International Journal of Computer Science and Information Technology
ISSN: 3005-9682 (Print), ISSN: 3005-7140 (Online) | Volume 4, Number 3, Year 2024
DOI: https://doi.org/10.62051/ijcsit.v4n3.21
Journal homepage: https://wepub.org/index.php/IJCSIT/index

A Comparative Study of Pattern Recognition Models on the PaviaU Dataset

Xuerui Wang, Senwei Liang, Hanjun Wu, Xinkun Wang

School of AIAC, Xi'an Jiaotong-Liverpool University, Suzhou, China

ABSTRACT

This report provides a comprehensive study focused on land cover classification and remote sensing image analysis using the PaviaU dataset. The report first introduces the basic characteristics and application background of the data set, and then discusses in detail the application of feature selection and dimensionality reduction techniques, especially linear discriminant analysis (LDA) and principal component analysis (PCA). In terms of model application, the report not only uses traditional machine learning models, such as support vector machines, Bayesian and KNN posterior probabilities, but also explores the performance of these models under different parameter settings. Through a series of experiments, the study found that optimizing feature selection and dimensionality reduction techniques can significantly improve the classification accuracy of the model. Finally, the report compares different models and suggests alternative improvements.

KEYWORDS

Pattern Recognition; Dimensionality Reduction; Model Evaluation

1. DATASET

1.1. Dataset Description

PaviaU is one of the two datasets obtained from Pavia, Northern Italy using the ROSIS sensor [1]. The PaviaU dataset captures the scene acquired by the ROSIS sensor during a flight campaign over Pavia, Northern Italy. It features a total of 103 spectral bands for Pavia University, which is originally 610x610 pixels in size. However, some samples within the images contain no information and need to be removed prior to analysis.

Figure 1. PaviaU dataset

#	Class	Samples
1	Asphalt	6631
2	Meadows	18649
3	Gravel	2099
4	Trees	3064
5	Painted Metal Sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682
9	Shadows	947

Figure 2. 9 distinct classes in ground truth map and number of samples

Pavia scenes were provided by Prof. Paolo Gamba from the Telecommunications and Remote Sensing Laboratory, University (Italy)

The PaviaU dataset is primarily used for land cover classification and remote sensing image analysis, offering high spatial and spectral resolution to enable detailed studies of the Earth's surface. It is particularly valuable for tasks such as urban land cover classification, vegetation monitoring, and object recognition, as well as for research in remote sensing image processing and computer vision [2].

The given PaviaU dataset is a high-spatial-resolution hyperspectral remote sensing dataset primarily used for land cover classification and remote sensing image analysis. It consists of a hyperspectral image with dimensions 610x340 pixels, capturing 103 spectral bands for different spectral features of the Earth's surface.

In addition to the hyperspectral image, the dataset is accompanied by a ground truth map of dimensions 610x340 pixels which differentiates 9 distinct classes. It is used to label different land cover categories within the image:

1.2. Feature Visualization

Here's a visualization of 8 different randomly selected bands:

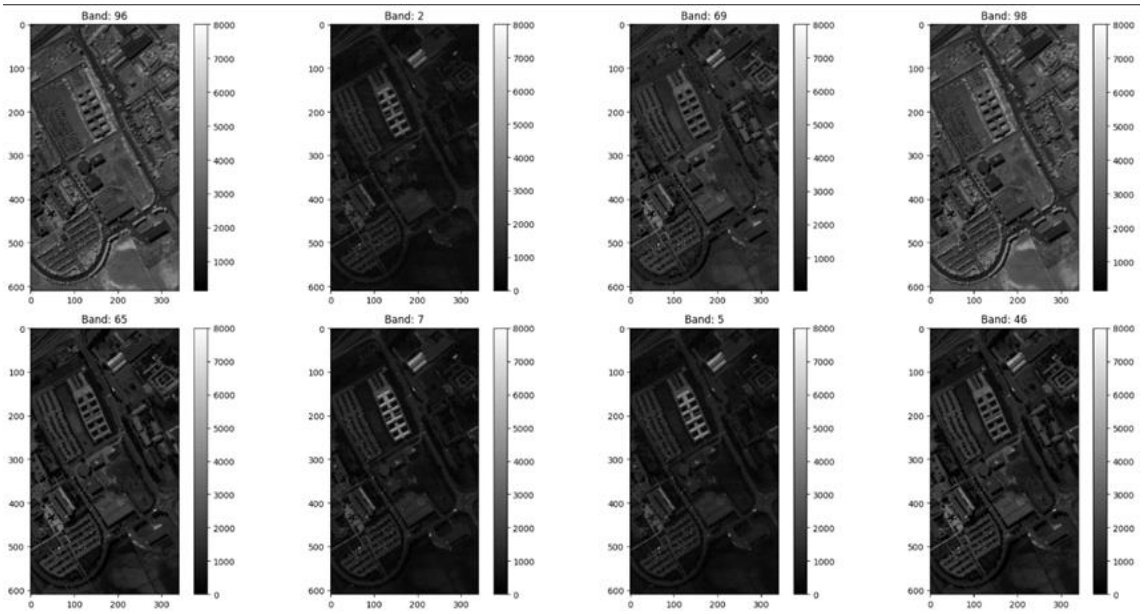

Figure 3. 8 different randomly selected bands

This is a single-band image with a color scale, we have picked three bands at random and used color mapping to enhance the contrast and differentiation of the display:

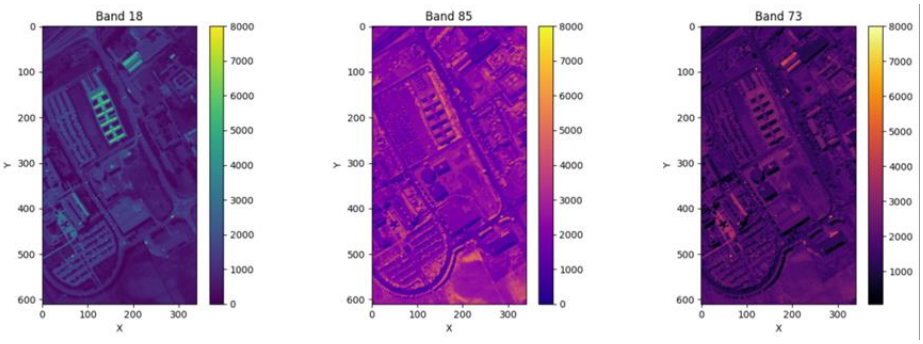

Figure 4. Single-Band image with color scale

This is a random selection of three bands, then composing them together into RGB channels:

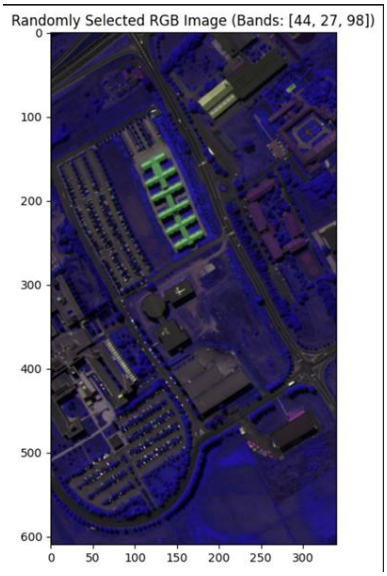

Figure 5. Random selected RGB image

This is a visualization of spectral curves. We randomly selected six different locations and plotted their spectral response curves. This can help understand the spectral characteristics and distinctiveness of different materials:

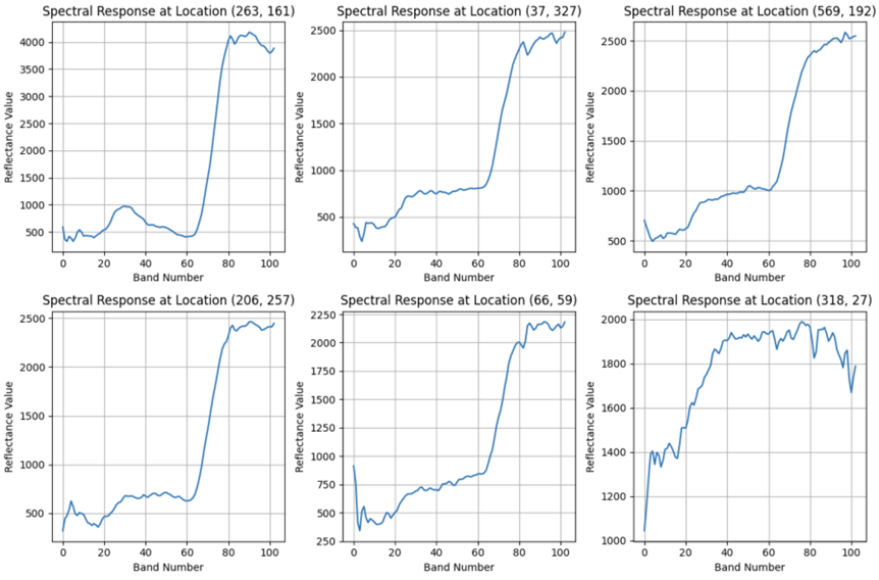

Figure 6. The spectral characteristics and distinctiveness of different materials

This is a 3D scatter plot in feature space. We display data points in three-dimensional space, with each dimension representing a different wavelength:

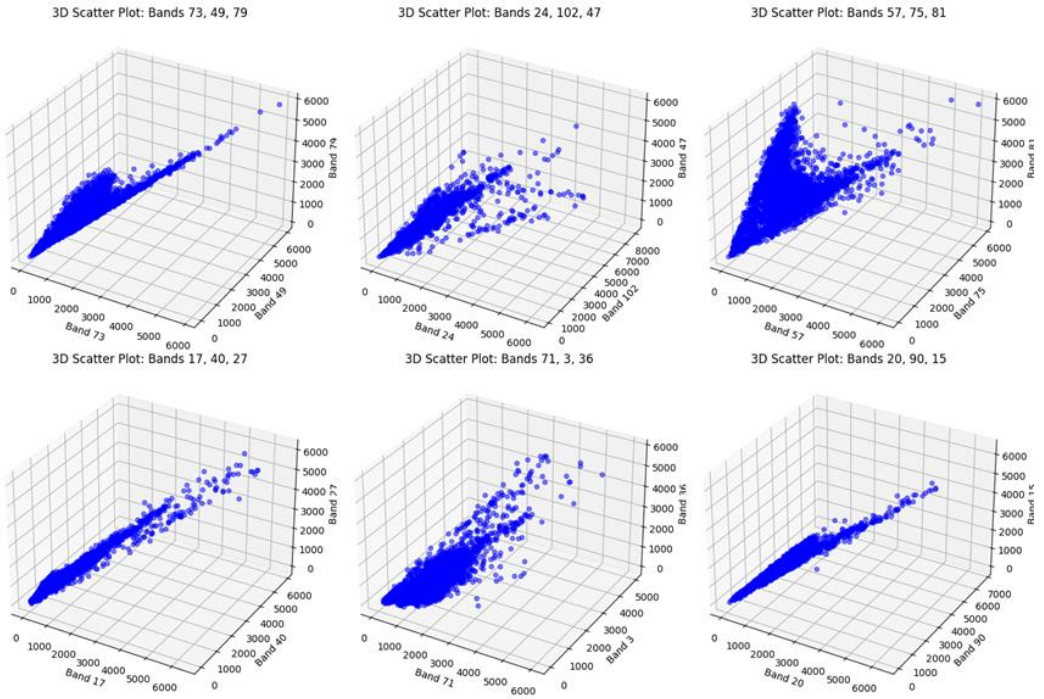

Figure 7. 3D scatter plot in feature space

This is a heatmap we have created, which provides an intuitive visualization of the correlation or covariance between different spectral bands. In this heatmap, the color intensity represents the magnitude of the correlation or covariance:

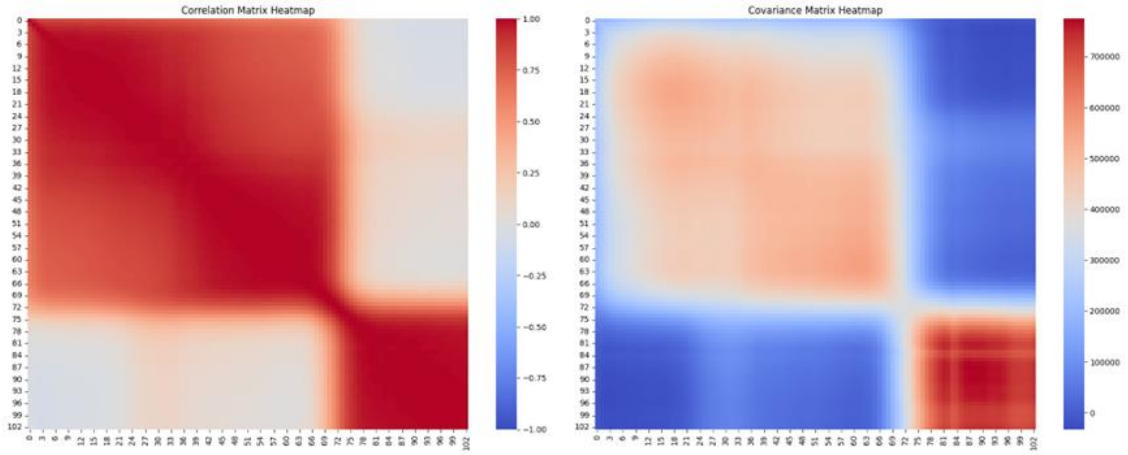

Figure 8. Heatmap of the correlation or covariance between different spectral bands

This is a network graph we have generated to display the relationships between spectral bands. In this graph, nodes represent spectral bands, and edges represent the correlations or covariances between spectral bands. The width or color of the edges can indicate the magnitude of the correlations or covariances:

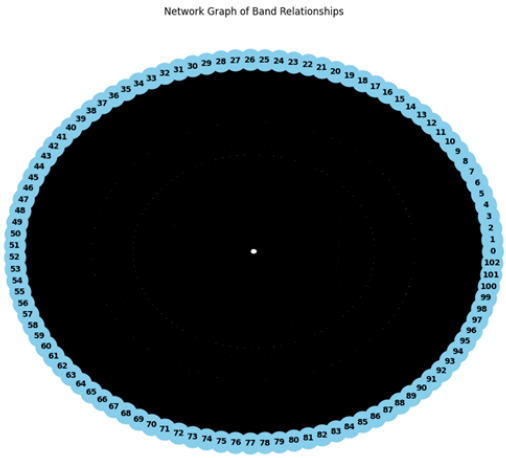

Figure 9. Network Graph of Band Relationships

For the ground truth:

This is a sample of the ground truth map. It displays the land cover classification labels for each pixel in the hyperspectral image using different colors:

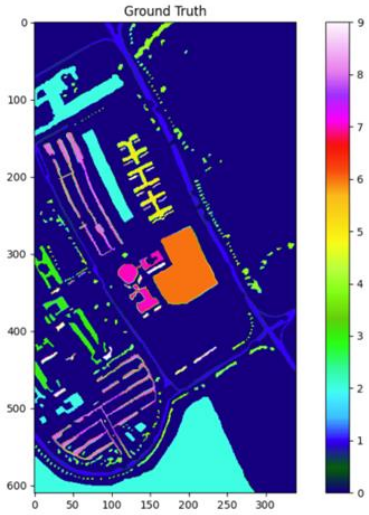

Figure 10. The Map of Ground Truth

This involves using histograms and pie charts to depict the distribution of each category within the Ground Truth image:

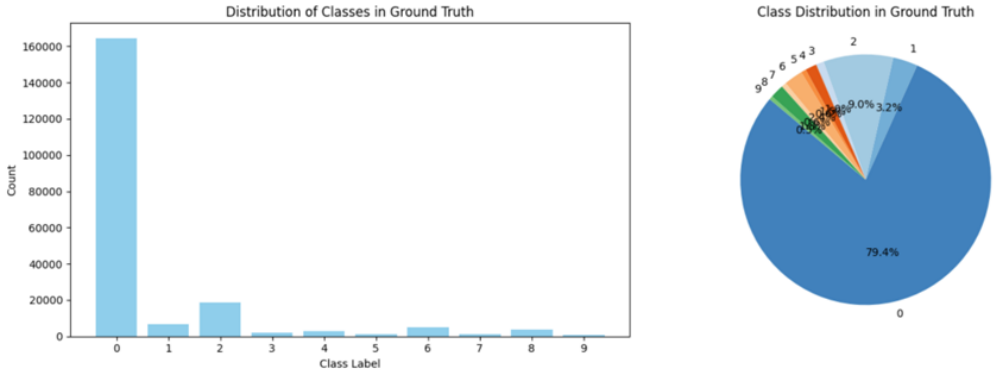

Figure 11. Distribution of Classes in Ground Truth

It can be observed that the part labeled as class 0 in the Ground Truth image accounts for 80% of the total data. This particular class typically represents the background or unlabeled regions, i.e., the parts of the image that are not of interest or relevant objects or regions. The background class is usually

represented by 0 because it signifies areas not assigned to any other class. Therefore, in the upcoming classification task, we will exclude class 0 to achieve a more efficient, clear, and easy-to-evaluate model.

This involves merging image data and ground truth data into a single DataFrame and then plotting a boxplot. This approach allows us to understand the differences in data distribution between different classes, identify potential outliers or anomalies, and gain insights into the data's distribution:

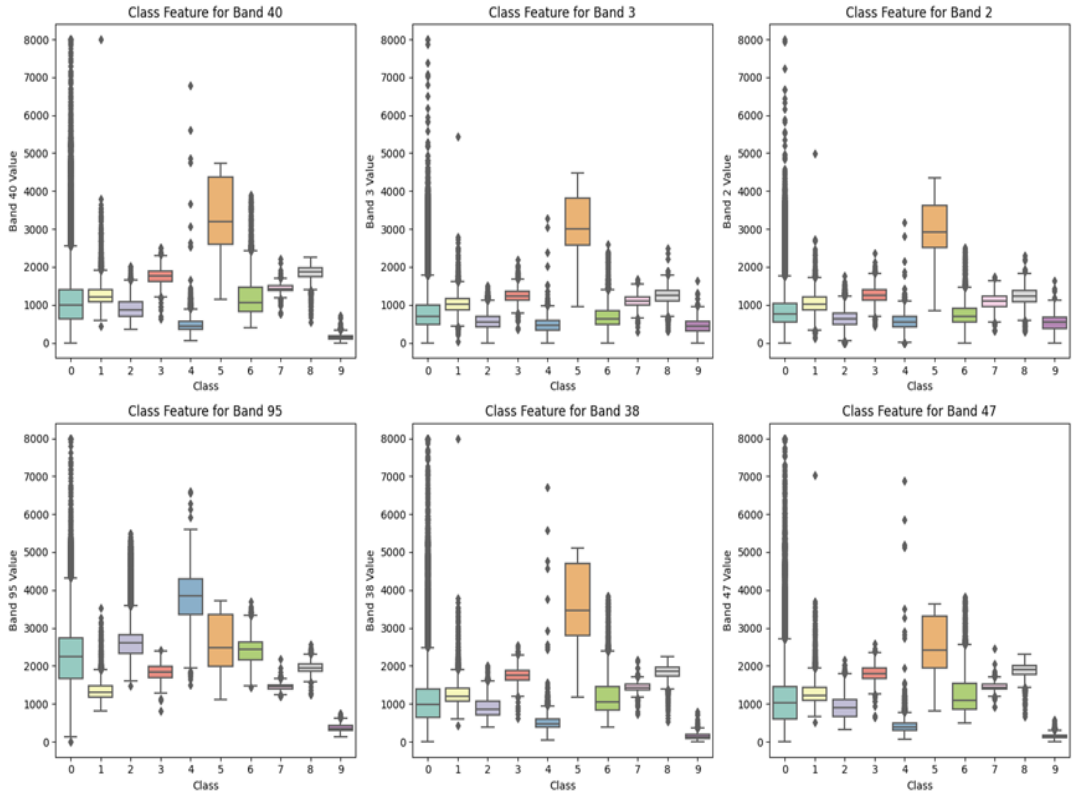

Figure 12. Boxplots of data distribution between different classes

2. IMPLEMENTATION

2.1. Workflow

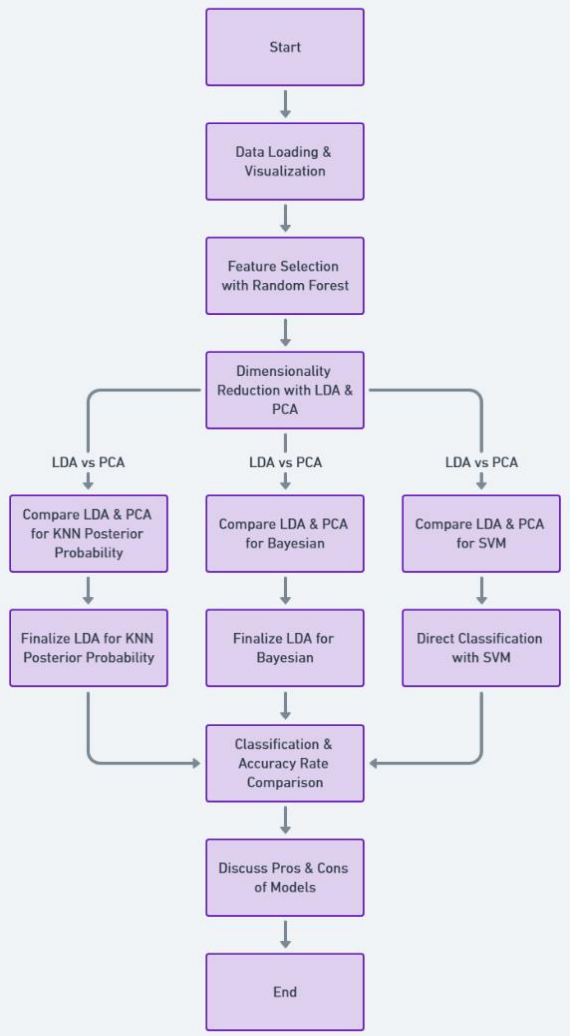

Figure 13. Data classification flow chart

2.2. Feature Selection

Why we need feature selection?

The dataset PaviaU.mat has 103 features, which is too large to make the model too complex and difficult to interpret. For faster training and prediction time, we must perform feature selection. At the same time, fewer features can help the risk of overfitting the training data.

2.2.1. Basic principles of Random Forest (RF)

We utilize the RF for selection features. RF build multiple decision trees, each of which is trained on a subset of the data and uses a random subset of features in its construction [3]. RF predictions are obtained by voting (classification problems) or averaging (regression problems) the predictions of all trees.

$$\mathrm{RF(x)} = \frac{1}{B}\sum_{b=1}^{B} T_b\,(x)$$

Where:

B is the number of the trees

Tb(x) is the prediction of sample x by the bth tree

2.2.2. Feature Selection

In addition to having good prediction accuracy, RF provides two feature selection methods: Mean Decrease Impurity (MDI) and Mean Decrease Accuracy (MDA). Our feature selection for PaviaU.mat is based on the MDI.

The MDI score of a variable is calculated by summing the reduction in node impurity (usually called Variance), especially when feature X^j is used for data partitioning. According to Scornet (2020) [4], this impurity reduction is defined as:

$$\Delta V(t) = V(t) - p_L V(t_L) - p_R V(t_R)$$

When data is partitioned at node t, two child nodes are generated, denoted tL and tR, tL= NtL / Nt and PR= NtR / Nt represent the ratio of data arriving at child nodes tL and tR, respectively.

This figure below shows the structure of MDI method:

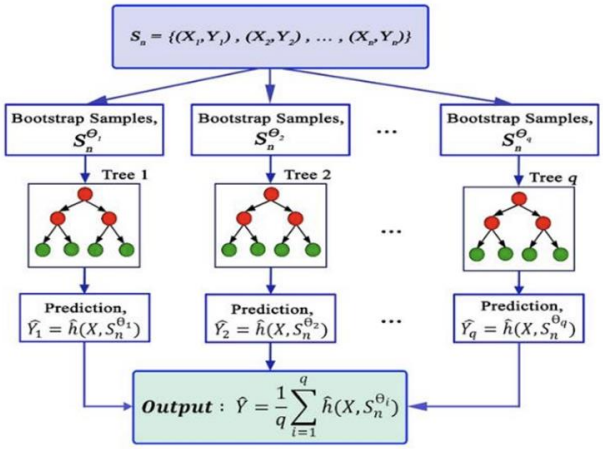

Figure 14. The structure of MDI method

In the case of the RF model, the score of MDI is acquired by averaging the scores of all q trees in the forest [5].

$$MDI = \frac{1}{q}\sum q \sum_{t \in q} p(t)\, \Delta V(t)$$

Where:

q is the number of trees

qP(t) is the node set of the bth tree

2.2.3. Algorithm Application

We iterate over different numbers of decision trees (n_estimators), from 100 to 200 with a step size of 50. For each number of decision trees (n_estimators), we create a random forest classifier and train it on the data. Then the feature importance is obtained and the median value is calculated as the threshold for feature selection. Select features based on a threshold, update the best number of features (best_num_features), the best number of decision trees (best_n_estimators), and a list that stores the number of selected features in each iteration (best_select_feature). Finally we get the optimal n_estimators value and the number of selected features.

According to the figure of the number of selected features and decision tree below, we can see that the quantities of decision trees are from 100 to 200, the step size is 50, and the number of filtered features is 51.

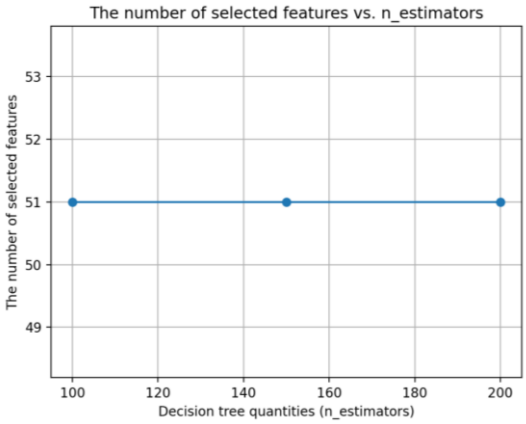

Figure 15. The number of selected features vs n_estimators

By using the median feature importance as a threshold, we can determine that the filtered feature has the best value when the decision tree is 100, and the shape of the filtered data becomes (42776, 51).

2.2.4. Strengths of Random Forest

We also use SelectKBest for feature selection and compare it with RF to illustrate the superiority of RF.

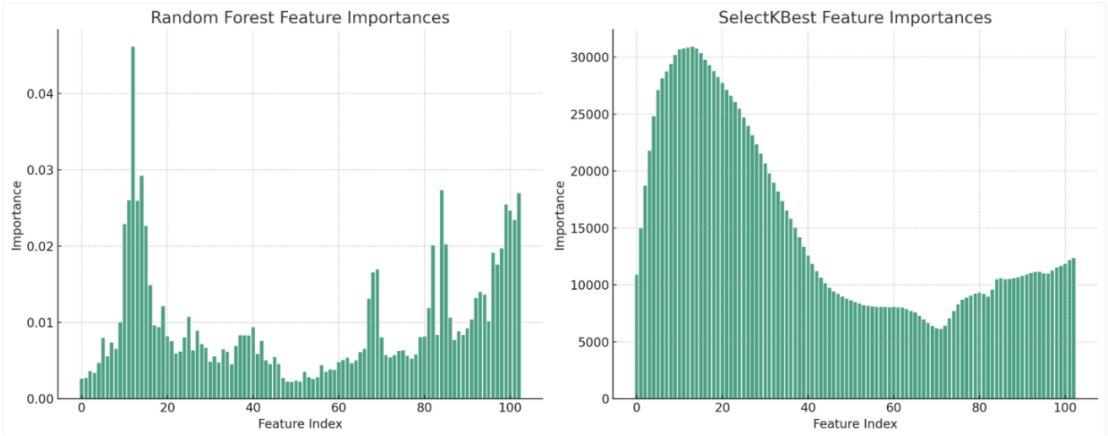

Figure 16. Compare with SelectKBest and RF

Advantages of Random Forest:

(1) Model complexity:

Random forest can handle complex data sets and non-linear relationships, while SelectKBest is mainly suitable for linear models.

(2) Robustness:

Random Forest is not susceptible to outliers and noise, which SelectKBest may be affected by.

(3) Generalization ability

Random forest generally has better generalization ability because it is an ensemble method that combines the predictions of multiple decision trees.

(4) Multi-objective support:

Random Forest can be used for multi-classification problems, while SelectKBest requires feature selection for each category, which can be more complex.

(5) No extra steps required:

Random Forest performs feature selection naturally during training, while SelectKBest requires an extra step to select features.

2.2.5. Investigate and experiment on the data

In order to analyze why the eigenvalue is optimal when the decision tree is 100, we calculated the mean, standard deviation and variance of these 51 eigenvalues.

Table 1. The Mean, standard deviation and variance of 51 eigenvalues

Feature	Mean	Standard Deviation	Variance
1	880.8961	660.1023	707.0
2	883.0966	685.5730	699.0
3	877.5243	726.7483	687.0
4	886.9314	751.6635	687.0
5	898.7503	775.7333	695.0
6	895.4971	789.0844	693.0
7	897.9141	805.6931	694.0
8	905.6715	820.4087	700.0
9	913.7094	830.2729	705.0
10	917.8967	836.2771	708.0
11	928.3730	842.4697	722.0
12	940.2070	845.0398	739.0
13	951.9673	842.5662	755.0
14	966.5009	838.6600	774.0
15	982.8947	832.2484	794.0
16	1079.6560	791.3762	899.0
17	1112.0332	786.2305	932.0

(Only show part of content)

At the same time, we employ the box plot to better observe the differences between these 51 eigenvalues, and we analyze these 51 eigenvalues from the mean, standard deviation and variace respectively.

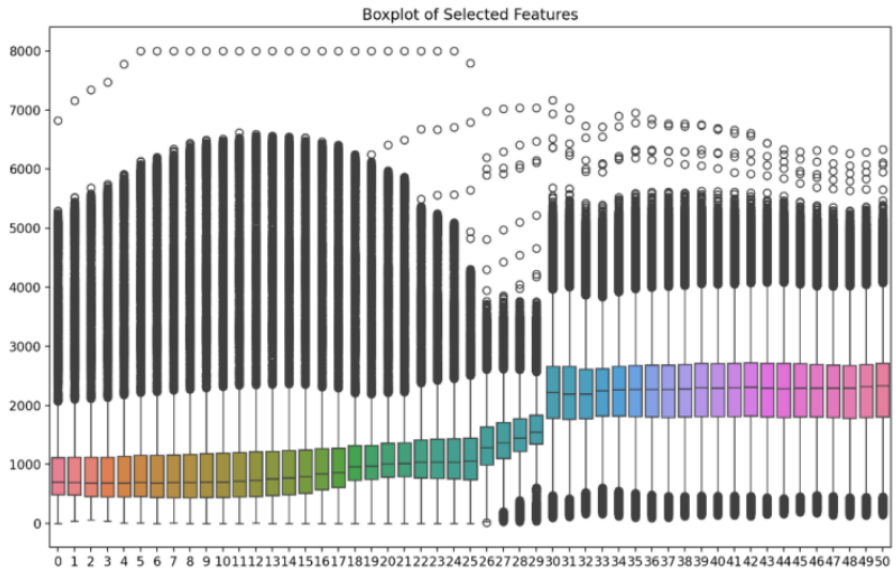

Figure 17. Boxplot of 51 eigenvalues from mean, standard deviation and variance

Mean:

Lower range: Features 1 to 5 have relatively low average values.

Midrange: Features 6 to 24 have gradually increasing average values.

Higher range: Features 25 to 51 have higher averages, with feature 30 showing a significant jump.

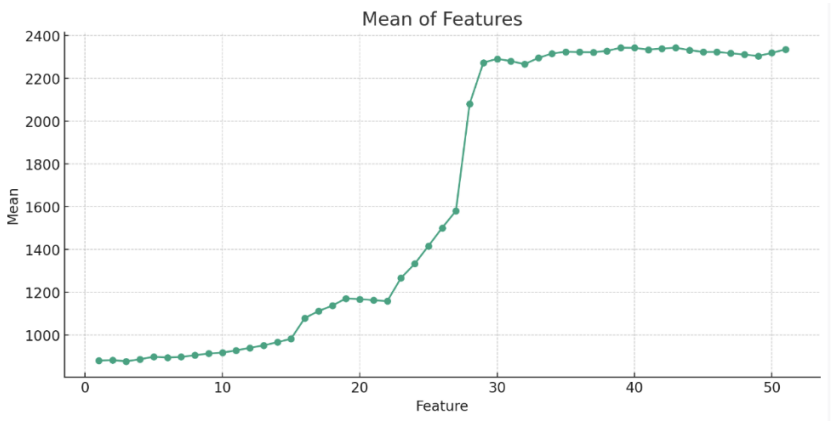

Figure 18. The mean of Feature

Standard deviation:

Increasing trend: The standard deviation generally increases from feature 1 to feature 9.

Downtrend: From feature 15 to feature 25, the standard deviation decreases slightly.

Lower variability: Features 26 to 29 have lower standard deviations, indicating a smaller spread around the mean.

Higher variability: Features 30 to 51 have higher standard deviations, indicating a larger spread around the mean.

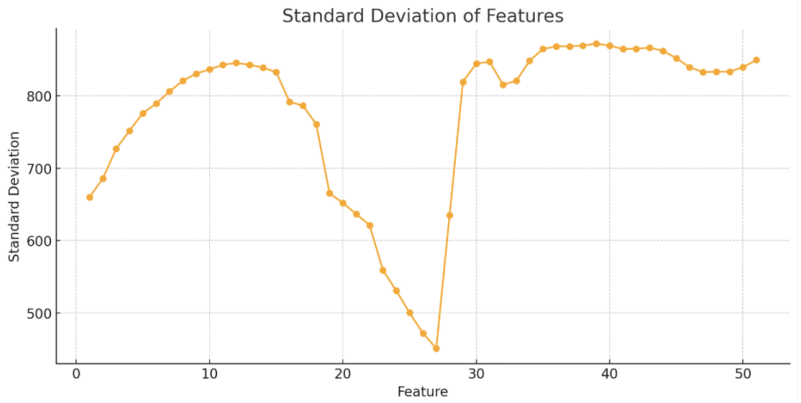

Figure 19. The Standard Deviation of Features

Variance:

General trend: Variance values seem to increase with more features, but with some fluctuations.

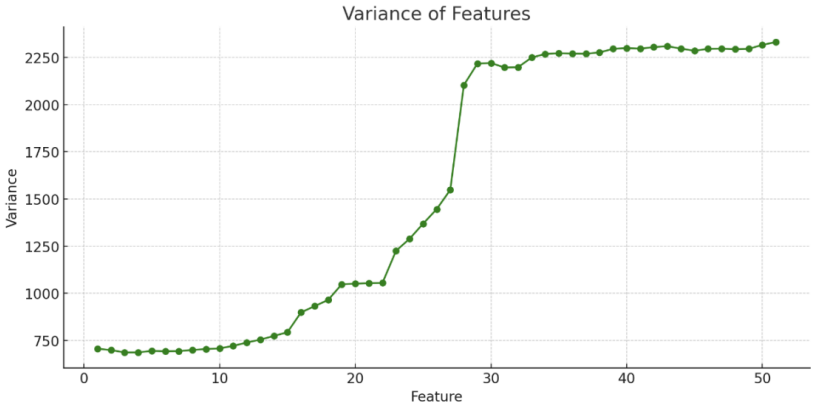

Figure 20. The Variance of Features

Conclusion:

The data shows a trend of increasing mean values, features can be divided into different groups based on their statistical properties, and there are clear differences between them.

2.3. Dimensionality Reduction

Why dimensionality reduction is needed?

Even after feature selection, the number of features may still be large, and dimensionality reduction can reduce the computational complexity. And helps remove noise and potentially improve model performance.

2.3.1. Linear Discriminant Analysis (LDA)

Principle of LDA

Between-class variance:

Between-class variance represents the difference between the means of different classes. The aim is to maximize this spread, making the classes as different as possible [8].

$$S_B = \sum_{i=1}^{C} N_i \, (\mu_i - \mu)(\mu_i - \mu)^T$$

Where:

C is the number of classes

Ni is the number of samples in class i

μi is the mean vector of class i

μis the overall mean vector of the data

(μi-μ)(μi-μ)^T is a matrix that represents the spread of class i's mean from the overall mean

For each class, we consider the number of samples in that class and observe the distance between its mean and the overall mean of the data. We then sum these values across all classes. The result is a matrix SB that represents the between-class scatter.

Within-class Variance:

Within-class variance represents the distribution within each class. The purpose of this is to minimize the distribution so that samples in the same class are as close to each other as possible [8].

$$S_W = \sum_{i=1}^{C} \sum_{x \in C_i} (x - \mu_i) \, (x - \mu_i)^T$$

Where:

Ci is the set of samples belonging to class i

x is a sample vector

(μi-μ)(μi-μ)^T is a matrix that represents the spread of sample x from the mean of its class

For each class, we observe the distance of each sample from the mean of the class and then sum the values of all samples in the class. We then sum these values across all classes. The result is a matrix Sw that represents the within-class scatter.

Objective of LDA:

The goal of LDA is to find a projection that maximizes inter-class variance and minimizes within-class variance. This is achieved by finding a matrix W that maximizes the following objective function [8]:

$$J(W) = \frac{Between-Class\ Variance}{Within-Class\ Variance} = \frac{W^T S_B W}{W^T S_W W}$$

Where:

W is the matrix containing the projection vectors

SB is the between-class scatter matrix

Sw is the within-class scatter matrix.

The optimal projection vectors W can be found by solving the generalized eigenvalue problem:

$$S_B W = \lambda S_W W$$

Dimensionality reduction with LDA:

Using LDA can reduce the data to at most c-1 dimensions, where c is the number of classes. This is because LDA tries to find directions that maximize the separation between categories, and if there are c categories, there can be at most c−1 such directions [9].

In the following content, we will select the dimensionality reduction number of LDA with the highest accuracy through cross-validation.

LDA suitable model:

Classification model: LDA itself is a classification algorithm and therefore works well with other classification models such as logistic regression, SVM, etc.

Gaussian distribution assumption: LDA usually performs well if the data (or features) roughly fit a Gaussian distribution.

Small sample size: LDA usually outperforms PCA for small sample size problems.

2.3.2. Principle Component Analysis (PCA)

(1) Theoretical Foundations of PCA

Principal Component Analysis (PCA) is a Dimensionality reduction technique in multivariate analysis. Its primary objective is to orthogonalize a set of potentially correlated variables into a reduced set of linearly uncorrelated variables [10].

For the dataset X, aiming to utilize linear transformation a and make the variance maximum, we get:

$$J_v = \frac{1}{N}\sum_{n=1}^{N}(a^T X_n - a^T \bar{X})^2 = a^T S a$$

Where:

$$S = \frac{1}{N}\sum_{n=1}^{N}(X_n - \bar{X})(X_n - \bar{X})^T = \frac{1}{N} X_c X_c^T$$

With the restriction of $a^T a = 1$, we can transform the problems into Lagrange Optimization problem:

Maximum $J = a^T S a - \lambda(a^T a - 1)$

After differentiation, we can get:

$$\frac{\partial J}{\partial a} = 2Sa - 2\lambda a = 0$$

It shows that the a is eigen vector of S.

(2) Core Objectives and Implications of PCA

In PCA method, there are three main objectives of data visualization, feature selection, and dimension reduction.

Dimensionality Reduction: High-dimensional data often suffers from the "disaster of dimensionality," making analyses computationally intensive and less interpretable [11]. PCA solves this by transforming the original variables into a new data set of uncorrelated principal components. It ensures the initial components encapsulate the majority of the data's variance.

Data Visualization: By reducing data to two or three dimensions, PCA offers a tangible representation. Visualization makes the representation more visible.

Feature Selection: Beyond mere reduction, PCA's transformative capabilities can distill salient features, potentially enhancing the performance of model.

(3) PCA implementation

We utilize “sklearn.decomposition” library to implement PCA method. It receives the data from the feature selection implemented by random forest and execute PCA.

Explained Variance:

Explained variance provides a quantifiable measure of how much information (variance) each principal component retains [12]. In essence, this could be used to evaluate the importance of each component in representing the original data's structure.

We plot a scree to delineate the explained variance across dimensions. It could guide the selection of the optimal number of components.

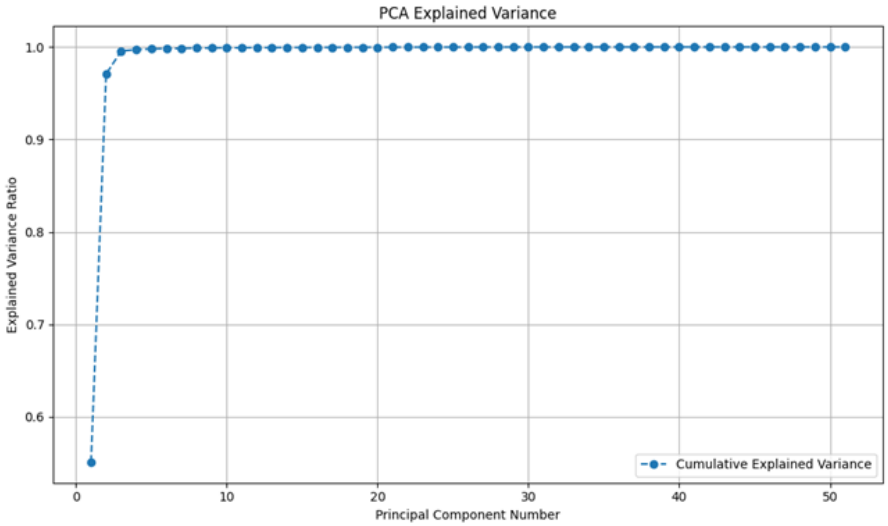

Figure 21. Use PCA to explain variance

From this figure, we can find that explained variance exceed 95% when we select 2 principle component. When component number equal 3, it approach to 99%. This indicates that 3 components are able to contain most of information for pattern recognition. Thus, in next step, we choose the three component and visualize them check whether these classes can be clearly classified.

Visualization Post PCA Transformation:

When the number of principle component equal to 2 or 3, the data was then transmuted and visualized. This figure offers a spatial representation, elucidating the distribution and potential clustering of data points.

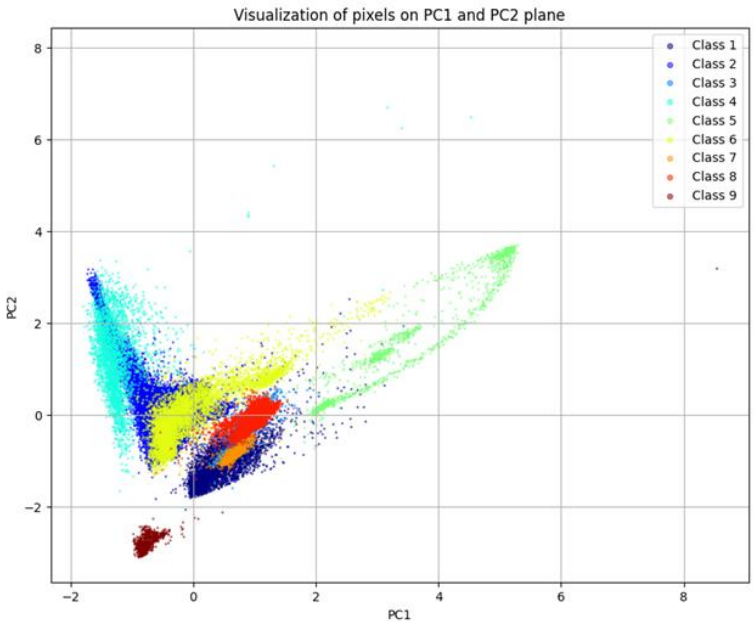

Figure 22. The Visualization of Pixels on PC1 and PC2 plane

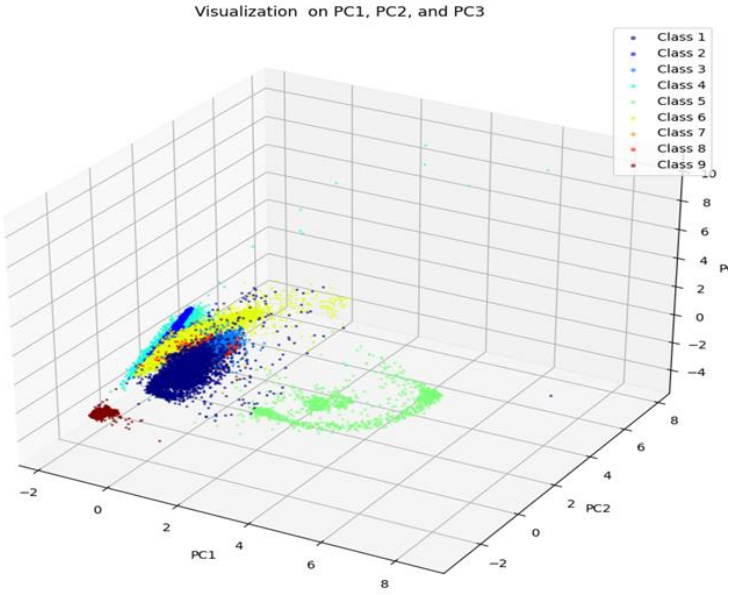

Figure 23. The Visualization of Pixels on PC1 and PC2 and PC3 plane

In these figures, we find that data points in different classes were not distinctly separable.

the data points exhibited significant overlap, suggesting that a two or three-dimensional representation might not be sufficient for clear class differentiation.

Augmented Analysis and Validation:

Initial visual inspections intimated suboptimal linear separability in reduced dimensions. To redress this, we need to choose save feature information. In next model section, we e will utilize classification accuracy as a metric and seek for an optimal principle component number for each model.

2.4. Model 1: Navie Bayes Classifiers

2.4.1. Model Description

The Naive Bayes classifier is a probabilistic classifier based on the Bayes theorem, with strong independence assumptions between the features. Despite its simplicity, it has shown remarkable success in various applications [13]. This section conducts a review of the basic theory of naive Bayes classifiers.

The primary goal in Bayesian inference is to maximize the posterior probability. In the context of classification, this means assigning a sample to the class that has the highest posterior probability given the observed features of the sample.

Mathematically, this can be represented as:

$$\omega_i = argmax(P(\omega_i|\mathrm{X}))$$

Where: ω_i is the class that maximizes the posterior probability.

X is the observed features points of the sample

Using Bayes theory, we can find that this formula can be written as:

$$P(\omega_i|\mathrm{X}) = \frac{P(x|\omega_i)\cdot P(\omega_i)}{P(X)}$$

Considering that P(X) is certain in one sample, we transfer the problems into

$$\omega_i = argmax(P(\omega_i|\mathrm{X})) \propto argmax(P(x|w_i)\cdot P(w_i))$$

Function and classification process:

We have introduced the basic principle of Bayes classifier. Then we will introduce the detailed working process of Bayes classifiers.

(1) Classification Ability:

The Naive Bayes is fundamentally a probabilistic classifier. It generally used for binary or multiclass classification tasks.

(2) Adaptable to Multiclass Problems:

Due to its inherent design, the Naive Bayes classifier can be seamlessly adapted to scenarios requiring multiclass predictions.

(3) Text Classification/Spam Filtering

Bayes classifiers show effectiveness in text categorization tasks, such as spam email filtering and sentiment analysis.

2.4.2. Environment and Hardware

Environment: Python 3.10, anaconda.

Hardware: 13th Gen Intel (R) Core (TM) i5-13500HX 2.50 GHz

Special Library: Intel extension for scikit-learn (accelerate the training on CPU.

2.4.3. Parameters and Training Procedure

In this section, we implement the parameters estimation procedure in Bayes classifier, which include mean μ and variance σ^2. Additionally, the reduce dimension optimization will be conducted and we

will choose the dimension for dimensionality reduction based on accuracy to achieve the optimal result.

(1) Parameters Estimation

Prior Probability Estimation

We utilize the default prior probability estimation method in sk-learn bayes classifiers. It calculates the percentage of one class in training set as its prior probability.

$$P(\omega_i) = \frac{N(w_i)}{N}$$

Mean and variance Estimation with Maximum Likelihood Estimation

MLE aims to find the parameter values that maximize the likelihood function, which measures how well the model explains the observed data. The likelihood for a set of parameter values is the probability of the observed data given those values [14].

Given a dataset, the goal is to estimate the parameters that maximize the likelihood of observing this data under a normal distribution assumption.

The likelihood function for a normal distribution is given by:

$$L(\mu, \Sigma) = \Pi_{i=1}^{n} \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(X_i - \mu)^T \Sigma^{-1}(X_i - \mu)\right)$$

To find the MLE estimates, we differentiate the log-likelihood with respect to

and equate to zero.

For mean:

$$\hat{\mu}_{ML} = \frac{1}{N}\sum_{k=1}^{N} X_k$$

For the variance:

$$\hat{\sigma}_{ML}^2 = \frac{1}{N}\sum_{k=1}^{N}(x_k - \mu)^2$$

(2) Dimension Optimization with PCA and LDA

This section presents a comprehensive study on the application of Bayesian classification post dimensionality reduction with Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). Through rigorous validation experimentation across varying dimensions, this research aims to identify the optimal dimensionality that maximizes classification accuracy.

(1) Dimension Optimization with PCA

Since we Since we have selects 51 spectrums from initial datasets with random tree algorithm, the PCA dimension will range from 1 to 51. In order to determine the best dimension, we conduct a cross validation experiment on bayes classifiers. As a result, we utilize a figure to measure the accuracy on different dimension with PCA.

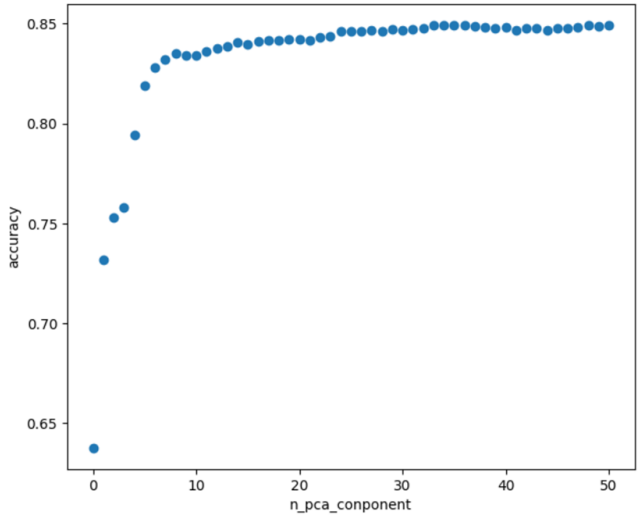

Figure 24. Meansure accuracy on different dimension with PCA

Upon applying PCA, it was observed that the accuracy of the Bayesian classifier varied across dimensions. In the case of PCA, when the dimension equal 33, the highest accuracy was 85%.

(2) Dimension Optimization with LDA

Since we only have 9 classes in datasets, the LDA dimension range from 1-9. Similar to PCA, we plot a figure to show the accuracy in different dimension.

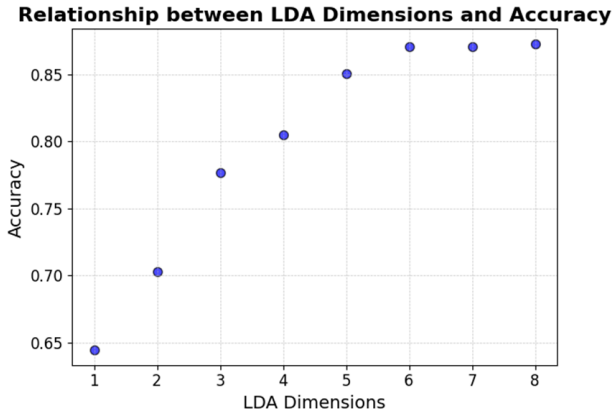

Figure 25. Relationship between LDA Dimensions and accuracy

From the plot, we can find that when dimension equal to 8, the bayes classifier achieve the highest accuracy of 87%.

2.4.4. Evaluation

After the dimension validation, we have confirmed that when the dimension is 8 with LDA method, bayes could achieve the highest accuracy. Here we will utilize confusion matrix and other metrics to conduct an evaluation.

Here we use confusion matrix to evaluate the model. The given confusion matrix presents the predictive performance of the model across various classes. Observing the diagonal values, it's evident that most predictions are accurate, specifically in class 2 and class 5. Furthermore, several classes exhibit misclassification counts of zero or very low, suggesting that the model distinctly recognizes them with high confidence. However, for classes 1, 3, and 7, although the correct predictions remain the highest, they present relatively higher misclassifications when compared to other classes. Overall, bayes classifiers exhibits a high predictive capability.

Confusion matrix is intuitive, now we could use data to analyze model more concretely.

Here we utilize accuracy, precision, recall and F1-score.

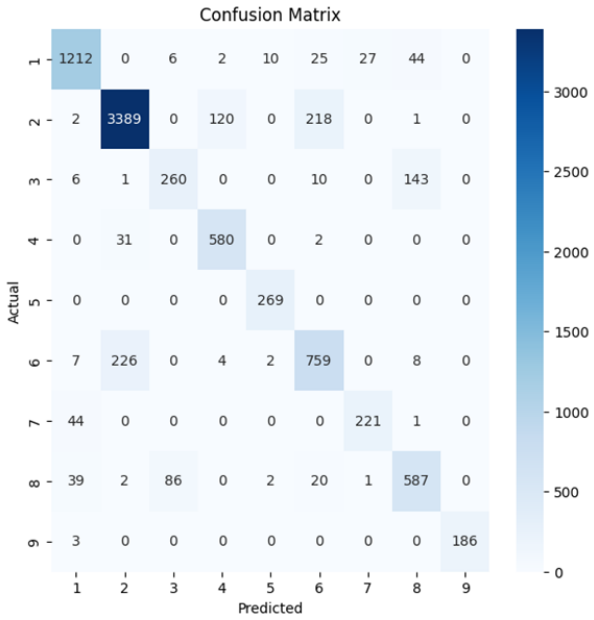

Figure 26. Condusion matrix to evaluate

Table 2. Precision, recall, F1-Score, and support

	Precision	Recall	F1-Score	Support
1	0.92	0.92	0.92	1326
2	0.91	0.91	0.91	3730
3	0.76	0.59	0.66	420
4	0.80	0.95	0.87	613
5	0.95	1.00	0.97	269
6	0.75	0.70	0.72	1006
7	0.88	0.82	0.85	266
8	0.75	0.82	0.79	737
9	1.00	0.99	0.99	189
Accuracy			0.87	8556
Macro Avg	0.86	0.86	0.86	8556
Weighted Avg	0.87	0.87	0.87	8556

2.5. Model 2: K-NearestNeighbor

2.5.1. Model Description

Theory

KNN (K-Nearest Neighbor Algorithm) is an instance-based learning algorithm commonly used for classification and regression tasks. For a new unknown sample, KNN finds the k samples closest to that sample in the training set (i.e., the "nearest neighbors"), and then predicts the properties of the unknown sample based on the properties of those neighbors [15].

Function

11. o determine which samples are the "nearest neighbors" of the new sample, we need a distance metric. Common distance measures are:

Euclidean Distance: For two points P (x1, y1) and Q (x2, y2).The Euclidean distance is defined as:

$$d(P,Q) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

For multi-dimensional Spaces, this formula can be extended to:

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

Or a classification task, KNN considers the number of samples from each class in the k nearest neighbors and assigns new samples to the class with the most number.

The posterior probability of KNN is the probability that, given an input sample, each class is predicted to be the correct class. For classification problems, not only the most likely category is represented, but also the probability of this prediction can be known [16].

Calculate a posteriori probability: For each class ki, calculate how often it occurs in these k neighbors, which can be expressed by the following formula:

$$P_n(X, \omega_i) = \frac{k_i}{nV}$$

$$P_n(\omega_i|X) = \frac{P_n(X,\omega_i)}{\sum_i^N P_n(X,\omega_i)} = \frac{k_i}{k}$$

Pn (x, wi):The sum of the joint probabilities of all possible classes equals the number k of all samples in the window.

2.5.2. Environment and Hardware

Hardware

CPU:

13th Gen Intel (R) Core (TM) i9-13980HX 2.20 GHz

RAM:

16GB

Environment

Sklearnex: To improve CPU utilization, the sklearnex package, a Python package developed by Intel to accelerate Scikit-learn, was installed in the environment.

2.5.3. Parameters and Training Procedure

(1) Select the reduce dimension

The dimensionality reduction methods were PCA and LDA. The two methods are traversed and the one with the highest accuracy is chosen as the dimensionality reduction method.

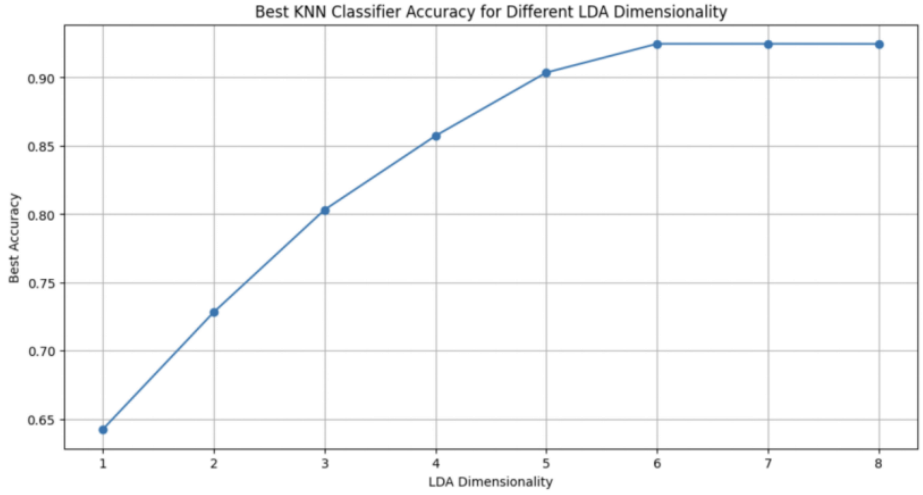

Figure 27. Best KNN classifier accuracy for different LDA dimensionality

This graph shows the best accuracy using K-nearest neighbor (KNN) classifiers in different linear discriminant anysis (LDA) dimensions.

The increased LDA dimension is positively correlated with the improved accuracy: As the LDA dimension increases from 1 to 6, the accuracy of the KNN classifier also increases.

Accuracy saturation: at LDA dimensions 6, 7, and 8, accuracy appears to have saturated and remained at the same level without further increase.

Initial gain is significant: The increase in accuracy is particularly significant as the LDA dimension increases from 1 to 3. This could mean that these preliminary dimensions capture major changes in the data and help improve the classifier's performance.

In linear discriminant analysis (LDA), the optimal dimension is 6. For this dimension, the optimal value of k is 13. The highest accuracy obtained is 92.46%.

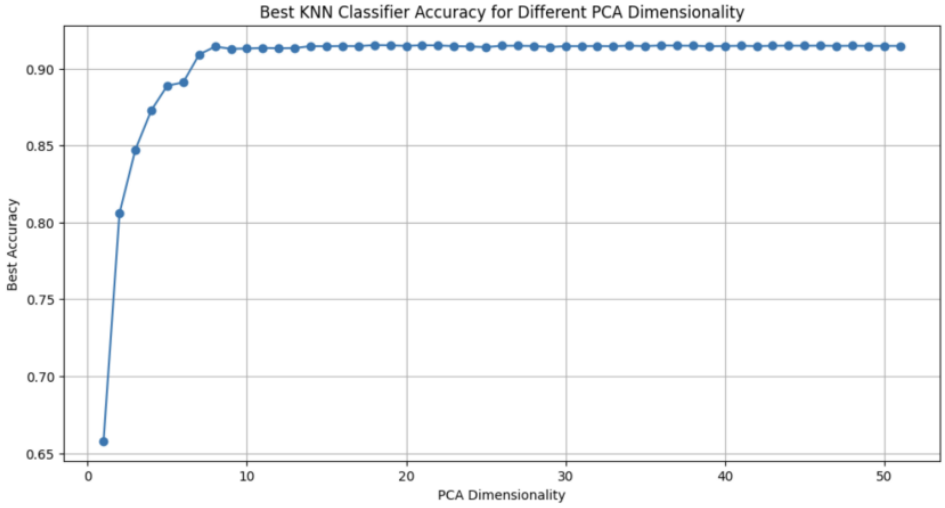

Figure 28. Best KNN classifier accuracy for different PCA dimensionality

This graph shows the best accuracy of a K-nearest neighbor (KNN) classifier in different principal component analysis (PCA) dimensions.

The best dimension of PCA is 22. The best k of the best dimension 22 is 5. The best accuracy is 91.60%.

Because the accuracy of dimensionality reduction with LDA is higher, we decided to use LDA.

(2) Select the Best Model Parameters

In order to find the best KNN parameters, I planned to traverse all the parameters, but found that there was not enough memory. Decide to use a distributed grid search to determine the best parameters for each in turn. Iterate through each parameter in turn, selecting the parameter with the highest accuracy.

n(1)_neighbors: The core parameter of KNN represents the value of "k", that is, how many neighbors are considered to make decisions.

W (2) eights: It is used to determine the neighbor weight.

The best weights option is: distance with an accuracy of 91.39%.

'uniform': In this case, each neighbor has the same voting rights. This means that the algorithm simply decides which category to belong to based on the number of neighbors, regardless of their distance to the query point [17].

'distance': In this setting, the weight of neighbors is inversely proportional to their distance to the query point [17]. This means that the closer neighbor has more decision-making power than the farther neighbor.

m(3)etric:The metric parameter is used in the KNN algorithm to determine the distance or similarity between data points. xi and yi are the coordinates of two points in the i - th dimension [18].

The best metric option is: manhattan with an accuracy of 91.77%.

'euclidean': Euclidean distance. This is the "straight line" distance between two points on a plane

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

'manhattan': Manhattan distance, also known as L1 distance or city block distance. It is the distance measured along the axis (not diagonally).

$$\sum_{i=1}^{n}|x_i - y_i|$$

'minkowski': Minkowski distance, when p=2, this is the Euclidean distance; When p=1, it's the Manhattan distance; As p→∞, it approaches the Chebyshev distance [19].

$$\left(\sum_{i=1}^{n}|x_i - y_i|^p\right)^{\frac{1}{p}}$$

(3) The Best Accuracy under the Best LDA

The best dimension of LDA is 6. The best k of the best dimension 6 is 12. The best accuracy is 92.58%.

2.5.4. Evaluation

(1) Confusion Matrix

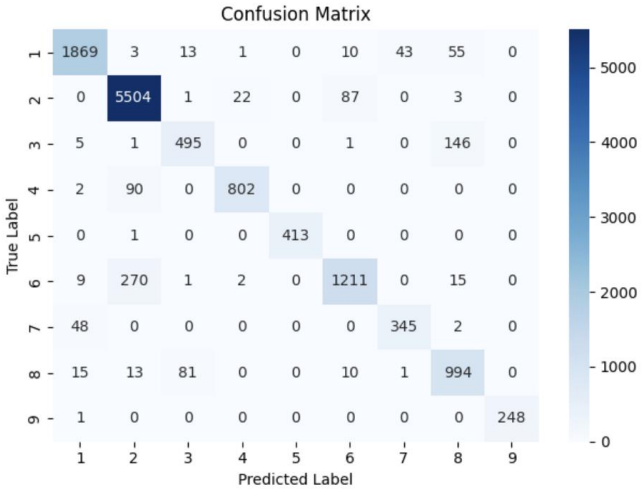

Figure 29. Confusion matrix

From this confusion matrix it can be observed that:

Category 2 (with a value of 5504) was predicted very well, and most of the samples were classified correctly.

Category 6 had a large misclassification, with 1211 samples misclassified into category 5.

For category 7, 345 samples were misclassified as Category 6.

Some other categories, such as categories 1, 3, 4, 5, 8, and 9, also have relatively good classification results, but there are still some misclassifications.

(2) Classification Report

Table 3. Classification report from precision, recall, F1-Score, and support

	Precision	Recall	F1-Score	Support
1	0.96	0.94	0.95	1994
2	0.93	0.98	0.96	5617
3	0.84	0.76	0.80	648
4	0.97	0.89	0.93	894
5	1.00	1.00	1.00	414
6	0.92	0.80	0.86	1508
7	0.89	0.87	0.88	395
8	0.82	0.89	0.85	1114
9	1.00	1.00	1.00	249
Accuracy			0.93	12833
Macro Avg	0.93	0.90	0.91	12833
Weighted Avg	0.93	0.93	0.92	12833

This is a category report that provides details on accuracy, recall, F1 scores and support for each category.

Precision:

For a particular category, accuracy is the ratio of the samples that are correctly predicted for that category to all the samples that are predicted for that category.

Recall:

For a particular category, recall is the ratio of the samples that were correctly predicted for that category to all the samples that actually belong to that category.

FF1 Score:

The F1 score is a harmonic average of accuracy and recall. It tries to provide a balanced measurement of both.

Support: For each category, support is the number of samples in the data that actually belong to that category. For example, category 1 has 1994 samples.

accuracy: This is the overall correct classification rate. Here, the model has an accuracy of 0.93, meaning that 93% of the sample was correctly classified.

macro avg: This is a simple average of accuracy, recall, and F1 scores for each category.

Weighted Avg:

This is a weighted average of class accuracy, recall, and F1 scores based on support for each category.

From the classification report:

Category 5 and Category 9 performed particularly well, as they achieved 1.00 on all three metrics (accuracy, recall and F1 score).

Category 3 had a lower recall rate of 0.76, which could mean that some samples in that category were misclassified into other categories.

Overall, the models performed relatively well, as most of the accuracy, recall and F1 scores were above 0.8.

2.6. Model 3: Support Vector Machines

2.6.1. Model Description

(1) SVM Theory principles and implementation process on the PaviaU dataset

SVM Intuition:

SVM in Hyperspectral Imaging: In the context of PaviaU (or any hyperspectral image), each pixel has multiple bands, providing a high-dimensional feature vector for each pixel. SVM attempts to find a hyperplane that separates pixels of one class from others in this high-dimensional space [20].

Support Vectors: The pixels that lie closest to the decision boundary and influence its orientation and position are termed support vectors. The hyperplane is driven by these pixels [21].

Formulization:

Decision Boundary: The hyperplane is mathematically represented as:

$$\omega \cdot x + b = 0$$

Where w is the weight vector and b is the bias.

Objective:

For a PaviaU image, considering its high dimensionality, the aim is to maximize the margin between different classes. If yi represents the label of the i th pixel and xi. Its feature vector:

$$y_i(\omega \cdot x_i + b) \geq 1$$

for every pixel i.

Lagrange Duality:

Lagrangian Formulation: The above can be turned into an optimization problem using Lagrange multipliers. The Lagrangian is:1

$$L(\omega, b, \alpha) = \frac{1}{2}||\omega||^2 - \sum_{i=1}^{N} \alpha_i [y_i(\omega \cdot x_i + b) - 1]$$

Where αi are the Lagrange multipliers and N is the number of pixels in the PaviaU image.

Dual Problem: By converting the primal problem to its dual, we have:

$$max_\alpha \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

Ensuring:

$$\alpha_i \geq 0 and \sum_{i=1}^{N} \alpha_i y_i = 0$$

Kernel Trick:

Higher Dimensionality in PaviaU: Some classes in PaviaU may not be linearly separable in the original band space. The kernel trick helps project these pixels into a space where they can be linearly separated [22].

Kernel Function: This function computes the dot product in the transformed space without actually doing the transformation. For pixel xi and xj:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$$

Some kernels include:

Linear:

$$K(x_i, x_j) = x_i \cdot x_j$$

Polynomial:

$$K(x_i, x_j) = (1 + x_i \cdot x_j)^d$$

RBF:

$$K(x_i, x_j) = e^{-\gamma||x_i - x_j||^2}$$

Soft Margin:

Handling Noise in PaviaU: Given the noise and inherent variability in hyperspectral images, it's hard to achieve perfect separation. Soft margin SVM introduces slack to tolerate some misclassifications [23].

The objective with Slack: By introducing Slack variables ξi:

$$\frac{1}{2}||\omega||^2 + C\sum_{i=1}^{N}\xi_i$$

Subject to:

$$y_i(\omega \cdot x_i + b) \geq 1 - \xi_i and \xi_i \geq 0$$

Where C strikes a balance between margin maximization and misclassification minimization.

(2) General Functionality and Applications of SVM

Multi-Class Classification: SVM is primarily used for multi-class classification in your PaviaU hyperspectral remote sensing image classification task. It aims to find the best hyperplane to effectively distinguish between multiple different land cover or feature classes in the image, allowing each pixel to be assigned to the corresponding class.

Regularization: SVM introduces the concept of regularization, which is crucial for controlling model complexity and preventing overfitting [24]. This regularization ensures that the model generalizes well to unseen data by balancing the trade-off between maximizing the margin and minimizing classification errors.

Soft Margin Classification: SVM introduces the concept of soft margin classification, allowing for a certain tolerance range for misclassifying some data points. This flexibility is particularly valuable when dealing with noisy or non-linearly separable data, enhancing the robustness of classification. This can be especially useful in your PaviaU image classification task, as remote sensing data often contains noise or complex non-linear features, and soft margin classification can improve classification robustness.

2.6.2. Parameters Estimation

Common Hyperparameters in SVM: Kernel Function, C (Regularization Parameter), γ (RBF Kernel Bandwidth Parameter), Degree (Degree of Polynomial Kernel), Coef0 (Sigmoid Kernel Parameter).

When using SVM for classification, we use different kernel functions matching different parameters to find the best performance to make the model perform. Below is the meaning of each kernel function and its corresponding parameters:

(1) Linear Kernel (linear)

Purpose: The linear kernel is the simplest kernel function. It assumes the data is linearly separable, meaning there exists a hyperplane that can separate the data.

Parameters:

C: Penalty parameter. It controls the penalty for misclassified points. A larger C means the classifier is more eager to ensure each data point is correctly classified, but may lead to overfitting; a smaller C allows the classifier to be more tolerant of misclassifications, potentially resulting in a larger decision boundary.

(2) Polynomial Kernel (poly)

Purpose: The polynomial kernel can capture the nonlinear patterns in the data. It allows for improved classification performance when the data isn't linearly separable by introducing polynomial terms.

Parameters:

C: Same as linear kernel

degree: Determines the order of the polynomial. A higher degree can result in more complex decision boundaries.

gamma: Controls the influence of each training sample.

coef0: A constant term in the kernel function. It can influence the shape of the decision boundary.

(3) RBF Kernel (rbf)

Purpose: The RBF kernel is a highly flexible kernel function capable of capturing complex nonlinear decision boundaries. It's based on the Euclidean distance between data points.

Parameters:

C: Same as linear kernel.

gamma: Controls the shape of the RBF kernel. A smaller gamma results in a wider decision boundary, while a larger gamma results in a narrower boundary.

(4) Sigmoid Kernel (sigmoid)

Purpose: The sigmoid kernel function is a common activation function in neural networks. In SVM, it can introduce nonlinear decision boundaries.

Parameters:

C: Same as linear kernel.

gamma: Same as RBF kernel.

coef0: Controls the threshold of the sigmoid function.

Hyperparameter Combination: Kernel, gamma, and C are crucial in SVM. Turning these enhances performance [25]. Besides, adjusting the other two would result in a lack of computing resources. Therefore, we choose to consider only the three most important hyperparameters, the kernel function, gamma, and C.

2.6.3. Environment and Hardware

We briefly describe our computing resource for the SVM model.

Environment: Python 3.8.17, anaconda.

Hardware: 13th Gen Intel(R) Core(TM) i9-13900HX 2.20 GHz

Library: Using Sklearnex to accelerate CPU

2.6.4. Parameters and Training Procedure

A. Hyperparameter value

(1) C (Slack Parameter)

We select the values [0.001, 0.01, 0.1, 1, 10, 100] for the C parameter for the following reasons:

1) Logarithmic Scale: Many hyperparameters have optimal values spanning a wide logarithmic range. A logarithmic scale ensures comprehensive search and efficiency.

2) Regularization Balance: Small C values (e.g., 0.001) offer strong regularization, useful against overfitting. Large C values (e.g., 10) provide weak regularization, suitable for intricate data.

3) Common Practice: These values are common starting points in literature and practice for initial hyperarameter searches [26].

(2) Gamma

We select the values [0.01, 0.1, 1, 10, 'auto', 'scale'] for the gamma parameter for the following reasons:

1) Varying Scale: The chosen gamma values cover a wide range, from small to large, testing the model's complexity. Small gamma means more flexibility, while large gamma results in a tighter fit to the data.

2) Computational Efficiency: These selections balance thoroughness and computational efficiency. Trying all possible values can be computationally expensive, especially with large datasets.

3) 'auto' and 'scale' Options:

'auto' uses 1/n_features as gamma, providing a heuristic for datasets with many features.

'scale' considers dataset variance, often leading to better generalization by default [6].

B. Parameters Cross-validation

After cross-validation with different kernel functions for different parameters, we find that the model performs best with an accuracy of 0.95 when the C parameter of 10 and gamma parameter of 0.1 are chosen for the rbf kernel function without using downscaling methods.

The accuracy is 95.14%

C. Selection of Downscaling Methods

In order to achieve the best accuracy, we separately test the accuracy under PCA and LDA. We kept the hyperparameters of the SVM fixed with the previous best parameters combination.

For the PCA model, we iterate through all possible values of the retained principal components and select the one that yields the highest accuracy through a for loop.

The best PCA components number is 9 and the best accuracy is 94.52%.

For the LDA model, the maximum possible dimension is one less than the number of classes. We iterate through all dimensions and find the dimension that results in the highest accuracy.

The best LDA dimension is 8 and the best accuracy is 92.82%.

Although the use of dimensionality reduction methods can significantly reduce the training duration, we found that the accuracy rate decreases after using dimensionality reduction methods. This could be due to the following reasons:

1) Loss of Important Variance (PCA): PCA aims to retain the dimensions (or features) that explain the most variance in the dataset. However, for classification, not all high-variance features are necessarily the most discriminative ones. By applying PCA, we might be discarding some features that are crucial for the SVM to draw optimal decision boundaries, even if they account for less variance.

2) Class Discrimination vs. Class Separation (LDA): While LDA attempts to maximize the distance between class means and minimize the spread (variance) within each class, it might not always be the best approach for complex datasets like remote sensing data. PaviaU, being a hyperspectral dataset, might have subtle spectral signatures that LDA fails to capture, leading to sub-optimal projections [27].

Finally, without using dimensionality reduction methods, we can print the confusion matrix and classification report:

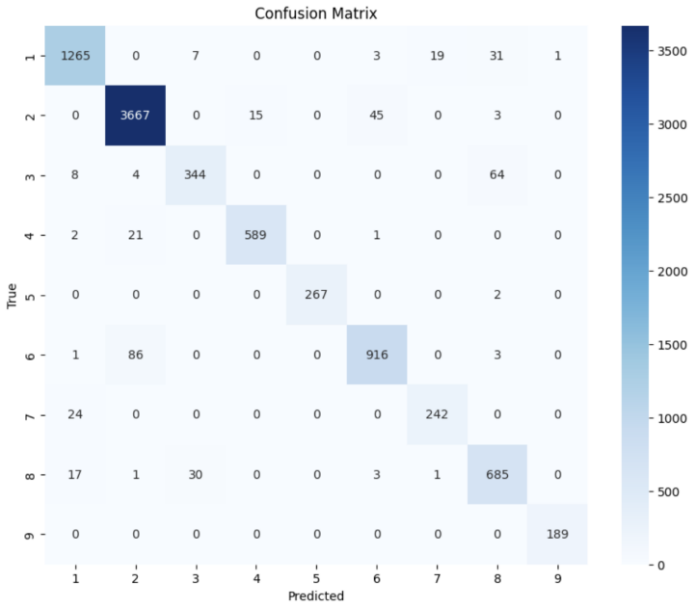

Figure 30. Confusion matrix

We can see that:

Class 2 and Class 9 are well-predicted with minor errors.

Class 3 has a notable misclassification with Class 7.

Class 4 has confusion primarily with Class 7.

Classes 5 and 6 are predicted with high precision.

Class 7 is largely correct but gets confused with Class 3.

Class 8 has minor confusion with Classes 2, 4, and 9.

Overall, the model is effective, but improvements can be made in distinguishing between Classes 2 and 6.

Table 4. Classification report

	precision	recall	f1-score	support
1	0.96	0.95	0.95	1326
2	0.97	0.99	0.98	3730
3	0.90	0.81	0.85	420
4	0.98	0.97	0.98	613
5	1.00	0.99	1.00	269
6	0.95	0.91	0.93	1006
7	0.91	0.90	0.90	266
8	0.87	0.93	0.90	737
9	0.99	1.00	1.00	189
accuracy			0.95	8556
macro avg	0.95	0.94	0.95	8556
weighted avg	0.95	0.95	0.95	8556

3. DISCUSSION

3.1. Bayes Model

Advantages:

(1) Simplicity

The theory of bayes is simpler than the other classifier. Besides, compared with other classifiers, bayes classifiers do not have complex hyperparameters tuning process. Due to the conditional

independence assumption between features, the model does not need to consider complex relationships between features. Therefore, it only needs to estimate the probability distribution of a single feature, which greatly simplifies the complexity of the model and avoid too much hyperparameters.

(2) Efficiency:

In order to achieve similar performance, Bayes classifiers consumes less time and computing resource. Firstly, bayes classifiers' core mechanism relies on the independence assumption between features, which streamlines the computational process and accelerates predictions. Secondly, bayes classifiers avoid complex linear algebra calculation. When applying Bayes' theorem, we are mainly concerned with probability and conditional probability, which usually involves calculating basic statistics such as mean, variance, probability, etc., but does not involve operations such as matrix operations.

Drawbacks:

(1) Assumptive Independence:

Bayesian classifiers assumes that data features are mutually independent.This requirement is strict and it doesn't always hold in real-world scenario. It means we need to do more process before we utilize Bayes Theory.

(2) Rely on prior probability

Even though bayes classifiers do not have complex mathematical theory derivation liked SVM, it strongly relies on the value of prior probability. In some cases, the choice of prior probability could significantly influence the classification performance.

(3) Over-Simplification:

Since it Relying heavily on the feature independence assumption, it might overlook nuanced relationships involving feature interactions. Thus, its intrinsic simplicity might make it ill-suited for capturing intricate data patterns.

3.2. KNN Model

Advantages:

(1) Simplicity:

The principle of the KNN algorithm is very simple, it has no training process, and only calculates when predicting. The basic principle of KNN algorithm is to predict the classification or output of a sample based on the classification or output of k nearest neighbors of the sample [15]. Since it does not involve a training process, the implementation of the algorithm is very intuitive. When there is a new data point that needs to be classified or predicted, simply calculate its distance from a known sample and then vote or average according to the nearest k samples.

(2) Data independent distribution:

KNN is instance-based learning, so it does not assume that data follows a particular distribution. Many traditional statistical methods require the assumption that the data follows a specific distribution, such as a normal distribution. If these assumptions are not true, the performance of the model may suffer. However, KNN does not require any assumptions about the distribution of the data. It is entirely based on the structure of the data itself, which makes KNN perform better on data that does not follow a particular distribution.

Drawbacks:

(1) High computational complexity:

For each prediction sample, the distance to all training samples needs to be calculated [28]. KNN works by calculating the distance between the new data point and all the training data points at the time of prediction, thus determining the nearest k neighbors. This can make predictions very slow, especially if the data set is large.

(2) High space complexity:

KNN needs to store all the training data, which can lead to insufficient memory when the data volume is large.

(3) Sensitivity:

KNN is sensitive to outliers. An outlier can cause a large deviation in the forecast. KNN relies entirely on training data to make predictions, so it is very sensitive to outliers or noisy data. This is because an outlier can skew the results of a prediction during voting, especially if the value of k is relatively small[28].

(4) Dimensionality disaster:

As features increase, the distance between samples tends to be consistent, which makes KNN less effective on high-dimensional data [29]. Mathematically, as dimension d increases, the difference in the distance between any two points decreases, approaching a constant.

3.3. SVM Model

Advantages:

(1) High-Dimensional Data Handling: SVM excels at handling high-dimensional datasets, such as the PaviaU dataset, which contains multiple spectral bands.

(2) Strong Performance with Small Samples: SVM's performance is not heavily reliant on the amount of data but on support vectors, making it suitable for scenarios with limited samples.

(3) Regularization Parameter: SVM provides a regularization parameter that helps prevent overfitting.

(4) Kernel Trick: SVM can employ kernel functions to address non-linear problems, which is valuable in remote sensing imagery where class boundaries may be complex and non-linear.

(5) Robustness: SVM is robust to noise and outliers commonly present in remote sensing data.

Drawbacks:

(1) Computational Complexity: SVM can be computationally intensive, especially for large-scale datasets, particularly when selecting appropriate kernel functions and parameters.

(2) Limited Interpretability: SVM models can be challenging to interpret, which can be a drawback in remote sensing applications where understanding model decisions is important.

(3) Sensitivity to Parameter Choices: SVM's performance is highly dependent on parameter choices, including the regularization parameter and kernel selection.

(4) Imbalanced Data: When dealing with imbalanced datasets, where certain classes have significantly more samples than others, SVM may not perform well without proper handling, such as sample weighting or resampling.

Table 5. Compare with KNN, Naive Bayes, and SVM

Parameter	KNN	Naive Bayes	SVM
Dataset Effectiveness	Limited for small datasets	Effective for large datasets	Effective for large datasets
Speed	Slower with large datasets	Faster	Faster
Handling Noisy Data	Sensitive to noisy data	Robust to noisy data	Sensitive to noisy data
Accuracy	High accuracy	Requires a large dataset for high accuracy	High accuracy
Interpretability	Limited interpretability	High interpretability	Moderate interpretability
Imbalanced Data Handling	Sensitive to imbalance	Handles imbalanced data	Sensitive to imbalance

4. CONCLUSION

The report successfully demonstrates the application of feature selection and dimensionality reduction techniques, specifically Linear Discriminant Analysis (LDA) and Principal Component Analysis (PCA), in the analysis of the PaviaU dataset. These techniques not only improve the efficiency of land cover classification but also offer valuable insights into remote sensing image analysis. The study concludes that optimizing these methods can significantly enhance data visualization and classification accuracy. Future work should focus on exploring other dimensionality reduction techniques and datasets to further validate these findings.

ACKNOWLEDGEMENTS

Four authors contributed equally to this paper.

REFERENCES

[1] "Hyperspectral Remote Sensing Scenes - Grupo de Inteligencia Computacional (GIC)," www.ehu.eus. https://www.ehu.eus/ccwintco/index.php/Hyperspectral_Remote_Sensing_Scenes#Pavia_Centre_and_University (accessed Oct. 26, 2023).

[2] P. Liu, K.-K. R. Choo, L. Wang, and F. Huang, "SVM or deep learning? A comparative study on remote sensing image classification," Soft Computing, vol. 21, no. 23, pp. 7053–7065, Jul. 2016, doi: https://doi.org/10.1007/s00500-016-2247-2.

[3] Md. A. M. Hasan, M. Nasser, S. Ahmad, and K. I. Molla, "Feature Selection for Intrusion Detection Using Random Forest," Journal of Information Security, vol. 07, no. 03, pp. 129–140, 2016, doi: https://doi.org/10.4236/jis.2016.73009.

[4] E. Scornet, "Trees, forests, and impurity-based variable importance in regression," Annales de l'Institut Henri Poincaré, Probabilités et Statistiques, vol. 59, no. 1, Feb. 2023, doi: https://doi.org/10.1214/21-aihp1240.

[5] M. Chaibi, E. M. Benghoulam, L. Tarik, M. Berrada, and A. El Hmaidi, "Machine Learning Models Based on Random Forest Feature Selection and Bayesian Optimization for Predicting Daily Global Solar Radiation," International Journal of Renewable Energy Development, vol. 11, no. 1, pp. 309–323, Nov. 2021, doi: https://doi.org/10.14710/ijred.2022.41451.

[6] J. Ye, Ravi Janardan, and Q. Li, "Two-Dimensional Linear Discriminant Analysis," Neural Information Processing Systems, vol. 17, pp. 1569–1576, Dec. 2004.

[7] G. T. Reddy et al., "Analysis of Dimensionality Reduction Techniques on Big Data," IEEE Access, vol. 8, pp. 54776–54788, 2020, doi: https://doi.org/10.1109/ACCESS.2020.2980942.

[8] W.-S. Zheng, J. H. Lai, P. C. Yuen, and S. Z. Li, "Perturbation LDA: Learning the difference between the class empirical mean and its expectation," Pattern Recognition, vol. 42, no. 5, pp. 764–779, May 2009, doi: https://doi.org/10.1016/j.patcog.2008.09.012.

[9] E. K. Tang, P. N. Suganthan, X. Yao, and A. K. Qin, “Linear dimensionality reduction using relevance weighted LDA,” Pattern Recognition, vol. 38, no. 4, pp. 485–493, Apr. 2005, doi: https://doi.org/10.1016/j.patcog.2004.09.005.

[10] A. Daffertshofer, C. J. C. Lamoth, O. G. Meijer, and P. J. Beek, “PCA in studying coordination and variability: a tutorial,” Clinical Biomechanics, vol. 19, no. 4, pp. 415–428, May 2004, doi: https://doi.org/10.1016/j.clinbiomech.2004.01.005.

[11] R. Bellman, “DYNAMIC PROGRAMMING AND LAGRANGE MULTIPLIERS,” Proceedings of the National Academy of Sciences of the United States of America, vol. 42, no. 10, pp. 767–769, Oct. 1956, doi: https://doi.org/10.1073/pnas.42.10.767.

[12] I. T. Jolliffe, “Principal Component Analysis,” Technometrics, vol. 30, no. 3, pp. 351–351, Aug. 1988, doi: https://doi.org/10.2307/1270093.

[13] D. W. Aha, D. Kibler, and M. K. Albert, “Instance-based learning algorithms,” Machine Learning, vol. 6, no. 1, pp. 37–66, Jan. 1991, doi: https://doi.org/10.1007/bf00153759.

[14] L. L. Cam, “Maximum Likelihood: An Introduction,” International Statistical Review / Revue Internationale de Statistique, vol. 58, no. 2, p. 153, Aug. 1990, doi: https://doi.org/10.2307/1403464.

[15] I. Rish, “An empirical study of the naive Bayes classifier,” Jan. 2001.

[16] R. O. Duda, D. G. Stork, and P. E. Hart, Pattern classification and scene analysis. Part 1, Pattern classification. New York ; Chichester: Wiley, 2000. Available: https://dl.acm.org/citation.cfm?id=954544

[17] S. A. Dudani, “A Note on Distance-Weighted k-Nearest Neighbor Rules,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 8, no. 4, pp. 311–313, 1978, doi: https://doi.org/10.1109/tsmc.1978.4309958.

[18] T. Cover and P. Hart, “Nearest neighbor pattern classification,” IEEE Transactions on Information Theory, vol. 13, no. 1, pp. 21–27, Jan. 1967, doi: https://doi.org/10.1109/tit.1967.1053964.

[19] Michel Marie Deza, E. Deza, and Springerlink (Online Service, Encyclopedia of Distances. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[20] J. Anthony Gualtieri and S. R. Chettri, “Support vector machines for classification of hyperspectral data,” Nov. 2002, doi: https://doi.org/10.1109/igarss.2000.861712.

[21] G. Camps-Valls and L. Bruzzone, Kernel Methods for Remote Sensing Data Analysis. John Wiley & Sons, 2009. Accessed: Oct. 26, 2023. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=_KhUMXQQkmQC&oi=fnd&pg=PA51&dq=J.+A.+Gualtieri

[22] S. Wan, C. Gong, P. Zhong, S. Pan, G. Li, and J. Yang, “Hyperspectral Image Classification With Context-Aware Dynamic Graph Convolutional Network,” IEEE Transactions on Geoscience and Remote Sensing, vol. 59, no. 1, pp. 597–612, Jan. 2021, doi: https://doi.org/10.1109/tgrs.2020.2994205.

[23] F. Kam, 2009. Accessed: Oct. 26, 2023. [Online]. Available: https://files.core.ac.uk/pdf/23/140536.pdf

[24] H. Xu et al., “Robustness and Regularization of Support Vector Machines,” Journal of Machine Learning Research, vol. 10, pp. 1485–1510, 2009, Available: https://www.jmlr.org/papers/volume10/xu09b/xu09b.pdf

[25] A. Tharwat, “Parameter investigation of support vector machine classifier with kernel functions,” Knowledge and Information Systems, vol. 61, no. 3, pp. 1269–1302, Feb. 2019, doi: https://doi.org/10.1007/s10115-019-01335-4.

[26] F. E. H. Tay and L. Cao, “Application of support vector machines in financial time series forecasting,” Omega, vol. 29, no. 4, pp. 309–317, Aug. 2001, doi: https://doi.org/10.1016/s0305-0483(01)00026-3.

[27] L. J. Cao, K. S. Chua, W. K. Chong, H. P. Lee, and Q. M. Gu, “A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine,” Neurocomputing, vol. 55, no. 1–2, pp. 321–336, Sep. 2003, doi: https://doi.org/10.1016/s0925-2312(03)00433-8.

[28] D. B. V, “Nearest neighbor (NN) norms: NN pattern classification techniques,” IEEE Computer Society Tutorial, 1991, Available: https://cir.nii.ac.jp/crid/1572261550010307072

[29] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, “When Is ‘Nearest Neighbor’ Meaningful?” Lecture Notes in Computer Science, pp. 217–235, 1999, doi: https://doi.org/10.1007/3-540-49257-7_15.