Predicting Income Levels Using the UCI Adult Census Dataset: A Machine Learning Approach

Shahzada Saeed

Module: Applied Machine Learning

Student Number: 230265789

Word Count: 2037

This report documents the steps undertaken to analyse the UCI Adult Census Dataset, to predict whether an individual's income exceeds $50,000 per year. The project involves data preprocessing, feature selection, and the application of machine learning algorithms: XGBoost, Self-Training with Decision Trees, and Support Vector Machines (SVM).

1. Data

Overview

The dataset contains 15 attributes, which encompasses demographic and income information for approximately 48,842 individuals. The binary nature of the target variable (income >50K) provides a clear and straightforward objective for classification models, and the dataset reflects real-world socio-economic factors, making it applicable for practical scenarios.

Relevance and Applications

This dataset can be utilised in:

- Marketing: Facilitating the identification of high-income consumer segments for targeted marketing campaigns.
- Business Intelligence: Enhancing customer segmentation and enabling the development of personalised services based on income predictions.
- Economic Research: Providing insights into income distribution and the socio-economic factors influencing earnings, which can inform public policy and economic strategies.
- Credit Scoring: Financial institutions can use income predictions to improve credit scoring models, enhancing risk assessment.

Source Reference

The dataset was retrieved from the UCI Machine Learning Repository: (https://archive.ics.uci.edu/ml/datasets/Adult).

Data Issues

Sampling Bias

The UCI Adult Census Dataset originates from the 1994 U.S. Census, meaning the data may reflect biases specific to that time period. These biases could affect the model's performance and generalisability. Therefore, predictions made using this dataset might not accurately represent current income distributions and demographic trends.

Attribute Representation

Certain demographic groups may be underrepresented in the dataset, and the model may not perform well for these groups, leading to skewed or biased predictions.

Missing Values

The dataset contains missing values for some attributes, which need handling. Specifically, attributes such as work class, occupation, and native country have missing entries.

Data Imbalance

Income distribution in the dataset is imbalanced, with fewer instances of individuals earning more than $50,000 compared to those earning less. This imbalance can affect the training process of machine learning models, leading them to be biased towards the majority class.

Pre-Processing Steps

The dataset is loaded from the UCI repository URL, and the 15 columns are defined. Figure 1 demonstrates the head of the dataset.

Missing values are then handled, such that rows with missing values are dropped to ensure a clean dataset. Categorical variables are converted into numerical format using label encoding, as many machine learning algorithms require numerical input, such as distance calculations or matrix multiplications.

Numerical features are standardised using StandardScaler to ensure that all features are on a comparable scale to improve model performance. It will also improve convergence for the SVM candidate gradient-based algorithm in this report.

The data is split into features and a target variable; therefore, ‘income’ and ‘fnlwgt’ are dropped from features and ‘income’ is defined as the target variable.

Dimensionality reduction is performed after feature selection.

2. Feature Selection

Feature selection helps improve model performance and reduce complexity by identifying the most relevant features from a dataset. Three techniques are utilised as functions followed by an aggregation mechanism to select the final set of features. The number of features is defined as input and the functions are called by the final workflow, such that multiple numbers are tested for the performance of algorithms.

Mutual Information

This filter method uses statistical tests to select features with the highest correlation to the target variable. It measures the reduction in uncertainty about the target variable given the feature. It is simple to interpret and evaluates each feature independently of others, however, it does not consider interactions between features.

Recursive Feature Elimination with a Decision Tree Classifier

This wrapper method evaluates feature subsets by training and evaluating a model iteratively. The report utilises a Decision Tree Classifier, ranks the features based on their importance, removes the least important features and retrains the model. The method considers interactions between features and their collective contribution to the model's performance; however, it is computationally expensive and may overfit.

Random Forest Classifier

Feature Importance from a Random Forest Classifier involves evaluating the significance of each feature based on its contribution to the model's predictions. Random forests, which are ensembles of decision trees, provide a measure of feature importance by assessing how much each feature decreases the impurity in the trees.

This report utilises Gini Impurity because it is faster to compute than entropy as it does not involve logarithms, and it provides for performance as it tends to create purer nodes compared to entropy.

Feature Selection

The results of the techniques are collated into core and extended features. Core Features are those that appear in at least two of the selection methods and so are more reliable. Extended Features only appear in one of the selection methods. The Selected Features combine both results; however, if this is higher than the number of features defined by the workflow, the exact number will be selected.

Dimensionality Reduction

The number of features is reduced by this step while retaining as much of the information as possible. High-dimensional spaces may lead to overfitting, increase computational cost, and reduce the generalisation ability of the model.

Principal Component Analysis (PCA) is utilised to achieve dimensionality reduction. It aims to capture the maximum variance in the data with the fewest number of components and achieves orthogonality. The number of components in this report is defined by the final workflow, such that the optimal number is chosen through iteration for model performance.

3. Candidate ML Algorithms

The primary task for candidate algorithms is binary classification, i.e. predicting whether income is >50K or <50K. The dataset has a mix of continuous (e.g. age) and discrete (e.g. occupation) attributes. The candidates should be able to handle both types of features effectively, exploit the structure of the data and provide interpretable results, achieve scalability to handle large datasets and be computationally efficient.

Three candidates are chosen for the report: XGBoost, Self-Training with Decision Trees, and Support Vector Machines (SVM). In addition, a fourth candidate, K-Means Clustering, is chosen to demonstrate relatively poor performance for the task.

XGBoost (Extreme Gradient Boosting)

This is a supervised learning algorithm and is used to predict an outcome variable based on input features, with labelled data. Gradient boosting is an ensemble technique that builds models sequentially; each new model corrects errors made by the previous models. Decision trees are used as the base learners in this framework.

Its hyperparameters are Learning Rate, Number of Trees, Maximum Depth, Minimum Child Weight, Gamma, Subsample, and Colsample_bytree.

The algorithm is chosen as it can handle both categorical and numerical features effectively, is known for its high performance and accuracy in classification tasks, provides insights into feature importance and therefore aides interpretability, is designed to be efficient and scalable, provides flexibility in hyperparameter turning, and is an ensemble method.

Self-Training with Decision Trees

A semi-supervised learning algorithm that leverages both labelled and unlabelled data for training. Self-training is a wrapped method that can be applied to any base classifier, which is a Decision Tree in this case.

The hyperparameters of self-training are Base Estimator, Threshold, Maximum Iterations, and the Decision Tree has Max Depth, Minimum Samples Split, and Minimum Samples Leaf.

The algorithm is chosen as decision trees provide clear and interpretable decision rules, can capture non-linear relationships effectively, and can use both label and unlabelled data, which in this case will deliberately be masked.

Support Vector Machines (SVM)

A supervised algorithm used for classification and regression tasks, with an objective to find the hyperplane that best separates the data points of different classes.

The hyperparameters are a Regularisation Parameter, Kernel Type, Kernel Parameters: Gamma and Degree, and Class Weight.

SVM is chosen as it performs well in high-dimensional spaces, is robust to overfitting, has high accuracy in binary classification tasks and has strong generalisation to unseen data.

K-Means Clustering

K-means clustering is an unsupervised learning algorithm that partitions data into K distinct clusters based on feature similarity, minimising the variance within each cluster. It iteratively assigns data points to the nearest cluster centroid and then recalculates the centroids until convergence.

It is likely to perform poorly as it is primarily a clustering algorithm rather than classification.

4. Workflow

The report utilises a workflow function that provides for variables and hyperparameters to run feature selection, dimensionality reduction, and candidate algorithms. It aims for a comprehensive computation of variables to identify the optimal configuration for each algorithm. The results are collectively exported to the project directory in 'results.csv'.

The first section of the workflow runs each algorithm on the following permutations:

- Number of features for selection methods: 3, 7, 10
- Number of PCA components: 2, 3, 5
- Test sizes: 20%, 30%
- Cross-validation values: 3, 5, 7

Due to the computational complexity of SVM, only the first two values are used for the first three variables, and the first value is used for the last two variables.

Within each of these permutations, a given model is run for the best configuration by employing a grid search over a dictionary of hyperparameters combined with number of cross-validation folds and a scoring metric, macro average F1. The metric is used so that performance on the minority class is adequately reflected, and that precision and recall are incorporated together.

5. Evaluating Models and Analysing Results

Feature Selection

Relationship, Age, Marital Status, and Capital-Gain were generally ranked with higher importance, whilst Race, Sex, and Native Country were less important. Figures 2-3 demonstrate the rankings by Mutual Information and Random Forest Classifier.

PCA

The variance captured by PC1 ranged from approximately 40-70%, with generally higher capture when lower features were selected. Figure 4 demonstrates PCA with 5 components when 10 features were selected. PC1 and PC2 together capture around 70% of the total variance, suggesting that a reduced set could potentially improve performance. The results output demonstrates the best models for each of the three candidate algorithms had PCA components between 3-5, with feature selection of 7.

Top Performing Models

Figures 5- 8 demonstrate the top-performing permutations and hyperparameters for the algorithms. XGBoost achieved the highest accuracy of 85.15%, whilst K-Means achieved the lowest of 63.75%. XGBoost and Self-Training had balanced precision-recall for both classes, whereas SVM had a slightly lower recall for the minority class. K-Means's performance indicates that is not suitable for the task.

Accuracy By Algorithm

Per Figure 9, XGBoost and SVM exhibit high and consistent accuracy around 0.83-0.84, making them reliable and effective for the dataset. Self-Training achieves slightly lower accuracy (0.81 to 0.82) with more variability.

Accuracy by Variable

Figures 10-13 demonstrate the accuracy of an algorithm across four variables permutated by the workflow. For XGBoost, accuracy improves with 7 features and 5 PCA components, with consistent performance across test sizes and CV values. Self-Training also shows higher accuracy with 7 features and 5 PCA components, but exhibits more variance with different test sizes and CV values. SVM accuracy peaks with 7 features and 3 PCA components. This indicates that XGBoost and SVM are more robust to changes in these variables.

6. Conclusion

Based on the evaluation conducted on the UCI Adult dataset, the performance of four machine learning algorithms have been identified across a range of parameters. The key focus was to optimize the macro average F1-score, which provides a balanced measure of precision and recall across different classes.

The best-performing model was XGBoost with seven features, five PCA components, test size of 0.2 and five-fold cross-validation, achieving the highest accuracy of 85.15% and a macro-average F1-

score of 0.78. The best hyperparameters used were Learning Rate: 0.2, Maximum Depth: 3, Number of Estimators: 200.

The second model was Self-Training with Decision Trees with seven features, five PCA components, test size of 0.2 and three-fold cross-validation, achieving an accuracy of 83.75% with a macro average F1-score of 0.74. The best hyperparameters used were: Max Depth: 7, Minimum Samples Split: 2, Threshold: 0.5.

The report therefore recommends the XGBoost as the primary model due to its superior performance in terms of accuracy and balanced precision-recall metrics. Second, it recommends Self-Training with Decision Trees as a complementary model, Its semi-supervised learning approach makes it adaptable and valuable for leveraging additional unlabelled data.

These models were chosen based on their ability to deliver high accuracy and balanced F1-scores, making them robust choices for predicting income levels in the UCI Adult dataset.

Diagrams

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex	capital-gain	capital-loss	hours-per-week	native-country	income
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	<=50K
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	<=50K
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	<=50K
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	<=50K
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	<=50K

Figure 1 - Head of UCI Adult dataset

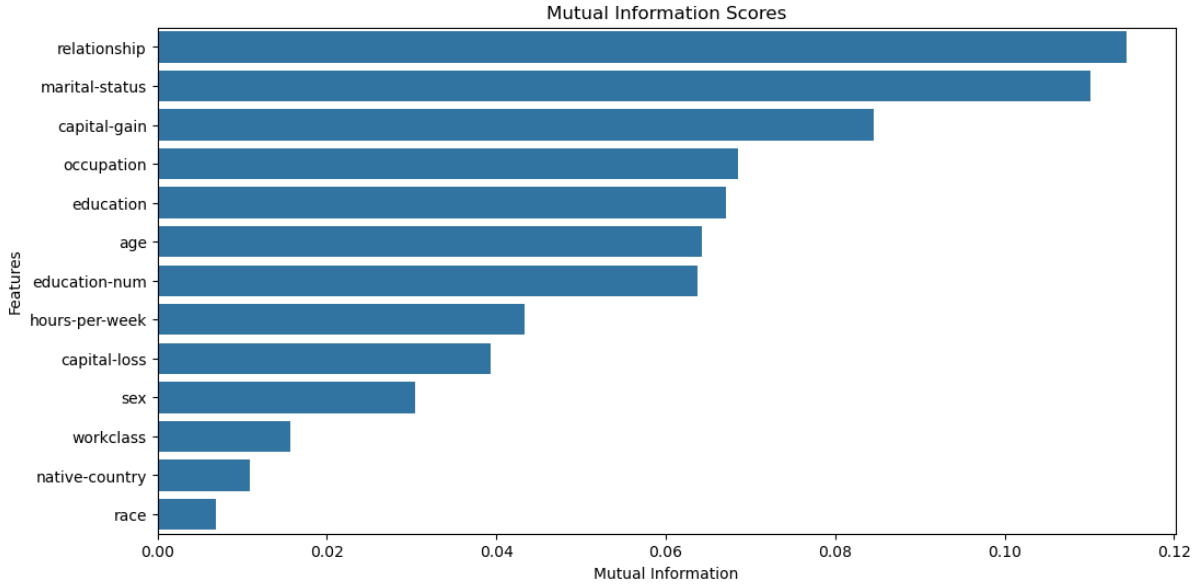

Figure 2 - Mutual Information Scores

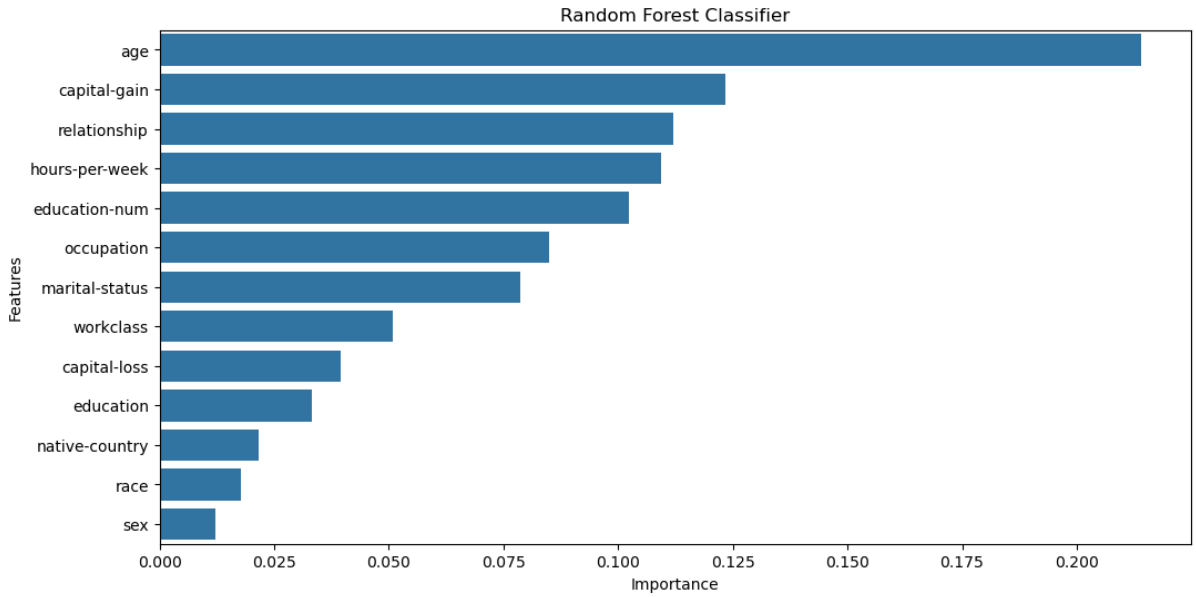

Figure 3 - Random Forest Classifier

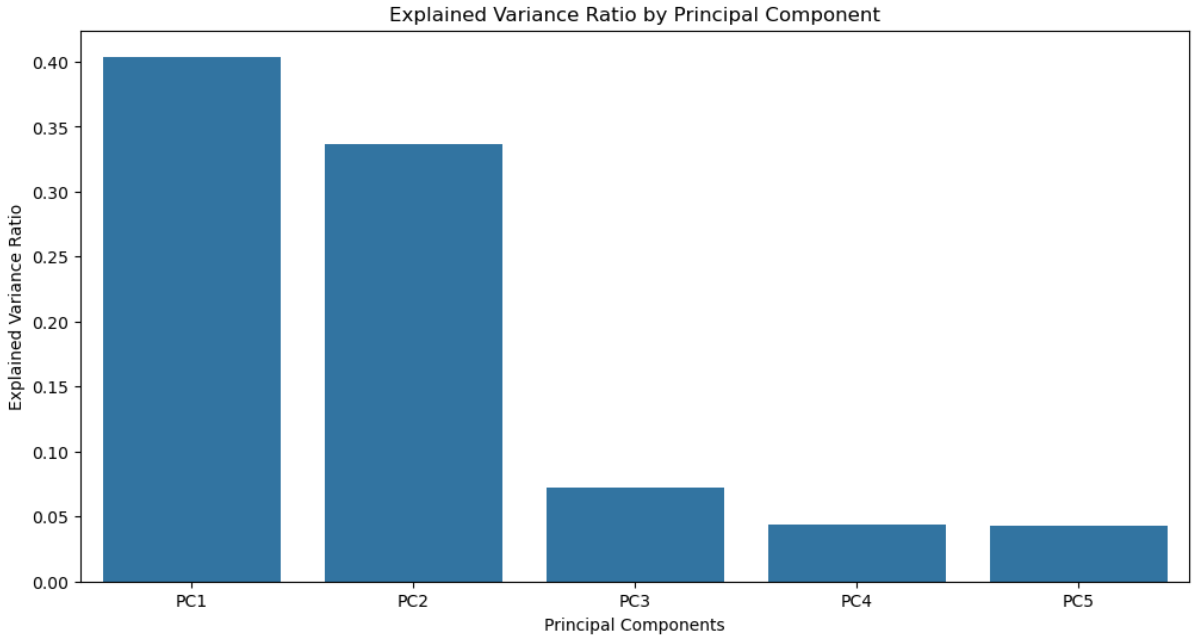

Figure 4 - Principal Component Analysis

Algorithm	Features	PCA	Test_Size	CV	Accuracy	Best_Params
XGBoost	7	5	0.2	5	85.15%	{'learning_rate': 0.2, 'max_depth': 3, 'n_estimators': 200}
	precision	recall	f1-score	support		
0	0.88	0.93	0.9	4942		
1	0.73	0.6	0.66	1571		
accuracy			0.85	6513		
macro avg	0.81	0.77	0.78	6513		
weighted	0.85	0.85	0.85	6513		

Figure 5 - XGBoost Best Performer

Algorithm	Features	PCA	Test_Size	CV	Accuracy	Best_Params
Self-Trainir	7	5	0.2	3	83.76%	{'base_estimator__max_depth': 7, 'base_estimator__min_samples_split': 2, 'threshold': 0.5}
	precision	recall	f1-score	support		
0	0.87	0.93	0.9	4942		
1	0.71	0.55	0.62	1571		
accuracy			0.84	6513		
macro avg	0.79	0.74	0.76	6513		
weighted av	0.83	0.84	0.83	6513		

Figure 6 - Self-Training Best Performer

Algorithm	Features	PCA	Test_Size	CV	Accuracy	Best_Params
SVM	7	3	0.2	5	81.64%	{'C': 10}
	precision	recall	f1-score	support		
0	0.86	0.91	0.88	4942		
1	0.65	0.51	0.57	1571		
accuracy			0.82	6513		
macro avg	0.75	0.71	0.73	6513		
weighted av	0.81	0.82	0.81	6513		

Figure 7 - SVM Best Performer

Algorithm	Features	PCA	Test_Size	CV	Accuracy	Best_Params
K-Means	3	2	0.2	3	63.75%	{'n_clusters': 2}
	precision	recall	f1-score	support		
0	0.75	0.79	0.77	4942		
1	0.19	0.16	0.18	1571		
accuracy			0.64	6513		
macro	0.47	0.47	0.47	6513		
weighted	0.61	0.64	0.62	6513		

Figure 8 - K-Means Best Performer

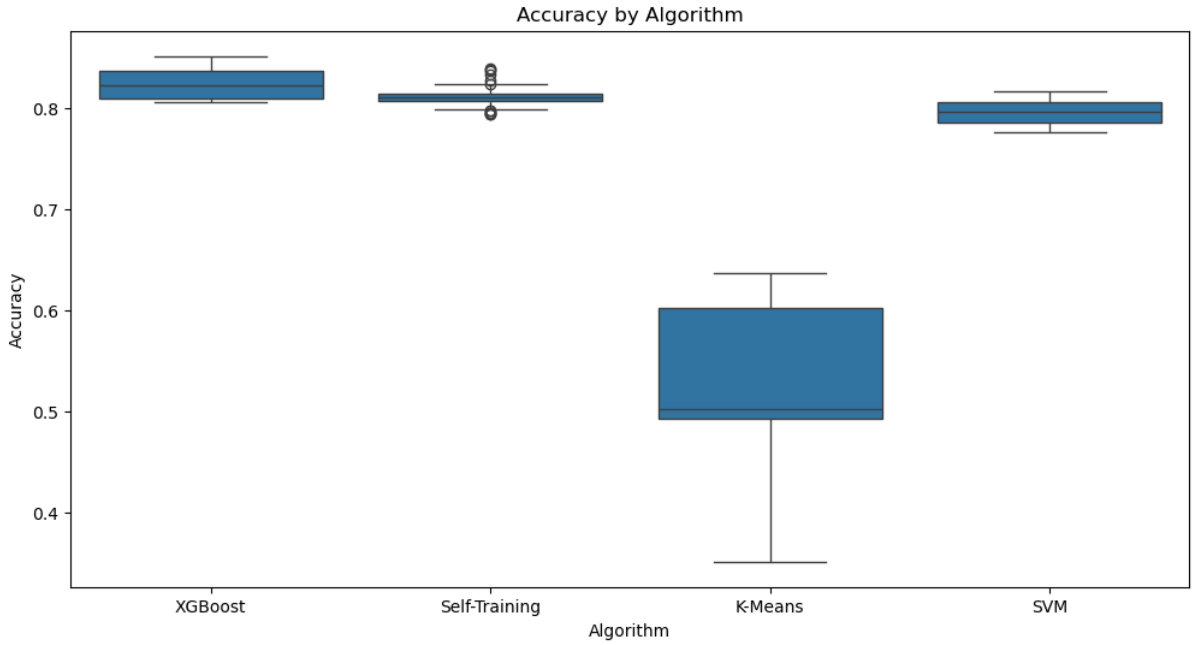

Figure 9 - Accuracy by Algorithm

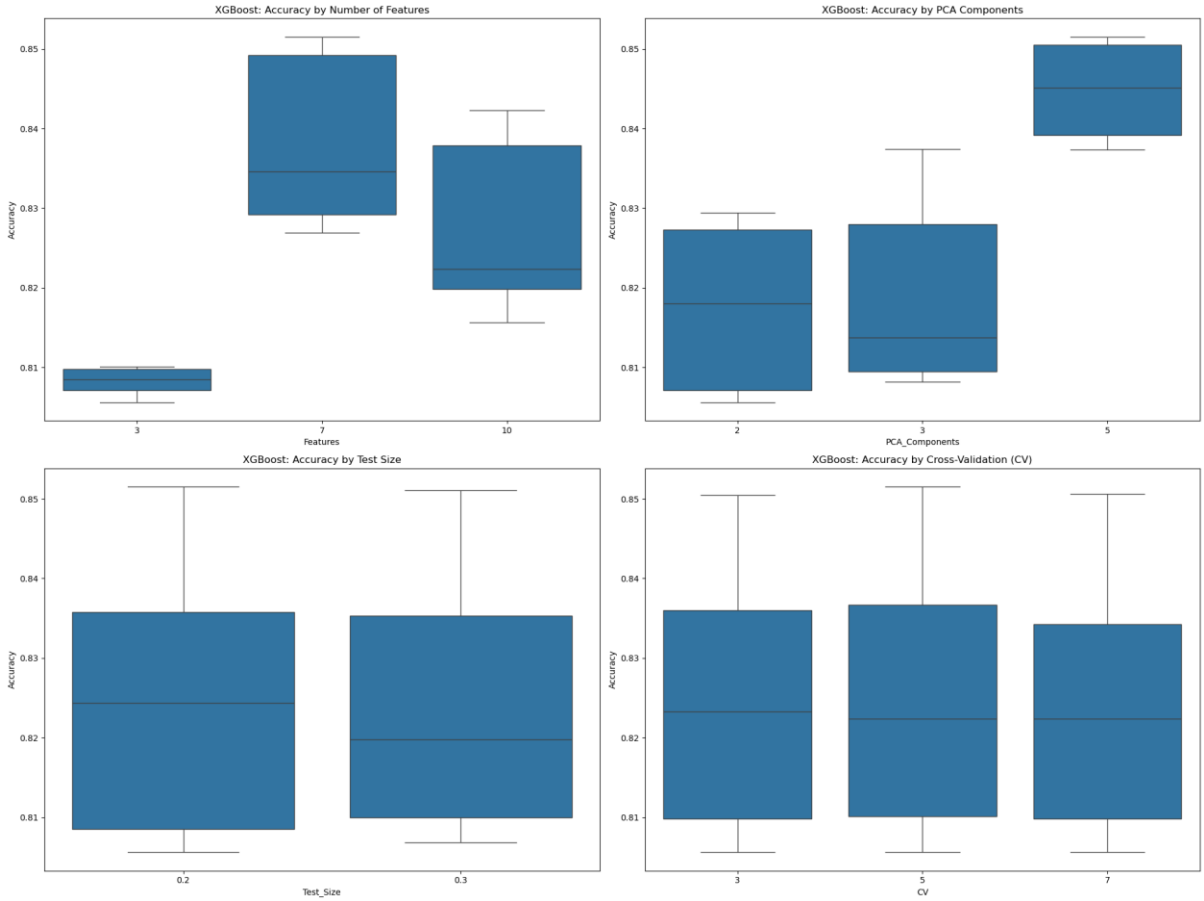

Figure 10 - XGBoost Accuracy

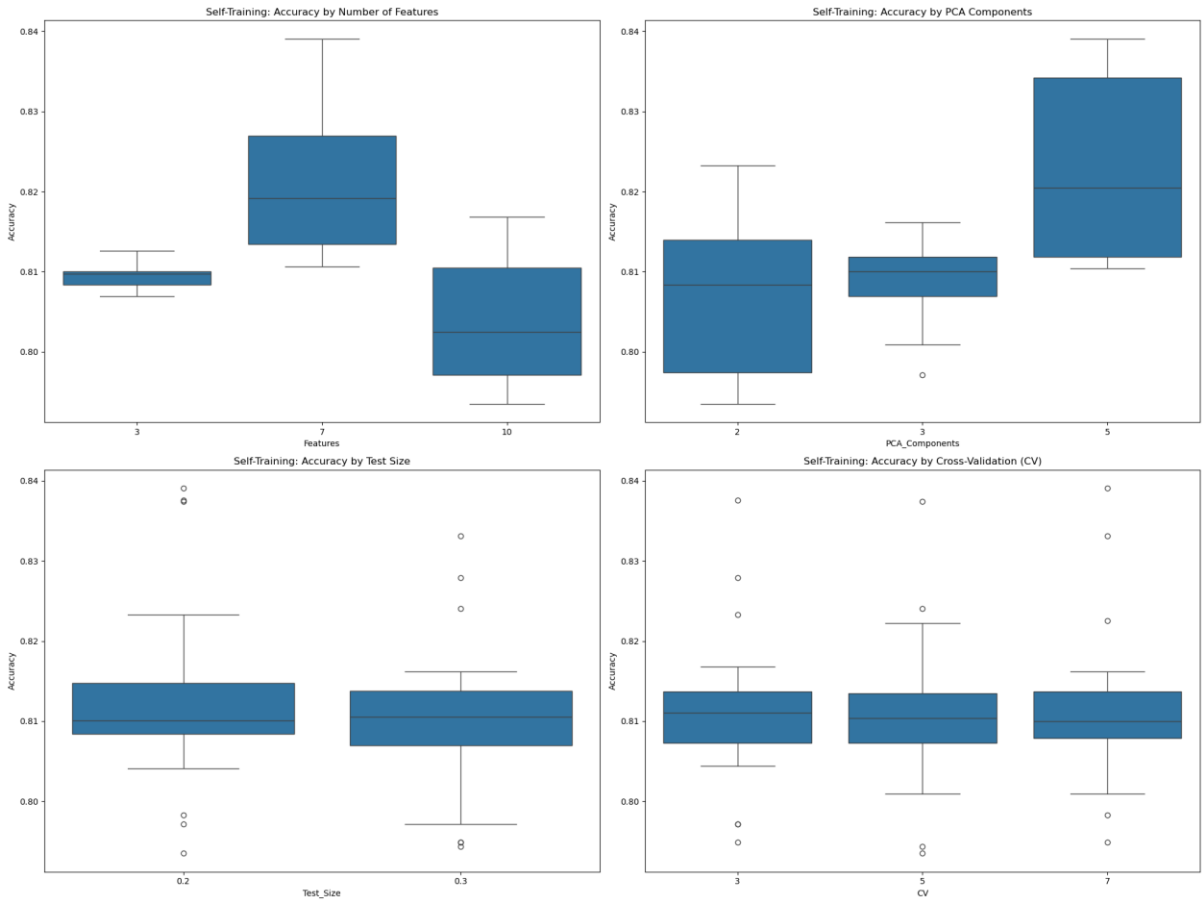

Figure 11 - Self-Training Accuracy

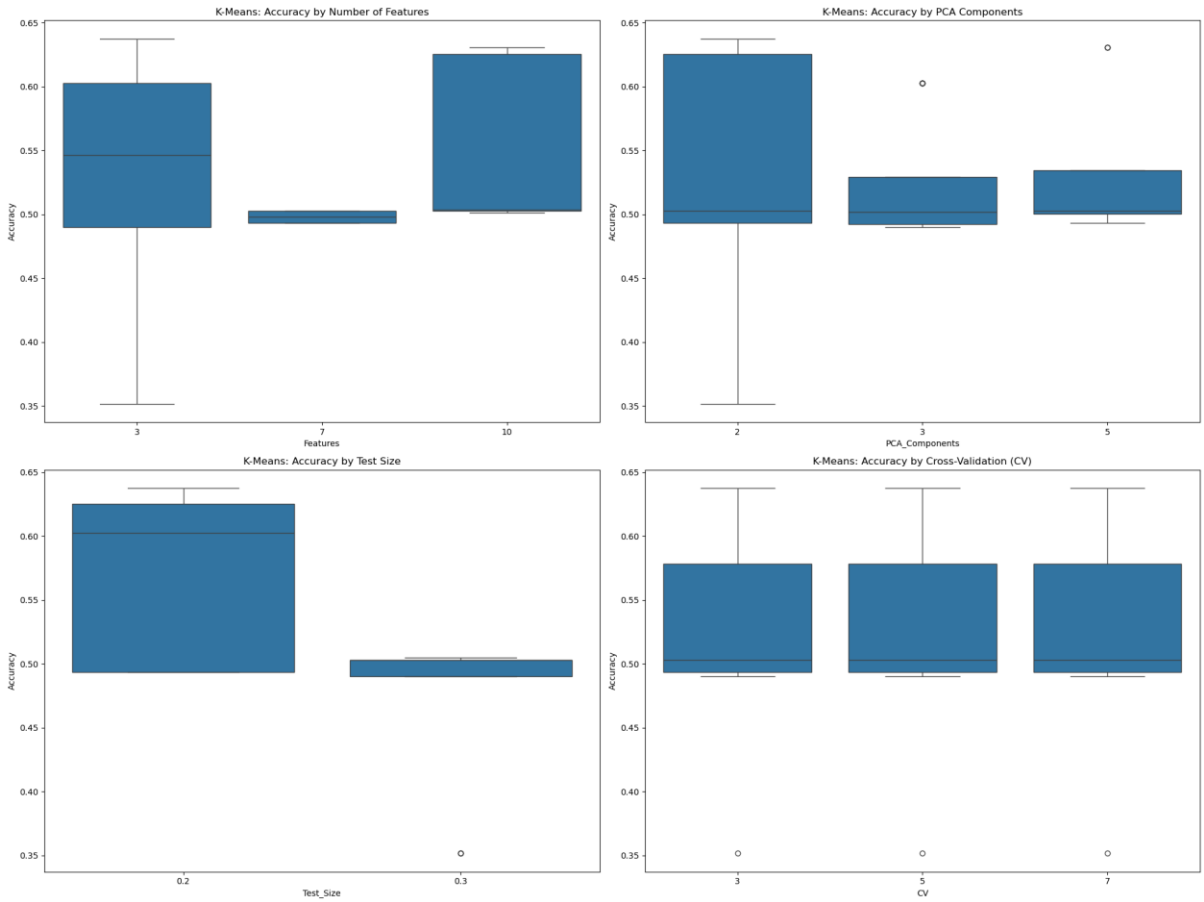

Figure 12 - K-Means Accuracy

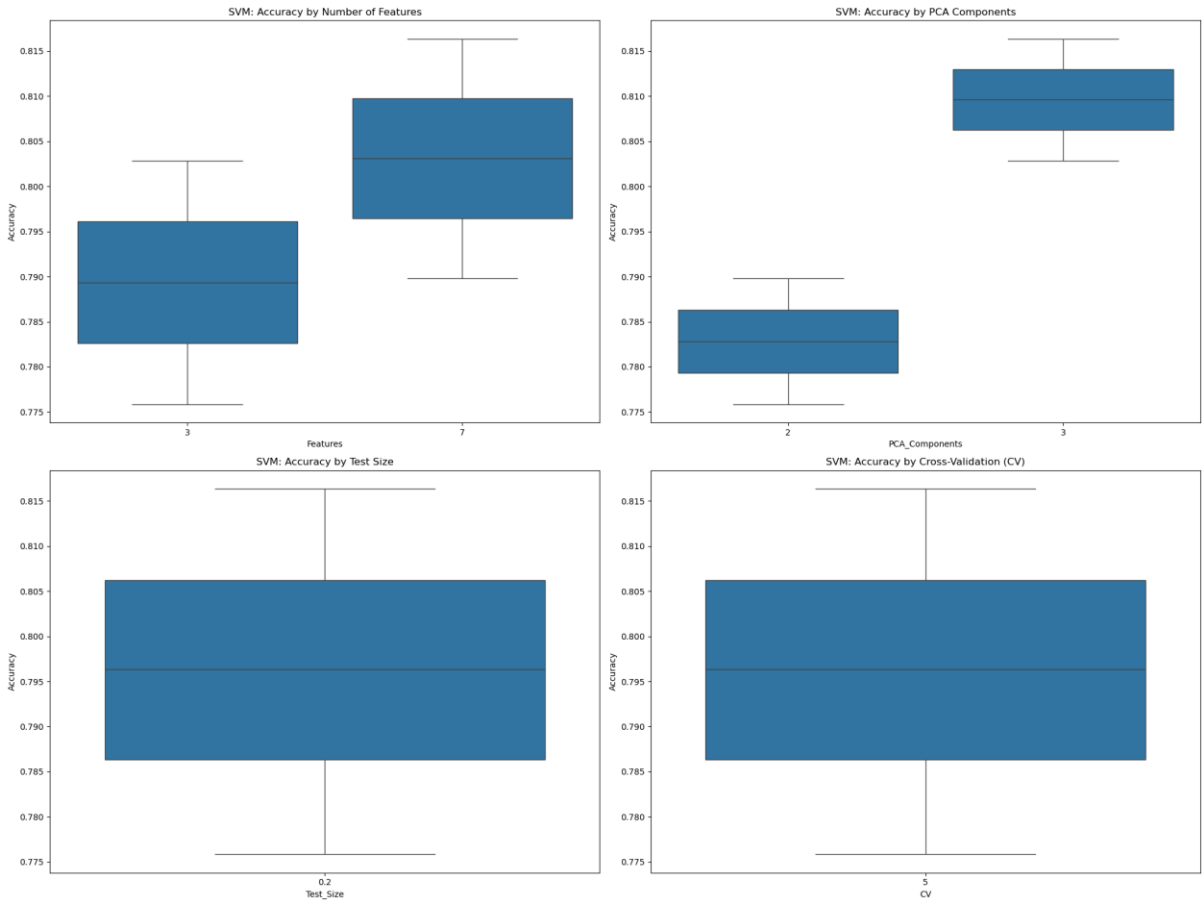

Figure 13 - SVM Accuracy