PROJECT REPORT

Title : Comparative Analysis of Regression Models for Predicting Unicorn Startup Valuations in India

Objective

The objective of this project is to compare the performance of different regression models — Linear Regression, Ridge Regression, and Lasso Regression — in predicting the valuation of Indian unicorn startups. By applying consistent data preprocessing and feature engineering techniques, the project evaluates each model using standard performance metrics such as Mean Squared Error and R^2 score. This comparative analysis aims to identify the most effective regression approach, understand the impact of regularization on prediction accuracy, and demonstrate how model selection influences predictive performance in real-world valuation problems.

Dataset

The dataset is a CSV file that provides an overview of Indian unicorn startups as of June 2023. It contains information about privately held startups valued at $1 billion or more, including details such as company name, sector, founding year, location, valuation and notable investors.

Data Preprocessing:

1. Identified and handled missing values using appropriate strategies such as removal or imputation
2. Separated categorical and numerical features for targeted preprocessing
3. Applied One-Hot Encoding to categorical variables (e.g., sector, location) using OneHotEncoder
4. Standardized numerical features using StandardScaler to normalize feature scales
5. Implemented preprocessing pipelines using ColumnTransformer and Pipeline to ensure reproducibility
6. Split the dataset into training and testing sets using train_test_split for unbiased model evaluation

Prediction or Output Expected

The output of this code is the predicted valuation of Indian unicorn startups, generated using regression models. The models make predictions on unseen test data and compare these predicted values with actual valuations. Instead of focusing on individual predictions, the code evaluates model performance using metrics such as Mean Squared Error (MSE) and R^2 score, enabling a comparative analysis of Linear, Ridge, and Lasso regression models to determine the most effective approach for valuation prediction.

Code

```
#Importing
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import mean_squared_error, r2_score
import warnings
warnings.filterwarnings("ignore", category=UserWarning)

#Data Loading
df = pd.read_csv("data/Indian Unicorn startups 2023 updated.csv")
df.head()

#Drop Irrelevant columns
df = df.drop(columns = ['No.', 'Company'])
#Rename columns
df = df.rename(columns = {'Entry Valuation^^ ($B)' : 'Entry_Valuation', 'Valuation ($B)' :
'Valuation'})

#Extract exact year
df['Entry_year'] = pd.to_datetime(df['Entry'], format = '%b/%Y').dt.year
df = df.drop(columns = ['Entry'])

#Investor Count feature
df['Investor_Count'] = df['Select Investors'].apply(lambda x: len(str(x).split(',')))
df = df.drop(columns = ['Select Investors'])

#Features and target
X = df.drop(columns = ['Valuation'])
y = df['Valuation']

numerical_features = ['Entry_Valuation', 'Entry_year', 'Investor_Count']
categorical_features = ['Sector', 'Location']
numeric_transformer = Pipeline(steps = [('scaler', StandardScaler())])
categorical_transformer = OneHotEncoder(drop = 'first', handle_unknown = 'ignore')
```

```
preprocessor = ColumnTransformer(transformers = [('num', numeric_transformer,
numerical_features), ('cat', categorical_transformer, categorical_features)])
#Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)

#function for visualizing actual vs predicted
def plot_actual_vs_pred(y_test, y_pred, model_name):
    plt.figure()
    plt.scatter(y_test, y_pred)
    plt.plot([y_test.min(), y_test.max()],
             [y_test.min(), y_test.max()])
    plt.xlabel("Actual Values")
    plt.ylabel("Predicted Values")
    plt.title(f"Actual vs Predicted - {model_name}")
    plt.show()

#Evaluation function for all models
def evaluate_model(name, model, X_train, X_test, y_train, y_test):
    model.fit(X_train, y_train)

    y_train_pred = model.predict(X_train)
    y_test_pred = model.predict(X_test)

    plot_actual_vs_pred(y_test, y_test_pred, name)
    train_rmse = np.sqrt(mean_squared_error(y_train, y_train_pred))
    test_rmse = np.sqrt(mean_squared_error(y_test, y_test_pred))
    r2 = r2_score(y_test, y_test_pred)

    return{'Model' : name, 'Train RMSE' : train_rmse, 'Test RMSE' : test_rmse, 'R2 Score' : r2}

models = []
#Linear regression
models.append((
    "Linear Regression",
    Pipeline(steps = [
        ('preprocessor', preprocessor),
        ('model', LinearRegression())
    ])
))
#Ridge Regression
models.append((
    "Ridge Regression",
    Pipeline(steps = [
        ('preprocessor', preprocessor),
```

```
        ('model', Ridge(alpha = 1.0))
    ])
))
#Lasso Regression
models.append((
    "Lasso Regression",
    Pipeline(steps = [
        ('preprocessor', preprocessor),
        ('model', Lasso(alpha = 0.01))
    ])
))

#run evaluation for all models
results = []
for name, model in models:
    results.append(
        evaluate_model(name, model, X_train, X_test, y_train, y_test)
    )

results_df = pd.DataFrame(results)
results_df

#Visual Comparison
results_df.set_index('Model')[['Train RMSE', 'Test RMSE']].plot(kind = 'bar')
plt.ylabel("RMSE")
plt.title("Train vs Test RMSE Comparison")
plt.show()

best_model = results_df.sort_values(by = 'Test RMSE').iloc[0]
print("Best Model:", best_model)
def get_user_input():
    data = {}
    for col in X.columns:
        value = input(f"Enter value for {col}: ")
        data[col] = [value]
    return pd.DataFrame(data)
user_input = get_user_input()
for name, model in models:
    prediction = model.predict(user_input)[0]
    print(f"{name} Prediction: {prediction:.2f}")
```

Output

Actual vs Predicted Values Scatter Plot

This graph visually shows how accurate each regression model is and helps compare how closely their predictions match real values.

1. Each dot represents one startup from the test dataset
2. X-axis: Actual valuation
3. Y-axis: Predicted valuation by the model
4. The diagonal line represents perfect prediction

For Linear Regression,

Data points are widely scattered around the diagonal reference line. Many points lie far from the line, especially at higher valuation values. This means that the model struggles to capture complex relationships in the data and prediction errors are relatively high.

For Ridge Regression,

Points are slightly closer to the diagonal line compared to Linear Regression. The spread of errors is more controlled which means regularization reduces the effect of extreme values and predictions are more stable than Linear Regression.

For Lasso Regression,

Points are more compressed, with many predictions clustered together. Less variation in predicted values. Lasso removes less important features by shrinking coefficients to zero. This leads to simpler but more conservative predictions.

MODEL	SCATTER PATTERN	PREDICTION QUALITY
Linear	Widely spread	Least accurate
Ridge	Moderately close to diagonal	Most balanced
Lasso	Highly compressed	Over-simplified

Train vs Test RMSE Bar Chart

This graph helps compare:

1. How well each model learns from data
2. Which model performs better on new, unseen data

In this graph, each model has two bars.

Train RMSE → error on training data

Test RMSE → error on unseen data

Inference:

- Lower RMSE bar = better performance
- If test RMSE is much higher than train RMSE → model may not generalize well
- Similar heights → model behaves consistently

Best Model Selection
The best model is selected by comparing RMSE and R^2 scores on test data and choosing the model that produces the lowest prediction error and most consistent results.

From the results, the Ridge Regression model shows a lower test RMSE compared to Linear and Lasso Regression, indicating smaller prediction errors. It also provides a better(less negative or higher) R^2 score, showing improved ability to capture patterns in the data. Linear Regression has a higher RMSE and lower R2 score, suggesting weaker prediction accuracy. Lasso Regression shows strong regularization, leading to simplified predictions but higher error and lower explanatory power.

Conclusion
This project applies machine learning regression techniques to analyze and predict the valuation of Indian unicorn startups using data from June 2023. By comparing Linear, Ridge, and Lasso regression models, the study highlights how different approaches affect prediction accuracy.
Among the models, **Ridge Regression** performs best, showing lower RMSE and better R^2 scores, indicating more reliable and stable predictions. The project demonstrates the importance of proper data preprocessing, model evaluation, and comparative analysis, while providing insights into how startup characteristics influence valuation.