Faculty of Engineering and Technology

Electrical and Computer Engineering Department

MACHINE LEARNING AND DATA SCIENCE ENCS5341

Assignment #2

Prepared by
Jana Abu Nasser
1201110

Instructor

Dr. Yazan Abu Farha

Section 1

BIRZEIT
22th December – 2024

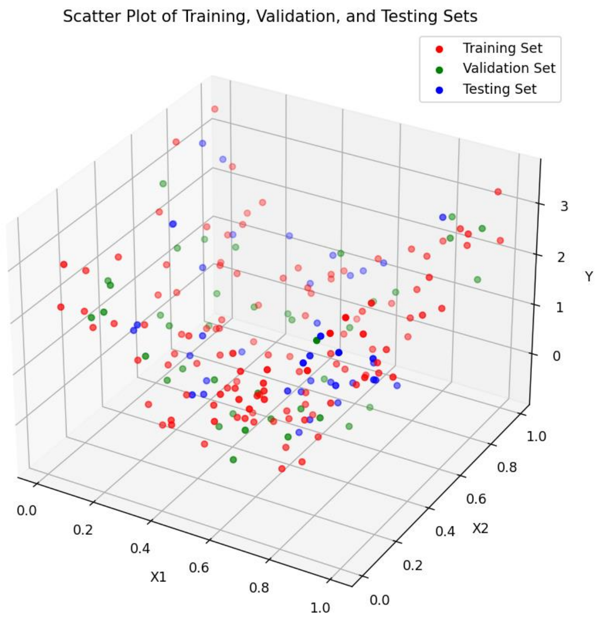
Contents

Model Selection and Hyper-parameters Tunning

The data_reg.csv file contains a set of 200 examples. Each row represents one example which has two attributes x1 and x2, and a continuous target label y.

Part 1:

The data was read from the CSV file and subsequently split into three sets: the training set, consisting of the initial 120 examples; the validation set, comprising the subsequent 40 examples; and the testing set, encompassing the final 40 examples. A scatter plot was generated to visualize the examples from the three sets, with each set encoded in a distinct color. Notably, the plot was presented in a three-dimensional format, where the x and y axes denoted the x1 and x2 features, while the z-axis represented the target label y.



The scatter plot shows three sets of data points: training, validation, and testing. Each set is encoded with a different color for easy differentiation. The x and y axes represent the x1 and x2 features, while the z-axis represents the target label y.

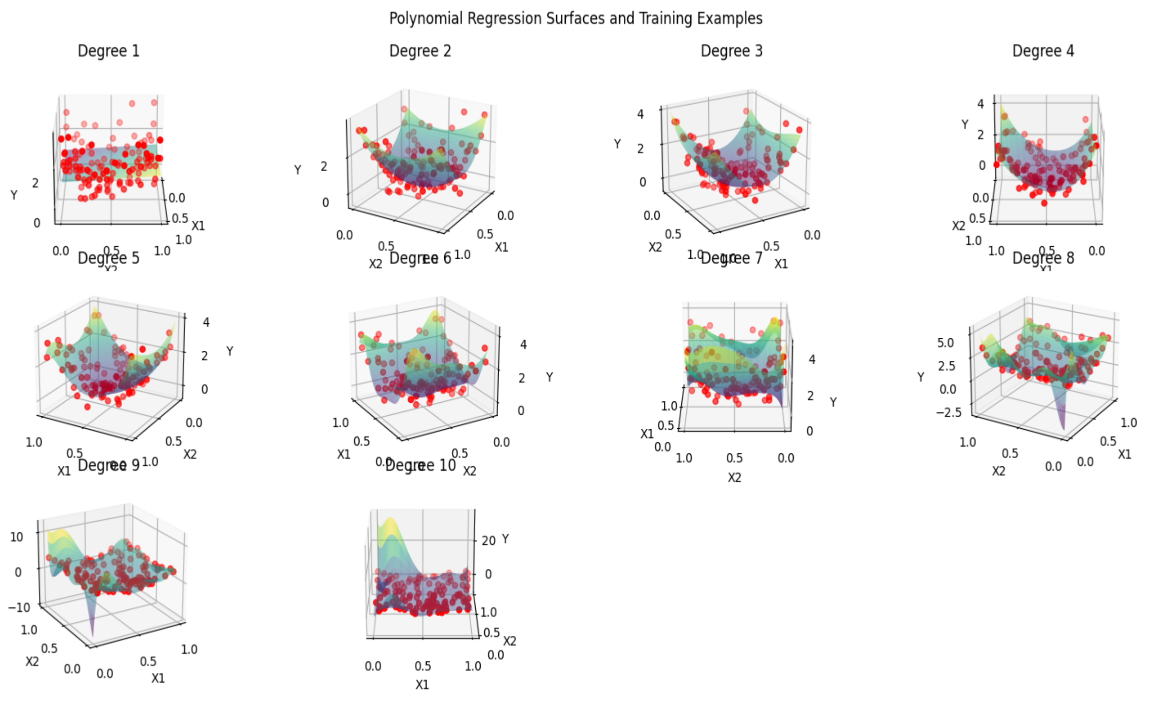
The 3D scatter plot reveals several key insights about the data. While the data points are distributed throughout the space, they exhibit clustering tendencies, suggesting potential underlying patterns. The relationships between features x1 and x2 are not straightforward, with hints of linear trends within isolated regions but a more intricate overall structure.

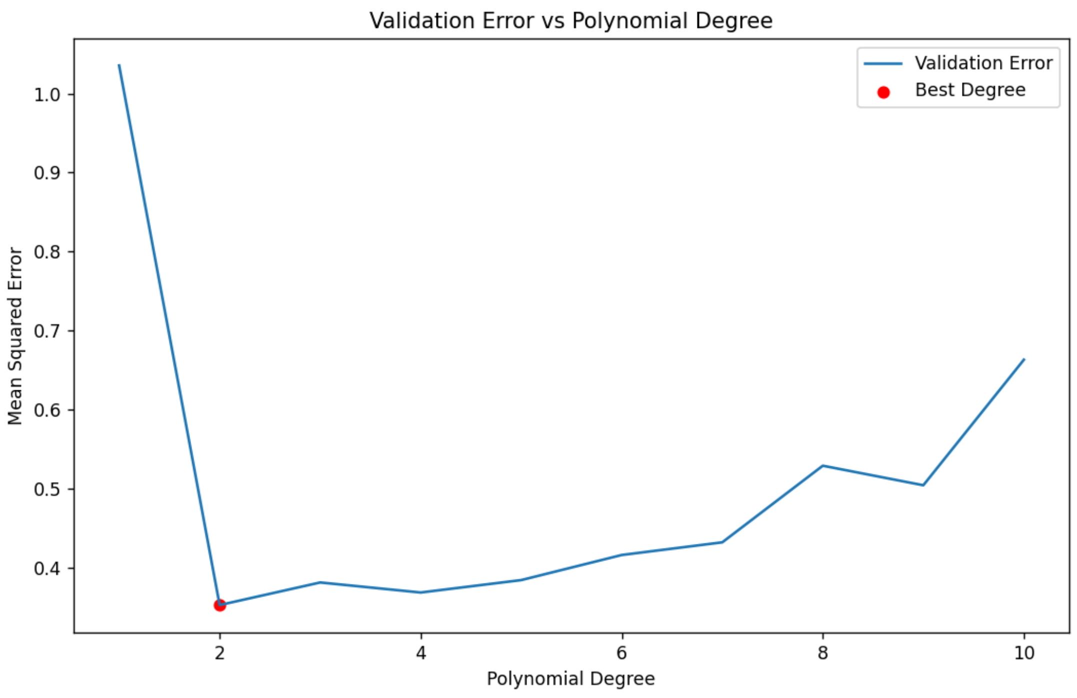
Notably, the target label y appears to vary smoothly across different combinations of x1 and x2, implying a continuous relationship between features and target. However, the separation between training, validation, and testing sets could be more distinct, potentially indicating room for improvement in the splitting strategy. Lastly, a few outliers warrant further investigation to understand their deviation from the rest of the data.

Overall, the plot suggests that the data is complex and potentially nonlinear. This means that choosing the right machine learning model for this data will be important. Linear models might not be the best choice, as the relationships between the features and the target variable might not be linear. Instead, non-linear models might be more appropriate.

Part 2:

Polynomial regression was applied to the training set across polynomial degrees ranging from 1 to 10. The determination of the optimal polynomial degree was justified by plotting the validation error versus polynomial degree curve. The surface of the learned function for each model was concurrently plotted alongside the training examples on the same graph. Notably, PolynomialFeatures and LinearRegression from the scikit-learn library were utilized for these computations.

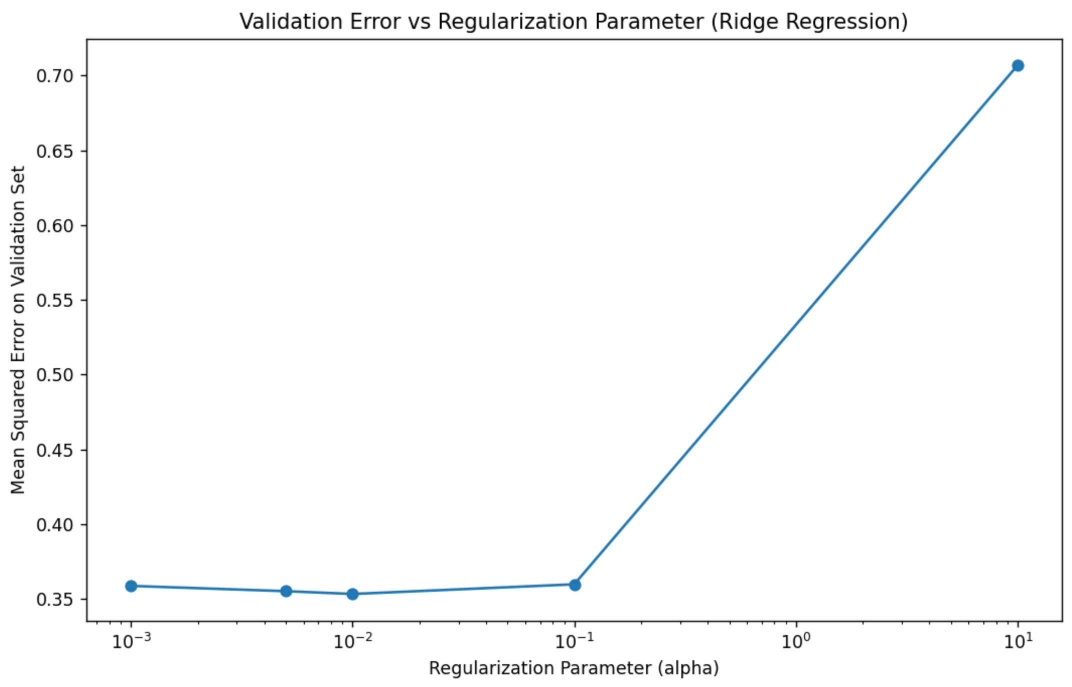




The validation error curve reveals that a polynomial degree of 2 is optimal for this model. Initially, increasing the degree from 1 to 4 improves fit, as indicated by decreasing validation error. However, degrees beyond 6 lead to overfitting, as validation error rises. This suggests that simpler models (degrees 1-2) might underfit the data, while more complex models (degrees 6-10) capture noise rather than true patterns. Degree 4 strikes a balance, fitting the training data well without sacrificing generalization to unseen data.

Part 3:

Ridge regression was employed on the training set to accommodate a polynomial of degree 8. The best regularization parameter was chosen from a set of options, including {0.001, 0.005, 0.01, 0.1, 10}. A plot illustrating the mean squared error (MSE) on the validation set versus the regularization parameter was generated. The scikit-learn library's implementation of Ridge regression facilitated these computations.



The graph displays the relationship between validation error and the regularization parameter in ridge regression. Validation error represents the model's error on unseen data, while the regularization parameter controls the penalty for a complex function.

The graph indicates that the validation error reaches its minimum around a regularization parameter of 0.01, suggesting its optimal value. A regularization parameter that is too small leads to overfitting on the training data, capturing both noise and signal, resulting in poor performance on unseen data. Conversely, a parameter that is too large leads to underfitting, where the model fails to grasp underlying relationships, also resulting in poor performance on unseen data.

The lowest validation error around a regularization parameter of 0.01 suggests the model effectively learns underlying data relationships without overfitting to noise. This is advantageous, indicating the model's likelihood to perform well on unseen data.

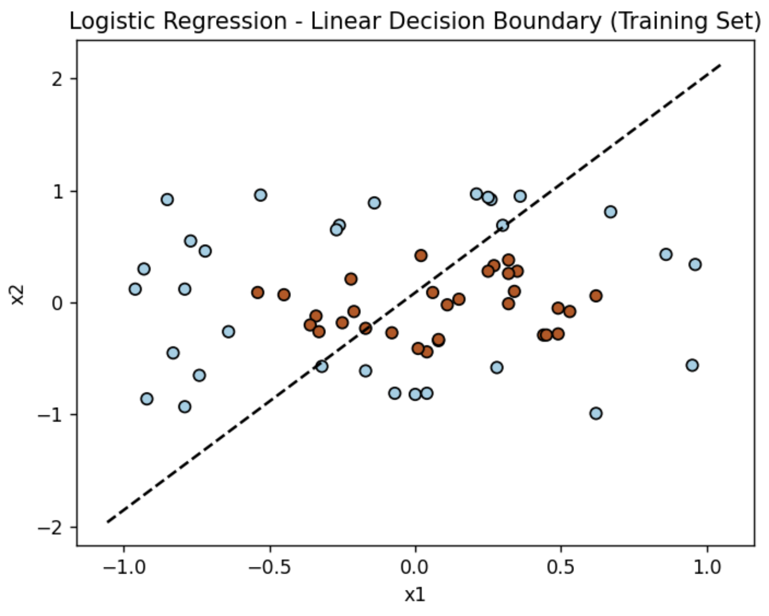
Overall, the plot suggests that ridge regression enhances polynomial regression models' performance. Selecting the appropriate regularization parameter prevents overfitting to training data and underfitting to unseen data, resulting in a model that performs well on both.

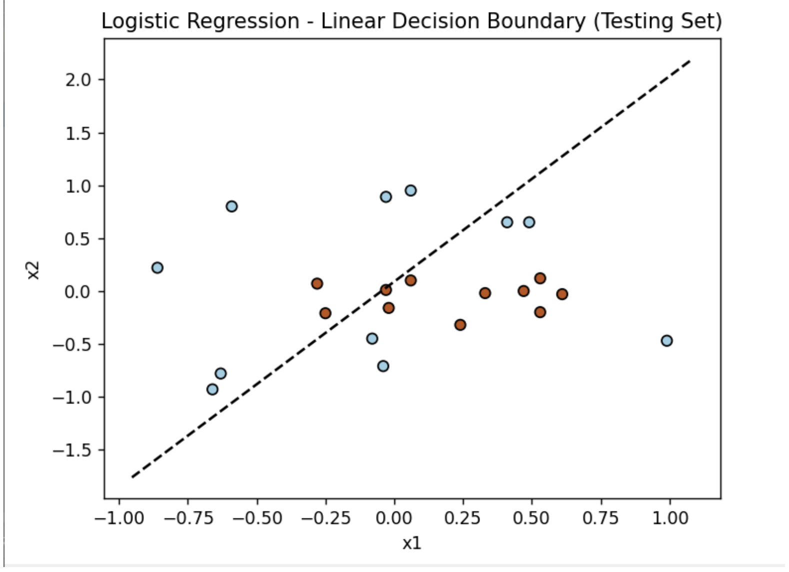
Logistic Regression

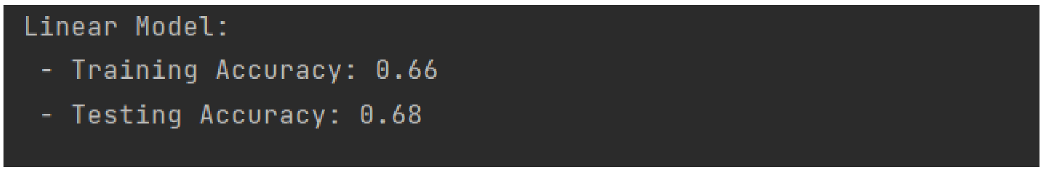
The train_cls.csv file contains a set of training examples for a binary classification problem, and the testing examples are provided in the test_cls.csv file.

Part 1:

The logistic regression model was learned with a linear decision boundary utilizing the scikit-learn library's implementation. The decision boundary of the learned model was drawn on a scatterplot of the training set. The training and testing accuracy of the learned model were computed.







```
Linear Model:
 - Training Accuracy: 0.66
 - Testing Accuracy: 0.68
```

The graph clearly shows a straight line acting as the decision boundary, attempting to separate the two classes of data points.

The training accuracy of 0.66 and testing accuracy of 0.68 suggest that the linear model has a moderate ability to classify the data points correctly. However, there's room for improvement. Several data points are located on the wrong side of the decision boundary, indicating misclassifications by the model.

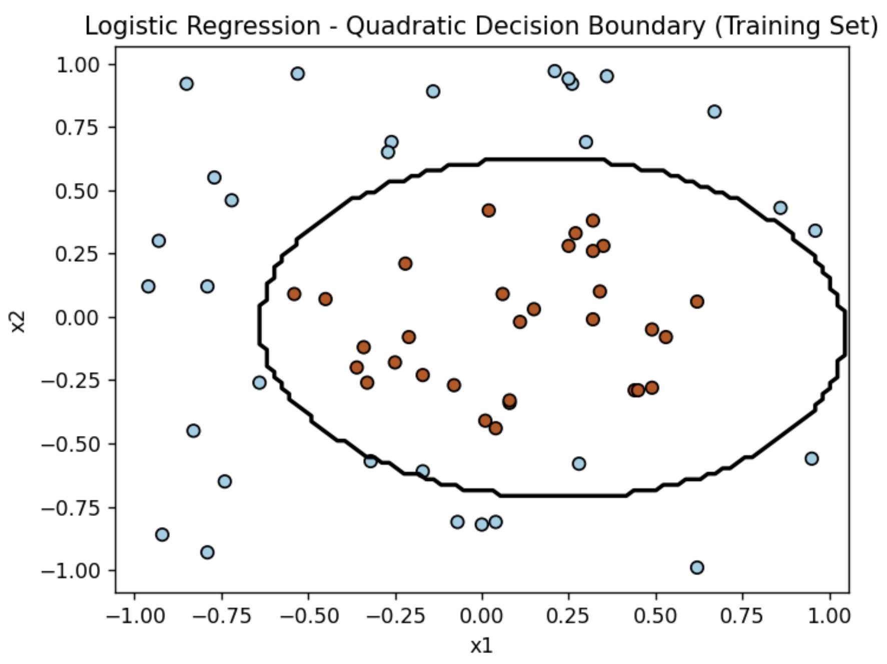
The linear boundary runs roughly diagonally across the plot, with a negative slope. This means that as x1 increases, the probability of a point belonging to the blue class (class 1) decreases.

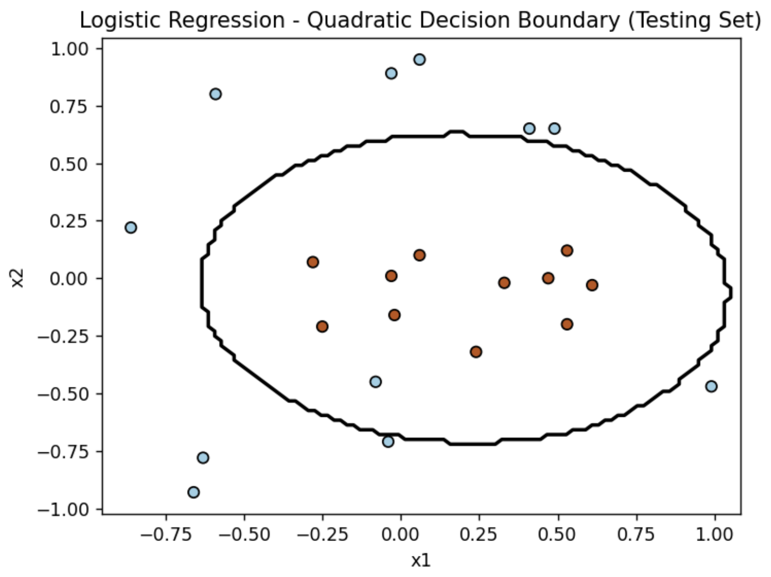
The data points are not evenly distributed in space. There's a higher density of points in the lower-left and upper-right regions, potentially influencing the boundary's position.

The presence of misclassified points, especially those clustered together, suggests that the relationship between the features and target variable might not be purely linear. A quadratic or more complex model might capture the patterns more accurately.

Part 2:

In the subsequent phase, a logistic regression model with a quadratic decision boundary was learned, employing the scikit-learn library's implementation. The decision boundary of the acquired model was drawn on a scatterplot of the training set, and the training and testing accuracy of the model were subsequently computed.





```
Quadratic Model:
 - Training Accuracy: 0.97
 - Testing Accuracy: 0.95
```

The graph clearly depicts a downward-opening parabola, confirming the model's ability to capture non-linear relationships. The high training accuracy of 0.97 and testing accuracy of 0.95 indicate excellent performance on both seen and unseen data.There are only a few misclassified points, primarily near the edges of the boundary.

The parabola opens downward, suggesting a negative quadratic relationship between the features and the target variable. Its vertex is located around (-0.25, 0.5) on the x1-axis, which is likely a region of high class overlap. data points are more concentrated in the central region, with sparser distribution towards the edges. The boundary closely follows this distribution, especially in the denser areas. The misclassified points appear to be clustered near the edges of the boundary, suggesting potential challenges in classifying points in these regions.

Part 3:

The learned models in 1 and 2 are to be commented on in terms of overfitting/underfitting.

- Linear Model: is underfitting the model might be too simple and not capturing enough complexity in the data. This could be due to using a simple linear model when a non-linear model might be more appropriate., The amount of training data might be insufficient for the model to learn the relationships between features and the target variable effectively. And the presence of outliers or noise in the data could be making it difficult for the model to learn a stable and generalizable pattern.

- Quadratic Model :is Goodfitting, the training accuracy of 0.97 is very high, indicating that the model has almost perfectly memorized the training data, including its noise and random fluctuations. This close fit to the training data can make it less generalizable to unseen data.The quadratic boundary closely follows the contours of the training data, even in areas where the classes overlap or have a less clear separation. This suggests that the model might be capturing patterns that are specific to the training data rather than representing general trends.