CSE342: Statistical Machine Learning Report

Prakhar Gupta
Roll No. 2021550
CSAI Dept.
prakhar21550@iiitd.ac.in

Shreyas Kabra
Roll No. 2021563
CSAI Dept.
shreyas21563@iiitd.ac.in

***Abstract*—This report presents a machine learning project completed for the course *CSE 342: Statistical Machine Learning* at the *Indraprastha Institute of Information Technology, New Delhi*. The project, a Kaggle competition, involved creating an accurate fruit classification model with 20 possible labels. The report briefly describes the problem statement, the data set provided, and the different pre-processing steps taken to prepare the given data for analysis. It then outlines the various machine learning algorithms tested during the project, including the respective results, observations, and evaluation metrics. The report concludes with a summary of the project and provides relevant references. Overall, the report serves as a comprehensive summary of the project, presenting the application of various machine learning algorithms.**

***Index Terms*—PCA, LDA, logistic regression, cross validation, hyperparameter tuning, agglomerative clustering, meanshift clustering, DBSCAN clustering.**

I. Introduction

The problem is a fruit classification problem where the aim of the project was to build a machine learning model to classify a fruit into one of 20 fruit labels. The dataset consists of 1216 samples, where each is classified into any label. The dataset and the competition is hosted on Kaggle [1].

II. Outlier Detection

The data set was checked for outliers by implementing the *Local Outlier Factor* algorithm. The hyperparameter for the number of neighbours in the *Local Outlier Factor* was set to 4 after performing *k-fold cross validation* on the training data set. The algorithm found 3 outliers in the data set and the outliers were removed from the training data set. When the *Local Outlier Factor* algorithm was implemented along with the other algorithms mentioned in this report, the validation accuracy of the model was found to increase from 85.5% to 85.99%.

III. Dimensionality Reduction

Dimensionality reduction is a crucial pre-processing step in machine learning. It is implemented to reduce the number of features while retaining the maximum amount of information. By reducing the number of features, the model is able to train faster and the model is less prone to overfitting. The following dimensionality reduction algorithms were implemented on the training data set to reduce the number of features-

A. Principal Component Analysis

The large number of features in the data set made it critical to perform *principal component analysis*[2][3] on the given data set to reduce the number of features. It is implemented to reduce the number of features while retaining the maximum amount of information. *Elbow method* and *k-fold cross validation* was implemented to determine the optimal number of principal components. *Elbow method* determined the optimal number of components to be 91 components, however, that resulted in a poor accuracy of around 74%. The *k-fold cross validation* found the optimal number of components to be around 417 components, which resulted in a better accuracy of about 80%. Due to higher accuracy, the number of components was set to 417 for the rest of the project.

B. Linear Discriminant Analysis

Linear discriminant analysis[4][5] was also performed on the data set to reduce the number of features. It is implemented to maximize the separation between the classes. The range of the number of components for the project was 1 to 19 since the data set had 20 classes. *Elbow method* and *k-fold cross validation* was implemented to determine the optimal number of principal components. *Elbow method* determined the optimal number of components to be 16 components, however, that resulted in a poor accuracy of around 76%. The *k-fold cross validation* found the optimal number of components to be around 19 components, which resulted in a better accuracy of about 81%. Due to higher accuracy, the number of components was set to 19 for the rest of the project.

IV. Unsupervised Clustering

Unsupervised clustering algorithms aim to create clusters for the sample points based on the training data set. Adding a cluster feature to the data set after performing unsupervised clustering algorithms identifies correlations between the sample data points, thus finding the trends and groupings in the dataset, which may help in the supervised classification of the data set. The following unsupervised clustering algorithms were implemented on the training data set to create unsupervised clusters-

A. Agglomerative Clustering

Agglomerative clustering[6] algorithm was implemented on the data set to cluster the training data and appending the unsupervised cluster labels to the training data. The unsupervised cluster labels were used as a feature in the training data to identify patterns and groupings in the dataset, which would otherwise not be apparent. Grouping the similar data together was expected to improve the accuracy of the model. For implementing agglomerative clustering, the hyperparameter for the number of clusters was set to 4 after performing *k-fold cross validation*[7] on the training data set.

B. DBSCAN Clustering

DBSCAN clustering[8] algorithm was implemented on the data set to cluster the training data and appending the unsupervised cluster labels to the training data. The hyperparameters set for implementing *DBSCAN clustering* were set by grid search and the hyperparameters were set to *eps* = 0.5 and *min_samples* = 5.

C. K-Means clustering

K-means clustering[9] was used on the data set to cluster the data set and compute a new feature using unsupervised clustering. The hyperparameter for the number of clusters was set to 19 after performing *k-fold cross validation* on the training data set.
Amongst *agglomerative, DBSCAN, and k-means clustering*, *agglomerative clustering* was found to be the most effective in improving the accuracy of the model and was used for the rest of the project.

V. Logistic Regression

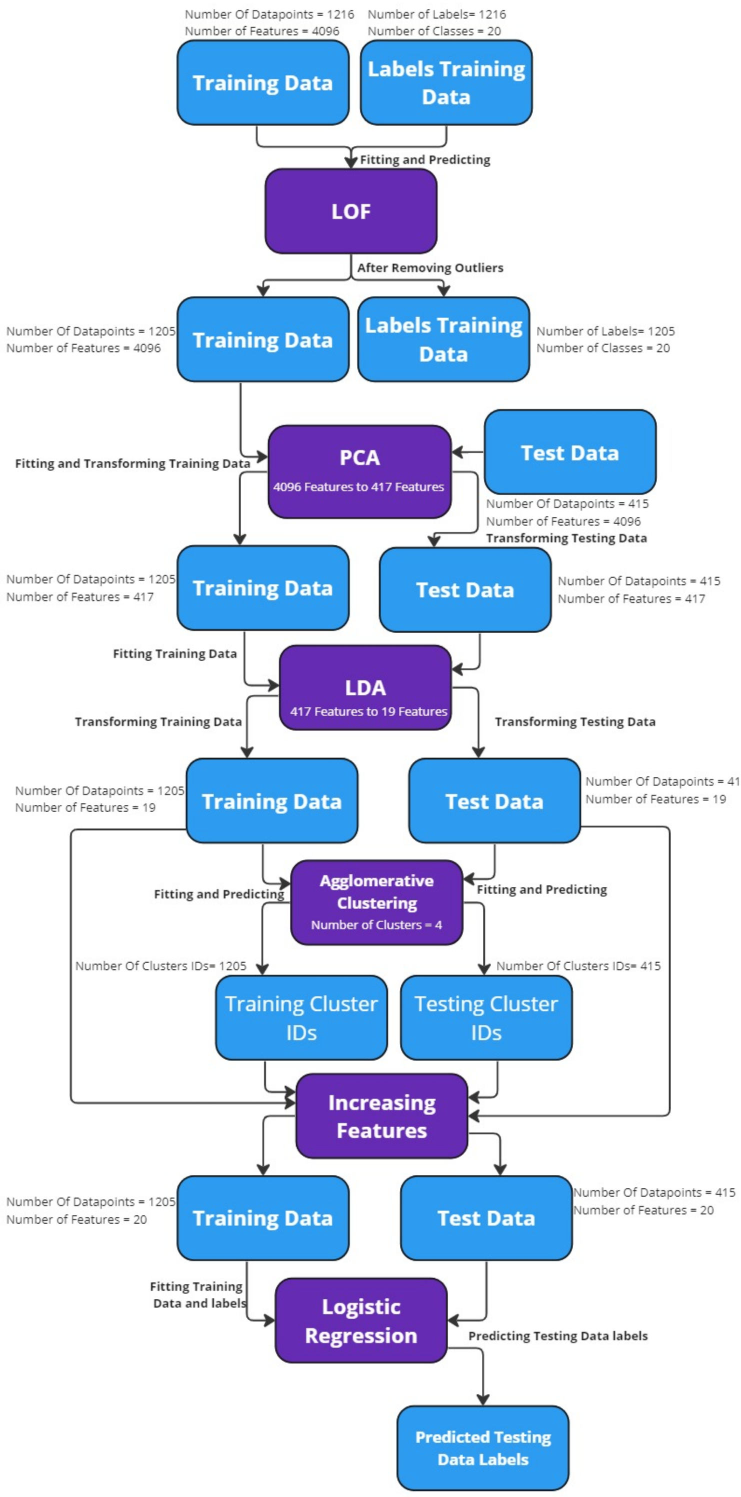
Logistic Regression[10] is a supervised classification algorithm implemented to classify data into two classes, which is called binary logistic classification. In this project, the *One vs All* method was used to implement *Logistic Regression* for the 20 class labels classification. Amongst other classification methods, *Logistic Regression* was found to be the most effective in improving the accuracy of the model.

VI. Ensembling Methods

Ensembling methods are used to combine the predictions of multiple models to improve the accuracy of the model. The *Voting classifier* ensembling method was implemented with *Logistic Regression and DBSCAN clustering*, *Logistic Regression and Agglomerative clustering*. The accuracy of the model fell from 85.5% to close to 79%, and hence, the ensembling methods were not performed for the creation of the model.

VII. References

[1] Kaggle Competition
[2] Principal Component Analysis
[3] PCA in Scikit-learn
[4] Linear Discriminant Analysis
[5] LDA in Scikit-learn
[6] Agglomerative Clustering
[7] K-Fold Cross Validation
[8] DBSCAN Clustering
[9] K-Means Clustering
[10] Logistic Regression



Number Of Datapoints = 1216
Number of Features = 4096
Number of Labels= 1216
Number of Classes = 20
Training Data
Labels Training Data
Fitting and Predicting
LOF
After Removing Outliers
Number Of Datapoints = 1205
Number of Features = 4096
Training Data
Labels Training Data
Number of Labels= 1205
Number of Classes = 20
Fitting and Transforming Training Data
PCA
4096 Features to 417 Features
Test Data
Number Of Datapoints = 415
Number of Features = 4096
Transforming Testing Data
Number Of Datapoints = 1205
Number of Features = 417
Training Data
Test Data
Number Of Datapoints = 415
Number of Features = 417
Fitting Training Data
LDA
417 Features to 19 Features
Transforming Training Data
Transforming Testing Data
Number Of Datapoints = 1205
Number of Features = 19
Training Data
Test Data
Number Of Datapoints = 41
Number of Features = 19
Fitting and Predicting
Agglomerative Clustering
Number of Clusters = 4
Fitting and Predicting
Number Of Clusters IDs= 1205
Number Of Clusters IDs= 415
Training Cluster IDs
Testing Cluster IDs
Increasing Features
Number Of Datapoints = 1205
Number of Features = 20
Training Data
Test Data
Number Of Datapoints = 415
Number of Features = 20
Fitting Training Data and labels
Logistic Regression
Predicting Testing Data labels
Predicted Testing Data Labels

Fig. 1. Figure for the model pipeline