“In The Name Of God”

HW#2 Report

Professor: Dr.Azmifar

Presenters:

Narges Dehghan

Masome Jafari

Table of Contents

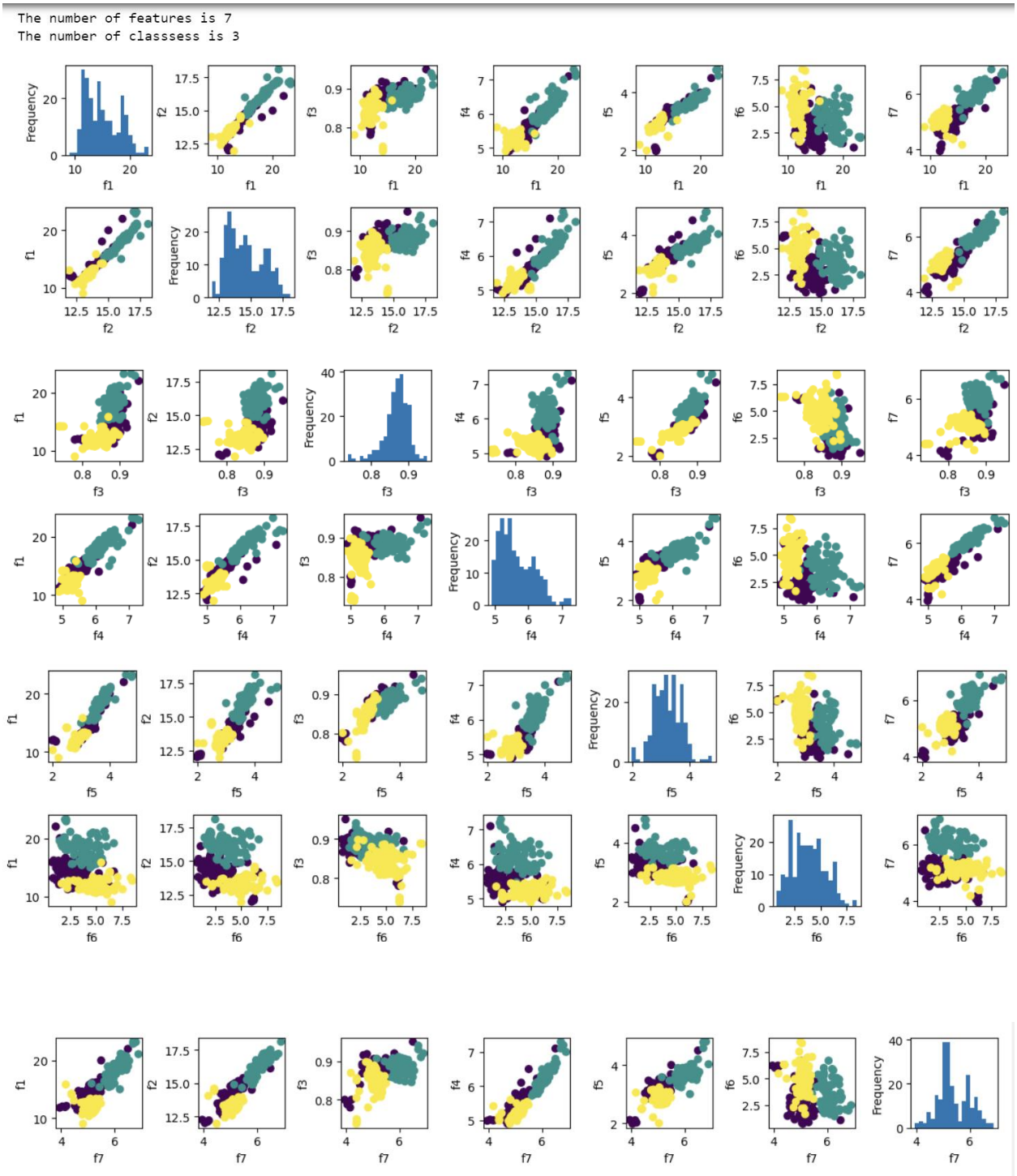
Abstract

This assignment encompasses a comprehensive exploration of OVO, OVR and SoftMax regression also binary classification , fundamental techniques in the realm of machine learning and data analysis. The primary objectives are twofold: to understand the fundamental concepts and application of these methods and to apply this knowledge to real-world datasets.

Logistic Regression for Multiclass Classification data

here we prepare our dataset for implementation. Put the result by the sequence of the questions.

Understanding the



Here we find number of outlier samples with different thresholds.

```
for threshold 2.75
class1 has 1 outliers.
class2 has 6 outliers.
class3 has 6 outliers.
--------------------------------
for threshold 3
class2 has 4 outliers.
class3 has 4 outliers.
--------------------------------
for threshold 2.5
class1 has 6 outliers.
class2 has 12 outliers.
class3 has 8 outliers.
--------------------------------
```

could you detect the outliers visually?

yes we can use scatter and visually detect not all but the obvious ones

What other methods can we use to detect the outliers?

Histogram can be used to detected outliers if we have normal distributed data a bar far from other are outliers

Do you get better classification performance (better accuracy) by omitting the outliers in any of

these three algorithms?

When we omit outlier the accuracy will get better.

softmax is more sensitive with outliers , ovo and ova is robust with outliers but finally we should calculate accuracy with outliers or without outlier to see which one is better result

Later in this homework you will need to implement the Mahalanobis distance for your Bayesian

classifier. Can you calculate any kind of distance between your samples now? why?

Here our implement is based on mahanobis distance.

Yes we can use other distance and calculate them but the important thing is usage of other distances

What is a nan value? How many nan values do you

have per feature?

Nan value when we have feature but we don’t have observation or value for it.or its unrepresntation for example the resut of divation/0 is nan

Implementation OVO & OVR

This code applies implements logistic regression for multiclass classification using a one-vs-all and one-vs-one approach.

2. Data Preprocessing:

The data is loaded from 'data.txt', and standardization (z-score normalization) is applied to the input features.

The dataset is then split into training and testing sets.

3. One-vs-All Logistic Regression:

The fit_onevsall function trains multiple binary classifiers, each distinguishing one class from the rest.

Sigmoid activation, gradient descent, and cost calculation functions are used iteratively.

The training progress is tracked by the cost for each class over iterations.

Training and testing accuracy are evaluated using the learned parameters.

4. One-vs-One Logistic Regression:

The fit_onevsone function constructs binary classifiers for all pairs of classes.

Similar to one-vs-all, it uses sigmoid activation, gradient descent, and cost calculation.The training progress is monitored by the cost for each class pair over iterations.

Training and testing accuracy are computed based on the learned parameters.

5. Prediction and Evaluation:

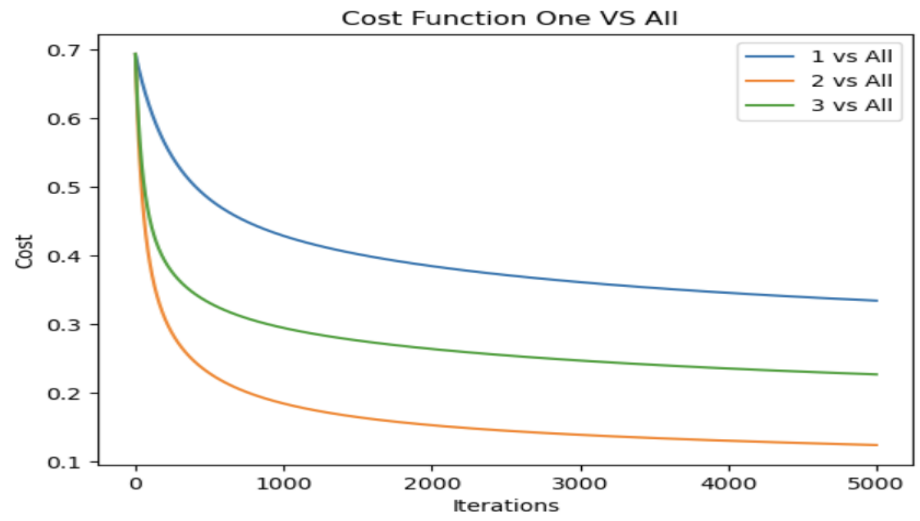
predict_ovr and predict_ovo functions predict the class labels using one-vs-all and one-vs-one strategies, respectively.Accuracy scores for both training and testing sets are computed and printed.

6. Cost Function Plots:

The code generates plots showing the cost function's progress over iterations for both one-vs-all and one-vs-one approaches.

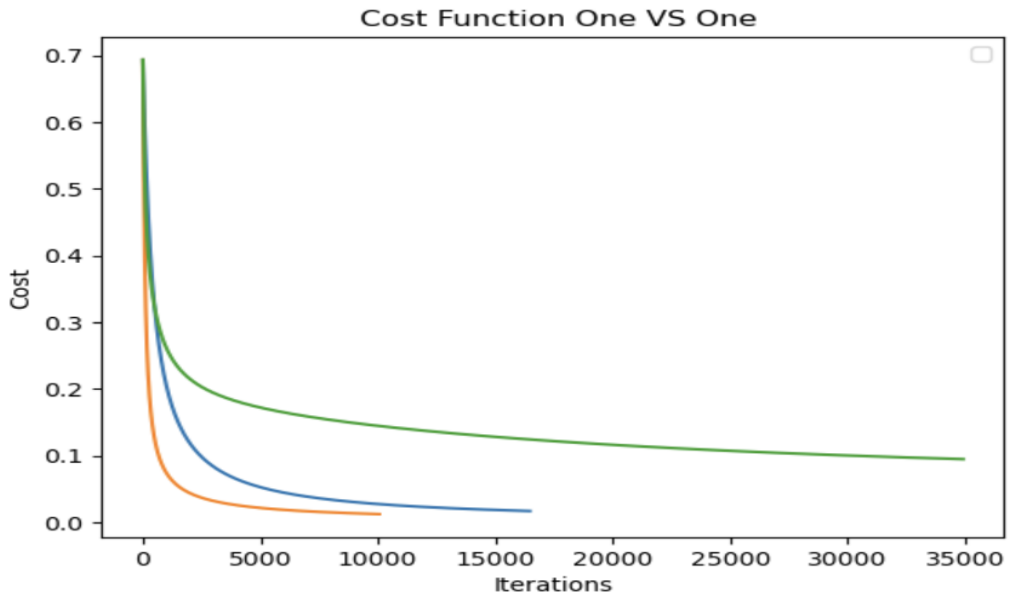
In summary, The code provides a comprehensive implementation of logistic regression for multiclass classification using both one-vs-all and one-vs-one strategies. Training and testing accuracies are reported, and cost function plots offer insights into the learning process.

```
5000
5000
5000
```




```
Training accuracy (One-vs-All): 87.93103448275862
Testing accuracy: 96.55172413793103
```

```
16466
10062
34912
```




```
Training accuracy (One-vs-One): 63.2183908045977
Testing accuracy: 65.51724137931035
```

Softmax Regression

for this part, the code implements Softmax Regression for multiclass classification.

2. Data Preprocessing:

The data is loaded from 'data.txt', and the input features and labels are extracted.

The dataset is split into training and testing sets using the train_test_split function.

One-hot encoding is applied to the labels to convert them into a binary matrix.

3. Softmax Activation and Loss Function:

The Softmax function converts raw scores (logits) into probabilities.

Cross-entropy loss is used to measure the difference between predicted and actual probabilities.

4. Model Training:

The fit function trains the Softmax Regression model using gradient descent.

Random weights and biases are initialized, and the model iteratively updates parameters to minimize the loss.

The training progress is tracked by the loss, and training stops when the loss converges.

5. Loss Function Plot:

The code generates a plot displaying the loss over the training iterations.

This provides insights into how well the model is learning.

6. Prediction and Evaluation:

The predict function applies the trained model to make predictions on both the training and testing sets.Training and testing accuracies are computed based on the predictions and actual labels.

7. Results:

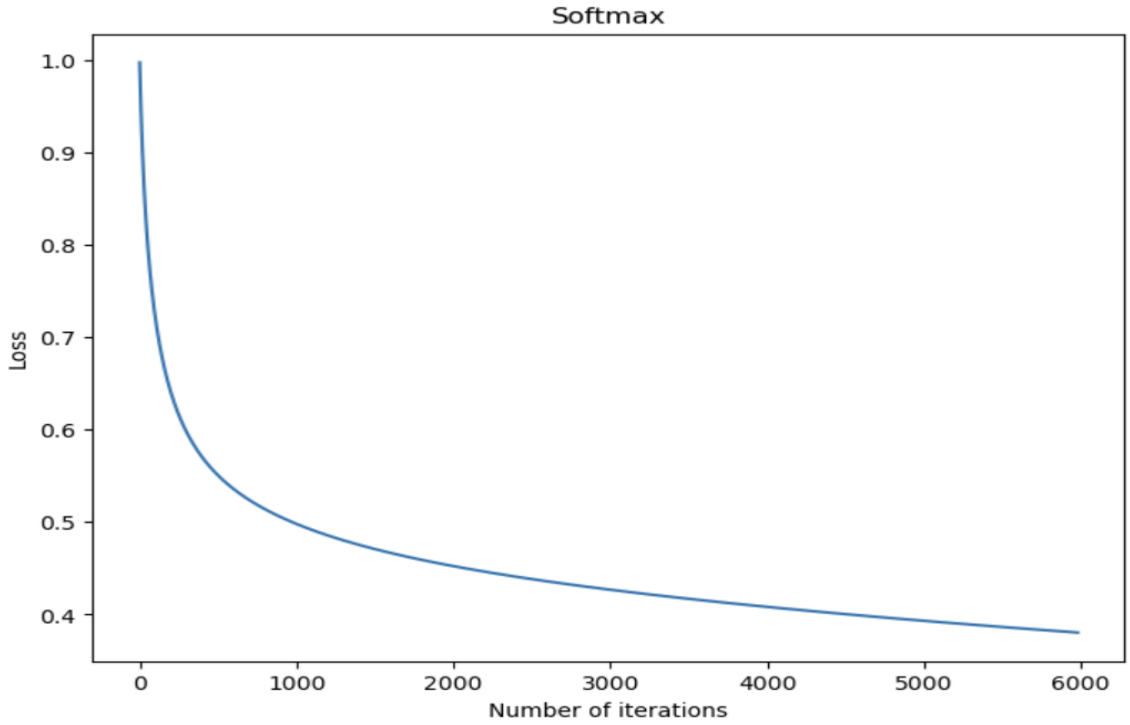
The code prints the number of iterations and displays a loss plot during training.

Training and testing accuracies are reported, providing an assessment of the model's performance.

The code successfully implements Softmax Regression for multiclass classification.

The training process is visualized through a loss plot, and accuracies on both training and testing sets are reported, indicating the model's performance.

```
5980
```




```
train accuracy is = 88.50574712643679
test accuracy = 91.37931034482759
```

Binary Classification using Bayesian Classifiers

here we prepare our dataset for implementation. Put the result by the sequence of the questions.

Can you calculate any kind of distance between your samples now? why? Yes for example i can calcuate qulidisian distance because i can take diiference between same feaure per 2 samples and the then make them ^2 sum them and filnaly take ~ of them

"what is the Bayes rule? How is it used in Bayesian classification?

What is each term of this rule called?

In this p(y|x)= p(x|y) p(y) / p(x) . in this formula the p(x|y) is the likelihood function and p(y) is prior and p(x) is the parameters that keep value small so it can be not consider. it is called beysian rule

What are the model parameters of LDA and QDA classifiers? What is the role of each parameter?

Between QDA and LDA, which one has has potentially more parameters? why?

We have 2 method linear discriminant analysis and quadratic discriminative analysis both are supervised learning for classification but their difference between nuber of parameter and qda has more parameters and they differ in data distribution consumering.

lda each class has mean vector per features.but all class share one covariance matrix for spread and correlation of all features

both have same priors qda has same class ean but qda has different covariance matrix for each class.

What causes the Covariance matrix to be singular? name 3 reason

1- one feature is mutipliction of other feature

2- we dont have much sample

3- we have more feature than saple

4 - we dont have mature data set or we have sae feature for one or more features

From confusion matrix it can be derived that in QDA the numbers of false is more less than LDA and also the number of true negetive is ucher than the number of true positive and i dont know why

Which one is more

important in the task of classifying poisonous mushrooms? I think fase negetaive is more important to detect nonclifier because it is bigger

finally report the model parameters for both classifiers. How many parameters are there? there are more paramteres for qda but at all we have mean parameter for mu per each class and we have covariance for both LDA and QDA but in QDA we have sigma or covariance matrix for each class it gave us good result and the result of QDA is better at all.

but in logistic regression familly number of parameters depend of the number of features because if n feature we have n phi or teta tha multiply to feature and also we have one bias so we have n + 1 parametr . for lda we have c number of meu and one covariance matrix so we have c +1 parameter which c is number of class and in qda we have c meu and c covarianse matrix so we have 2*c parameters

Creating our model: LDA And QDA

This Python code implements a Bayesian classification using Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) for mushroom classification based on various features. Here is a breakdown of the key components and functionality:

Data Preprocessing:

Reads feature names from "mushroom_feature_names.csv" and the dataset from "agaricus-lepiota.data.txt". Encodes categorical features using LabelEncoder. Handles missing values, drops a column ("veil-type") with zero variance, and splits the data into training and testing sets.

Bayesian Classification (LDA and QDA):

Implements LDA and QDA classifiers. Calculates mean and covariance matrices for each class. Uses the Bayes rule to compute posterior probabilities.Applies thresholding to classify samples.

Accuracy Calculation:

Computes accuracy for both LDA and QDA classifiers on the test set.

Confusion Matrix and Performance Metrics:

Computes confusion matrices for LDA and QDA.Calculates precision, recall, F1-score, and the number of misclassified samples for each classifier.

Visualization:

The code includes some visualization, such as displaying the confusion matrices and plotting the cost during the gradient ascent.

Report Generation:

The code generates a report containing the accuracy, confusion matrices, and various performance metrics for both LDA and QDA.

Conclusion:

The code successfully implements Bayesian classification with LDA and QDA, providing insights into accuracy and performance metrics for mushroom classification. we demonstrate our result as follow.

```
[1625 rows x 1 columns]
Accuracy LDA for test: 0.9335384615384615
Accuracy QDA for test: 0.9815384615384616
Confusion matrix for LDA:

[['true_positive', 'false_positive']]
[['false_negative', 'true_negetive']]
[[721  32]
 [ 76 796]]

 Precision for LDA:
 0.9575033200531209

 Recall for LDA:
 0.904642409033877

 f1-score for LDA:
 0.46516129032258063

 Number of misclasifiered in LDA:
 108

 Consusion matrix for QDA:
 [['true_positive', 'false_positive']]
 [['false_negative', 'true_negetive']]
 [[767   0]
  [ 30 828]]

 Consusion matrix for QDA:
 [['true_positive', 'false_positive']]
 [['false_negative', 'true_negetive']]
 [[767   0]
  [ 30 828]]

 Precision for QDA:
 1.0

 Recall for QDA:
 0.9623588456712673

 f1-score for QDA:
 0.49040920716112535

 Number of misclasifiered in QDA:
 30
```

Conclusion

Both codes serve as educational examples of how to implement and apply fundamental machine learning algorithms. They highlight the importance of data preprocessing, parameter optimization, and result visualization in the context of regression and classification tasks. These codes can be a starting point for more advanced modeling and analysis of the respective domains they address.

*__Bones__:

Naive Bayes Classifier

This code implements a Naive Bayes Classifier for classifying mushrooms based on various features.

The dataset used is 'agaricus-lepiota.data.txt,' containing information about mushroom characteristics.

2. Data Preprocessing:

The dataset is loaded into a Pandas DataFrame and preprocessed.

Columns are labeled, and 'veil-type' is dropped as it has the same value for all instances.

Label encoding is applied to convert categorical data into numerical format.

The data is split into training and testing sets using the train_test_split function.

3. Naive Bayes Classifier Implementation:

The NaiveBayesClassifier class is implemented to handle the Naive Bayes algorithm.It includes methods to calculate prior probabilities, mean, and variance for each class.The Gaussian density function is used to estimate the likelihood of a given feature belonging to a particular class.Posterior probabilities are calculated, and predictions are made based on the class with the highest posterior probability.

4. Training and Prediction:

The fit method of the Naïve Bayes Classifier is used to train the model on the training data.The predict method is then applied to the testing set to obtain predictions.

5. Evaluation:

The code utilizes metrics such as confusion matrix to evaluate the classifier's performance.The classification_report and confusion_matrix functions from sklearn.metrics are employed for this purpose.

The code prints a detailed classification report, including precision, recall, and F1-score for each class.

The confusion matrix provides insights into the model's performance on the testing set.

6. Conclusion:

The Naive Bayes Classifier demonstrates its effectiveness in classifying mushrooms based on the given features.

The evaluation metrics offer a comprehensive view of the model's accuracy and its ability to correctly identify different classes.

```
train metircs:              precision    recall  f1-score   support

           0       0.90      0.49      0.64      3382
           1       0.63      0.94      0.76      3117

    accuracy                           0.71      6499
   macro avg       0.77      0.72      0.70      6499
weighted avg       0.77      0.71      0.69      6499

[[1666 1716]
 [ 175 2942]]
test metircs:              precision    recall  f1-score   support

           0       0.92      0.52      0.66       826
           1       0.66      0.95      0.78       799

    accuracy                           0.73      1625
   macro avg       0.79      0.74      0.72      1625
weighted avg       0.79      0.73      0.72      1625

[[430 396]
 [ 39 760]]
```

Yes, with consider the confusion matrix indicates misclssifications. To determine the number of misclassified instances, we sum the off-diagonal elements of the

matrix so for first one has(1716+175) and for second one has (396 + 39). And also with respect of the confusion matrix in general form the count of FP and FN : for first one : FP = 1716 / second one FP= 396 and FN = 175 / FN = 39, so we have both.

For naive Bayes generally have mean(n*m) , variance(n*m) and prior probabilities(m) so if we have n feature and m classes so : 2*(n*m)+m.

From confusion matrix it can be derived that in QDA the numbers of false is more less than LDA and also the number of true negetive is ucher than the number of true positive and i dont know why\n\nWhich one is more\nimportant in the task of classifying poisonous mushrooms? I think fase negetaive is more important to detect nonclifier because it is bigger\n\nfinally report the model parameters for both classifiers. How many parameters are there? there are more paramteres for qda but at all we have mean parameter for mu per each class and we have covariance for both LDA and QDA but in QDA we have sigma or covariance matrix for each class it gave us good result and the result of QDA is better at all.\nbut in logistic regression familly number of parameters depend of the number of features because if n feature we have n phi or teta tha multiply to feature and also we have one bias so we have n + 1 parametr . for lda we have c number of meu and one covariance matrix so we have c +1 parameter which c is number of class and in qda we have c meu and c covarianse matrix so we have 2*c parameters\n.