William Woodard

801069657

Matthew Zahn

801055398

GitHub: https://github.com/willwoodard16/Intro-to-Ml-4105/tree/main/Final%20Project

https://github.com/mzahn4/IntroToML/tree/main/FinalProject

Car Acceptability Classification Final Report

Introduction

The objective of this project is to create a machine learning algorithm to predict whether a car is an acceptable purchase. Car acceptability will be based on three main subfactors: price, tech, and comfort respectfully. These three subfactors will be based on factors such as safety rating, cost of maintenance, number of doors, etc. This algorithm will allow people to quickly determine if a car is worth purchasing or allow people to compare separate cars to determine which is the best fit for them.

Project Details

This project was completed using the Car Evaluation data set obtained form UCI, which can be found at http://archive.ics.uci.edu/ml/datasets/Car+Evaluation. This dataset was developed for the demonstration of DEX, M. Bohanec, V. Rajkovic: Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990.). It was derived from a simple hierarchical decision model.

This dataset was used to create a machine learning algorithm to determine the viability, or acceptability, of purchasing a car. Acceptability is based on six key characteristics; buying price, maintenance cost, number of doors, number of passengers that can fit in the car, the size of the trunk space, and the safety rating.

The algorithm will incorporate multiple methods and approaches to find the most accurate prediction model. The methods used are logistic regression, logistic regression using K-fold method, Gaussian Naive Bayes classifier, Gaussian Naive Bayes classifier with K-fold method, and gradient descent. Also, Principal Component Analysis (PDA) and Linear Discriminant Analysis (LDA) were used to extract features and reduce dimensionality. After each method was implemented, the values for accuracy, precision, and recall were output to determine the most optimal method. Plots for the loss over iteration were also made for each variable of the project using two different values of alpha.

Project Results

In this section, all the results from running the code will be shown. This includes all graphs, confusion matrices, and values for accuracy, precision and recall. Results will be discussed in the discussion section of the report.

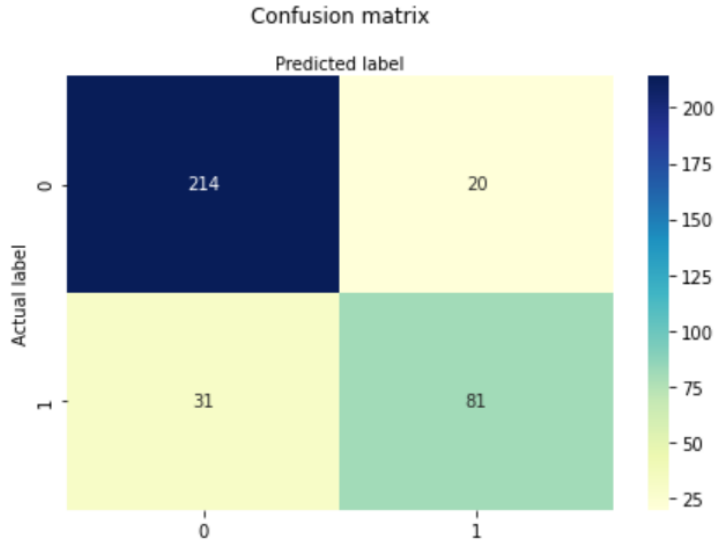
Logistic Regression:

The following section shows the results obtained from the logistic regression portion of the project. The accuracy, precision, and recall were all found to be around 85% for this method.

Accuracy: 0.8526011560693642

Precision: 0.8503283783610166

Recall: 0.8526011560693642



Logistic Regression w/ K-Fold Method:

Logistic regression was repeated using the K-Fold method. The highest accuracy was found to be at K=12, with an accuracy of 87.34%. The precision and recall at K=12 was found to be 80.83% and 74.62% respectively.

K = 12

Accuracy: 0.8733570714642679

Precision: 0.8083306226944242

Recall: 0.7462431161207093

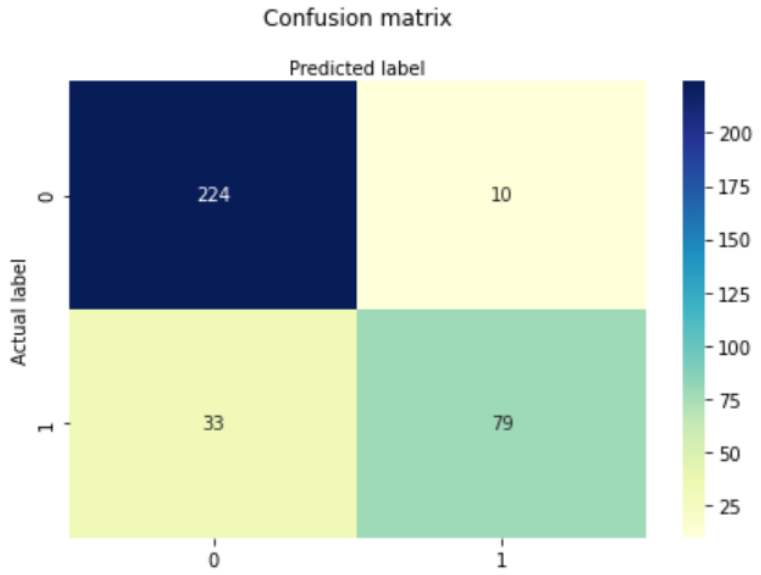
Gaussian Naive Bayes:

This section shows the results found using the Gaussian Naive Bayes classification method. The accuracy, precision, and recall were all found to be around 87.6-87.7% for this method.

Accuracy: 0.8757225433526011

Precision: 0.8767891263874993

Recall: 0.8757225433526011



Gaussian Naive Bayes w/ K-Fold Method:

Gaussian Naive Bayes was repeated using the K-Fold method. The highest accuracy was found at K=14. This accuracy was at 89.15%, with precision and recall at 89.26% and 71.56% respectively. This is the highest accuracy found out of all methods tested for this project.

K = 14

Accuracy: 0.8914804016844834

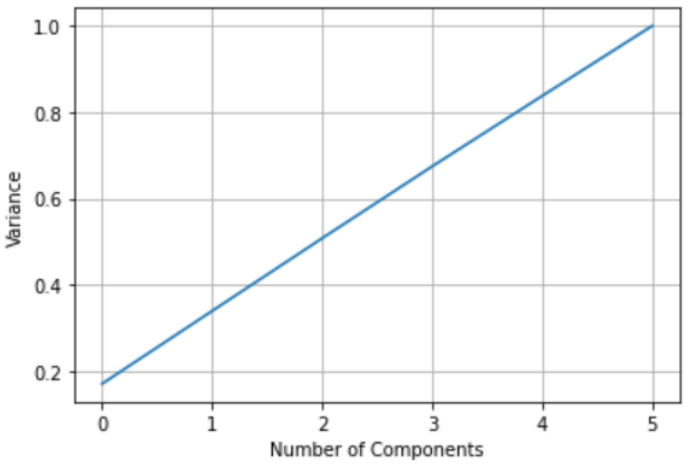
Precision: 0.8926676699096365

Recall: 0.7156151679764519

PCA:

This section describes the results obtained from the PCA portion of this project. The graph below shows how the variance changes with the number of components. The higher the variance the better as a high variance means more information is stored in each component.

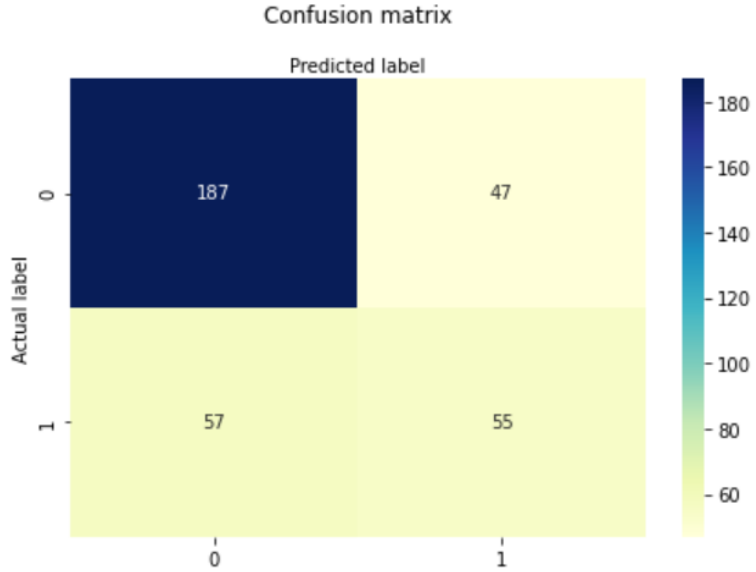
The accuracy, precision, and recall were found to be around 69-70% for this method. This is the lowest accuracy found out of all methods implemented for this project.



Accuracy: 0.6994219653179191

Precision: 0.6928561342095826

Recall: 0.6994219653179191



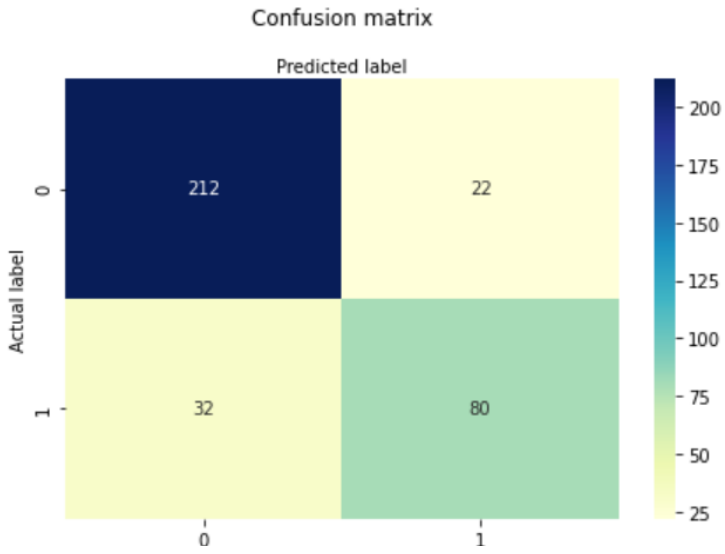
LDA:

This section describes the results found using the LDA method. The accuracy, precision, and recall were all found to be around 84% for this method.

Accuracy: 0.8439306358381503

Precision: 0.8414873198402832

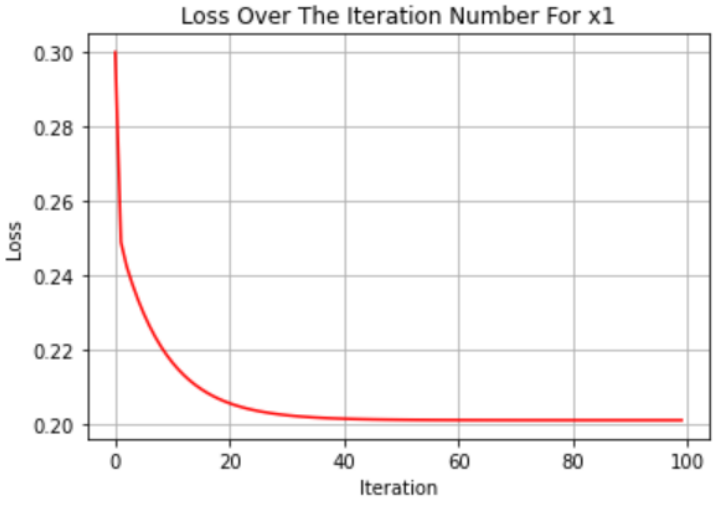
Recall: 0.8439306358381503



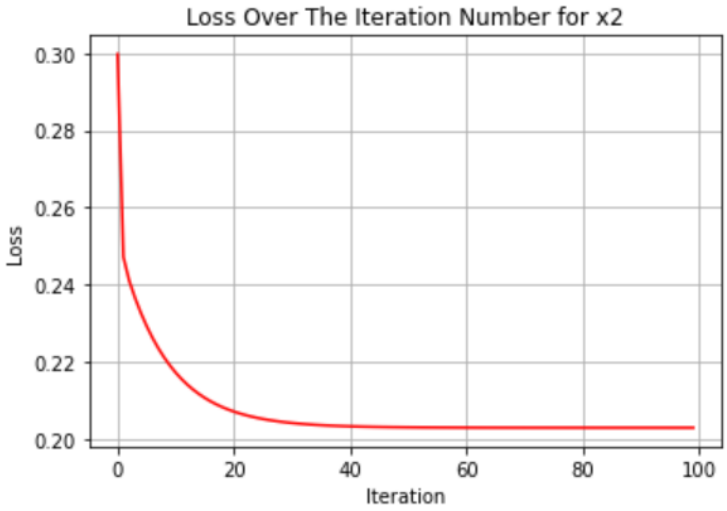
Gradient Descent (loss over iterations) for alpha = 0.1:

This section shows the results obtained from conducting gradient descent with an alpha value of 0.1. The following graphs show the loss over iteration for each variable in the project. Variable X1 corresponds to the buying price of the car. X2 corresponds to the maintenance cost. X3 refers to the number of doors the car has. X4 refers to the number of people the car can hold. X5 refers to the trunk space. Finally, X6 refers to the safety rating of the car.

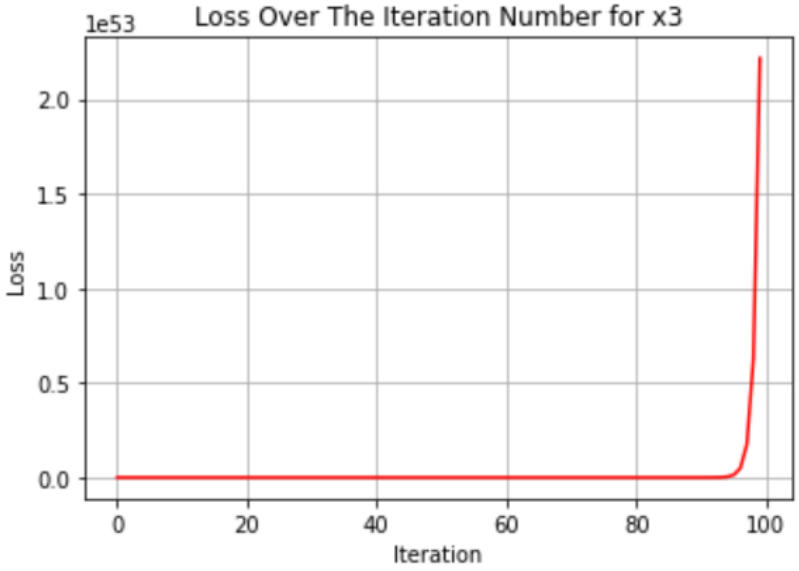
X1:



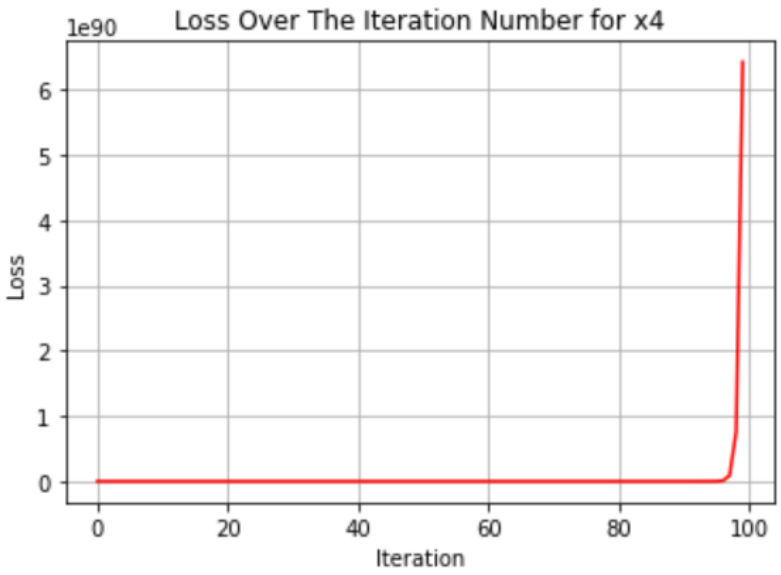
X2:


Loss Over The Iteration Number for x2
0.30
0.28
0.26
0.24
0.22
0.20
Loss
0
20
40
60
80
100
Iteration

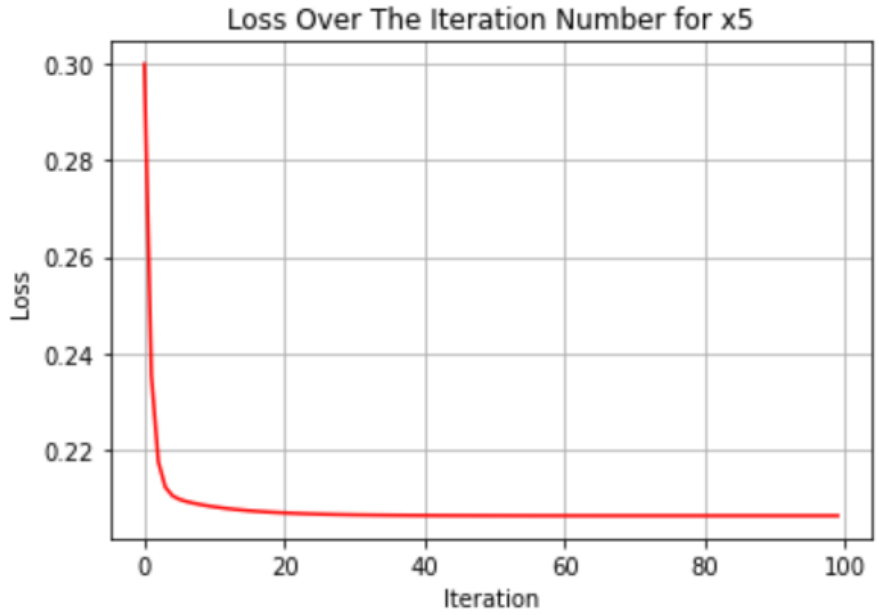
X3:


1e53
Loss Over The Iteration Number for x3
2.0
1.5
1.0
0.5
0.0
Loss
0
20
40
60
80
100
Iteration

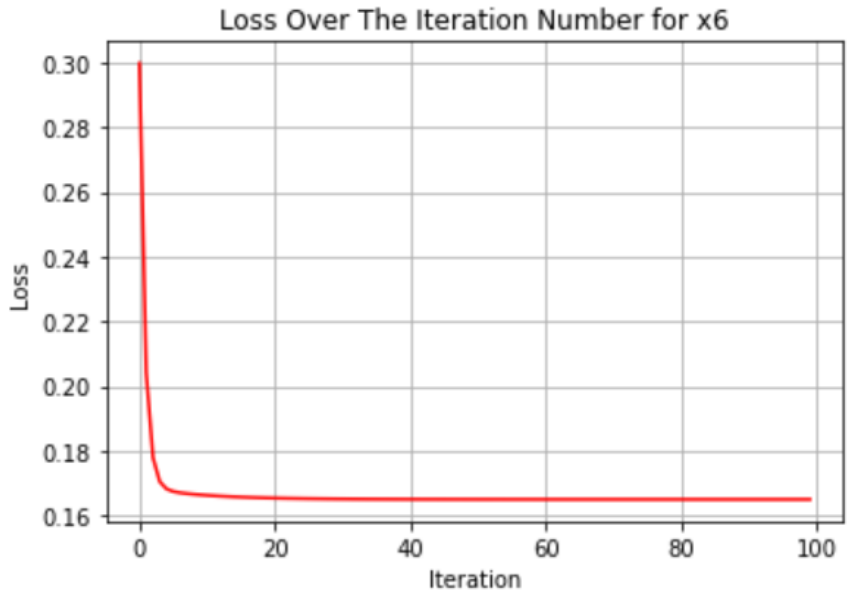
X4:


Loss Over The Iteration Number for x4
1e90
6
5
4
3
2
1
0
Loss
0
20
40
60
80
100
Iteration

X5:


Loss Over The Iteration Number for x5
0.30
0.28
0.26
0.24
0.22
Loss
0
20
40
60
80
100
Iteration

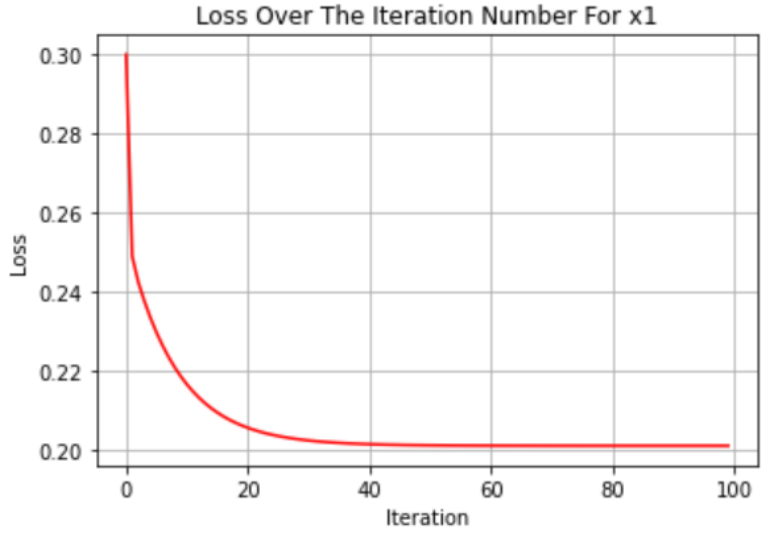
X6:



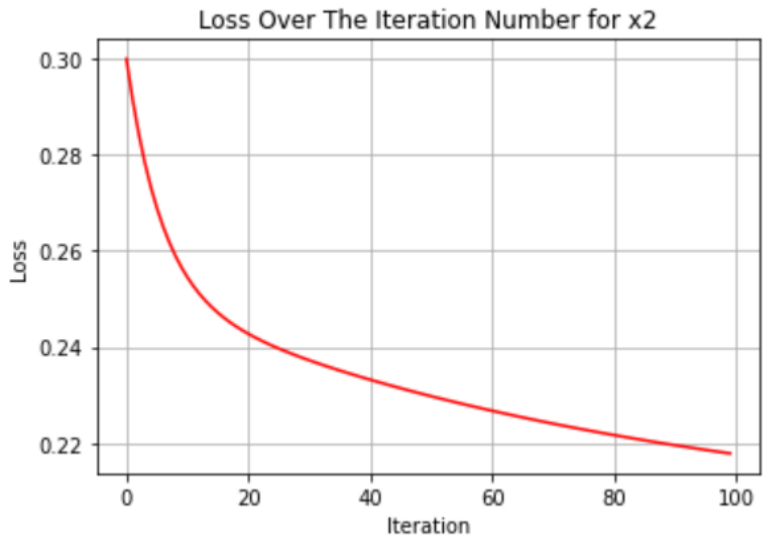
Gradient Descent (loss over iterations) for alpha = 0.01:

This section shows the results found using gradient descent with an alpha value of 0.01. The following graphs show the loss over iteration for each variable in the project. The variables are the same variables used for the gradient descent graphs using the alpha value of 0.1.

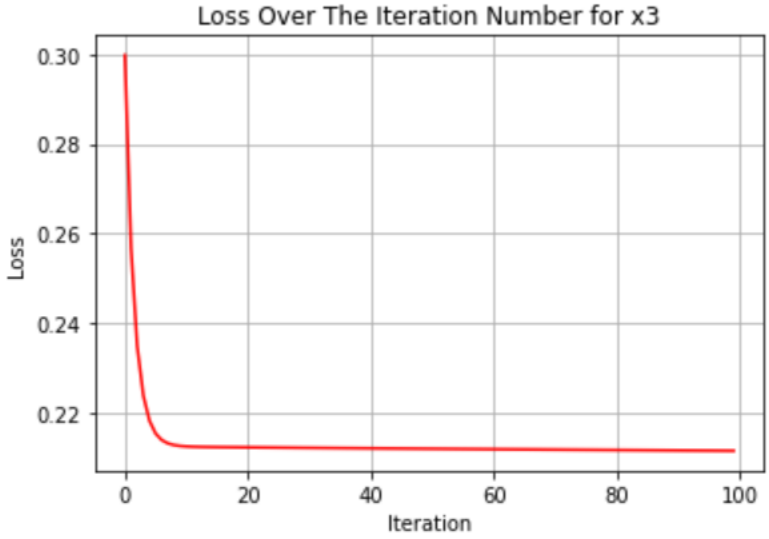
X1:



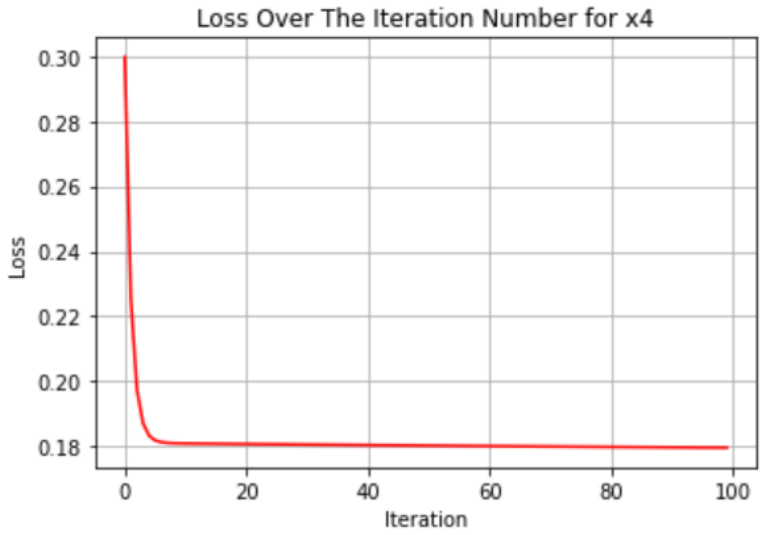
X2:


Loss Over The Iteration Number for x2
0.30
0.28
0.26
0.24
0.22
Loss
0
20
40
60
80
100
Iteration

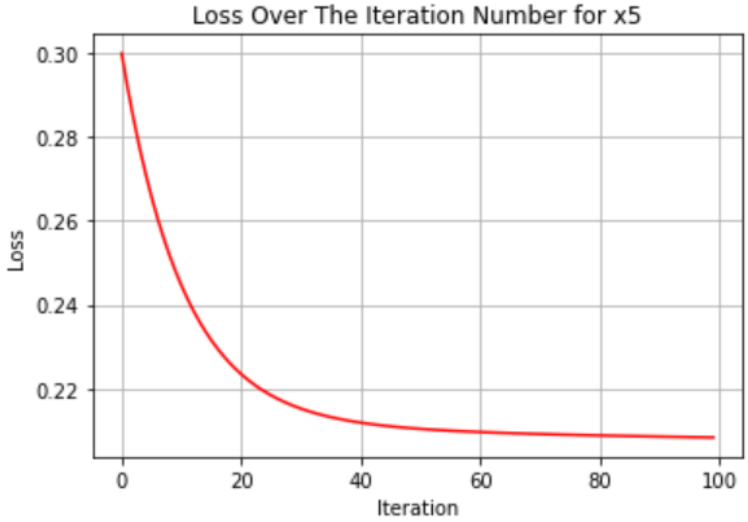
X3:


Loss Over The Iteration Number for x3
0.30
0.28
0.26
0.24
0.22
Loss
0
20
40
60
80
100
Iteration

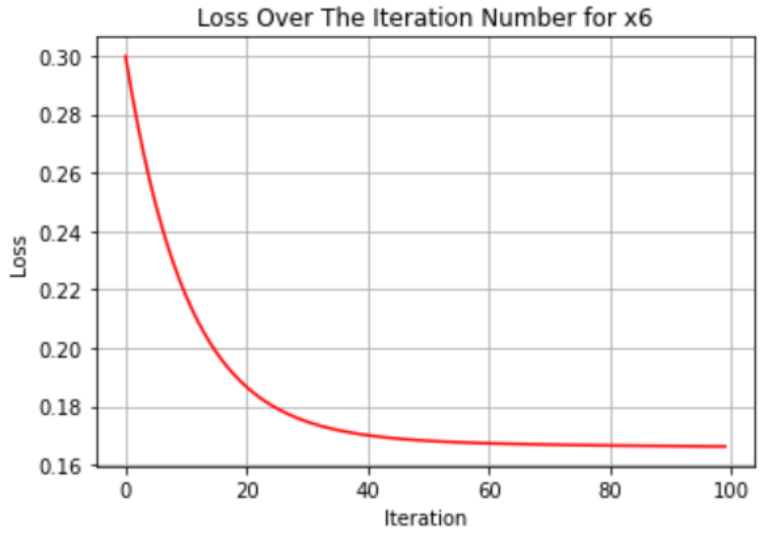
X4:


Loss Over The Iteration Number for x4
0.30
0.28
0.26
0.24
0.22
0.20
0.18
Loss
0
20
40
60
80
100
Iteration

X5:


Loss Over The Iteration Number for x5
0.30
0.28
0.26
0.24
0.22
Loss
0
20
40
60
80
100
Iteration

X6:



Discussion

This project started off as multivariate but was changed to binary after the presentation to minimize the bias towards unacceptable. This was due to the feedback we received from Dr. Tabkhi. Additionally, loss over iterations graphs were added for each of the individual variables which are buying price, cost of maintenance, number of doors, capacity, trunk space, and safety rating. The data was split into a 80-20 split between train and test for all of the training models used. After splitting the dataset, the data itself was standardized to have zero mean and unit variance.

After running the code multiple times to accommodate for the random state set when the dataset was set, we observed that Gaussian Naive Bayes performed better than logistic regression without any method to avoid overfitting. When implementing K-fold cross validation, we observed that Gaussian Naive Bayes was the most accurate and had the highest precision, but a lower recall value.

When implementing a method to reduce dimensionality, the following two were chosen: PCA and LDA. Using principal component analysis, we found very low values for accuracy, precision, and recall averaging in the mid to high 60's. Furthermore, when implementing linear discriminant analysis, we found much higher values in the mid to high 80's.

After running all the previous methods, gradient descent was implemented to obtain graphs for loss over iterations at two different alpha values. For the first alpha value, which was 0.1, you can see some errors in the graphs for x3 and x4. This is presumed to result from a very high alpha value which skewed that data which seemed to maximize loss. Below is a screenshot of the final values of the loss for each of the variables. In this you can clearly see the code error that shows a ridiculously high value for x3 and x4.

```
final loss for the first column: 0.20113036015205932
final loss for the second column: 0.20296290031600567
final loss for the third column: 2.2160407069501533e+53
final loss for the fourth column: 6.42380655133589e+90
final loss for the fifth column: 0.20652963888568068
final loss for the sixth column: 0.16499707812768777
```

For the second alpha value, which was 0.01, you can see the previous errors in the graphs for x3 and x4 were fixed. This is fixed by a lower value of alpha which seemed to play better with gradient descent and mean loss error calculations. Below is a screenshot of the final values of the loss for each of the variables. As you can see the final loss values are all around the same value which is very low for this data.

```
final loss for the first column: 0.20113036015205932
final loss for the second column: 0.2178840718943801
final loss for the third column: 0.21149317775625842
final loss for the fourth column: 0.17956480646028997
final loss for the fifth column: 0.20844060449792876
final loss for the sixth column: 0.16631268945694924
```

Conclusion

Since the dataset was initially biased towards unacceptable results, we reformatted the data to be binary which reduced the bias towards unacceptable values, which resulted in higher values of accuracy, precision, and recall. The objective of this project was to create a machine learning algorithm to predict whether a car is acceptable to purchase. This was based on buying price, cost of maintenance, number of doors, capacity, trunk space, and safety rating. Through the implementation of the various machine learning algorithms, we were able to determine the best method was Gaussian Naive Bayes classification without K-fold cross validation as it had the highest accuracy, precision, and recall values among all the rest of classification algorithms. Additionally, if a dimension reduction method was to be used for this dataset we found that linear discriminant analysis worked much better than principal component analysis. Finally, we ran the code through gradient descent to obtain loss over iterations where we found an alpha of 0.01 worked the best with our standardized data using descent and mean loss error calculations.