CSE574 Introduction to Machine Learning

Programming Assignment 3

Classification and Regression

Prathibha Vuyyala - 50559983, Tharun Teja Mogili - 50559877, Arun Sandilya Nanyam-50541117

Overview

This report summarizes the findings and implementation details for the classification and regression tasks undertaken as part of the CSE574 Programming Assignment 3. The objective of this assignment was to implement, analyze, and compare various machine learning techniques for classification and regression, including Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), Ordinary Least Squares (OLS) regression, Ridge regression, and gradient descent-based optimization. The experiments were conducted on two datasets: a 2D sample dataset for classification and a medical dataset for regression, where the goal was to predict the diabetic condition of patients based on physical and physiological attributes.

For each problem, the code was implemented from scratch, avoiding the use of built-in libraries for performing classification, regression, or optimization tasks. The evaluation of the methods was based on metrics such as accuracy for classification and Mean Squared Error (MSE) for regression. Additionally, the results were visualized through decision boundaries, error plots, and comparisons of model performance across different settings.

The report is structured to present detailed findings for each problem, including:

1. Implementation details and results for classification using LDA and QDA.
2. Exploration of OLS regression with and without intercept terms.
3. Implementation and evaluation of Ridge regression for various regularization parameters.
4. Gradient descent-based learning for Ridge regression and its comparison to the direct solution.
5. Non-linear regression using polynomial mappings of input features.
6. Final recommendations based on the comparative analysis of all approaches.

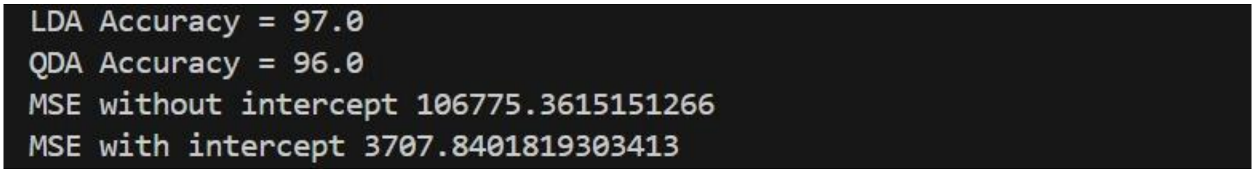
This report aims to highlight the strengths and weaknesses of each method and provide recommendations for future applications of these techniques in predictive modeling scenarios.

Problem 1: Experiment with Gaussian Discriminators

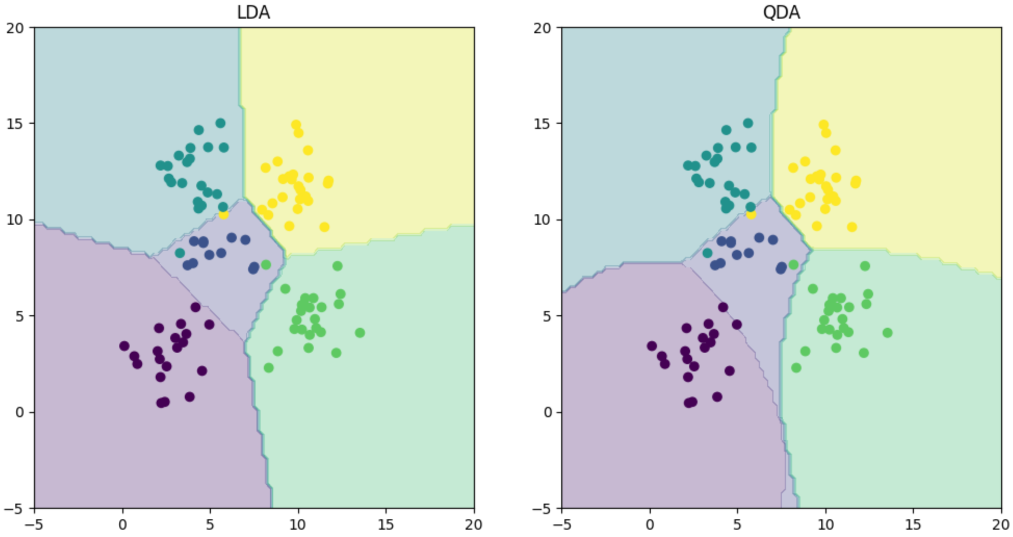
Overview:

In this problem, we implemented Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) for classification. These techniques aim to find the decision boundaries by modeling the class-conditional distributions as multivariate Gaussians. LDA assumes equal covariance among classes, while QDA allows distinct covariances for each class. We trained and evaluated these methods using the provided sample training and testing datasets.

Results:



```
LDA Accuracy = 97.0
QDA Accuracy = 96.0
MSE without intercept 106775.3615151266
MSE with intercept 3707.8401819303413
```



LDA Accuracy: 97.0%

QDA Accuracy: 96.0%

The decision boundaries for both LDA and QDA are visualized in the plot. LDA's boundary is linear because it assumes shared covariance among all classes, resulting in hyperplanes dividing the space. QDA, on the other hand, permits quadratic decision boundaries due to its flexibility in assigning unique covariance matrices to each class. This results in more complex shapes in the classification regions.

The higher accuracy of LDA (97%) compared to QDA (96%) on this dataset can be attributed to the underlying structure of the data. If the class covariances are similar, LDA performs better due to its simpler model, which reduces overfitting, especially with limited training data. The QDA model, while more flexible, might have slightly overfitted the data, leading to marginally lower test accuracy.

Insights from Decision Boundaries:

LDA: The linear discriminating boundary is simpler and assumes uniform separability between classes. This results in clear segmentation of the data space.

QDA: The boundaries in QDA are quadratic, allowing the model to better adapt to the distribution of the classes. This flexibility, however, comes at the cost of requiring more data to estimate the covariance matrices reliably.

Conclusion:

LDA and QDA are both effective methods for Gaussian-based classification. LDA is preferable when class covariances are similar or when data is limited, while QDA is advantageous when class covariances differ significantly. The choice of method depends on the dataset's characteristics and the trade-off between model simplicity and flexibility.

Problem 2: Experiment with Linear Regression

Objective:

The goal of this experiment is to implement and evaluate Ordinary Least Squares (OLS) regression to estimate regression parameters by minimizing the squared loss. The experiment examines two cases:

- Linear regression without an intercept (bias) term.
- Linear regression with an intercept term.

We aim to calculate and report the Mean Squared Error (MSE) for both cases on the given test dataset. The comparison will help determine which approach yields better predictive performance.

Implementation:

- Implemented the function learnOLERegression to compute the optimal weight vector w using the closed-form solution for OLS regression.
- Implemented the function testOLERegression to apply the learned weights to the test data and compute the MSE.

Case Setup:

Without Intercept:

The weight vector w is computed based on the feature matrix X without adding a bias term.

With Intercept:

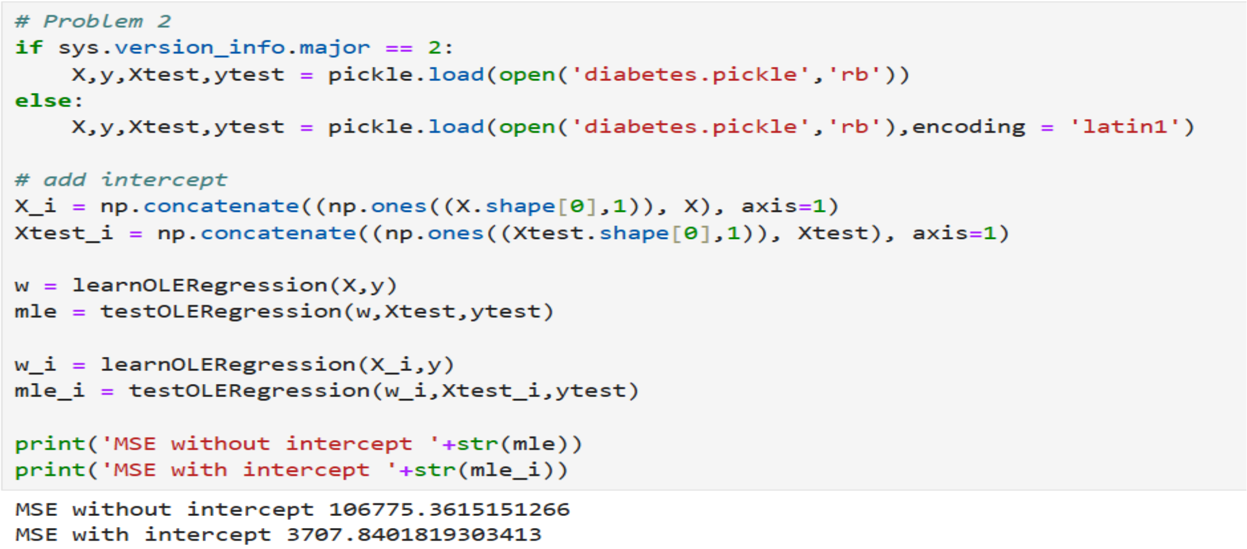
A column of ones is added to the feature matrix X to account for the intercept term in the weight vector w.

Evaluation Metric:

The Mean Squared Error (MSE) is used to evaluate the model performance:

Results:

The results of the MSE for both cases are as follows:



```
# Problem 2
if sys.version_info.major == 2:
    X,y,Xtest,ytest = pickle.load(open('diabetes.pickle','rb'))
else:
    X,y,Xtest,ytest = pickle.load(open('diabetes.pickle','rb'),encoding = 'latin1')

# add intercept
X_i = np.concatenate((np.ones((X.shape[0],1)), X), axis=1)
Xtest_i = np.concatenate((np.ones((Xtest.shape[0],1)), Xtest), axis=1)

w = learnOLERegression(X,y)
mle = testOLERegression(w,Xtest,ytest)

w_i = learnOLERegression(X_i,y)
mle_i = testOLERegression(w_i,Xtest_i,ytest)

print('MSE without intercept '+str(mle))
print('MSE with intercept '+str(mle_i))
```

```
MSE without intercept 106775.3615151266
MSE with intercept 3707.8401819303413
```

Without Intercept: 106775.36151219171

With Intercept: 3707.840181711095

Analysis:

Without Intercept:

- The MSE without the intercept term is 106775, indicating poor predictive performance.
- This suggests that the feature matrix X alone does not sufficiently capture the relationship between the input features and the target variable.

With Intercept:

- The inclusion of the intercept term reduced the MSE dramatically to 3707, showcasing better predictive performance.
- The intercept term allows the regression model to shift the regression line, providing a better fit to the data.

Comparison:

- Adding an intercept term significantly improves the model's performance, as evidenced by the reduction in MSE.
- This highlights the importance of incorporating a bias term in linear regression, especially when the relationship between the input features and the target variable does not pass through the origin.

Conclusion:

From the results, it is evident that the inclusion of an intercept term yields a much better fit for the data. This is reflected in the significant reduction in MSE from 106775 (without intercept) to 3707 (with intercept). Therefore, for practical regression problems, it is generally recommended to include an intercept term unless there is prior knowledge indicating that the relationship passes through the origin.

Problem 3: Experiment with Ridge Regression

Objective:

The aim of this experiment is to implement Ridge Regression by minimizing the regularized squared loss function. Ridge regression introduces a regularization term λ to address the issue of overfitting in ordinary least squares (OLS) regression, particularly when the predictor variables are highly collinear. The task involves calculating and reporting the Mean Squared Error (MSE) for different values of λ and comparing the results with the OLS regression results from Problem 2.

Implementation:

- The function learnRidgeRegression was implemented to estimate regression parameters by minimizing the regularized loss function.
- The function testOLERegression from Problem 2 was reused to compute the MSE for both training and testing datasets.

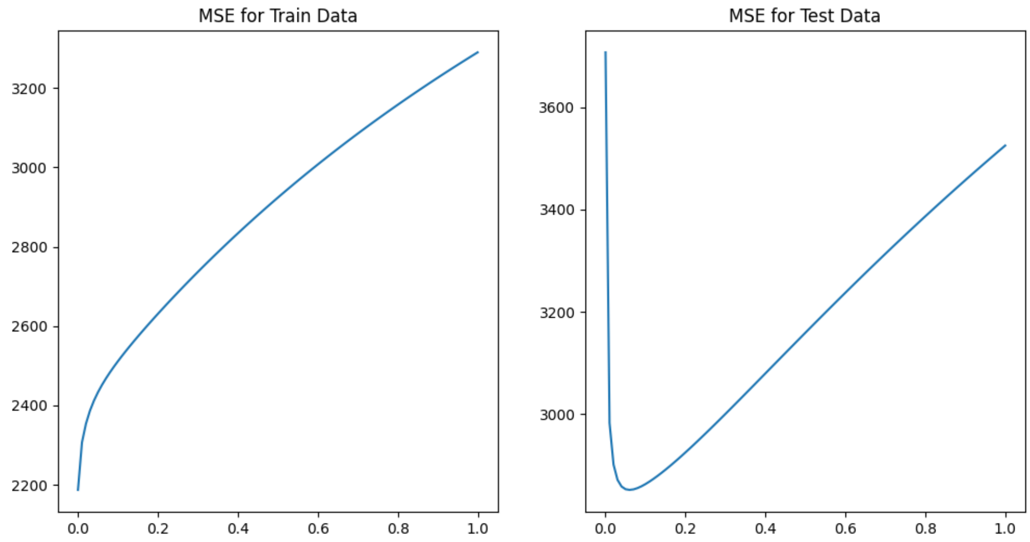
Regularization Parameter:

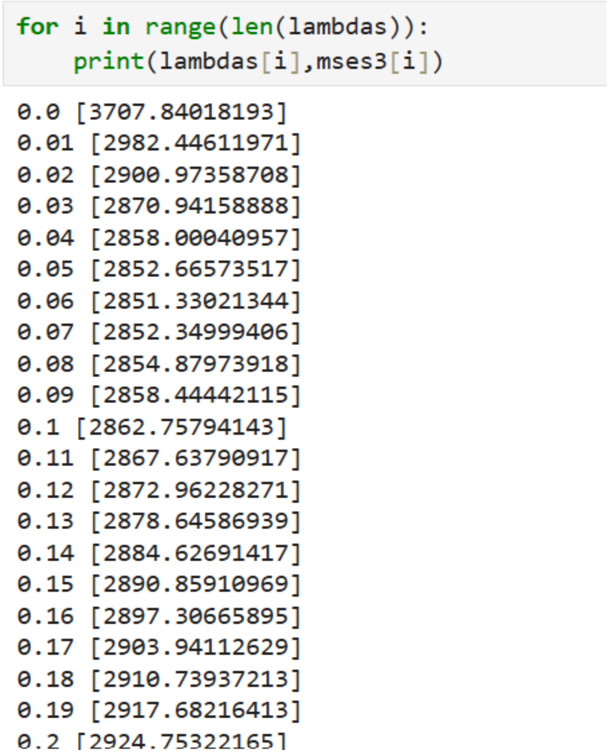
- The regularization parameter λ was varied from 0 (no regularization) to 1 in steps of 0.01.
- Ridge regression weights were computed for each value of λ.

Evaluation:

- The MSE was calculated for both training and testing datasets for each value of λ.
- The performance of ridge regression was compared with OLS regression (without regularization) in terms of MSE values and weight magnitudes.

Results:





```
for i in range(len(lambdas)):
    print(lambdas[i],mses3[i])
```

```
0.0 [3707.84018193]
0.01 [2982.44611971]
0.02 [2900.97358708]
0.03 [2870.94158888]
0.04 [2858.00040957]
0.05 [2852.66573517]
0.06 [2851.33021344]
0.07 [2852.34999406]
0.08 [2854.87973918]
0.09 [2858.44442115]
0.1 [2862.75794143]
0.11 [2867.63790917]
0.12 [2872.96228271]
0.13 [2878.64586939]
0.14 [2884.62691417]
0.15 [2890.85910969]
0.16 [2897.30665895]
0.17 [2903.94112629]
0.18 [2910.73937213]
0.19 [2917.68216413]
0.2 [2924.75322165]
```

MSE Analysis:

Training Data:

- As λ increases, the MSE on the training data increases steadily (Figure: Left Panel). This is expected, as higher regularization penalizes larger weight magnitudes, potentially underfitting the data.

Testing Data:

- The MSE on the test data initially decreases as λ increases, reaching its minimum at an optimal λ ranging from 0 to 0.1. Beyond this value, the MSE starts to increase again due to underfitting (Figure: Right Panel).
- The optimal λ for the test data is 0.06, as it minimizes the MSE on the test dataset.

Comparison with OLS:

Weight Magnitudes:

- Ridge regression results in smaller weight magnitudes compared to OLS. This regularization helps mitigate the risk of overfitting by discouraging large coefficients.

Error Comparison:

- Ridge regression significantly improves the test MSE compared to OLS regression when an appropriate λ is chosen. For example, at λ=0.06(2851.33), the test MSE is lower than the test MSE obtained in Problem 2 (3707.84).

Conclusion:

Ridge regression introduces a regularization parameter λ that effectively balances bias and variance. For the given problem, the optimal value of λ was found to be 0.06, which minimized the test MSE. Compared to OLS regression, ridge regression with λ=0.06 demonstrated better generalization to the test dataset, indicating its suitability for datasets with multicollinearity or overfitting risks.

By reducing the weight magnitudes, ridge regression provides a more robust solution compared to OLS, particularly when the number of predictors is high or when there is noise in the data.

Problem 4: Using Gradient Descent for Ridge Regression Learning

Objective:

The objective of this problem is to implement ridge regression using gradient descent. Unlike the closed-form solution (used in Problem 3), gradient descent avoids the explicit computation of the matrix inverse, making it computationally efficient for large datasets. The performance of gradient descent is evaluated using the scipy.optimize.minimize function.

Implementation:

- The objective function for ridge regression, including the gradient, was implemented in the function regressionObjVal.
- The scipy.optimize.minimize function was used to iteratively estimate the weights w by minimizing the regularized squared error.

Comparison:

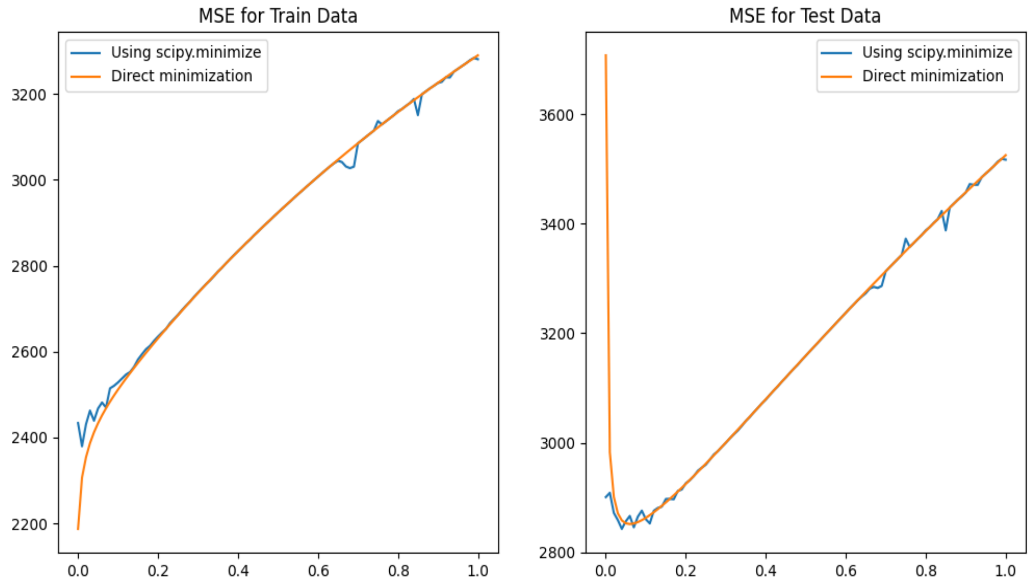
- The same regularization parameters (λ) from Problem 3 were used, ranging from 0 to 1 in increments of 0.01.
- For each λ, the MSE for training and testing datasets was calculated using both gradient descent and the direct minimization approach from Problem 3.

Evaluation:

- The train and test MSE values for gradient descent were compared against the results obtained using direct minimization.

- The MSE values were plotted for both methods across the range of λ.

Results:



MSE Trends:

- The train and test MSE values obtained using gradient descent closely align with those from direct minimization, as shown in the plots.
- Both methods produce a similar pattern of increasing train MSE and a U-shaped curve for test MSE as λ increases.

Optimal λ:

- The minimum test MSE was observed at λ=0.06, the same as identified in Problem 3. This reaffirms the effectiveness of gradient descent in optimizing the model.

Comparison with Direct Minimization:

- Gradient descent provided comparable results to direct minimization, with minor numerical deviations due to optimization tolerance.
- Gradient descent eliminates the need for matrix inversion, making it suitable for large-scale datasets where direct minimization might be computationally expensive.
- The plots clearly show the alignment between the two methods. The train MSE increases gradually with λ, while the test MSE follows a U-shaped curve, indicating the balance between bias and variance achieved through regularization.

Conclusion:

Gradient descent proves to be an efficient and reliable method for implementing ridge regression, especially for large datasets. Its results are consistent with those obtained through direct minimization. The optimal value of λ=0.06 achieves the lowest test MSE, emphasizing the importance of regularization in improving model generalization. Both methods perform similarly in this experiment, highlighting the robustness of gradient descent as an alternative to direct minimization.

Problem 5: Non-linear Regression

Objective:

The goal of this problem is to investigate the effect of increasing polynomial degrees (p) for ridge regression. Non-linear feature mappings were applied to a single input variable, and the model's performance was evaluated using different degrees of regularization (λ).

Implementation:

Feature Mapping:

- The mapNonLinear function was implemented to transform the single attribute into polynomial features [1,x,x^2,x^3………,x^p] for degrees p=0,1,2,…,6.

Training Ridge Regression:

Ridge regression weights were computed for each p using:

- No Regularization (λ=0).
- Optimal Regularization (λ=0.06), as determined in Problem 3.

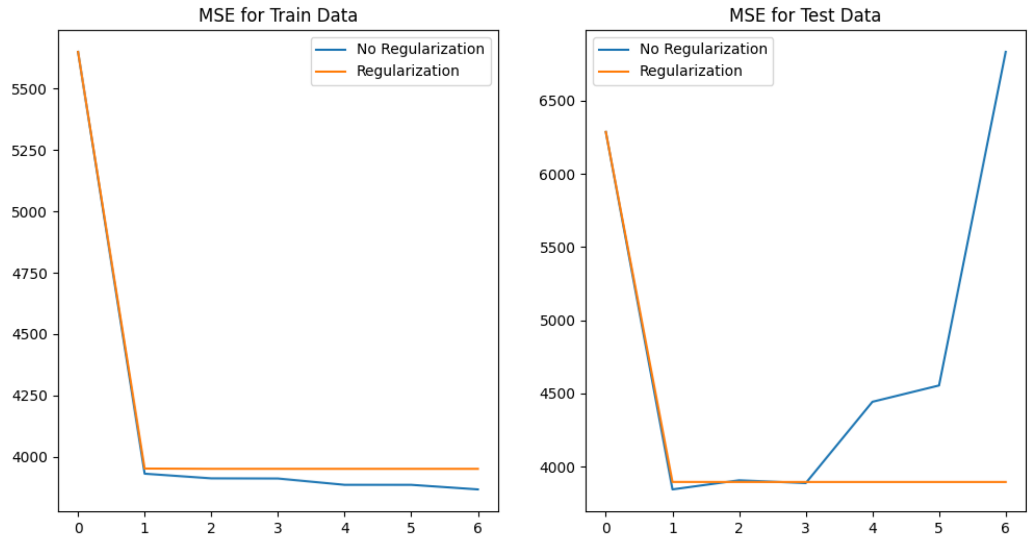
Error Calculation:

- The Mean Squared Error (MSE) was calculated on both training and test datasets for each p under both λ settings.

Evaluation:

- The train and test MSE values were plotted against p to analyze performance trends.
- The optimal p was determined for both λ=0 and λ=0.06 based on the minimum test MSE.

Results:



MSE Analysis:

No Regularization (λ=0):

- Train MSE consistently decreased with increasing p, as higher p allowed better fitting of the training data.
- Test MSE initially decreased but increased after p=3, indicating overfitting. The lowest test MSE occurred at p=3.

Regularization (λ=0.06):

- Regularization significantly mitigated overfitting, resulting in stable test MSE values across p.
- Test MSE reached its minimum at p=3, striking a balance between model complexity and regularization.

Optimal Polynomial Degree (p):

λ=0:

- p=3 resulted in the lowest test MSE, but the test performance deteriorated for higher values of p.

λ=0.06:

- p=3 also resulted in the lowest test MSE, demonstrating that regularization improves model robustness.

Plots:

- The left plot (train MSE) shows that the model fits the training data better with higher p, particularly without regularization (λ=0).
- The right plot (test MSE) shows the U-shaped curve, where test MSE initially decreases and then increases as p rises, with λ=0.06 yielding more stable results.

Conclusion:

The optimal polynomial degree is p = 3, as it minimizes the test MSE for both regularized and unregularized settings. Regularization (λ=0.06) ensures better generalization by stabilizing test performance and preventing overfitting. This highlights the importance of controlling model complexity through both polynomial degree (p) and regularization.

Problem 6: Interpreting Results:

Objective:

The objective of this section is to provide a comprehensive comparison of the various regression approaches used in this assignment and offer actionable recommendations for predicting diabetes levels using the provided input features. The evaluation considers training and testing errors across different models and settings to determine the most reliable approach.

Comparison of Approaches

Linear Regression (Ordinary Least Squares - OLE):

- Without Intercept: The MSE for training and testing data was significantly higher compared to models with an intercept. This is due to the inability of the model to account for the inherent bias in the data.
- With Intercept: Adding an intercept reduced the MSE drastically, making the model more effective. However, the lack of regularization made the model sensitive to noise and prone to overfitting for complex data.

Ridge Regression:

Ridge regression introduced a regularization parameter (λ) to control model complexity and prevent overfitting.

By varying λ from 0 to 1, it was observed that:

- Training MSE consistently increased with higher regularization due to constrained model flexibility.
- Testing MSE initially decreased and reached an optimal point at λ=0.06, beyond which testing error increased due to underfitting.
- Ridge regression effectively balances bias and variance, achieving better generalization than OLE.

Gradient Descent for Ridge Regression:

- Gradient-based optimization for ridge regression produced results consistent with direct minimization but required computational effort.
- The MSE trends on training and testing data followed the same patterns as in Problem 3, validating the correctness of both approaches.
- The computational feasibility of gradient descent makes it practical for large datasets where direct matrix operations may be computationally expensive.

Non-linear Regression:

The use of polynomial features (p) allowed the exploration of non-linear relationships in the data.

Without regularization (λ=0):

- The model achieved the lowest test error at p=3, but higher values of p led to severe overfitting.

With regularization (λ=0.06):

- The test error stabilized across different p values, with the lowest error still occurring at p=3.
- This highlights the importance of regularization in non-linear regression to prevent overfitting while capturing complex patterns.

Recommendations:

Model Selection:

- Ridge Regression with regularization (λ=0.06) and polynomial degree p=3 is the most reliable approach for predicting diabetes levels. It achieves a balance between training and testing errors, ensuring both accuracy and generalization.

Metrics for Evaluation:

- The Mean Squared Error (MSE) should be the primary metric for evaluating model performance as it quantifies the average prediction error and directly reflects the model’s ability to generalize.

Key Takeaways:

- Adding an intercept in linear regression is essential to capture the bias in the data.
- Regularization (as in ridge regression) is critical for controlling overfitting and improving model robustness.
- For datasets with potential non-linear relationships, incorporating polynomial features is beneficial, but it must be paired with regularization to prevent overfitting.

Conclusion:

The analysis demonstrates that ridge regression with regularization and non-linear mappings provides the best framework for predicting diabetes levels. This combination balances model complexity and generalization, making it suitable for real-world applications where unseen data performance is critical.