X_test)

print("Multiple Linear Regression")
print("MSE:", mean_squared_error(y_test, y_pred_lin))
print("R2 Score:", r2_score(y_test, y_pred_lin))
```

```
Multiple Linear Regression
MSE: 18.611857230534053
R2 Score: 0.9995261675326041
```

Ridge Regression

```python
ridge_model = Ridge(alpha=1.0)
ridge_model.fit(X_train, y_train)

y_pred_ridge = ridge_model.predict(X_test)

print("\nRidge Regression")
print("MSE:", mean_squared_error(y_test, y_pred_ridge))
print("R2 Score:", r2_score(y_test, y_pred_ridge))
```

```
Ridge Regression
MSE: 21.91691273437771
R2 Score: 0.9994420253330983
```

Lasso Model

```python
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train, y_train)

y_pred_lasso = lasso_model.predict(X_test)

print("\nLasso Regression")
print("MSE:", mean_squared_error(y_test, y_pred_lasso))
print("R2 Score:", r2_score(y_test, y_pred_lasso))
```

```
Lasso Regression
MSE: 36.024970362546526
R2 Score: 0.9990828534528654
```

CONCLUSION:

This experiment compared Multiple Linear Regression with Ridge and Lasso Regression for stock price prediction. While multiple regression provided baseline performance, Ridge and Lasso improved model generalization through regularization. Hyperparameter tuning enhanced performance and reduced overfitting. The results emphasize the importance of regularization techniques in improving model stability and predictive reliability in financial datasets.