MACHINE LEARNING (E0 270) ASSIGNMENT 1

Analysis of PCA Components and SVM Performance Metrics in Multi-class Classification

Introduction:

In this study, we examine the relationship between the number of Principal Component Analysis (PCA) components (k) and the performance metrics of a Support Vector Machine (SVM) classifier in a multi-class classification problem. The performance metrics evaluated include accuracy, precision, recall, and F1-score. The objective is to determine if there is a consistent pattern in the performance metrics as the number of PCA components varies, and to discuss the implications of the observed trends.

Methodology:

1. We employed a dataset (MNIST) with multiple classes and features, loaded the splitted training and testing data and normalized it.
2. The PCA technique was applied to the training data to reduce its dimensionality.
3. Implementing a Soft SVM using Stochastic gradient descent.
4. The number of PCA components (k) was varied, and for each k, the multi-class SVM classifier was trained using the reduced-dimensional training data.
5. The trained classifier was then used to predict the test data, and the performance metrics were calculated.
6. The relationship between the number of PCA components and the performance metrics was analyzed by plotting them together.

Observations:

Upon analyzing the output from the given graph below, we observed that all the metric curves follow a similar trend for every k value up to a certain number of components, after which the curve flattens. The key observations are as follows:

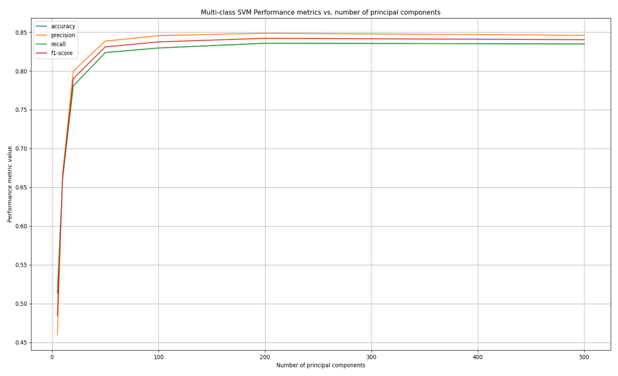

Fig 1: with Our implementation (C = 1)

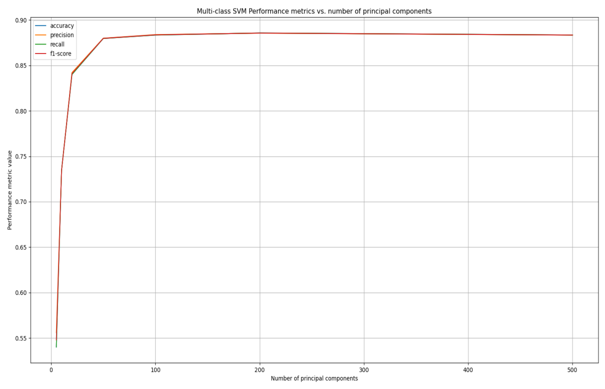

Fig 2: with Our implementation (C = 5)

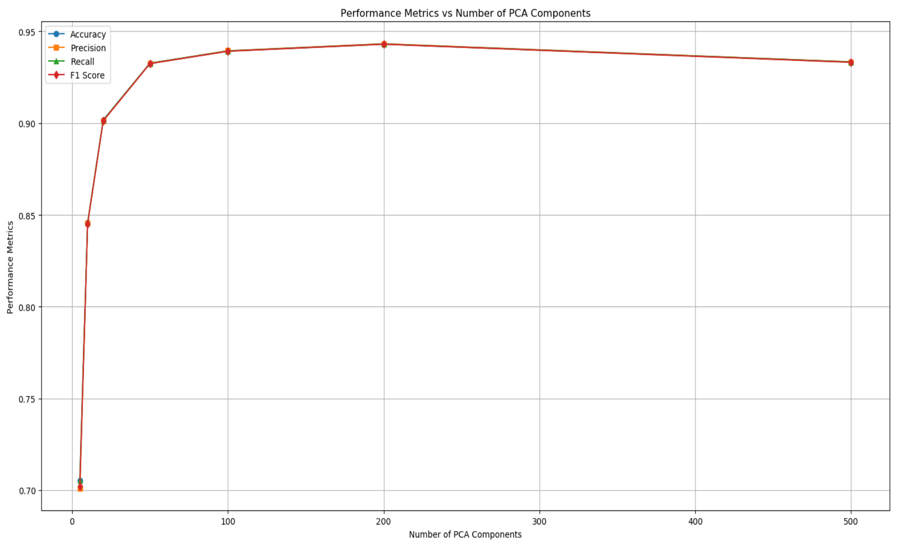

Fig 3: with Scikit-learn Implementation.

1. As the number of PCA components increases, the performance metrics initially increase, reaching an optimal point where the classifier performs the best. The optimal point here is for k=200 with our implementation and also for k = 200 with scikit-learn implementation.

2. Beyond the optimal point, the performance metrics tend to plateau, showing little to no improvement as more PCA components are added. This suggests that after a certain number of components, the additional dimensions may not contribute to the overall classification performance. This behavior is known as the "elbow" or "knee" point, and it indicates the optimal number of principal components to use for the given dataset and classification task.

3. The optimal number of PCA components varies depending on the specific dataset and classification problem. It is crucial to find the right balance between retaining useful information and reducing dimensionality for computational efficiency.

4. Comparison with Scikit-learn library Implementation: With our implementation, the performance metric curves follows similar pattern to the sciki-learn Implementation. But the maximum accuracy with our implementation is 88.56 (for k=200) but using scikit-learn, the maximum accuracy is 94.27 (for k=200). It is important to note that, in order to improve performance of the model, we need to optimize our implementation. The following techniques are recommended:
Optimize the learning rate and parameter C: Instead of using a fixed learning rate and the parameter C, we can perform a grid search or random search to find the best hyperparameters for our dataset.
Use better optimization techniques: Instead of using stochastic gradient descent (SGD), we can use more advanced optimization algorithms like Adam or RMSProp. These optimizers typically converge faster and can improve model performance.
Use early stopping: To avoid overfitting, we can use early stopping by monitoring the validation loss or performance metrics during training. Stop the training if the validation metric does not improve for a certain number of iterations.
Use k-fold cross-validation: To get a more reliable performance estimate, we can use k-fold cross-validation when evaluating the performance of the SVM classifier.

5. Comparing the results between hyperparameter C=5 and C=1 (keeping iterations and learning_rate constant):

- the hyperparameter C determines the balance between allowing misclassification and forcing strict separation of classes. A higher value of C prioritizes minimizing misclassifications over finding a larger decision boundary margin.
- The fact that C=5 performs better than C=1 for this particular dataset suggests that the data points are not linearly separable and the classes are tightly mixed. With a lower C value (C=1), the SVM tries to maximize the decision boundary margin, allowing for more misclassifications. But in a scenario where the classes are closely intertwined, this leads to a less optimal model as more instances are misclassified.
- On the other hand, when C=5, the SVM algorithm prioritizes minimizing misclassifications over finding a larger margin. This results in a more complex decision boundary that may better fit the complex distribution of the data, thus leading to better model performance.
- It's important to note that while C=5 performs better in this instance, it doesn't imply that a higher C value will always lead to better performance. The optimal value for C is problem-dependent and should be chosen using methods like cross-validation. Overly high values for C can cause the model to overfit to the training data, which can lead to poor performance on unseen data. Therefore, careful tuning of C is crucial in model development.

Analysis:
Based on the observed patterns in the performance metrics, it can be concluded that the output graph is reasonable for the test data. The consistent trends in accuracy, precision, recall, and F1-score up to a certain number of PCA components, followed by a flattening of the curve, suggest that the classifier performance is sensitive to the number of PCA components used.

It is important to select an appropriate number of PCA components that maximizes the performance metrics while minimizing the risk of overfitting and maintaining computational efficiency. Cross-validation and other model selection techniques can be employed to determine the optimal number of PCA components for a given dataset and classification problem.

In terms of model bias, using the fact that the performance metrics are relatively balanced (i.e., approximately similar values for precision, recall, and f1_score, i-e not much large difference) suggests that the model is not significantly biased towards any particular class.

Conclusion:
In summary, the analysis of PCA components and SVM performance metrics in multi-class classification provides valuable insights into the importance of feature reduction and its impact on classifier performance. This study highlights the need to carefully consider the trade-offs between dimensionality reduction and model performance in machine learning applications, as well as the potential limitations of adding more PCA components beyond a certain point.