Xi'an Jiaotong-Liverpool University
西交利物浦大学

XJTLU Entrepreneur College (Taicang) Cover Sheet

Module code and Title	**DTS201TC Pattern Recognition**
School Title	**School of AI and Advanced Computing**
Assignment Title	**Coursework (Groupwork)**
Submission Deadline	**23:59, 29th Oct.**
Final Word Count	**NAN**
If you agree to let the university use your work anonymously for teaching and learning purposes, please type **"yes"** here.	**yes**

I certify that I have read and understood the University's Policy for dealing with Plagiarism, Collusion and the Fabrication of Data (available on Learning Mall Online). With reference to this policy I certify that:

- My work does not contain any instances of plagiarism and/or collusion.
- My work does not contain any fabricated data.

By uploading my assignment onto Learning Mall Online, I formally declare that all of the above information is true to the best of my knowledge and belief.

Scoring – For Tutor Use	
Student ID	**2141480, 2142116, 2142457, 2141434**

Stage of Marking	**Marker Code**	**Learning Outcomes Achieved (F/P/M/D) (please modify as appropriate)**				**Final Score**
		A	**B**	**C**	**D**	
1st Marker – red pen						
Moderation – green pen	**IM Initials**	The original mark has been accepted by the moderator (please circle as appropriate):				Y / N
		Data entry and score calculation have been checked by another tutor (please circle):				Y
2nd Marker if needed – green pen						

For Academic Office Use			**Possible Academic Infringement (please tick as appropriate)**	
Date Received	**Days late**	**Late Penalty**	☐ **Category A**	Total Academic Infringement Penalty (A,B, C, D, E, Please modify where necessary) _____
			☐ **Category B**	
			☐ **Category C**	
			☐ **Category D**	
			☐ **Category E**	

The assignment must be submitted via Learning Mall Online to the correct drop box. Only electronic submission is accepted and no hard copy submission. All students must download their file and check that it is viewable after submission. Documents may become corrupted during the uploading process (e.g. due to slow internet connections). However, students themselves are responsible for submitting a functional and correct file for assessments.

DTS201TC Coursework Report

A Comparative Analysis of Pattern Recognition Models for Remote Sensing Data Classification

Group 12

Contents

Abstract

This study conducts a comparative analysis of multiple Pattern Recognition (PR) algorithms on a remote sensing dataset, including K-Nearest Neighbors (KNN), Random Forest, and a novel model that combines K-Means and Support Vector Machine (SVM).

In this study, we first provide a comprehensive dataset description and visualization analysis to ensure data integrity and usability. We employ feature selection methods to precisely choose features for model training and offer detailed explanations.

Next, we provide in-depth descriptions of the implemented PR models, including KNN, Random Forest, and the new model that combines K-Means and SVM. We explain the theoretical foundations and functionalities of these models, including descriptions of parameter estimation and training procedures. We also introduce the hardware environment used for experiments, ensuring code correctness and alignment with reported results.

We evaluate the performance of these models, presenting experimental results through graphs and data. We discuss the strengths and weaknesses of each model and the rationale behind selecting them. Particularly, the performance of the new model is subject to detailed analysis.

1 Investigating the dataset

1.1 Dataset description

1.1.1 Dataset Loading

In this study, we utilized the PaviaU dataset and the PaviaU_gt (ground truth) dataset as the foundation for our research. These datasets contain crucial information about remote sensing images, providing the necessary data for our pattern recognition algorithm studies. We began by using the SciPy library in Python to load this data and conducted an analysis of the following steps.

1.1.2 PaviaU Dataset

The PaviaU dataset contains rich information from remote sensing images, including pixel values from different spectral bands. This dataset is in a 3D format, encompassing multiple channels and pixel values. By loading the `PaviaU.mat` file, we successfully imported the dataset into our research environment. It includes information about various land cover types such as vegetation, buildings, and water bodies, which are essential for remote sensing image classification. The content and structure of the dataset form the foundation for subsequent analysis and model development.

1.1.3 PaviaU_gt Dataset

Corresponding to the PaviaU dataset is the `PaviaU_gt` dataset, which contains the ground truth map of the PaviaU dataset. This dataset will be used for assessing the performance of our established pattern recognition algorithms. We also employed the SciPy library to load the `PaviaU_gt.mat` file, successfully obtaining this critical dataset. Ground truth maps are vital for verifying and comparing the classification accuracy of our models.

```
PaviaU Data Dimensions: (610, 340, 103)
PaviaU Data Type: uint16
PaviaU Ground Truth Dimensions: (610, 340)
PaviaU Ground Truth Data Type: uint8
Missing values in PaviaU Data: 0
Missing values in PaviaU Ground Truth: 0
Unique labels in PaviaU Ground Truth: [0 1 2 3 4 5 6 7 8 9]
Counts for each label: [164624   6631  18649   2099   3064   1345   5029   1330   3682    947]
```

Overview of PaviaU Dataset and Ground Truth Statistics

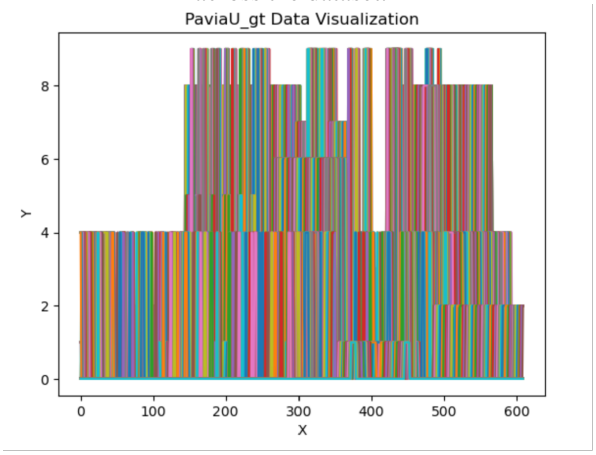
1.2 Visualization

1.2.1 Data Visualization

In this section, we conduct a visual exploration of the PaviaU dataset and its associated ground truth labels. Visualization is an essential step in understanding the data and gaining insights into its characteristics.

1.2.2 PaviaU_gt Data Visualization

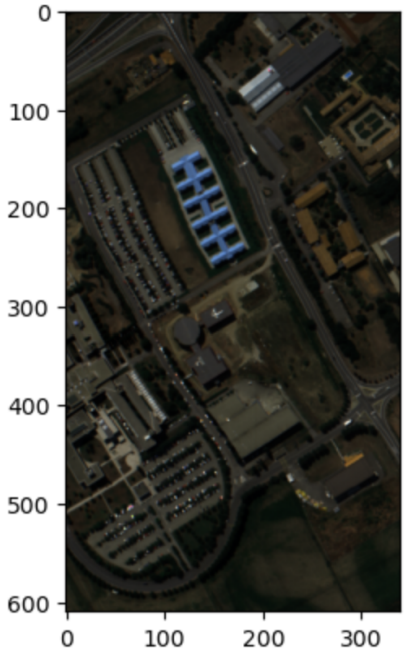
To start our analysis, we visualize the PaviaU_gt dataset, which serves as the ground truth map for the PaviaU dataset. This map contains class labels corresponding to different land cover types. The following plot provides an overview of the data's structure, showcasing the distribution of classes across the dataset.



PaviaU_gt Data Visualization

1.2.3 Spectral Band Visualization

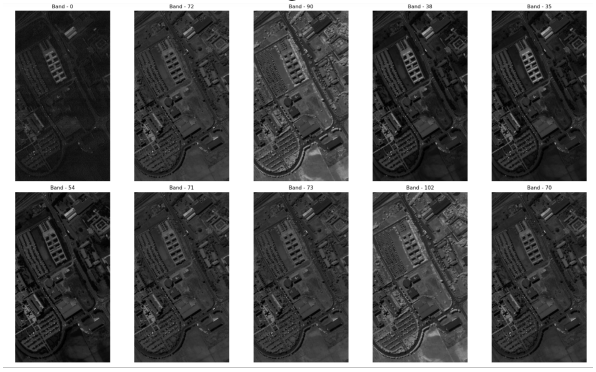
Next, we explore the spectral bands of the PaviaU dataset. We display a subset of spectral bands to gain insights into the data's characteristics. Spectral bands are a critical component of remote sensing data, and visualizing them can help identify unique features in the dataset.



Visualization of Spectral Bands in the PaviaU Dataset

1.2.4 Random Band Sample

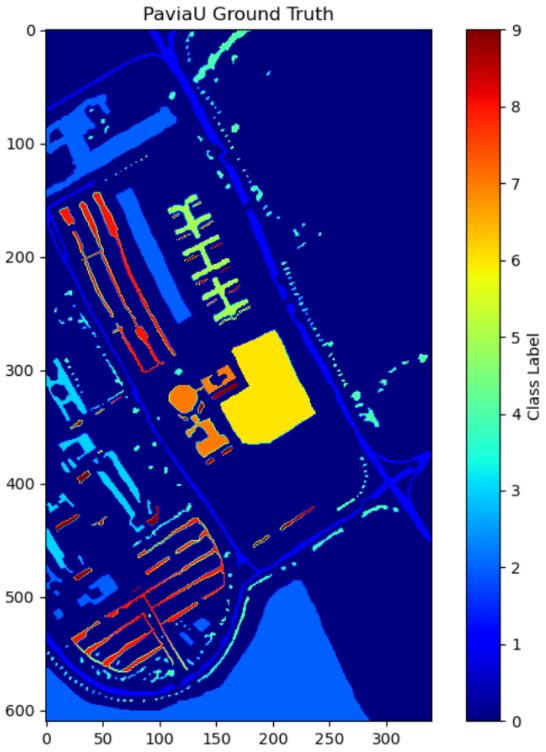
To delve deeper into the data, we randomly select a single spectral band for visualization. This allows us to examine the pixel values within that band and gain a better understanding of the data's details.



Random Sample from the PaviaU Data

1.2.5 Ground Truth Visualization

Finally, we visualize the ground truth labels associated with the PaviaU dataset. This image displays class labels using a color-coded representation. Visualizing the ground truth is essential for understanding the labeled data, which is vital for evaluating classification models.



PaviaU Ground Truth Visualization

The visual exploration of the PaviaU dataset and its associated components provides valuable insights into the data's characteristics and sets the stage for our pattern recognition and classification analyses.

1.2.6 Proper References

mat-file-parsing[1].
Read Matlab mat Files in Python[2].
mat file conversion and reading using Python[3].
Hyperspectral classification[4].

1.3 Possibility of Using Feature Selection Methods

In our analysis, we work with the PaviaU dataset, where X represents the features or input variables. These features describe the characteristics of each data point in our dataset. In our case, each data point is an image pixel, and the features can include various spectral bands or other relevant attributes.
For example, the features in our dataset may include:

- Spectral Reflectance Values: These values represent the reflectance of different spectral bands, capturing information about the surface materials in the image.
- Spatial Coordinates: Depending on the dataset, you may also have spatial coordinates (X and Y)

representing the pixel's location within the image.

- Derived Features: We may have derived features, such as principal components obtained through dimensionality reduction techniques like PCA, as discussed earlier.

1.3.1 Explanation

The labels or targets (y) in our dataset represent the ground truth information associated with each data point. In the context of the PaviaU dataset, the labels typically represent the land cover or land use class of each pixel. These labels are crucial for supervised learning tasks such as classification or segmentation. For example, the labels may include:

- Land Cover Classes: These can include categories like vegetation, urban areas, water bodies, roads, and more, each represented by a numeric code.
- Ground Truth Images: In some cases, the ground truth labels may be provided as a separate image, where each pixel corresponds to a specific land cover class.
- One-Hot Encoding: For machine learning tasks, we might use one-hot encoding to represent these classes as binary vectors, where each class is represented by a unique combination of 0s and 1s.

Understanding the features (X) and labels/targets (y) is crucial for effectively analyzing and modeling our dataset. We can use this information to train machine learning models for tasks such as classification, segmentation, or regression, depending on our research goals.

1. **Filter Methods:**

Description: Filter methods assess the relevance of each feature to the target variable based on statistical properties. These statistical properties can include metrics such as Pearson correlation coefficient, chi-squared test, or mutual information.

Advantages: These methods are computationally simple, fast, and can quickly filter out irrelevant features before feature selection.

2. **Wrapper Methods:**

Description: Wrapper methods use the performance of a prediction model as the evaluation criterion for feature subset quality. They iteratively try different feature subsets to find the optimal combination.

Advantages: These methods consider relationships between features and are typically more accurate in the feature subset search.

3. **Embedded Methods:**

Description: Embedded methods perform feature selection during the model training process. They determine whether to retain a feature by assessing its importance.

Advantages: These methods directly consider feature importance within the model and typically have high accuracy in feature selection.

For example code related to each method, please refer to our code package.

Reasons for Slow Execution:

- **Large Dataset:** Training a Random Forest model on a large dataset with numerous features can be time-consuming.
- **Complex Model:** Random Forest is a complex ensemble model consisting of many decision trees, which leads to longer training times.
- **Feature Importance Calculation:** The process of computing feature importances for a large number of features contributes to extended training time.
- **Hyperparameters:** Model hyperparameters, such as the number of estimators and tree depth, can have a significant impact on training time.
- **Hardware Resources:** The available computational resources, including CPU and memory, directly affect the speed of training.

Handcrafted Feature Selection Techniques for Pattern Recognition[5].

1.4 Feature Analysis

1.4.1 Feature Extraction Methods

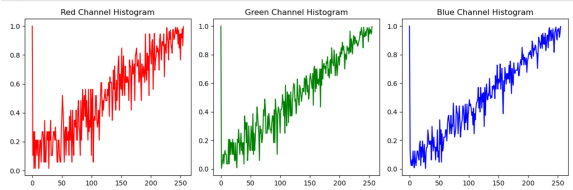
1.4.2 Color Histograms

Description: Color histograms extract color information by calculating the histogram for each color channel in the image. The histogram can show the frequency of each color in the image, forming a feature vector.

Advantages: Color histograms are a simple and effective method of color feature extraction that can reflect the color distribution of an image to a certain extent.

Disadvantages: Color histograms cannot reflect the spatial relationship between colors. For images with the same color distribution but different structures,

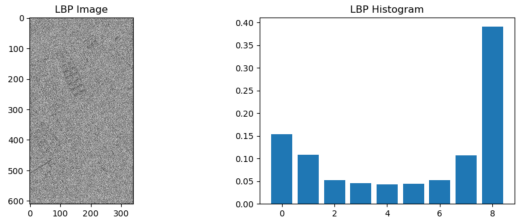
their color histograms may be the same. *color-histograms-pinecone*[6]. *color-feature-extraction-springer*[7].



Visualization of Color Histograms

1.4.3 Local Binary Patterns (LBP)

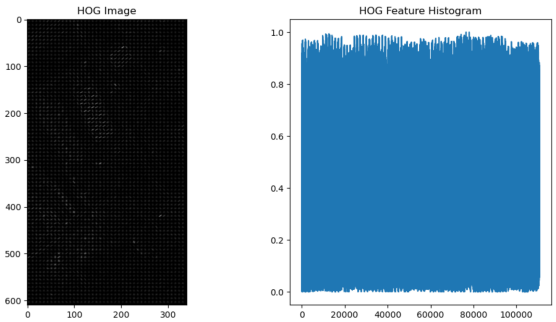
Description: LBP is a texture descriptor that encodes local structural information by comparing pixel intensity with its neighborhood.
Advantages: The LBP method has good robustness to changes in illumination and is simple to calculate and easy to implement.
Disadvantages: The LBP method is sensitive to noise and easily affected by it. *Local binary patterns*[8]. *lbp-springer*[9]. *lbp-mdpi*[10].



Visualization of Local Binary Patterns (LBP)

1.4.4 Histogram of Oriented Gradients (HOG)

Description: The HOG descriptor is used to capture shape information in an image. It does this by calculating and counting the gradient direction histogram of local areas in the image.
Advantages: The HOG method has a good description ability for local shape changes and has some robustness to changes in illumination.
Disadvantages: The HOG method requires a large amount of computational resources and is sensitive to scale changes. *Local binary patterns*[8]. *hog-wikipedia*[11]. *hog-learnopencv*[12]. *hog-scikit-image*[13].



Visualization of Histogram of Oriented Gradients (HOG)

1.4.5 Principal Component Analysis (PCA)

Description: PCA is a dimensionality reduction technique that can transform high-dimensional data into low-dimensional data while retaining as much of the original data's variability as possible.
Advantages: PCA can effectively reduce data dimensions and can remove noise and redundant information from the data.
Disadvantages: PCA assumes that the main variability of the data can be captured by orthogonal bases, which may not hold in some cases.

1. **Covariance Matrix:**

   The first step in PCA is to compute the covariance matrix for the data. The covariance between two variables, X and Y, is calculated using the formula:

$$\text{Cov}(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

   Here, $\text{Cov}(X,Y)$ represents the covariance between X and Y, X_i and Y_i are the values of X and Y for data point i, $\bar{X}$ and $\bar{Y}$ are the means of X and Y, and n is the number of data points.

2. **Eigenvalue Decomposition:**

   The next step in PCA is to perform eigenvalue decomposition on the covariance matrix. The formula for eigenvalue decomposition is as follows:

$$\text{Covariance Matrix} = V \cdot \Lambda \cdot V^T$$

   Here, Λ is a diagonal matrix containing the eigenvalues of the covariance matrix, and V is a matrix containing the eigenvectors of the covariance matrix.

3. **Selecting Principal Components:**

   PCA selects the principal components based on the magnitude of the eigenvalues. Typically, the top k eigenvectors with the largest eigenvalues are chosen to form a new feature space.

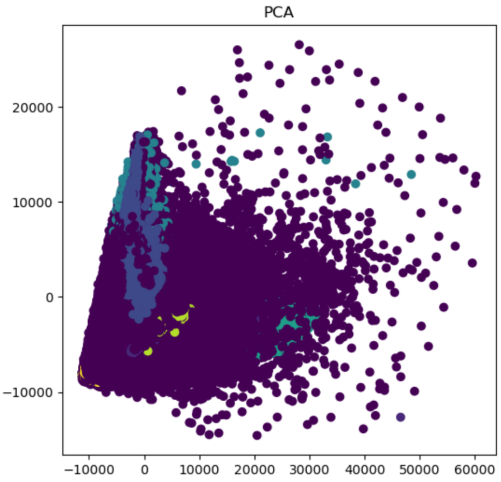
4. **Projecting Data:**

   Finally, data is projected into the new feature space. For a data point X, its projection in the new feature space can be calculated using the following formula:

$$\text{Projected Data} = X \cdot V_k$$

Here, V_k contains the selected top k eigenvectors.

pca-wikipedia[14] *pca-geeksforgeeks*[15] *pca-mathworks*[16] *pca-builtin*[17]



Visualization of Principal Component Analysis (PCA)

1.4.6 Linear Discriminant Analysis (LDA)

Description: LDA is a supervised learning method that tries to find a feature space of linear combinations where samples of the same class are as close as possible and samples of different classes are as far apart as possible.
Advantages: LDA can effectively perform feature dimensionality reduction and can consider class information, so it performs well in classification tasks.
Disadvantages: LDA assumes that data follows a Gaussian distribution and that covariances are the same for all classes, which may not hold in practical applications. In addition, LDA is sensitive to outliers and noise. *lda-geeksforgeeks*[18] *lda-analyticssteps*[19] *da-365datascience*[**?**] *lda-vitalflux*[20]

1. **Fisher's Discriminant Criterion:**

   Fisher's Discriminant Criterion is the core of LDA, aiming to find a feature space where samples of the same class are as close as possible, and samples of different classes are as far apart as possible. Given a sample dataset X containing n samples, each with m features, for a binary classification problem, we can define two classes as Class 0 and Class 1. Fisher's Discriminant Criterion can be expressed as:

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

Here, w is a weight vector, S_B is the between-class scatter matrix, and S_W is the within-class scatter matrix. Maximizing $J(w)$ leads to finding the optimal weight vector w for optimal class separation.

2. **Within-Class Scatter Matrix:**

   The within-class scatter matrix S_W represents the scatter within the same class samples. For a binary classification problem, S_W can be computed as follows:

$$S_W = \sum_{i=0}^{1} \sum_{x \in X_i} (x - \mu_i)(x - \mu_i)^T$$

   where X_i represents the sample set of Class i, and μ_i is the sample mean of Class i.

3. **Between-Class Scatter Matrix:**

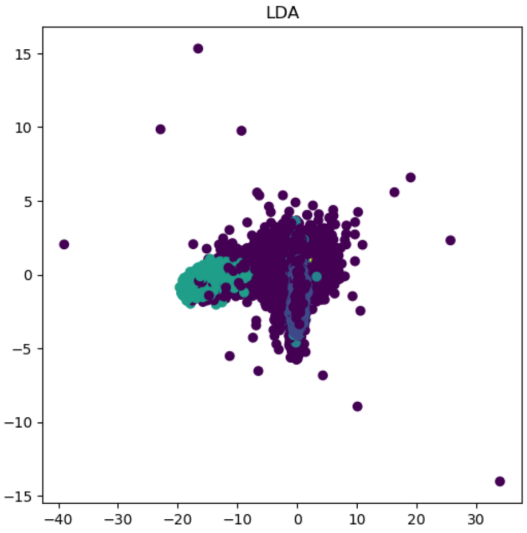
   The between-class scatter matrix S_B represents the scatter between different classes. For a binary classification problem, S_B can be computed as follows:

$$S_B = (\mu_1 - \mu_0)(\mu_1 - \mu_0)^T$$

   where μ_0 and μ_1 are the sample means of Class 0 and Class 1, respectively.

4. **Weight Vector:**

   The optimal weight vector w can be obtained by solving the eigenvalue problem associated with Fisher's Discriminant Criterion.



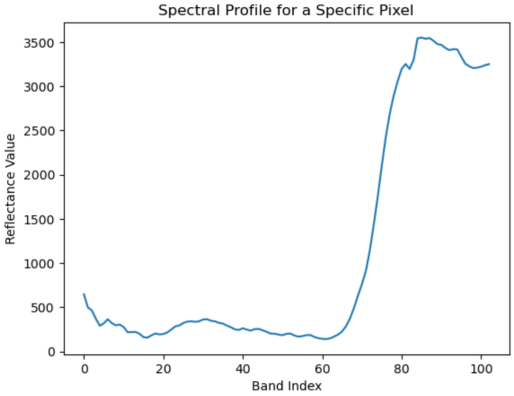
Visualization of Principal Component Analysis (lda)
Please note that we selected Principal Component Analysis (PCA) among these five methods for several reasons:

- **Dimensionality Reduction:** PCA is primarily used for dimensionality reduction. It transforms the original data into a new set of uncorrelated variables, known as principal components. By selecting a subset of these components, you can effectively reduce the dimensionality of your data while retaining most of the variance. This can be particularly useful if you have high-dimensional data and want to reduce computational complexity.
- **Decorrelation:** PCA ensures that the resulting principal components are orthogonal (uncorrelated). This is advantageous because it simplifies the interpretation of the transformed data. Other methods like color histograms, LBP, and HOG do not inherently provide decorrelation.
- **Data Visualization:** PCA allows you to visualize the data in a lower-dimensional space, making it easier to observe the relationships between data points. This can be valuable for exploratory data analysis and pattern recognition.
- **Noise Reduction:** By retaining the top principal components that capture the most variance, you can effectively reduce the influence of noise in your data. This can enhance the signal-to-noise ratio in your analysis.
- **Computationally Efficient:** PCA is computationally efficient, making it suitable for large datasets.

1.5 Data Preprocessing

Data Exploration

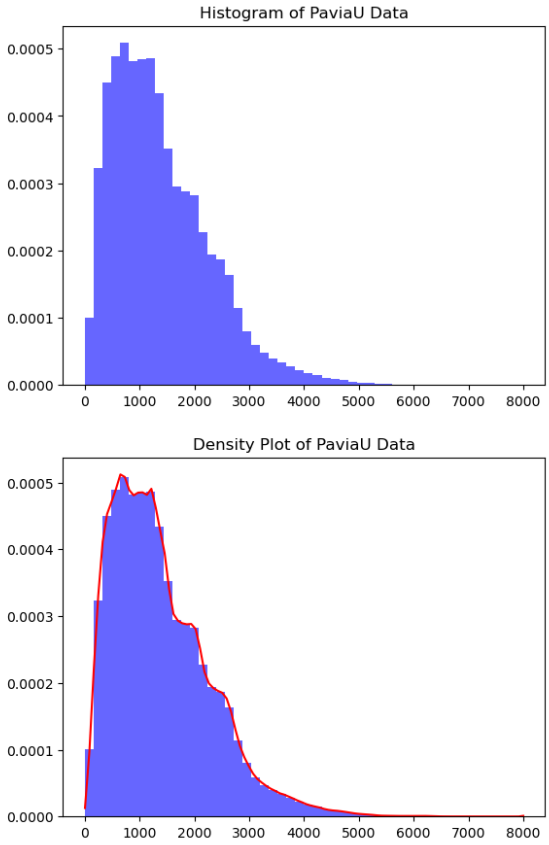
To understand the dataset and its features, we conducted some exploratory data analysis. We visualized a specific feature's spectral profile for a pixel to get an idea of the data's characteristics. Additionally, we analyzed the distribution and statistics of the dataset.



Spectral Profile for a Specific Pixel

Histogram and Density Plot

We examined the dataset's distribution using histograms and density plots. The histogram provides a visual representation of the data distribution, and the density plot overlays a kernel density estimate to visualize the probability density function.



Histogram and Density Plot of PaviaU Data

Statistical Analysis

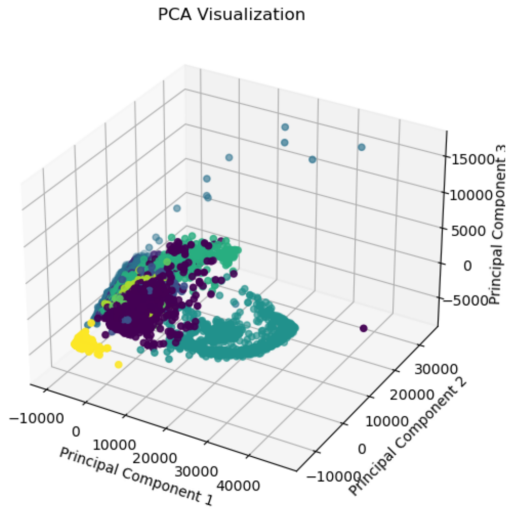
We performed statistical analysis to gain insights into the dataset's characteristics. This included calculating the mean, median, standard deviation, and quartiles of the data. We also conducted the Kolmogorov-Smirnov and Shapiro-Wilk tests to assess the normality of the data.

Mean	1389.13
Median	1215.0
Standard Deviation	897.66
Q1	702.0
Q3	1917.0

Statistical Descriptions of PaviaU Data

PCA Visualization

We applied Principal Component Analysis (PCA) for dimensionality reduction. After performing PCA, we visualized the reduced data in a three-dimensional space to explore the distribution of data points in the new feature space.



PCA Visualization in Three Dimensions

PCA Feature Analysis

We conducted further analysis of the principal components by plotting their relationships with the labels. These scatterplots help understand how the principal components relate to the classification labels.

PCA Feature Analysis vs. Labels

Data Splitting

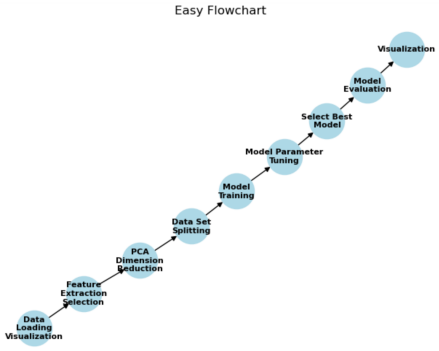
The dataset was divided into three subsets for the purpose of model development and evaluation: training, validation, and test sets. A common practice was followed in the split:

- **Training Set**: Approximately 70% of the data was allocated to the training set. This larger portion of the data was used to train the machine learning models, enabling them to learn from the data patterns.
- **Validation Set**: The remaining 30% of the data was divided equally between the validation and test sets. The validation set was used during model training for hyperparameter tuning and performance evaluation.
- **Test Set**: The test set, also containing 15% of the data, remained untouched during the model development phase. It was solely employed for the final evaluation of model performance, ensuring that the model's generalization capabilities could be properly assessed.

Data Standardization

Standardization is essential for machine learning, ensuring that features have a mean of 0 and a standard deviation of 1. It prevents issues related to feature scales, making the data more suitable for many machine learning algorithms.

2 Workflow



Easy Flowchart By Python

2.1 Nodes and Edges

The workflow consists of the following nodes:

- *Data Loading Visualization*
- *Feature Extraction Selection*
- *PCA Dimension Reduction*
- *Data Set Splitting*
- *Model Training*
- *Select Best Model*
- *Model Evaluation*
- *Visualization*

The edges connecting these nodes represent the logical sequence of tasks to be performed in the workflow, as follows:

- Data Loading Visualization → Feature Extraction Selection
- Feature Extraction Selection → PCA Dimension Reduction
- PCA Dimension Reduction → Data Set Splitting
- Data Set Splitting → Model Training
- Model Training → Model Parameter Tuning
- Model Parameter Tuning → Select Best Model
- Select Best Model → Model Evaluation
- Model Evaluation → Visualization

3 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) is a fundamental supervised learning algorithm used for both classification and regression tasks. It operates based on the principle of proximity, where a new sample's class or value is determined by the classes or values of its nearest neighbors.

3.1 Theoretical Foundation

KNN identifies the K nearest neighbors of a new sample in the feature space, typically using distance measurements. In classification tasks, it classifies the new sample by majority voting among its neighbors. In regression, it predicts the new sample's target value by calculating the average (or weighted average) of the target values of its K nearest neighbors.

3.2 Functionality and Characteristics

- Non-parametric Nature: KNN is non-parametric, making it versatile for various data types.
- Simplicity and Interpretability: KNN's straightforward concept makes it beginner-friendly.
- Suitable for Small Datasets: It performs well with small datasets but can be computationally expensive for larger ones.
- Choosing K: Selecting the appropriate value of K is crucial, as it impacts the model's performance.
- Sensitivity to Noisy Data: KNN is sensitive to noise; outliers can influence predictions.
- Suitable for Low-Dimensional Data: KNN is effective when dealing with a moderate number of features.

3.2.1 Mathematical Formulas

KNN for Classification:

Given a dataset of labeled samples $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$ where x_i represents the feature vectors, and y_i represents the class labels, KNN classifies a new sample x_q into one of the classes based on the majority class among its K nearest neighbors. The formula for KNN classification can be expressed as:

$$\hat{y}_q = \arg\max_j \sum_{i=1}^{K} \mathbf{1}(y_i = j)$$

where $\hat{y}_q$ is the predicted class for sample x_q, j iterates over the class labels, and $\mathbf{1}(y_i = j)$ is an indicator function that equals 1 if y_i is equal to j and 0 otherwise.

KNN for Regression:

In the case of KNN regression, the predicted value for a new sample x_q is determined as the average (or weighted average) of the target values of its K nearest neighbors. The formula for KNN regression can be expressed as:

$$\hat{y}_q = \frac{1}{K} \sum_{i=1}^{K} y_i$$

where $\hat{y}_q$ is the predicted value for sample x_q, K is the number of nearest neighbors, and y_i represents the target values of the neighbors.

Distance Metric:

KNN relies on a distance metric (e.g., Euclidean distance, Manhattan distance) to determine the proximity of samples. The distance between two samples x_a and x_b can be computed using a distance function, such as the Euclidean distance:

$$\text{Distance}(x_a, x_b) = \sqrt{\sum_{i=1}^{n} (x_{a,i} - x_{b,i})^2}$$

where n is the number of features in the dataset.

Choosing the Value of K:

The choice of the value of K in KNN is a critical parameter. It can be determined through methods such as cross-validation to find the optimal K for a specific problem.

3.3 Application Areas

KNN is applied in various domains, including:

- Pattern Recognition: Recognizing patterns in tasks like image and character recognition.
- Recommendation Systems: Building recommendation systems based on user or item similarities.
- Data Mining: Used in data classification and clustering.
- Medical Field: Applied in disease prediction and drug design.
- Finance: Employed in credit scoring, fraud detection, and financial applications.

knn-geeksforgeeks[21] *knn-medium*[22] *knn-realpython*[23]

3.4 Model Performance Considerations

KNN's performance depends on factors like the choice of K and the distance metric. Careful hyperparameter tuning and data preprocessing are crucial when using KNN.

3.5 KNN Parameters Estimation Procedure

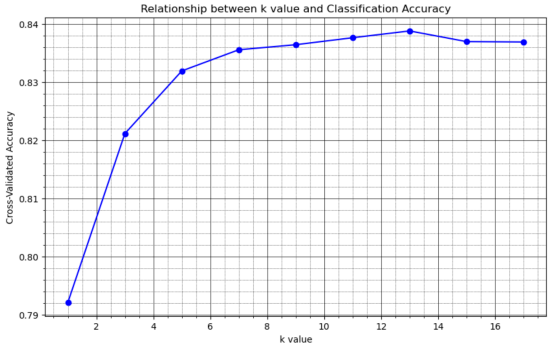
In the following section, we describe the procedure for estimating the optimal value of K for the KNN model.

3.5.1 Choosing the Best K Value

We experimented with different values of K (e.g., 1, 3, 5, 7, 9, 11, 13, 15, 17) to find the best K for our KNN classifier. We used K-Fold cross-validation to evaluate the model's performance for each K value.

The KNN model was trained using the training data, and the accuracy was computed on the validation set for each fold. The average accuracy for each K value was calculated, and the K value that resulted in the highest average accuracy was selected as the best K.

The best K value was found to be 13, with an accuracy of 84%.



Relationship between k value and Classification Accuracy

3.6 Training Procedure Description

3.6.1 KNN Model Training

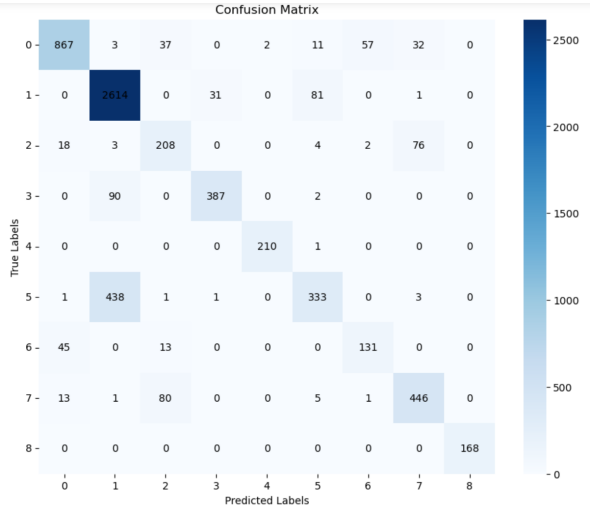
We created a KNN classifier using the best K value (K=13) obtained during the parameter estimation procedure.

The KNN classifier was trained on the entire training set using standardized data (`X_train_Standardization`).

3.7 Demonstrate Results with Figures and Plots

3.7.1 Confusion Matrix

To evaluate the KNN model's performance, we generated a confusion matrix using the test set and the predictions made by the trained classifier. The confusion matrix is shown in Figure.



Confusion Matrix

3.8 Classification Report

The classification report provides a detailed evaluation of the model's performance, including metrics such as precision, recall, and F1-score for each class. The results are summarized in the following table:

Class	Precision	Recall	F1-Score	Support
1	0.92	0.86	0.89	1009
2	0.83	0.96	0.89	2727
3	0.61	0.67	0.64	311
4	0.92	0.81	0.86	479
5	0.99	1.00	0.99	211
6	0.76	0.43	0.55	777
7	0.69	0.69	0.69	189
8	0.80	0.82	0.81	546
9	1.00	1.00	1.00	168

Classification Report

4 Random Forest

Random Forest serves as an ensemble learning technique for classification and regression. It harnesses the power of numerous decision trees, each built with a random subset of features and trained on a Bootstrap sample drawn with replacement from the original dataset. This approach enhances model robustness, addresses overfitting concerns, and consistently achieves high predictive accuracy.

4.1 Theoretical Foundation

The theoretical underpinning of Random Forest is rooted in ensemble learning, specifically the amalgamation of multiple decision trees. Decision trees serve as fundamental components, with each tree constructed using a distinct random subset of features and trained on a Bootstrap sample drawn with replacement from the original dataset. The introduction of randomness in feature selection and data sampling aims to decorrelate individual trees, fostering the development of a resilient and precise ensemble model. Through the amalgamation of diverse tree predictions, Random Forest effectively addresses overfitting concerns and exhibits strong generalization capabilities in both classification and regression scenarios.

4.2 Functionality and Characteristics

- **Ensemble Synergy:** Random Forest leverages ensemble learning, synergizing predictions from multiple decision trees to enhance overall model efficacy.
- **Task Adaptability:** Versatile across classification and regression scenarios, Random Forest accommodates a broad spectrum of predictive modeling requirements.
- **Feature Diversity:** Each decision tree is crafted with a distinct subset of features, ensuring diversity and reducing redundancy within the ensemble.
- **Consistent Predictive Precision:** Through amalgamating diverse tree predictions, Random Forest consistently attains heightened predictive accuracy.
- **Adaptable to Complexity:** Inherently diverse, the model adapts well to intricate data relationships, demonstrating versatility across various datasets and scenarios.

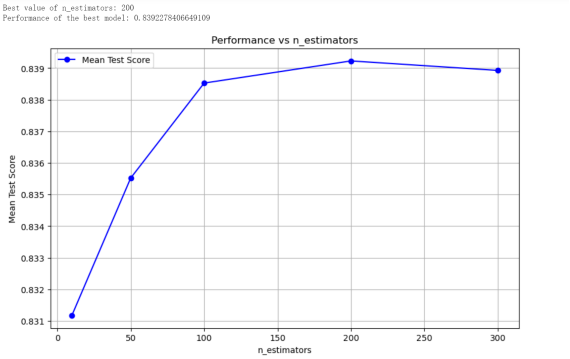
4.3 Application Areas

- **Finance:** In finance, Random Forest is employed for credit scoring, aiding in the assessment of creditworthiness and managing financial risk.
- **Ecology:** Applied in ecology for species classification, Random Forest helps analyze and predict patterns in biodiversity.
- **Marketing:** Widely used in marketing for customer segmentation, targeted advertising, and the development of personalized marketing strategies.
- **Image Analysis:** In computer vision, Random Forest is applied to tasks such as object recognition, image classification, and feature extraction.
- **Remote Sensing:** Used for land cover classification and environmental analysis in studies involving remote sensing data.

randomforest-wikipedia[24]
randomforest-geekculture[25]

4.4 Hyperparameter Tuning

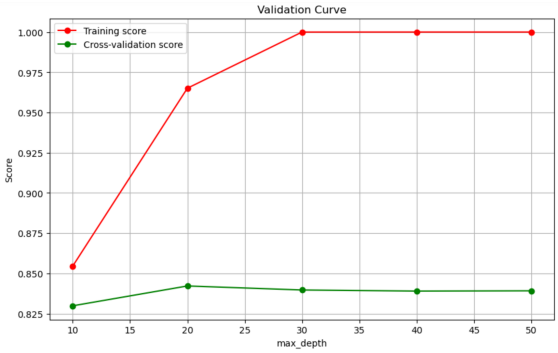
We utilized the `GridSearchCV` method from `scikit-learn` to find the optimal value for the `n_estimators` hyperparameter, which represents the number of trees in the forest. The best value of `n_estimators` was found to be 200, and the model achieved a performance score of approximately 0.8392.



Performance vs n$_e stimators$

4.5 Model Validation

To further refine the model, we performed a validation curve analysis for the `max_depth` hyperparameter. By plotting the training score and cross-validation score against different values of `max_depth`, we identified the optimal `max_depth` value as 20.



Validation Curve

4.6 Model Implementation

With the optimal hyperparameters (`n_estimators = 200` and `max_depth = 20`), we created an

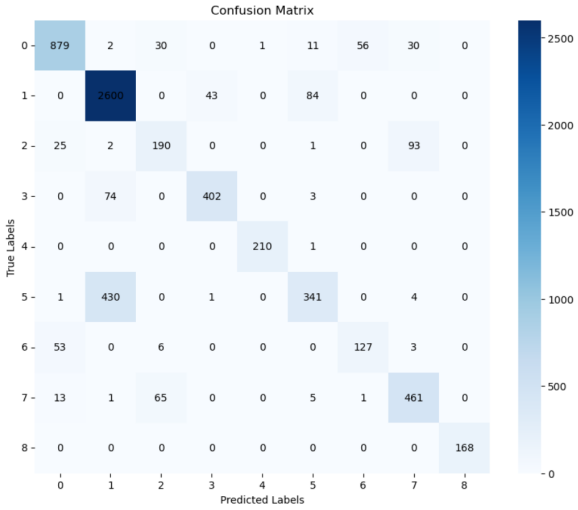
instance of the `RandomForestClassifier` and trained it on the standardized training data.

4.7 Model Evaluation

We evaluated the model's performance using the test data, and the results are as follows:

4.7.1 Confusion Matrix

The confusion matrix visually represents the model's ability to classify instances correctly. Each cell in the matrix indicates the number of true positive, true negative, false positive, and false negative predictions.



Confusion Matrix

4.7.2 Classification Report

The classification report provides detailed metrics for each class and an overall assessment of the model's performance.

Class	Precision	Recall	F1-Score	Support
1	0.91	0.87	0.89	1009
2	0.84	0.95	0.89	2727
3	0.65	0.61	0.63	311
4	0.90	0.84	0.87	479
5	1.00	1.00	1.00	211
6	0.76	0.44	0.56	777
7	0.69	0.67	0.68	189
8	0.78	0.84	0.81	546
9	1.00	1.00	1.00	168
Accuracy			0.84	6417
Macro Avg	0.84	0.80	0.81	6417
Weighted Avg	0.83	0.84	0.83	6417

Classification Report

In summary, the Random Forest model, with carefully tuned hyperparameters, achieved an accuracy of 84% and demonstrated strong performance across multiple classes. It is a robust and versatile model suitable for a wide range of applications.

5 K-Means Clustering and SVM Classification

5.1 K-Means Clustering

K-Means is an unsupervised machine learning algorithm used for clustering, which involves grouping data points into clusters based on their similarity. Here's a comprehensive overview of K-Means:

5.1.1 Theoretical Foundation

K-Means aims to partition a dataset into K clusters, where each cluster is represented by its center (centroid). The algorithm works iteratively, assigning data points to the nearest centroid and then updating the centroids based on the assigned points. This process continues until convergence is reached.

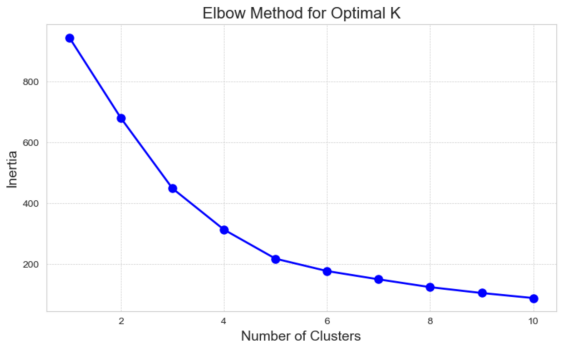
5.1.2 Functionality and Characteristics

- Clustering Method: K-Means is used for data clustering, where it tries to find groups of similar data points.
- Objective Function: It minimizes the sum of squared distances between data points and their respective cluster centroids.
- Initialization: The algorithm starts with an initial guess for the cluster centroids, and the final result may depend on the initial values.
- Sensitivity to K: The choice of K (the number of clusters) can significantly affect the clustering outcome.
- Sensitive to Initializations: Different initializations may lead to different solutions, making multiple runs with different initializations common.
- Euclidean Distance: By default, K-Means uses the Euclidean distance to measure the dissimilarity between data points.
- Scaling Matters: Data scaling can impact K-Means results. Features with different scales may disproportionately affect the clustering.

kmeans-geeksforgeeks[26] *kmeans-wikipedia*[27]

5.1.3 Elbow Method for Optimal K

The optimal number of clusters, as determined by the elbow method, is 5.



Elbow Plot for Optimal K Selection

5.1.4 Clustering Quality Evaluation-Davies-Bouldin Score

To assess the quality of the K-Means clustering, we calculated the Davies-Bouldin Score. This score is a metric used to measure the separation and distinctness of clusters produced by a clustering algorithm.

A lower Davies-Bouldin Score indicates better separation and more distinct clusters. In our case, the Davies-Bouldin Score is 0.69228, which suggests that the clusters generated by the K-Means model exhibit a reasonable level of separation.

However, it's essential to note that the interpretation of clustering quality metrics can be context-dependent. Further analysis and domain-specific knowledge may be required to fine-tune and optimize the clustering results to best suit the specific problem or application.

```
Davies-Bouldin Score: 0.6922783408919738
```

Davies-Bouldin Score

5.2 Support Vector Machines (SVM)

Support Vector Machines are supervised machine learning algorithms used for classification and regression tasks.

5.2.1 Theoretical Foundation

SVM finds the optimal hyperplane that best separates data points of different classes. It aims to maximize the margin between classes while minimizing classification errors.

5.2.2 Functionality and Characteristics

SVM is effective in high-dimensional spaces and can handle both linear and non-linear relationships through kernel functions. It prioritizes data points near the decision boundary (support vectors).

5.2.3 Mathematical Formulas

SVM for Binary Classification:

Given a training dataset (X, y) where X is the feature matrix and y contains binary class labels (typically -1 and 1), the goal of SVM is to find a hyperplane $w \cdot x + b = 0$, where w is the normal vector (vector) and b is the intercept (scalar) that separates the data points into two classes. The formula for the separating hyperplane is given by:

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

The objective of SVM is to maximize the distance (margin) between the support vectors and the separating hyperplane. The distance from the support vectors to the separating hyperplane can be expressed as:

$$\frac{1}{\|w\|}$$

where $\|w\|$ is the norm of the normal vector w.

SVM Optimization Problem:

The SVM optimization problem can be formulated as follows:

Maximize:

$$\frac{1}{2}\|w\|^2$$

Subject to:

$$y_i(w \cdot x_i + b) \geq 1, \text{ for all } i = 1, 2, \ldots, n$$

This is a convex quadratic optimization problem, typically solved using the Lagrange multiplier method.

The following is the Lagrangian dual form of the SVM optimization problem:

Maximize:

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

where α is the Lagrange multiplier.

SVM Classifier:

The SVM classifier can be represented as:

$$f(x) = \text{sign}(w \cdot x + b)$$

where $f(x)$ represents the classification prediction for the input sample x, and the sign$(\cdot)$ function returns the sign (positive or negative) of the input.

5.2.4 Application Areas

SVM finds applications in pattern recognition, image classification, text classification, bioinformatics, and medical diagnosis.

5.2.5 Model Training and Evaluation

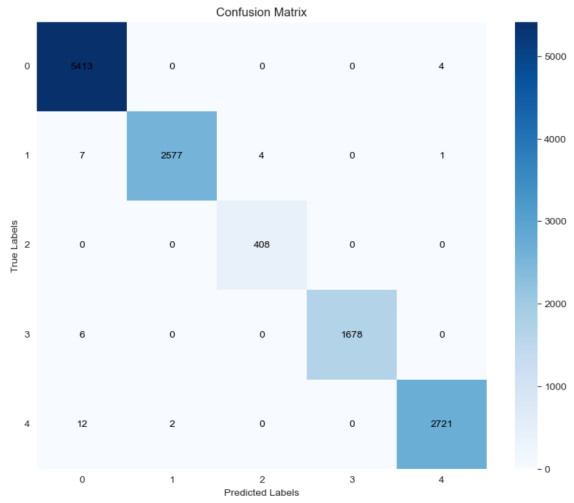
We applied SVM to classify the data with the following parameters:

- Kernel Function: rbf
- C Value: 10
- Gamma Value: 10

The accuracy of the SVM model on the test set is 0.9972

5.2.6 Confusion Matrix

The confusion matrix visually represents the model's ability to classify instances correctly. Each cell in the matrix indicates the number of true positive, true negative, false positive, and false negative predictions.



Confusion Matrix

5.2.7 Classification Report

The classification report provides detailed metrics for each class and an overall assessment of the model's performance.

Class	Precision	Recall	F1-Score	Support
1	0.9954	0.9993	0.9973	5417
2	0.9992	0.9954	0.9973	2589
3	0.9903	1.0000	0.9951	408
4	1.0000	0.9964	0.9982	1684
5	0.9982	0.9949	0.9965	2735
Accuracy			0.9972	12833

classification report

6 Hardware Specifications

The hardware used for this project includes the following:

6.1 CPU

- 11th Gen Intel(R) Core(TM) i5-11300H @ 3.10GHz
- AMD Ryzen 5 5600H with Radeon Graphics @ 3.30 GHz
- 11th Gen Intel(R) Core(TM) i7-11800H @ 2.30GHz

6.2 GPU

- NVIDIA GeForce RTX 3050 Laptop GPU
- NVIDIA GeForce RTX 3060 Laptop GPU

7 Discussion

7.1 Possible Reasons for High Accuracy after K-Means and SVM Processing

When the accuracy of a model approaches 100% after undergoing K-Means clustering and SVM processing, several potential explanations should be considered:

1. **Data Distribution:** One possible reason is that the data may exhibit a highly distinct separation between different classes. This inherent separation allows K-Means clustering to group data into tight clusters, and SVM can easily find linear boundaries. When the data naturally presents clear boundaries in the feature space, SVM efficiently creates linear separations.

2. **Effective Feature Engineering:** After K-Means clustering, the data may have been transformed in a way that enhances the separability between classes. Effective feature engineering or selection techniques might have been applied to make the data more suitable for SVM classification.

3. **Small Dataset Size:** If the dataset is relatively small, SVM may have the capability to memorize all the training examples, resulting in 100% accuracy on the training data. However, it's crucial to note that high accuracy on a small dataset does not guarantee good generalization to unseen data.

4. **Hyperparameter Tuning:** The model's hyperparameters, including the regularization parameter (C) and the choice of kernel functions, may have been meticulously tuned to optimize its performance. Proper parameter selection can lead to high accuracy.

While achieving near-perfect accuracy is desirable, it is essential to conduct a more comprehensive evaluation of the model. This may involve using cross-validation, considering additional performance metrics such as precision, recall, and F1-score, and addressing potential issues related to data imbalance. Additionally, it is crucial to ensure that the data splitting and feature engineering processes are conducted correctly to minimize potential pitfalls.

8 Pros and Cons of the Models

8.1 K-Nearest Neighbors (KNN)

8.1.1 Advantages and Disadvantages:

- **Advantages:**
  - **Simplicity:** KNN is easy to understand and implement.
  - **Versatility:** It can be used for both classification and regression tasks.
  - **Small Datasets:** Well-suited for small datasets.
- **Disadvantages:**
  - **Sensitivity to K:** Performance depends on the choice of the parameter K.
  - **Computational Cost:** Can be computationally expensive for large datasets.

8.2 Random Forest

8.2.1 Advantages and Disadvantages:

- **Advantages:**
  - **High Accuracy:** Random Forest often achieves high accuracy in both classification and regression tasks.
  - **Overfitting Control:** It is robust against overfitting due to ensemble techniques.
  - **Data Flexibility:** Handles both numerical and categorical data effectively.
  - **Feature Importance:** Provides feature importance scores for interpretability.
- **Disadvantages:**
  - **Computational Resources:** Requires more computational resources compared to simpler models.
  - **Interpretability:** May not be as straightforward to interpret as simpler models like linear regression.

8.3 Combined K-Means and SVM Model

8.3.1 Advantages and Disadvantages:

- **Advantages:**
  - **Synergy:** Combines the strengths of K-Means for data clustering and SVM for classification.
  - **Task Flexibility:** Adaptable to both unsupervised clustering and supervised classification tasks.
  - **Enhanced Feature Engineering:** Improves feature extraction for SVM classification by utilizing cluster assignments.
- **Disadvantages:**
  - **Parameter Tuning:** Requires careful parameter tuning for both K-Means and SVM components.
  - **K Sensitivity:** Sensitivity to the choice of K in K-Means may impact performance.
  - **High-Dimensional Data:** May face challenges when handling high-dimensional data.

9 Reasons for Model Combination

The decision to combine K-Means and Support Vector Machines (SVM) in a novel model is grounded in the aspiration to harness the respective strengths of clustering and classification. Clustering with K-Means allows for effective feature engineering, data pattern discovery, and grouping of similar data points. The subsequent classification through SVM empowers the model with informed decision-making capabilities, as it's well-suited for distinguishing between various data clusters. This symbiotic relationship enables our model to gain a deeper understanding of the underlying data structures and, as a result, substantially enhance classification performance.

10 Reasons for Models' Pros and Cons

10.1 K-Nearest Neighbors (KNN)

10.1.1 Advantages - Reasons:

- **Simplicity:** KNN is easy to understand and implement.

  This simplicity arises from its intuitive concept of making predictions based on the nearest neighbors, which requires minimal complex modeling or training.

- **Versatility:** It can be used for both classification and regression tasks.

  KNN's ability to handle classification and regression tasks makes it versatile, adapting to a wide range of predictive modeling problems.

- **Small Datasets:** Well-suited for small datasets.

  KNN performs well on small datasets as it doesn't rely on extensive training data, reducing the risk of overfitting.

10.1.2 Disadvantages - Reasons:

- **Sensitivity to K:** Performance depends on the choice of the parameter K.

  The choice of the K parameter significantly influences KNN's performance. An improper K value selection may lead to underfitting or overfitting, necessitating careful parameter tuning.

- **Computational Cost:** Can be computationally expensive for large datasets.

  KNN's computational cost becomes substantial with large datasets because it necessitates distance calculations between all data points, especially in high-dimensional data.

10.2 Random Forest

10.2.1 Advantages - Reasons:

- **High Accuracy:** Random Forest often achieves high accuracy in both classification and regression tasks.

  The high accuracy of Random Forest results from its robust ensemble techniques, which mitigate overfitting and enhance predictive performance.

- **Overfitting Control:** It is robust against overfitting due to ensemble techniques.

  Random Forest's ensemble methods, including random feature selection and majority voting, effectively control overfitting and improve generalization.

- **Data Flexibility:** Handles both numerical and categorical data effectively.

  Random Forest can efficiently handle mixed data types, including numerical and categorical features, with minimal data preprocessing.

- **Feature Importance:** Provides feature importance scores for interpretability.

  Random Forest offers feature importance scores, facilitating model interpretation and understanding.

10.2.2 Disadvantages - Reasons:

- **Computational Resources:** Requires more computational resources compared to simpler models.

  Random Forest demands increased computational resources due to multiple decision trees, leading to longer training times and higher memory usage.

- **Interpretability:** May not be as straightforward to interpret as simpler models like linear regression.

  The interpretability of Random Forest models is relatively lower compared to simpler models like linear regression, making the decision process less intuitive.

10.3 Combined K-Means and SVM Model

10.3.1 Advantages - Reasons:

- **Synergy:** Combines the strengths of K-Means for data clustering and SVM for classification.

  This combination leverages K-Means' clustering capabilities and SVM's powerful classification performance, resulting in enhanced overall model effectiveness.

- **Task Flexibility:** Adaptable to both unsupervised clustering and supervised classification tasks.

  The model's adaptability to unsupervised clustering and supervised classification tasks makes it highly versatile, suitable for diverse problem types.

- **Enhanced Feature Engineering:** Improves feature extraction for SVM classification by utilizing cluster assignments.

By utilizing K-Means clustering results, feature extraction for SVM classification is improved, leading to enhanced classification performance.

10.3.2 Disadvantages - Reasons:

- **Parameter Tuning:** Requires careful parameter tuning for both K-Means and SVM components.

  This combined model necessitates careful tuning of K-Means parameters (e.g., K value) and SVM parameters, which can be a complex and time-consuming process.

- **K Sensitivity:** Sensitivity to the choice of K in K-Means may impact performance.

  The choice of the K value in K-Means significantly influences the model's performance. An improper K value selection may lead to suboptimal clustering results.

- **High-Dimensional Data:** May face challenges when handling high-dimensional data.

  Dealing with high-dimensional data can be challenging and may require additional computational resources and feature selection techniques to ensure the model's performance remains effective.

11 Conclusion and Future Directions

In the course of this study, we introduced an innovative hybrid model that amalgamates K-Means with Support Vector Machines (SVM). And we compared with K-Nearest Neighbors (KNN) and Random Forest. During the model execution, we encountered notable computational demands, with the SVM component being particularly resource-intensive. This study represents a significant step in exploring the integration of multiple machine learning techniques for classification and clustering tasks.

Looking ahead, future improvements should be aimed at optimizing the training and tuning processes to mitigate the computational demands of the SVM component. Careful parameter selection for both K-Means and SVM will be crucial. Furthermore, addressing the potential challenges associated with high-dimensional data is essential for making this hybrid model more adaptable to a broader range of practical applications.

12 Novelty and Contributions

The primary contribution of this study lies in its comprehensive comparison of KNN, Random Forest, and the innovative combination of K-Means and SVM models. This unique fusion of diverse machine learning techniques offers valuable insights for practical applications, particularly in the domain of remote sensing data processing and analysis. The model's potential for enhanced feature engineering, data clustering, and informed classification presents a novel approach to solving complex problems in the field.

References

[1] N. A. Provided, "Pythonmat," *CSDN Blog*, No Date, accessed: 10 October 2023. [Online]. Available: https://blog.csdn.net/sinat_29957455/article/details/105436000

[2] M. Narula, "Read matlab mat files in python," *DelftStack*, No Date, accessed: 10 October 2023. [Online]. Available: https://www.delftstack.com/howto/python/read-mat-files-python/

[3] N. A. Provided, "python.mat," *CSDN Blog*, No Date, accessed: 10 October 2023. [Online]. Available: https://blog.csdn.net/qq_53860947/article/details/131478681

[4] nadagamal3. (2023) Hyperspectral classification. Accessed: 10 October 2023. [Online]. Available: https://www.kaggle.com/code/nadagamal3/hyperspectral-classification

[5] A. da Silva and C. Silveira. (2022) Handcrafted feature selection techniques for pattern recognition: A survey. Accessed: 10 October 2023. [Online]. Available: https://search-ebscohost-com-s.elink.xjtlu.edu.cn:443/login.aspx?direct=true&db=edsarx&AN=edsarx.2209.02746&site=eds-live&scope=site

[6] N. A. Provided, "Color histograms in image retrieval," *Pinecone Blog*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.pinecone.io/learn/series/image-search/color-histograms/

[7] D. Zhang, "Color feature extraction," *SpringerLink*, 2019, accessed: 27 October 2023. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-17989-2_4

[8] N. A. Provided, "Local binary patterns," *Wikipedia*, No Date, accessed: 27 October 2023. [Online]. Available: https://en.wikipedia.org/wiki/Local_binary_patterns

[9] ——, "A texture descriptor: Background local binary pattern (bglbp)," *SpringerLink*, 2015, accessed: 27 October 2023. [Online]. Available: https://link.springer.com/article/10.1007/s11042-015-2588-3

[10] ——, "On the application lbp texture descriptors and its variants for no-reference image quality assessment," *MDPI J. Imaging*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.mdpi.com/2313-433X/4/10/114

[11] ——, "Histogram of oriented gradients," *Wikipedia*, No Date, accessed: 27 October 2023. [Online]. Available: https://en.wikipedia.org/wiki/Histogram_of_oriented_gradients

[12] S. Mallick, "Histogram of oriented gradients explained using opencv," *LearnOpenCV Blog*, 2016, accessed: 27 October 2023. [Online]. Available: https://learnopencv.com/histogram-of-oriented-gradients/

[13] N. A. Provided, "Histogram of oriented gradients — skimage 0.22.0 documentation," *Scikit-Image Documentation*, No Date, accessed: 27 October 2023. [Online]. Available: https://scikit-image.org/docs/stable/auto_examples/features_detection/plot_hog.html

[14] ——, "Principal component analysis," *Wikipedia*, No Date, accessed: 27 October 2023. [Online]. Available: https://en.wikipedia.org/wiki/Principal_component_analysis

[15] ——, "Principal component analysis(pca)," *GeeksforGeeks*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.geeksforgeeks.org/principal-component-analysis-pca/

[16] ——, "Principal component analysis of raw data - matlab pca," *MathWorks Documentation*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.mathworks.com/help/stats/pca.html

[17] ——, "A step-by-step explanation of principal component analysis (pca)," *Built In Blog*, No Date, accessed: 27 October 2023. [Online]. Available: https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[18] ——, "Ml — linear discriminant analysis," *GeeksforGeeks*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/

[19] ——, "Introduction to linear discriminant analysis in supervised learning," *Analytics Steps Blogs*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.analyticssteps.com/blogs/introduction-linear-discriminant-analysis-supervised-learning

[20] ——, "Pca vs lda differences, plots, examples," *Analytics Yogi*, No Date, accessed: 27 October 2023. [Online]. Available: https://vitalflux.com/pca-vs-lda-differences-plots-examples/

[21] ——, "K-nearest neighbor(knn) algorithm," *GeeksforGeeks*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.geeksforgeeks.org/k-nearest-neighbours/

[22] A. Hussain, "K-nearest neighbors (knn) and its applications," *Medium Blog*, 2020, accessed: 27 October 2023. [Online]. Available: https://medium.com/@arman_hussain786/k-nearest-neighbors-knn-and-its-applications-7891a4a916c6

[23] N. A. Provided, "The k-nearest neighbors (knn) algorithm in python," *Real Python*, No Date, accessed: 27 October 2023. [Online]. Available: https://realpython.com/knn-python/

[24] ——, "Random forest," *Wikipedia*, No Date, accessed: 27 October 2023. [Online]. Available: https://en.wikipedia.org/wiki/Random_forest

[25] Yamini, "Random forest — ensemble method," *Geek Culture, Medium Blog*, 2021, accessed: 27 October 2023. [Online]. Available: https://medium.com/geekculture/random-forest-ensemble-method-860aaf4fcd16

[26] N. A. Provided, "K means clustering - introduction," *GeeksforGeeks*, No Date, accessed: 27 October 2023. [Online]. Available: https://www.geeksforgeeks.org/k-means-clustering-introduction/

[27] ——, "k-means clustering," *Wikipedia*, No Date, accessed: 27 October 2023. [Online]. Available: https://en.wikipedia.org/wiki/K-means_clustering