Project Report

Niharika Singh(2021545)
Computer Science and Artificial Intelligence (of Aff.)
Indraprastha Institute of Information and Technology Delhi (of Aff.)
Delhi, India
Niharika21545@iiitd.ac.in

***Abstract*—This report presents the implementation of a Local Outlier Factor (LOF) algorithm with Bagging MLP Classifier for classification. The dataset is preprocessed using LOF to remove outliers and StandardScaler for scaling. The dimensionality of the features is reduced using PCA. The model is trained using Bagging MLP Classifier and tested using cross-validation. The predicted target variable is added to the test dataset and saved as a CSV file.**

***Index Terms*—Local Outlier Factor, Bagging MLP Classifier, PCA, cross-validation, classification.**

I. Introduction

This report represents an overview of implementation of Local Outlier Factor with Bagging MLP Classifier for classification on training and test dataset used for SML kaggle competition.

II. Literature Review

A. Local Outlier Factor (LOF)

In anomaly detection, the local outlier factor (LOF) is an algorithm proposed by Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng and Jörg Sander in 2000 for finding anomalous data points by measuring the local deviation of a given data point with respect to its neighbours.[1]
LOF computes the local density of each data point and compares it with the densities of its neighboring points to determine whether the point is an outlier or not.

B. Principal Component Analysis (PCA)

Principal component analysis (PCA) is a popular technique for analyzing large datasets containing a high number of dimensions/features per observation, increasing the interpretability of data while preserving the maximum amount of information, and enabling the visualization of multidimensional data.[2]
PCA is commonly used to reduce the number of features in high-dimensional datasets and improve the performance of machine learning models.

C. BaggingClassifier

Bootstrap aggregating, also called bagging (from bootstrap aggregating), is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression.[3]
BaggingClassifier uses the bootstrap sampling method to generate multiple datasets and trains each base classifier on a different dataset. The predictions of all base classifiers are then aggregated to produce the final prediction.

D. Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) are a type of machine learning algorithm that are modeled on the human brain's structure and function. ANNs are widely used in various fields such as image and speech recognition, prediction, classification, and many others.[4]

E. MLPClassifier (Multi-Layer Perceptron)

The MLPClassifier (Multi-Layer Perceptron) is a feedforward neural network that consists of an input layer, hidden layers, and an output layer. Each layer is composed of a set of nodes (also known as neurons) that compute the weighted sum of inputs and pass them through an activation function. MLPClassifier is a popular and powerful tool for solving complex classification problems.[5]

III. Methodology

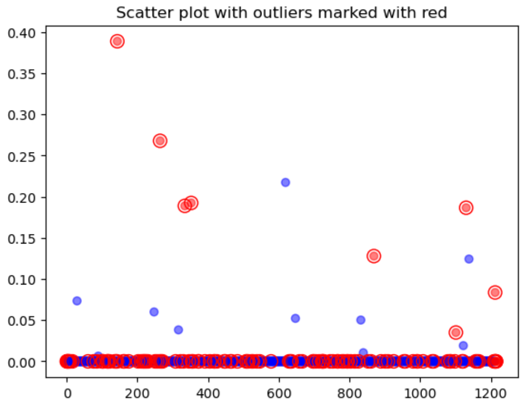
Methodology for applying MLPClassifier with Bagging and PCA for classification using Python's scikit-learn library[6]:

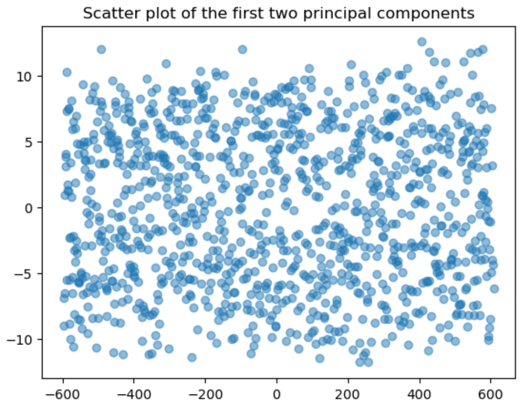
A. Implementation

1) **Load the required libraries:** Import the necessary libraries for implementing the classification model. In this case, import MLPClassifier for building a neural network model, accuracy_score to evaluate model performance, pandas to handle data frames,PCA for dimensionality reduction, LocalOutlierFactor for outlier detection, BaggingClassifier for ensemble learning, cross_val_score to evaluate model performance using cross-validation, and StandardScaler for data scaling.
2) **Load the dataset:** Import the Load the training data as a CSV file using the read_csv() function in Pandas. Store the feature variables in X and the target variable in y.
3) **Remove outliers from the dataset:** Use Local Outlier Factor (LOF) algorithm to remove any possible outliers from the dataset using the LocalOutlierFactor() function from sklearn.neighbors. Set n_neighbors to 20 and contamination to 0.1 to remove 10% of the samples as outliers. Then, apply the LOF algorithm to the dataset using the fit_predict() function.

4) **Scale the data:** Standardize the feature variables using StandardScaler() from sklearn.preprocessing. Fit the scaler on the training data using fit_transform().
5) **Reduce dimensionality of the features:** Use PCA to reduce the dimensionality of the features to 100 using the PCA() function from sklearn.decomposition. Fit the PCA on the standardized data using fit_transform().
6) **Build the model:** Build an MLPClassifier model with default parameters..
7) **Apply ensemble learning:** Apply bagging, which is an ensemble learning technique that aggregates multiple base models to form a more accurate model. Use BaggingClassifier() from sklearn.ensemble with base_estimator set to the MLPClassifier model and n_estimators set to 10 to build 10 models.
8) **Train the model:** Fit the bagging classifier on the PCA-reduced and scaled data using the fit() function.
9) **Evaluate the model:** Predict the target variable using cross-validation using the cross_val_score() function with cv set to 5. Also, predict the target variable on the training data using the predict() function and calculate the accuracy of the model using the accuracy_score() function.

IV. Evaluation





V. Result

Accuracy of the submitted model

TABLE I
MLP-Classifier Result

Accuracy	**Result**
Accuracy on training data	*0.99725*
Mean cross-validation accuracy	*0.76872*
Accuracy on test data	*0.79227*

Accuracy using different Classification Algorithms

TABLE II
Accuracy with different Algorithms

Classification Algorithm	**Accuracy**		
	Accuracy on training data	***Mean cross-validation accuracy***	***Accuracy on test data***
Decision Tree Classifier	*0.99725*	*0.48991*	*0.50261*
Logistic Regression	*0.98537*	*0.78243*	*0.79227*
k-NN	*0.77696*	*0.59686*	*0.60143*

Conclusion

On the basis of the result presented, I chose the MLP-Classifier to proceed with my project among all the other classification algorithms because of the accuracy results.
The MLP-Classifier also provided better accuracy without using PCA with
Accuracy on training data: 0.99085
Mean cross-validation accuracy: 0.76691
Accuracy on test data: 0.80193
The Local Outlier Factor with Bagging MLP Classifier provides a promising approach for classification by removing outliers and reducing the dimensionality of the features for my project after implementing classification algorithms like decision tree, logistic regression, random forest classifier and outliers algorithms like DBSCAN and dimension reductionality algorithms like LDA.

References

[1] Breunig, M. M.; Kriegel, H.-P.; Ng, R. T.; Sander, J. (2000). LOF: Identifying Density-based Local Outliers (PDF). Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data. SIGMOD. pp. 93–104. doi:10.1145/335191.335388. ISBN 1-58113-217-4.

[2] Jolliffe, Ian T.; Cadima, Jorge (2016-04-13). ”Principal component analysis: a review and recent developments”. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences. 374 (2065): 20150202.

[3] ”The bootstrap predictive distribution is considered to be an approximation of the Bayesian predictive distribution”. Bayesian bootstrap prediction, Tadayoshi Fushiki, http://dx.doi.org/10.1016/j.jspi.2009.06.007.

[4] Hardesty, Larry (14 April 2017). ”Explained: Neural networks”. MIT News Office. Retrieved 2 June 2022.

[5] https: // www.sciencedirect.com / topics/computer-science/multilayer-perceptron

[6] https://scikit-learn.org/stable/

[7] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.