Introduction

In this project, we explore the application of linear regression and logistic regression models for binary classification tasks. The objective is to predict binary labels {0, 1} based on two-dimensional input vectors. We have two datasets, Dataset A and Dataset B, generated from bivariate normal distributions with distinct characteristics.

Problem Formulation

The problem is formulated as a binary classification task, where the input is a two-dimensional vector, and the output is either 0 or 1. The goal is to train classifiers that can accurately predict the binary labels for each input vector.

Datasets

1. Dataset A:
   - Positive samples are generated from a bivariate normal distribution (mean = [0.2, 0.2], covariance matrix same as positive samples).
   - Negative samples are generated from another bivariate normal distribution (mean = [-0.2, -0.2], covariance matrix same as positive samples).
   - The dataset contains a total of 400 samples, with 200 positive and 200 negative samples.

2. Dataset B:
   - Half of the positive samples from Dataset A are shifted to the upper right quadrant (mean = [3.2, 3.2], covariance matrix same as positive samples).
   - The other half remains in the same position as in Dataset A.
   - The dataset size and class distribution are the same as Dataset A.

Methodology

We approach the binary classification problem using two methods: linear regression and logistic regression. The implementation is carried out in Python, utilizing the scikit-learn library for logistic regression. For linear regression, we use custom functions for training and prediction.

Linear Regression Classifier

1. Training:
   - The linear regression classifier is trained using the closed-form solution.
   - The training data is augmented with an additional bias term.
   - The weight and bias values are computed using the normal equation.

2. Prediction:
   - For prediction, the input data is augmented with the bias term.
   - The prediction is obtained by computing the dot product of the weight vector and input vector.
   - The predicted values are thresholded using 0.5 to determine the final binary labels.

Logistic Regression Classifier

1. Training:
   - We implement logistic regression training using gradient descent.
   - The learning rate is set to 0.1, and the number of epochs is set to 5000 for stable convergence.
   - The weights and bias are updated iteratively based on the logistic function.

2. Prediction:
   - The logistic regression model predicts probabilities of positive class labels.
   - The predicted probabilities are thresholded using 0.5 to obtain the final binary labels.

Results

The training accuracy of the classifiers on both datasets is as follows:

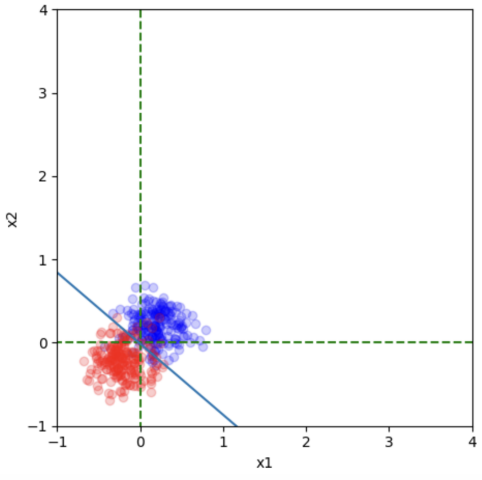
1. Training accuracy of linear regression on Dataset A: 0.9325
2. Training accuracy of logistic regression on Dataset A: 0.9025
3. Training accuracy of linear regression on Dataset B: 0.75
4. Training accuracy of logistic regression on Dataset B: 0.95

Decision Boundaries

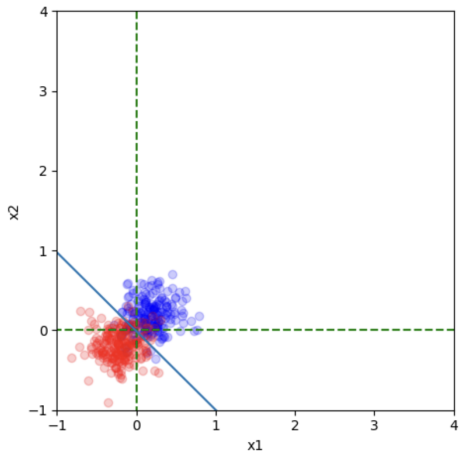
Below are the decision boundary plots for both classifiers on both datasets:

Dataset A

Linear Regression

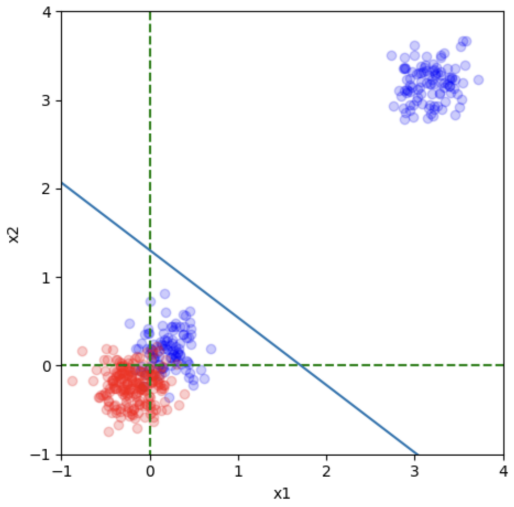


Logistic Regression

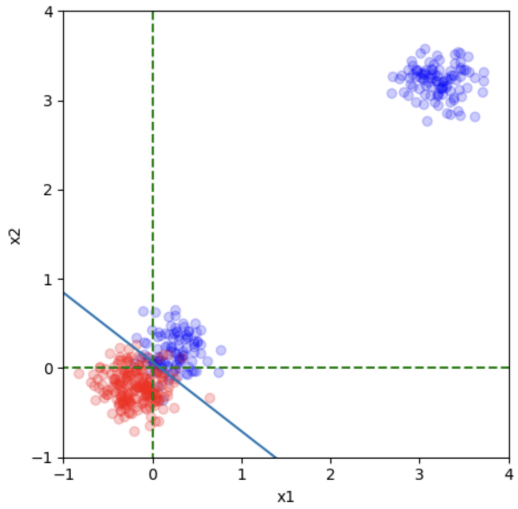


Dataset B

Linear Regression



Logistic Regression



Conclusion

The project demonstrates the performance of linear regression and logistic regression models in binary classification tasks. Logistic regression outperforms linear regression on both datasets, showcasing the superiority of logistic regression in handling binary classification problems. The decision boundary plots illustrate the effectiveness of each classifier in separating positive and negative samples.

The implementation of both classifiers using Python and scikit-learn provides valuable insights into the application of machine learning models in real-world scenarios. Overall, the project successfully highlights the importance of choosing appropriate models for specific classification tasks.