The Effectiveness of PCA in KNN, Gaussian Naive Bayes Classifier and SVM for Raisin Dataset

Agnieszka Polowczyk[1], Alicja Polowczyk[1]

[1]Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44-100 Gliwice, Poland

Abstract

Supervised learning is one of the main types machine learning in which model is trained from data which consists of features (input data) and labels, that is target values. Using our training data, the parameters of our model will be adjusted until loss function reaches a low value or until we get high accuracy on the validation data. Before we start building model, we need make data preprocessing. PCA is often used, to reduce numbers of dimensions our data. Models in which data have been reduced using PCA often have high accuracy. In this article, we will look at how well-known classifiers work such as: K-Nearest Neighbors, Gaussian Naive Bayes and Support Vector Machines, that using PCA. We will also check the performance of the classifiers for which the data has been reduced to fewer dimensions by analyzing correlation tables and we will look at models whose data contain the original number of features. We will evaluate their effectiveness based on the Raisin database and show how decision boundaries built in models that were constructed after our analysis.

Keywords

Machine learning, pca, knn, gaussian naive bayes, svm, classifiers

1. Introduction

Supervised learning is used in many areas, such as: classification [1, 2], regression [3], patterns recognition, natural language processing and image encryption [4, 5, 6]. Examples of problems that can be solved using supervised learning are: classifying whether an email is spam or not, weather forecasting, classify text, whether a review is positive or negative[7, 8].

The popular algorithm used during training model for example in the case of regression or SVM classifier, is the gradient descent, which minimizes loss function, by adjusting the parameters of our model in the direction of the decreasing gradient of the loss function [9, 10]. The goal of supervised learning is to achieve high accuracy to make right predictions on unknown data. There are many interesting improvements to such models for application systems. In [11] was presented how to use machine learning for imbalanced data inputs, while in [12, 13] was presented positioning of technical systems for power electric models. We can find also many applications for complex input data structure ie. [14] gave it for the graph based input relations compositions.

In classification problem we also distinguish models based on deep neural networks[15]. The architecture of neural networks is: weights, activation functions [16], loss function and optimizer. In the case of classifier KNN and Gaussian Naive Bayes there is no learning with weights. Using KNN model on large dataset, it can lead to high consumption of computing resources. In [17] was proposed strategy, which improve the efficiency of KNN classifier on Big Data.

In this paper we will compare three classifiers: KNN, Gaussian Naive Bayes and SVM, that were built on three various data:

- model uses PCA to reduce the dimensionality of the data
- model uses two features selected by us
- model uses all the features

We will check the effectiveness of the above models, in the case of the KNN for different metrics and for the SVM model we will test the performance for various kernels. We will summarize whether reduction of the dimensions of our data allows us to get satisfactory results, leading to a decrease in computational complexity.

2. Raisin database

The database that we used to build various classifiers contains samples that were described by 7 morphological features. These features were obtained after previously processing the photos.Values are continuous and we can see that each feature has value from different ranges. There are also high values of standard deviations for example, for Area and ConvexArea features, indicating that the values for these features are highly dispersed from their mean.

SYSYEM 2023: 9th Scholar's Yearly Symposium of Technology, Engineering and Mathematics, Rome, December 3-6, 2023

ap307985@polsl.pl (A. Polowczyk); ap307986@polsl.pl (A. Polowczyk)

2.1. Standardization

Normalization or standardization are used to improve the efficiency and effectiveness of the model. In the case of KNN model, that uses distance measures to classify data samples, if the values weren't normalized or standardization, features with higher values could have greater impact on model's result, which could lead to low accuracy. Therefore, an important and recommended action is to use one of the data processing techniques before creating KNN and SVM models. Mainly for Gaussian Naive Bayes doesn't use data standardization, because this algorithm doesn't depend on distance, so doesn't require scaling of features. We used standardization exceptionally in Gaussian Naive Bayes classifiers in which the PCA technique were used and in classifiers in which used two-dimensional data to plot decision boundaries.We used standardization, which transforms our data in a way that the mean is equal to 0 and the standard deviation is equal to 1. At the beginning for everyone feature we calculated the mean value and standard deviation and then we used the results to calculate new values using the formula below:

$$x_{new} = \frac{x - \mu}{\sigma} \tag{1}$$

2.2. Model based on PCA

One of the popular dimensionality reduction techniques is PCA. The task of PCA is to return n-features that we can create a model with high accuracy. Interesting improvements to PCA models composed for graph based classifiers were presented in [18]. In our models were used PCA, which returns to us new training and test data reduced from seven to two dimensions.

2.3. Model based on two features

Another way to prepare data for the model is to reduce dimensionality based on correlation analysis. Correlation defines the relation between two variables. Correlation value close to 1 or -1 mean a strong correlation, but value close to 0 mean weak correlation. The Extent feature was removed from our training and testing data, because its correlation value with our target feature was only 0.28. Additionally, the following features were eliminated: ConvexArea, Perimeter, Area, MinorAxisLength, because these attributes had strong relation with other features and didn't contribute relevant information to the classification models. Finally, our classifiers were built on other two features: MajorAxisLength and Eccentricity. The Fig. 1 shows correlation plots between two features.

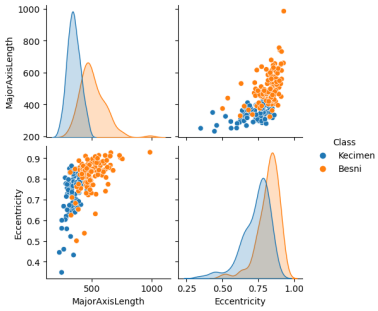


Figure 1: Correlation graphs of two features

2.4. Model based on all features

For each classifier, we also built a model based on all seven features. Sometimes training a model on the basis of all attributes can be a disadvantage, because this approach lead to slower learning of our classifier. However, the advantage of including all features is that in some cases it can lead to very high efficiency of our machine learning algorithm, because we don't lose any relevant information. Fig. **??** illustrates our feature and correlation graphs.

3. Methods

3.1. KNN

3.1.1. Formulas

Euclidean distance:

$$D(x,y) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2} \tag{2}$$

Manhattan distance:

$$D(x,y) = \sum_{i=1}^{m}|x_i - y_i| \tag{3}$$

Minkowski distance:

$$D(x,y) = \left(\sum_{i=1}^{m}|x_i - y_i|^r\right)^{\frac{1}{r}} \tag{4}$$

Canberra distance:

$$D(x,y) = \sum_{i=1}^{m}\frac{|x_i - y_i|}{|x_i| + |y_i|} \tag{5}$$

Chebyshev distance:

$$D(x,y) = \max_{i=1}^{m}|x_i - y_i| \tag{6}$$

Cosine distance:

$$D(x,y) = 1 - \frac{\sum_{i=1}^{m} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{m} x_i^2} \cdot \sqrt{\sum_{i=1}^{m} y_i^2}} \quad (7)$$

3.1.2. Algorithm

KNN classifier is mathematical model, that doesn't require training. New, unknown points are predicted based on the k-nearest points voting. When classifying a new sample, model calculates distances between the sample and each point in the specified n-dimensional space. Among all the distances the model chooses k-smallest and voting takes place. The class that occurs most frequently among the selected points becomes the predicted class for the new sample. We compared performance of KNN classifier for different distance measures (metrics): Euclidean, Manhattan, Minkowski for $r = 3$, Canberra, Chebyshev and Cosine.

3.2. Gaussian Naive Bayes

3.2.1. Formulas

Bayes' Theorem in our model:

$$P(y|x_1, x_2, ..., x_n) = \frac{P(y) \cdot \prod_{i=1}^{n} P(x_i|y)}{P(y|x_1, x_2, ..., x_n)} \quad (8)$$

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \cdot \exp(-\frac{(x_i - \mu_y)}{2\sigma_y^2}) \quad (9)$$

Sample variance:

$$\sigma^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \bar{x})^2 \quad (10)$$

3.2.2. Algorithm

Gaussian Naive Bayes is probabilistic model, that uses Bayes' Theorem to determine the probability of sample belonging to a specific class. The classifier assumes, that the features are independent. We used this type of classifier, because our data is continuous and the data is approximately normally distributed. During the training process, our model calculated the mean and variance for each attribute from every class and the "a priori" probabilities for each class. When predicting a test data, two probabilities are returned because we have binary classification. We chose the highest probability with its label.

3.3. SVM

3.3.1. Formulas

Hinge Loss:

$$\varepsilon_i = \max(0, 1 - y_i f(x_i)) \quad (11)$$

Updating weights and bias:

$$w_t = w_t - \eta \nabla_w Cost(w_t) \quad (12)$$

$$b = b - \eta \nabla_b Cost(w_t) \quad (13)$$

Minimizing the cost function using Stochastic Gradient Descent (SGD):

$$minCost(w_i) = \lambda ||w||^2 + \max(0, 1 - y_i f(x_i)) \quad (14)$$

$$\lambda = \frac{1}{NC} \quad (15)$$

3.3.2. Algorithm

SVM classifier (Support Vector Machines) creates a hyperplane that maximizes the distance between the closest points of two classes (support vectors). When creating a hyperplane, two techniques are used: soft margin and hard margin. Soft margin during the process of training allows our algorithms to make mistakes, so it allows points to be on the wrong side of the hyperplane or inside the margin. Hard margin during the process of training doesn't tolerate any errors, so points cannot be on the wrong side of the hyperplane or inside the margin. In our case, where our data isn't completely linear separable, we used the soft margin technique and used various kernels to transform our data to higher dimensionality. We also used regularization parameter C. We created models, in which one of the classes is equal to 1 and the other is equal to -1. Then, using Stochastic Gradient Descent, we updated our weights and b after each data sample. Finally, we tested our models, if the predicted values were negative, they were assigned the label -1, if non-negative, they are assigned the label 1. We compared the performance of classifiers using different kernels such as: linear, poly, rbf, laplacian and sigmoid.

4. Experiments

4.1. KNN

We created the KNN models for different metrics for which the prediction is based on 3 nearest neighbors.

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.89	0.87	0.88	39
Kecimen	0.88	0.90	0.89	39
accuracy			0.88	78
macro avg	0.88	0.88	0.88	78
weighted avg	0.88	0.88	0.88	78

Figure 2: Classification reports for KNN with PCA. The results are shown in order for the metrics: euclidean, manhattan, minkowski, canberra, chebyshev, cosine

	precision	recall	f1-score	support
Besni	0.82	0.85	0.84	39
Kecimen	0.84	0.82	0.83	39
accuracy			0.83	78
macro avg	0.83	0.83	0.83	78
weighted avg	0.83	0.83	0.83	78

	precision	recall	f1-score	support
Besni	0.83	0.87	0.85	39
Kecimen	0.86	0.82	0.84	39
accuracy			0.85	78
macro avg	0.85	0.85	0.85	78
weighted avg	0.85	0.85	0.85	78

	precision	recall	f1-score	support
Besni	0.82	0.85	0.84	39
Kecimen	0.84	0.82	0.83	39
accuracy			0.83	78
macro avg	0.83	0.83	0.83	78
weighted avg	0.83	0.83	0.83	78

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.85	0.85	0.85	39
Kecimen	0.85	0.85	0.85	39
accuracy			0.85	78
macro avg	0.85	0.85	0.85	78
weighted avg	0.85	0.85	0.85	78

	precision	recall	f1-score	support
Besni	0.86	0.82	0.84	39
Kecimen	0.83	0.87	0.85	39
accuracy			0.85	78
macro avg	0.85	0.85	0.85	78
weighted avg	0.85	0.85	0.85	78

Figure 3: Classification reports for KNN with two features. The results are shown in order for the metrics: euclidean, manhattan, minkowski, canberra, chebyshev, cosine

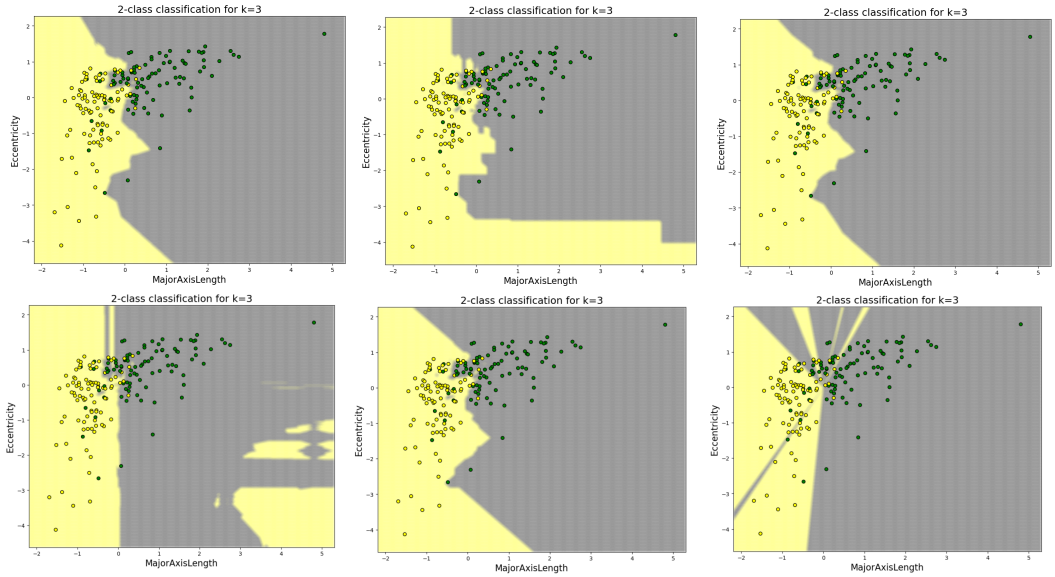


Figure 4: Decision boundaries for KNN with two features. The results are presented in order for the metrics: euclidean, manhattan, minkowski, canberra, chebyshev, cosine

	precision	recall	f1-score	support
Besni	0.86	0.92	0.89	39
Kecimen	0.92	0.85	0.88	39
accuracy			0.88	78
macro avg	0.89	0.88	0.88	78
weighted avg	0.89	0.88	0.88	78

	precision	recall	f1-score	support
Besni	0.86	0.92	0.89	39
Kecimen	0.92	0.85	0.88	39
accuracy			0.88	78
macro avg	0.89	0.88	0.88	78
weighted avg	0.89	0.88	0.88	78

	precision	recall	f1-score	support
Besni	0.87	0.85	0.86	39
Kecimen	0.85	0.87	0.86	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.84	0.92	0.88	39
Kecimen	0.91	0.82	0.86	39
accuracy			0.87	78
macro avg	0.88	0.87	0.87	78
weighted avg	0.88	0.87	0.87	78

	precision	recall	f1-score	support
Besni	0.80	0.85	0.83	39
Kecimen	0.84	0.79	0.82	39
accuracy			0.82	78
macro avg	0.82	0.82	0.82	78
weighted avg	0.82	0.82	0.82	78

	precision	recall	f1-score	support
Besni	0.86	0.92	0.89	39
Kecimen	0.92	0.85	0.88	39
accuracy			0.88	78
macro avg	0.89	0.88	0.88	78
weighted avg	0.89	0.88	0.88	78

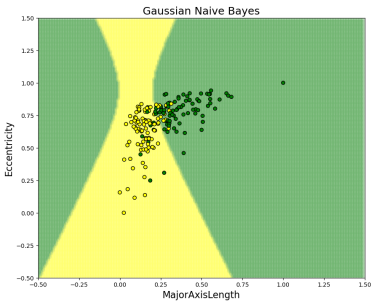
Figure 5: Classification reports for KNN with all features. The results are shown in order for the metrics: euclidean, manhattan, minkowski, canberra, chebyshev, cosine

4.2. Gaussian Naive Bayes

	precision	recall	f1-score	support
Besni	0.91	0.77	0.83	39
Kecimen	0.80	0.92	0.86	39
accuracy			0.85	78
macro avg	0.85	0.85	0.85	78
weighted avg	0.85	0.85	0.85	78

	precision	recall	f1-score	support
Besni	0.89	0.79	0.84	39
Kecimen	0.81	0.90	0.85	39
accuracy			0.85	78
macro avg	0.85	0.85	0.85	78
weighted avg	0.85	0.85	0.85	78

Figure 6: Classification reports for Gaussian Naive Bayes in order using PCA and all features



	precision	recall	f1-score	support
Besni	0.82	0.92	0.87	39
Kecimen	0.91	0.79	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

Figure 7: Classification report for Gaussian Naive Bayes with two features and decision boundaries of this model

4.3. SVM

We created SVM models for different kernels for specific parameters. These nuclei are: linear, polynomial with degree of 7, rbf with a gamma of 2, laplacian with a gamma of 2 and sigmoid with a gamma of 1.

	precision	recall	f1-score	support
Besni	0.78	1.00	0.88	39
Kecimen	1.00	0.72	0.84	39
accuracy			0.86	78
macro avg	0.89	0.86	0.86	78
weighted avg	0.89	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.78	1.00	0.88	39
Kecimen	1.00	0.72	0.84	39
accuracy			0.86	78
macro avg	0.89	0.86	0.86	78
weighted avg	0.89	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.79	0.85	0.81	39
Kecimen	0.83	0.77	0.80	39
accuracy			0.81	78
macro avg	0.81	0.81	0.81	78
weighted avg	0.81	0.81	0.81	78

	precision	recall	f1-score	support
Besni	0.62	1.00	0.76	39
Kecimen	1.00	0.38	0.56	39
accuracy			0.69	78
macro avg	0.81	0.69	0.66	78
weighted avg	0.81	0.69	0.66	78

	precision	recall	f1-score	support
Besni	0.86	0.92	0.89	39
Kecimen	0.92	0.85	0.88	39
accuracy			0.88	78
macro avg	0.89	0.88	0.88	78
weighted avg	0.89	0.88	0.88	78

Figure 8: Classification reports for SVM with PCA. The results are shown in order for the kernels: linear, poly, rbf, laplacian and sigmoid

	precision	recall	f1-score	support
Besni	0.77	0.95	0.85	39
Kecimen	0.93	0.72	0.81	39
accuracy			0.83	78
macro avg	0.85	0.83	0.83	78
weighted avg	0.85	0.83	0.83	78

	precision	recall	f1-score	support
Besni	0.65	0.90	0.75	39
Kecimen	0.83	0.51	0.63	39
accuracy			0.71	78
macro avg	0.74	0.71	0.69	78
weighted avg	0.74	0.71	0.69	78

	precision	recall	f1-score	support
Besni	0.77	0.95	0.85	39
Kecimen	0.93	0.72	0.81	39
accuracy			0.83	78
macro avg	0.85	0.83	0.83	78
weighted avg	0.85	0.83	0.83	78

	precision	recall	f1-score	support
Besni	0.80	0.95	0.87	39
Kecimen	0.94	0.77	0.85	39
accuracy			0.86	78
macro avg	0.87	0.86	0.86	78
weighted avg	0.87	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.80	0.95	0.87	39
Kecimen	0.94	0.77	0.85	39
accuracy			0.86	78
macro avg	0.87	0.86	0.86	78
weighted avg	0.87	0.86	0.86	78

Figure 9: Classification reports for SVM with two features. The results are shown in order for the kernels: linear, poly, rbf, laplacian and sigmoid

	precision	recall	f1-score	support
Besni	0.78	1.00	0.88	39
Kecimen	1.00	0.72	0.84	39
accuracy			0.86	78
macro avg	0.89	0.86	0.86	78
weighted avg	0.89	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.83	0.87	0.85	39
Kecimen	0.86	0.82	0.84	39
accuracy			0.85	78
macro avg	0.85	0.85	0.85	78
weighted avg	0.85	0.85	0.85	78

	precision	recall	f1-score	support
Besni	0.77	0.92	0.84	39
Kecimen	0.90	0.72	0.80	39
accuracy			0.82	78
macro avg	0.83	0.82	0.82	78
weighted avg	0.83	0.82	0.82	78

	precision	recall	f1-score	support
Besni	0.83	0.90	0.86	39
Kecimen	0.89	0.82	0.85	39
accuracy			0.86	78
macro avg	0.86	0.86	0.86	78
weighted avg	0.86	0.86	0.86	78

	precision	recall	f1-score	support
Besni	0.84	0.95	0.89	39
Kecimen	0.94	0.82	0.88	39
accuracy			0.88	78
macro avg	0.89	0.88	0.88	78
weighted avg	0.89	0.88	0.88	78

Figure 10: Classification reports for SVM with all features. The results are shown in order for the kernels: linear, poly, rbf, laplacian, sigmoid

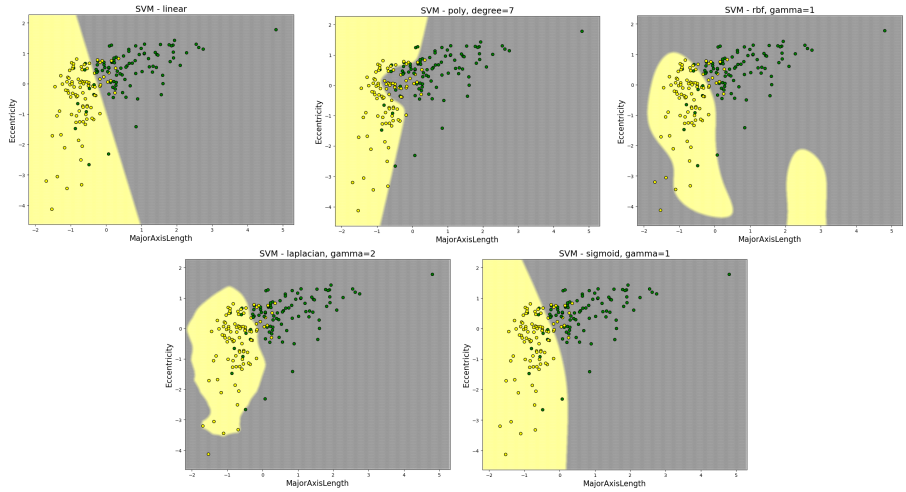


Figure 11: Decision boundaries for SVM with two features

5. Conclusion

After analyzing our results for three classifiers, we conclude that using the PCA technique to reduce the dimensionality from 7 to 2 features supported performance of our models, also achieving high accuracies, comparable to the results of models built on all features. After analyzing the correlation of our data, we were able to find two features for which the models had similar accuracy compared to the PCA-based models, these features are: MajorAxisLength and Eccentricity. The accuracy for the KNN classifiers with and without PCA are very similar, ranging from 82% to 88% depending on the metric. In the case of Gaussian Naive Bayes classifiers the accuracy result obtained using PCA and using 7 features gave the same value of 85% , which only confirms the fact that the reduction in dimensions didn't contribute to the loss of significant information. The last type of classifier, that was analyzed was SVM. After analyzing for different kernels, the sigmoid kernel turned out to be the best, which in models with and without PCA indicated the best accuracy of 88%.

References

[1] K. Thirunavukkarasu, A. S. Singh, P. Rai, S. Gupta, Classification of iris dataset using classification based knn algorithm in supervised learning, 2018 4th International Conference on Computing Communication and Automation (ICCCA) (2018) 1–4.

[2] G. De Magistris, R. Caprari, G. Castro, S. Russo, L. Iocchi, D. Nardi, C. Napoli, Vision-based holistic scene understanding for context-aware human-robot interaction 13196 LNAI (2022) 310 – 325. doi:10.1007/978-3-031-08421-8_21.

[3] V. Amaresh, R. R. Singh, R. Kamal, A. Kulkarni, Linear regression models based housing price forecasting, 2022 International Conference on Industry 4.0 Technology (I4Tech) (2022) 1–5.

[4] W. Feng, J. Zhang, Y. Chen, Z. Qin, Y. Zhang, M. Ahmad, M. Woźniak, Exploiting robust quadratic polynomial hyperchaotic map and pixel fusion strategy for efficient image encryption, Expert Systems with Applications 246 (2024) 123190.

[5] M. Wozniak, C. Napoli, E. Tramontana, G. Capizzi, G. Lo Sciuto, R. K. Nowicki, J. T. Starczewski, A multiscale image compressor with rbfnn and discrete wavelet decomposition, volume 2015-September, 2015. doi:10.1109/IJCNN.2015.7280461.

[6] G. Capizzi, C. Napoli, S. Russo, M. Woźniak, Lessening stress and anxiety-related behaviors by means of ai-driven drones for aromatherapy, volume 2594, 2020, pp. 7 – 12.

[7] N. Brandizzi, S. Russo, R. Brociek, A. Wajda, First studies to apply the theory of mind theory to green and smart mobility by using gaussian area clustering, volume 3118, 2021, pp. 71 – 76.

[8] V. Ponzi, S. Russo, A. Wajda, R. Brociek, C. Napoli, Analysis pre and post covid-19 pandemic rorschach test data of using em algorithms and gmm models, volume 3360, 2022, pp. 55 – 63.

[9] D. P. Hapsari, I. Utoyo, S. W. Purnami, Fractional gradient descent optimizer for linear classifier support vector machine, 2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE) (2020) 1–5.

[10] G. Capizzi, G. L. Sciuto, C. Napoli, R. Shikler, M. Wozniak, Optimizing the organic solar cell manufacturing process by means of afm measurements and neural networks, Energies 11 (2018). doi:10.3390/en11051221.

[11] M. Woźniak, M. Wieczorek, J. Siłka, Bilstm deep neural network model for imbalanced medical data of iot systems, Future Generation Computer Systems 141 (2023) 489–499.

[12] A. Sikora, A. Zielonka, M. F. Ijaz, M. Woźniak, Digital twin heuristic positioning of insulation in multimodal electric systems, IEEE Transactions on Consumer Electronics (2024).

[13] G. Capizzi, C. Napoli, L. Paternò, An innovative hybrid neuro-wavelet method for reconstruction of missing data in astronomical photometric surveys 7267 LNAI (2012) 21 – 29. doi:10.1007/978-3-642-29347-4_3.

[14] Q. Ke, X. Jing, M. Woźniak, S. Xu, Y. Liang, J. Zheng, Apgvae: Adaptive disentangled representation learning with the graph-based structure information, Information Sciences 657 (2024) 119903.

[15] N. A. Al-Sammarraie, Y. M. H. Al-Mayali, Y. A. B. El-Ebiary, Classification and diagnosis using back propagation artificial neural networks (ann), 2018 International Conference on Smart Computing and Electronic Enterprise (ICSCEE) (2018) 1–5.

[16] D. Gangadia, Activation functions: experimentation and comparison, 2021 6th International Conference for Convergence in Technology (I2CT) (2021) 1–6.

[17] P. H. Progga, M. J. Rahman, S. Biswas, M. S. Ahmed, D. M. Farid, K-nearest neighbour classifier for big data mining based on informative instances, 2023 IEEE 8th International Conference for Convergence in Technology (I2CT) (2023) 1–7.

[18] W. Dong, M. Woźniak, J. Wu, W. Li, Z. Bai, Denoising aggregation of graph neural networks by using principal component analysis, IEEE Transactions on Industrial Informatics 19 (2022) 2385–2394.