INT104 Coursework 2 Report

Mingyuan.Li(2145618)

Lab-D1/1-C

May 17, 2023

1 Introduction

This study contains three sections to meet the assignment requirements. Firstly, to remove redundant features from the dataset, Principal Component Analysis(PCA) was used to reduce the dimensionality of the patient data. Secondly, build 3 classifiers using the reduced dimensional data and evaluate the generalization ability of the model using cross-validation. Lastly, Clustering using unlabeled dimensionality-reduced data and testing the quality of the results.

1.1 Restatement of the Problem and Data Analysis

According to the assignment, the provided data set consists of a table containing questionnaire results for more than 5000 patients and labels from two groups. The data observation shows a total of 15 dimensions, representing the scores of 15 questionnaires. The study of Rehman [1] indicates that learning from high-dimensional data, whether supervised or unsupervised has drawbacks, including greater computing effort and easy overfitting. In addition, the data contains outliers, such as patient data with a label of 2, so exception handling is necessary.

1.2 Classification Selection

After preprocessing the dataset, it can be seen that the individual features in the sample are independent and have a concentrated numerical size. Given that the dataset contains more than 5000 patients, utilizing deep neural network(DNN) is an excellent approach [2]. Meanwhile, DNN models have strong generalization abilities and automated feature extraction capabilities [3]. After that, as XGBoost is based on gradient-boosting trees and has a greater capacity to prevent overfitting for multidimensional datasets, it should also be utilized [4]. Then, due to the high dimensionality of features in the dataset, high-performance classification can be achieved using Naive Bayes [5]. Last, for unsupervised learning the choice of k-means clustering is acceptable. Its low computational cost makes it beneficial for processing massive amounts of data, and its simplicity and intuitiveness make tweaking it straightforward [6]. However, DBSCAN was also considered for use, but the clustering results were too poor to be adopted in the end.

1.3 Classification and Clustering Results

The cross-validation scores of each classifier are as Table 1:

Table 1: Cross-validation Scores

Classifier	Train Set Score	Test Set Score
DNN	0.87	0.84
XGBoost	0.75	0.70
Bayes	0.71	0.69

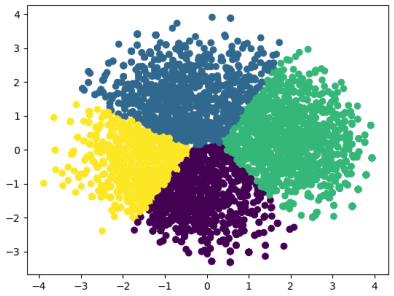


Figure 1: K-means Clustering Result

The silhouette coefficient of k-means clustering reached 0.42. The results are visualized as Figure 1:

2 Dimensionality Reduction

2.1 Reasons for Dimensionality Reduction

A basic inspection of the raw data reveals that there are 17 columns consisting of the ID of the patient, 15 dimensions, and a group label. According to what was declared in Rai's [7] study, dimensionality reduction should be utilized properly for datasets with too many dimensions to prevent a fall in data point density and an increase in sample sparsity. Furthermore, having a model with too many dimensions may cause it to overfit, which will make it simpler to access dataset noise and limit the model's capacity to generalize. In addition, it is a given that as the dimension rises, the computational complexity of model training and prediction will likewise rise exponentially, increasing the use of computer resources. Therefore, it is required to do dimensionality reduction for this data set.

2.2 Principal Component Analysis (PCA)

PCA is one of the most used techniques for reducing the dimensionality of data. In this study, PCA was selected as the dimensionality-reduction method.

2.2.1 Data Normalization

Data needs to be normalized before being reduced in dimension, which helps bring disparate data sets into the same size and make them comparable. This study uses maximum-minimum normalization with the following equation:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{1}$$

However, it is very sensitive to outliers, so it is also necessary to remove outliers before normalization.

2.2.2 Principle of PCA

The main idea of PCA is to map n-dimensional characteristics to k-dimensional space. This k-dimensional feature is a completely new orthogonal feature that was created using the original n-dimensional features as its foundation. It is also referred to as a main component [8]. In PCA, It is possible to assess how effectively the principal components account for the variety in the data by computing the variance of each principal component. To achieve dimensionality reduction, only the principal components with high variance are typically kept, while the rest are removed.

In particular, PCA's task is to systematically extract from the original space a collection of coordinate axes that are orthogonal to one another. The direction with the highest variance in the original data is chosen as the first new coordinate axis, and the direction with the highest variance in the plane perpendicular to the first coordinate axis is chosen as the second new coordinate axis, and so on until n such coordinate axes are obtained. The variation for the additional coordinate axes created in this manner is mostly contained in the first k coordinate axes, and the variance in the later coordinate axes is virtually zero. As a result, the other coordinate axes may be disregarded, leaving just the first k coordinate axes, which contain the majority of the variation.

2.2.3 PCA Specific Steps

1. **Data preprocessing:**

   To meet the task requirements, make the following preprocessing

   (a) Read the CSV file with pandas and drop the null values with `dropna()`.

   (b) Traverse each row and its corresponding label, if the label is 2, delete them.

   (c) Extract 2 to 16 columns of data as Data

2. **Get eigenvalues and Eigenvectors:**

   Normalize the data to eliminate scale disparities between various aspects so that each feature is correctly taken into account and weighted. Using the scaler method to standardization.

   Use `np.cov` to calculate the covariance matrix of the data, and then use `np.linalg.eig` to calculate the eigenvalues and eigenvectors of the data.

3. **Interpretability Variance Calculation:**

   To determine the relative relevance of each characteristic, divide the magnitude of each eigenvalue by the sum of all its eigenvalues. Then, computes the cumulative sum of the explained variance rates for the principal components by using `np.cumsum`.

   Use the `Matplotlib` library to draw bar and step plots to visualize the cumulative sum of the explained variance rates of the principal components. The result is shown in Figure x.

4. **Determine k Value:**

   According to Serrao [9], by evaluating the cumulative explained variance percentage following PCA, the final dimension reduction dimension may be determined, and it is acceptable to disregard the principal components whose cumulative explained variance ratio is more than 95% since they have a negligibly little effect on the feature dimension. In this way, the principal components accounting for the top 95% are retained to achieve dimensionality reduction.

   It is beneficial to lay up the following variance explanation table to gain particular information about the principal components of the cumulative explained variance ratio:

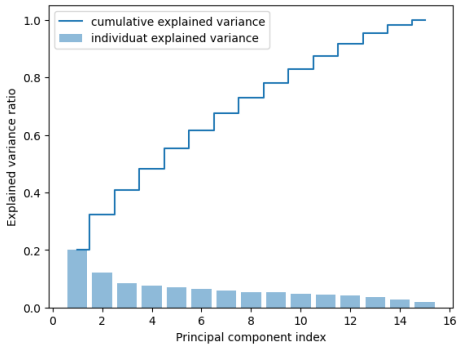


Figure 2: Cumulative Explained Variance Ratio

Furthermore, the scree plot after PCA should also be considered. It is used to show the contribution of each principal component to the total variance. The principal component is represented by an evident (inflection point) in this graph, which typically

depicts the number of principal components (x-axis) and the related variance (y-axis) [10]. Before this position, the principal components' contribution to the overall variance is considerable; however, following this position, the principal components' contribution to the overall variance starts to decline dramatically. The scree plot is as Figure 3 left one:

(a) Scree Plot

Element	characteristic root	Variance explained rate (%)	Cumulative variance explained rate (%)
1	2.999	19.995	19.995
2	1.81	12.069	32.064
3	1.259	8.394	40.458
4	1.103	7.354	47.812
5	0.998	6.655	54.467
6	0.939	6.262	60.728
7	0.88	5.87	66.598
8	0.849	5.663	72.261
9	0.792	5.28	77.541
10	0.761	5.076	82.617
11	0.704	4.692	87.309
12	0.624	4.161	91.469
13	0.574	3.823	95.293
14	0.403	2.689	97.982
15	0.303	2.018	100

(b) Detail of Total Variance

Figure 3

By observing Figure 2, Figure 3, the first 13 principal components are chosen since the contribution rates of components 14 and 15 to variable explanation are quite low. It is reasonable to choose k to be 13, that is, to reduce the dimension to 13 dimensions.

2.3 Results after PCA

To calculate each principal component's contribution to the variance of the data set in PCA, utilize the explained variance ratio. After examining the cumulative explained variance ratio, it is fair to lower the dimension to 13 dimensions to save more original feature data. Following dimensionality reduction, the explained variance plot is as follows:

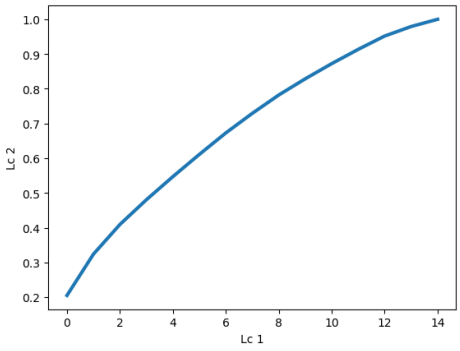


Figure 4: Explained Variance Plot

Analyzing principal component weights is also important to take into account. It is a more logical method to describe how the principal components' contributions to the data were made. The weight is defined as the variance explanation rate divided rotated cumulative variance explanation rate.

name	Variance explained rate (%)	Cumulative variance explained rate (%)	Weights(%)
Principal Component 1	19.995	19.995	19.995
Principal Component 2	12.069	32.064	12.069
Principal Component 3	8.394	40.458	8.394
Principal Component 4	7.354	47.812	7.354
Principal Component 5	6.655	54.467	6.655
Principal Component 6	6.262	60.728	6.262
Principal Component 7	5.87	66.598	5.87
Principal Component 8	5.663	72.261	5.663
Principal Component 9	5.28	77.541	5.28
Principal Component 10	5.076	82.617	5.076
Principal Component 11	4.692	87.309	4.692
Principal Component 12	4.161	91.469	4.161
Principal Component 13	3.823	95.293	3.823

Figure 5: Weight of Each Component

Figure 5 shows that principal component 13 only accounts for 4.012, showing that the contribution of the subsequent principal components is extremely seldom, demonstrating the viability of dimensionality reduction.

3 Training Classifiers in a Supervised Way

In this study, a total of three classifiers were employed: Deep Neural Network(DNN) Based on Pytorch, XGBoost, and Naive Bayes. A combination of numerous characteristics led to the selection of DNN as the classifier.

3.1 Deep Neural Network(DNN) Based on PyTorch

A neural network mimics how neurons and neural networks are connected in a biological nervous system. It functions by sending the input to several nodes for processing and transformation before producing the output. Python-based Pytorch is an open-source machine learning framework that offers a wide range of tools and features. Utilizing DNN is advantageous because of their automated feature extraction capabilities and the benefits of large-scale data processing.

3.1.1 Processing of Dataset

Define the `getxy(train,fold)` method for preparing the data set

- Loading and pre-processing of data sets:

  Source from Task1. To create a data set, eliminate the row with a value of 2 in the Label column, normalize the remaining information, and then apply PCA to decrease the data dimension to 13 dimensions.

  Change the order of the data using `torch.randperm` to lessen data dependencies and overfitting.

- Segmentation of the dataset:

K-fold Cross Validation: This model considers the use of cross-validation and is divided into a total of 5 folds. The validation set is not divided since each function has already been evaluated on it. To establish whether a set is a train set or a test set and the number of the test set in cross-validation, use the incoming train and fold parameters. As an illustration, if the fold is 3, the third fold will serve as the test set and the remaining folds as the training set. Repeat the preceding procedure five times to make each of the five folds a test set. Finally, according to the value of train, selectively return the `xy` of the train set or test set.

Using cross-validation, it is feasible to train on the full dataset for better evaluation. Additionally, as alternative data folds are permitted for training, overfitting can be reduced.

- Customized data sets:

  Rewrite three of PyTorch's Dataset class's methods: `init`, `getitem`, and `len`, using a custom dataset class that inherits from PyTorch. To enter the data into the PyTorch model for training or testing, this is required.

3.1.2 Construction of Neural Network

- Model Structure and Hierarchy

  The neural network model has four levels: an input layer, three hidden layers, and an output layer. It is a feed-forward neural network.

  There are 13 neurons in the input layer, 2048, 1024, and 512 neurons in each hidden layer, and 13 neurons in the output layer.

  - Each layer has batch normalization `nn.BatchNorm1d`, which improves the generalizability of the model, stabilizes the input data distribution of each layer, and successfully addresses the issue of gradient disappearance.
  - Use `nn.Dropout` at the end of each layer to exclude certain neurons, which significantly reduces overfitting

- Activation Function `ReLU`:

  - It is a nonlinear function that is added between linear layers to increase the model's capacity for expressiveness.
  - This function prevents the gradient from decreasing and maintains it at 1 in the positive interval, assisting the backpropagation method in determining the proper gradient.

3.1.3 Model Training and Evaluation

- Training Steps

  1. In the main function, use `getxy` to obtain the data set, and then call the data loader `DataLoader` to divide the batch
  2. instantiated model, optimizer, learning rate adjuster: The optimizer can reduce the loss function, and the learning rate scheduler `StepLR` dynamically modifies the learning rate to improve model convergence.
  3. Create an epoch, run the training method and the test method repeatedly while calculating the loss and accuracy, and then, after each iteration, apply the optimization strategy to improve the model's parameters.

- Model Tuning

  The model has methods and hyperparameters that may be changed to improve training outcomes.

  1. optimizer selection

     Multiple optimizers are considered:

     SGD: Each optimization only considers the gradient of a single sample.

     Adam: Each parameter's learning rate is dynamically changed to better match the gradient of that parameter.

     BGD: Similar to SGD but considers the gradient of the entire data each time
  2. batch size

     `batch_size` is the amount of data used for each batch of training. Although the training pace will be enhanced if the `batch_size` is too big, this will result in a memory-intensive process. There will be a slowdown in the training pace if the `batch_size` is too small.
  3. Learning rate:

     If the learning rate is too large, it will be difficult for the data to converge. If it is too small, more iterations will be needed to reach the optimal solution.
  4. Optimization Strategy:

     After a certain number of epochs, the learning rate will be reduced to make the model converge better.

  For efficiency, use 10epoch to find the best accuracy. The result is as Table 2:

- Overfitting Problem

  To reduce overfitting, the following methods are used:

Table 2: Hyperparameter Adjustment for Best

Hyperparameter	Value	Accuracy Score
Batch size	18	0.70
Learning rate	0.8	0.73
Optimization Strategy	100/0.9	0.72

– Add regularization method dropout to limit data complexity

– Stop training early, when model performance is no longer improving, stop training to avoid overfitting

3.1.4 Visualization Analysis

The training results which epoch is 200 on the dataset may be displayed as a line chart of accurate rate and a line chart of error using the chosen hyperparameters.

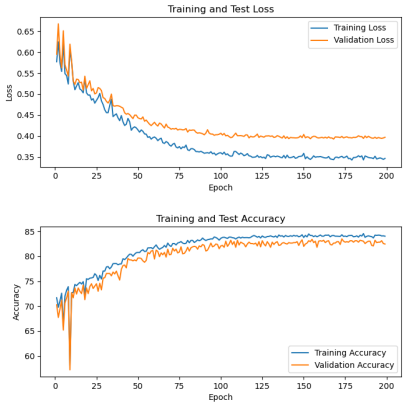


Figure 6: Accurate Rate and Error Variation

The accuracy of the test set is seen in the picture 6 to be continuously growing until it eventually stabilizes at 84% dynamically, which reflects that the epoch selection of 200 is feasible. The error is also shown to have decreased from the initial value of 0.65 to no more than 0.45, indicating that the model has performed well. However, overfitting has taken place since the accuracy of the training set has always been greater than that of the test set, although the difference often varies about 1%, which is a typical phenomenon brought on by model fitting noise.

3.2 XGBoost

XGBoost is a gradient-boosting decision tree algorithm, it improves classification accuracy via gradient boosting and regularization.

Each tree is used as a weak classifier to categorize the data based on the decision tree model. The model changes the parameters to complete further iterations after each round of iteration, using bias to calculate the gradient to assist it in better identifying the overall optimal parameters.

3.2.1 Processing of dataset

- Use the exact same data set as task1
- Segmentation of the dataset

  Different from DNN, this model calls the KFold method in `sklearn` for cross-validation. Create a KFold instance, which uses random shuffling and 5-fold cross-validation, with a 100-bit random seed. After cross-validation separation, which is used to divide the test set and the training set, the for loop iterates over the index.

3.2.2 Model Evaluation

Use the XGBoost model in `sklearn` to train a classifier. Use the trained model to predict the training set and return the prediction result. The accuracy of the test set was calculated using the accuracy_score function, and the result was 70.9, suggesting that the model is qualified but not exceptional. Additionally, the model's F-score of 70.1 indicates that it performs rather well, as demonstrated by a thorough review of the accuracy and recall rate of the model's prediction findings.

3.2.3 Visualization Analysis

Analysis of the feature importance figure is required for XGBoost outcomes. The feature importance diagram of the selected test set is as follows:

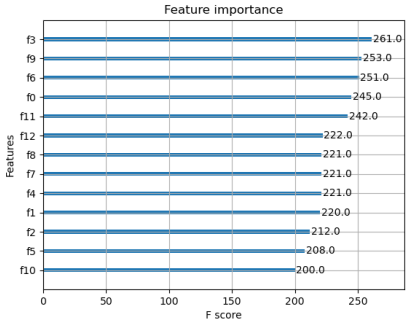


Figure 7: Feature Importance

The f3 feature in the figure has the highest value, which denotes that it has the most impact on the outcomes of the predictions. Additionally, the F-score of

the first 5 features is significantly higher than that of the final 8, indicating that these aspects require greater focus in order to enhance model performance.

3.3 Naive Bayes

Since Naive Bayes is based on Bayes' theorem, which has the advantages of high efficiency and high accuracy for data sets with many dimensions, it is extremely beneficial to apply to data sets that are still 13-dimensional after dimensionality reduction.

3.3.1 Processing of dataset

Use 13-dimensional data consistent with task 1. Cross-validation is similar to XGBoost, calling the KFold method to shuffle the data set and divide it into 5 folds.

3.3.2 Model Optimization

Construct a Naive Bayesian classifier by calling `MultinomialNB`. Since the accuracy rate with the initial hyperparameter training data is just 62.2%, we ought to consider altering the hyperparameters to improve performance.

- grid search technique: Using the grid search technique is effective in finding suitable hyperparameters, due to the combination of hyperparameters in a certain range to create a grid, which is afterward assessed using cross-validation to identify the optimal value. If each hyperparameter has m_i values and there are n hyperparameters to alter, then the formula is:

$$\prod_{i=1}^{n} m_i \tag{2}$$

  Using `GridSearchCV` to find the best alpha value in the training set, and finally find the best alpha value is 1.6. When the test set is predicted using the updated model, which has greatly enhanced model performance, it may attain an accuracy rate of 69

3.3.3 Visualization Analysis

Adjust the model to the best performance, its learning curve and Receiver Operating Characteristic Curve (ROC) are in Figure 8 and Figure 9

By observing the learning curve, the scores of the training set and the test set gradually converge and reached about 69%. This indicates that there not exist underfitting phenomenon, and the scores tend to be stable, therefore the training can be stopped to prevent overfitting.

For ROC, It takes the false positive rate(FPR) on the horizontal axis and the true positive rate(TPR) on the

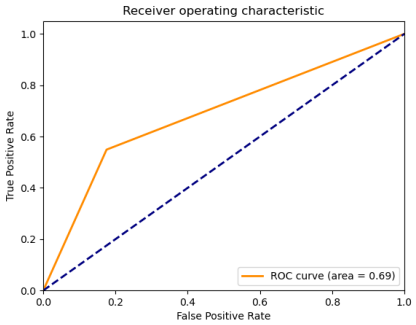


Figure 8: ROC

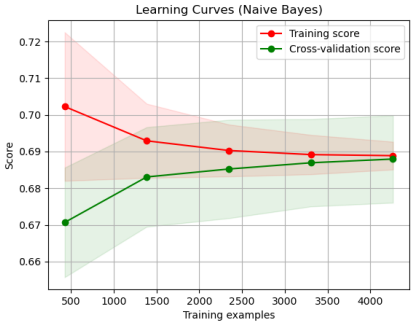


Figure 9: Learning Curve

vertical axis to generate a curve. The equations for FPR and TPR are as follows:

$$TPR = \frac{TP}{TP + FN} \tag{3}$$

$$FPR = \frac{FP}{FP + TN} \tag{4}$$

where TP is the number of true examples, FN is the number of false counterexamples, FP is the number of false positive examples, and TN is the number of true counterexamples.

When analyzing the ROC curve, the AUC index can be considered to measure the performance of the model. The outcomes of the model fulfill the stochastic prediction when the AUC is closer to 1. Figure x's AUC was 0.69, and the model's performance was satisfactory.

3.4 Classifier Selection

Since all three classifiers utilize cross-validation, the accuracy comparison may be used to determine which model performs the best. DNN is chosen because it achieves 84% accuracy, which is significantly more accurate than the other two classifiers(70%, 69%).

After selecting a DNN, it is also necessary to analyze its confusion matrix as follows:

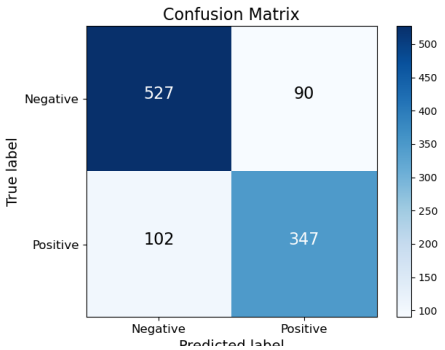


Figure 10: Confusion Matrix of the DNN

Based on the confusion matrix, an accuracy rate of 0.4, a recall rate of 0.77, and an F1 score of 0.53 can be calculated. Although the accuracy rate is 84%, the recall and precision rates are unsatisfactory.

4 Unsupervised Classification

The unsupervised learning method applied in this study is K-means clustering, DBSCAN. However, the silhouette coefficient of DBSCAN is unsatisfactory, so it is not considered to use DBSCAN

4.1 K-means clustering algorithm

K-means clustering divides a dataset into k non-overlapping categories. The basic idea is as follows:

- Pick k cluster centers at random.
- Determine the categorization cluster's center for each set of data.
- Update the location of the cluster center, which is the mean value of all the data in the cluster, in accordance with the categorized cluster.
- As long as the cluster center doesn't move, repeat steps 2 and 3 again.

The aforementioned procedures demonstrate that k-means clustering is straightforward and effective, making it worthwhile to use with the current dataset.

4.2 Data Preprocessing

The preprocessing idea is the same as task1 but different. First, Process outliers and remove the information with label 2. Second, the dataset is processed using maximum-minimum normalization, and then the data is downscaled to 5 dimensions using PCA, because in practice the clustering results in higher dimensions are not satisfactory, but the 5 dimensions before the weight break are chosen in order to retain more data. Last, remove labels from the data set for unsupervised learning.

4.3 Clustering Result Analysis

This study uses the KMeans model in `sklearn`. `K-means++` was chosen as the initialization algorithm in the model because it can select better clustering centers and help avoid convergence to locally optimal solutions.

4.3.1 Elbow Method

Using the elbow method to find the optimal k value is beneficial. In most circumstances, as additional clustering centers are fitted to the data, the variance of the findings lowers as the number of clusters for k-mean clustering rises. However, once the value of k is increased beyond a certain point, the improvement in the number of clusters for explaining the variance will be reduced. Therefore, observing the plot of the squared sum of distances from all data to the cluster centers with respect to the k-value, the ultimate k-value can be determined by finding the elbow where it starts to level off.

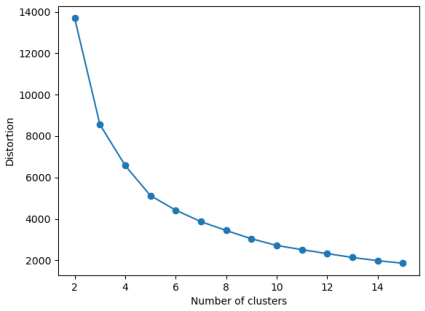


Figure 11: Cluster Number Comparison(elbow method)

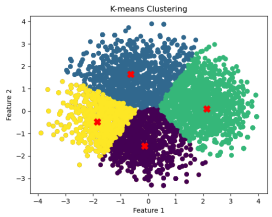
By analyzing the figure 11, it is possible to conclude that the value of k should be 4.

4.3.2 Quality of Clustering Results

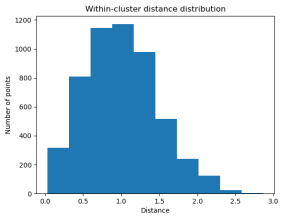
By setting k to 4, the quality of the results can be judged by the intra-class dispersion, the center of clustering.

Observe the location and distribution of cluster centers in Figure 12 left, uniformly distributed in the center of each cluster, while there is a clear distance between different cluster centers.

Analysis of the cluster center distance distribution can drive the intra-class dispersion. The intra-class distance is primarily distributed before 1.5, suggesting little intra-class dispersion and tightly clustered data, resulting in excellent clustering outcomes.



(a) Location and Distribution of Cluster Centers



(b) Cluster Center Distance Distribution

Figure 12

Using `Silhouette_sample` to calculate the resulting silhouette coefficient, the formula for the Silhouette Coefficient is as follows:

$$S = \frac{(b-a)}{max(a,b)} \tag{5}$$

a denotes intra-class distance, b denotes inter-class distance.

The silhouette coefficient reach 0.42, which also proves that the results are valuable.

4.4 Visualization of Results and Application of Results

Visualization of clustering results is a way to present the results visually, and the results are as follows:

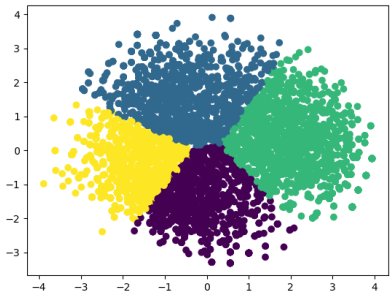


Figure 13: K-means Clustering Result

We can decide to divide the clusters into two groups with diagonal borders based on the distribution pattern of the clusters in order to satisfy the grouping requirements. The clustering effect is ensured and the criteria are met when classes with similar attributes are combined. The result is that the data points are blue and yellow for one group and the others for another group.

4.5 DBSCAN

As DBSCAN detects areas of high and low density, it may be utilized for high-density data. The primary concept is to start with any data as the center of sphere A and discover the region within this sphere that is related by density; if there are additional points B in this region, they all belong to the same class.DBSCAN offers the benefit of discovering clusters of any shape without requiring the cluster count to be specified. However, this dataset is not used since the findings are too subpar.

5 Evaluation of Weaknesses

Although the requirements of the assignment have been met, there are still many shortcomings in the study:

- The best hyperparameters for the neural network were manually selected; the grid search approach was not used.
- The accuracy of XGBoost is not excellent enough, and methods such as L2 regularization should be considered to avoid overfitting.
- Naive Bayes recall and precision rate is not ideal, more detailed preprocessing and more ideal hyperparameters should be considered
- DBSCAN has not found a suitable hyperparameter adjustment method to achieve suitable results
- Only cross-validation was used throughout the classifier design, and no methods such as sensitivity analysis or robustness testing were used to demonstrate the generalizability of the model.

6 Conclusion

This study accomplished three tasks:

In the first section, PCA is used to choose features with high variance and decrease the data's dimensionality to 13 dimensions. In the second section, DNN, XGBoost, and Naive Bayes were used to build supervised learning classifiers, and cross-validation was used to test model performance. Their accuracy rates were 0.84, 0.70.9, 0.69. In the third section, unsupervised learning clustering utilizing k-means clustering produced a final result with a silhouette coefficient of 0.42, demonstrating the accuracy of the grouping. In the last section, analyzed what flaws existed in the study.

References

[1] A. Rehman, A. Khan, M. A. Ali, M. U. Khan, S. U. Khan, and L. Ali, "Performance analysis of pca, sparse pca, kernel pca and incremental pca algorithms for heart failure prediction," in *2020 International Conference on Electrical, Communication, and Computer Engineering (ICECCE)*, 2020, pp. 1–5.

[2] Y. Pei, X. Hongyan, and D. Yuan, "Classification of marine noise signals based on dnn (deep neural networks) model," in *2017 13th IEEE International Conference on Electronic Measurement Instruments (ICEMI)*, 2017, pp. 465–470.

[3] S. Srikar, P. Gowtham, D. Swetha, M. Sri, B. B. Kumar, and S. Bulla, "Deep neural networks (dnn) based sports balls classification," in *2023 7th International Conference on Computing Methodologies and Communication (ICCMC)*, 2023, pp. 373–377.

[4] N. Zhou, Q. Shao, J. Zhou, and H. Changjie, "Fault classification of proton exchange membrane fuel cells for vehicles based on xgboost," in *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, 2021, pp. 1054–1058.

[5] D. P. He, Z. L. He, and C. Liu, "Recommendation algorithm combining tag data and naive bayes classification," in *2020 3rd International Conference on Electron Device and Mechanical Engineering (ICEDME)*, 2020, pp. 662–666.

[6] L. Wenchao, Z. Yong, and X. Shixiong, "A novel clustering algorithm based on hierarchical and k-means clustering," in *2007 Chinese Control Conference*, 2007, pp. 605–609.

[7] N. Rai, P. Meena, and C. Agrawal, "Improving the hate speech analysis through dimensionality reduction approach," in *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2020, pp. 321–325.

[8] N. Cartocci, G. Costante, M. Napolitano, P. Valigi, F. Crocetti, and M. Fravolini, "Pca methods and evidence based filtering for robust aircraft sensor fault diagnosis," 2022, aircraft sensors;D-PCA;Data driven;Evidence based filter;Evidence-based;Fault estimation;Fault isolation;PCA;PCA method;Sensor fault diagnosis;. [Online]. Available: (http://dx.doi.org/10.48550/arXiv.2212.06688)

[9] R. Girao Serrao, M. R. Oliveira, and L. Oliveira, "Theoretical derivation of interval principal component analysis," *Information Sciences*, vol. 621, pp. 227 – 247, 2023, algebraic structures;Covariance matrices;Eigenvalue;Interval algebraic structure;Interval score;Principal Components;Principal-component analysis;Symbolic covariance matrix;Symbolic data;Symbolic data analysis;. [Online]. Available: (http://dx.doi.org/10.1016/j.ins.2022.11.093)

[10] U. Radojičić, N. Lictzén, K. Nordhausen, and J. Virta, "Dimension estimation in two-dimensional pca," in *2021 12th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2021, pp. 16–22.