MSCS 634 – Lab 4: Regression and Regularization on the Diabetes Dataset

https://github.com/baralsamrat/MSCS634_Lab_4

Overview

This lab explores a variety of regression techniques using the **Diabetes Dataset** from `sklearn.datasets`. The objective is to understand how different regression models perform, how regularization affects model behavior, and how overfitting can be identified and mitigated.

The lab includes:

- Simple Linear Regression
- Multiple Linear Regression
- Polynomial Regression (degrees 1–5)
- Ridge Regression
- Lasso Regression

All models are evaluated using **MAE**, **MSE**, **RMSE**, and **R^2**. Visualizations and performance comparisons help demonstrate the strengths and weaknesses of each modeling approach.

Dataset Description

The **Diabetes Dataset** is a built-in dataset from Scikit-Learn containing:

- **10 standardized numeric features**
- **A continuous target variable** representing disease progression one year after baseline measurements

Features include: `age`, `sex`, `bmi`, `bp`, and six blood serum measurements (`s1`–`s6`).

This dataset is widely used for demonstrating regression techniques and model evaluation.

What This Lab Covers

1. Data Preparation

- Load dataset with `load_diabetes()`
- Explore features and target distribution
- Check for missing values (none in this dataset)
- Create train-test splits for modeling

2. Simple Linear Regression

- Uses **BMI** as the single predictor
- Establishes a baseline model

- Visualization: actual vs predicted scatter plot

3. Multiple Linear Regression

- Uses **all 10 features**
- Significant improvement over simple regression
- Visualization: predicted vs actual values

4. Polynomial Regression (Degrees 1–5)

- Adds polynomial feature expansion
- Demonstrates **underfitting** (low degrees) and **severe overfitting** (high degrees)
- Visualization: multiple polynomial curves

5. Regularization Models

- **Ridge Regression** (L2 penalty)
- **Lasso Regression** (L1 penalty + feature selection)
- Evaluation across different α values
- Visualization: Ridge/Lasso predicted vs actual

6. Model Performance Comparison

- Consolidated metrics table
- Identification of best-performing models
- Analysis of overfitting, feature redundancy, and dataset behavior

---

Performance Summary

Below is the full performance table generated in the lab:

Model	MAE	MSE	RMSE	R-squared
Simple Linear Regression (BMI)	52.26	4061.83	63.73	0.23
Multiple Linear Regression	42.79	2900.19	53.85	0.45
Polynomial Regression (Degree 1)	42.79	2900.19	53.85	0.45
Polynomial Regression (Degree 2)	43.58	3096.03	55.64	0.42
Polynomial Regression (Degree 3)	164.85	82446.05	287.13	-14.56
Polynomial Regression (Degree 4)	273.63	178909.29	422.98	-32.77
Polynomial Regression (Degree 5)	182.60	68178.08	261.11	-11.87
Ridge Regression (alpha=1.0)	42.81	2892.01	53.78	0.45
Lasso Regression (alpha=0.1)	42.81	2884.62	53.71	0.46

Discussion and Analysis of Model Performance

1. Review of Performance Metrics

The summary table shows that:

- **Simple Linear Regression** with only BMI performs poorly ($R^2 = 0.23$).
- **Multiple Linear Regression** significantly improves performance ($R^2 = 0.45$).
- **Lasso Regression ($\alpha = 0.1$)** provides the **best performance** with:
  - **Highest $R^2 = 0.46$**
  - **Lowest MSE = 2884.62**
  - **Lowest RMSE = 53.71**
- Polynomial models with degrees ≥ 3 perform extremely poorly due to overfitting.

---

2. Best Performing Model: Lasso Regression ($\alpha = 0.1$)

Lasso Regression achieved:

- **Best R^2 (0.46)**
- **Lowest error values**
- Benefits from **feature selection**, shrinking irrelevant coefficients
- Handles multicollinearity better than plain linear regression

This makes Lasso the best generalization model for this dataset.

---

3. Polynomial Regression: Underfitting & Overfitting

Polynomial models indicate:

- **Degree 1 (same as multiple regression):** $R^2 = 0.45$ (good fit)
- **Degree 2:** Training performance improves, test performance declines slightly → Light overfitting begins
- **Degrees 3, 4, 5:**
  - Training R^2 approaches or reaches **1.0**
  - Test R^2 becomes **highly negative**
  - → **Severe overfitting**

This shows that the Diabetes dataset does **not support complex nonlinear modeling**.

---

4. Regularization with Ridge & Lasso

Ridge Regression (L2 penalty)

- Smoothly shrinks all coefficients
- Performance nearly identical to Multiple Linear Regression
- Helps stabilize the model but does not improve accuracy significantly at α = 1.0

Lasso Regression (L1 penalty)

- Performs **feature selection** by zeroing out lesser-important coefficients
- Slightly improves generalization
- α = 0.1 yields the best overall performance

Lasso is preferred when some features may be redundant or weak predictors.

---

5. Key Insights About the Diabetes Dataset

✔ 1. No single feature strongly predicts the target

Simple Linear Regression with BMI yields $R^2 = 0.23$.

✔ 2. Combined features significantly improve accuracy

Multiple Linear Regression increases R^2 to 0.45.

✔ 3. Relationships are mostly linear

Polynomial degrees > 2 cause extreme overfitting.

✔ 4. Feature redundancy exists

Lasso improves performance by removing weaker features.

✔ 5. Regularization is beneficial

Light regularization improves stability and reduces overfitting risk.

---

6. Overall Conclusions

- **Lasso Regression (α = 0.1)** is the best-performing model.
- **Multiple Linear Regression** performs strongly and is a solid baseline.
- **Polynomial Regression beyond degree 2 should be avoided** due to overfitting.
- **Ridge and Lasso Regression** help control overfitting, with Lasso giving the greatest improvement.
- The Diabetes dataset appears **mostly linear** with modest feature interactions.
- Regularization improves generalization and model robustness.

Repository Contents

Your GitHub repository should contain:

```
MSCS_634_Lab_4/
│
├── lab4.ipynb   # Full notebook with code, plots, and results
├── README.md                          # Documentation (this file)
└── images/                            # Saved figures/plots for reference
```

How to Run This Notebook

1. Install required libraries:

```
pip install numpy pandas matplotlib seaborn scikit-learn
```

2. Launch Jupyter Notebook:

```
jupyter notebook

or

https://colab.research.google.com/
```

3. Open and run:

```
lab4.ipynb
```
