Faculty of Engineering & Technology

Electrical & Computer Engineering Department

Intelligent Systems Laboratory ENCS5141

Case Study #1

Data Cleaning and Feature Engineering for Diabetes Dataset

Prepared by:

Hiba Jaouni 1201154

Instructor: Dr. Mohammad Jubran

Section: 3

Date: Nov 30th 2024

Abstract

This report presents a comparative analysis of three machine learning models—Random Forest, Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP)—trained on both dimensionality-reduced and non-reduced datasets. The primary objective is to evaluate the models based on their classification performance metrics, including accuracy, precision, recall, and F1-score, while also assessing their computational efficiency in terms of training and testing times. Dimensionality reduction using Principal Component Analysis (PCA) was applied to examine its effect on model performance. The results indicate that the Random Forest model, when trained without dimensionality reduction, outperforms the other models in terms of both predictive accuracy and computational efficiency. Despite the use of PCA, the performance improvements were marginal, suggesting that dimensionality reduction may not always be beneficial, particularly when the number of features is relatively manageable. This analysis concludes that the Random Forest model without PCA offers the best trade-off between prediction accuracy and computational resources for this dataset.

Table of Contents

Table of Figures

Table of Tables

Introduction

Diabetes is a chronic disease that affects millions of individuals worldwide and poses a substantial burden on both patients and healthcare systems. Understanding the factors that contribute to the various forms of diabetes is essential for early diagnosis, targeted treatment, and effective prevention strategies. This study utilizes a comprehensive dataset encompassing 13 distinct types of diabetes, including Type 1 Diabetes, Type 2 Diabetes, Gestational Diabetes, and rare conditions such as Wolfram Syndrome and Steroid-Induced Diabetes. By examining a range of patient attributes—including blood pressure, age, body mass index (BMI), and smoking habits and more—this research aims to identify patterns and correlations that could assist in predicting the type of diabetes an individual may have. The ability to forecast diabetes types based on these attributes is crucial for personalized medical interventions and for improving patient outcomes.

Machine learning has emerged as a powerful tool for predictive modelling and disease diagnosis within the healthcare sector. This study employs three prominent machine learning models:

-Support Vector Machines (SVM): A supervised learning model that excels in classification by identifying an optimal hyperplane to distinguish between different classes of data.

- Multilayer Perceptron (MLP): A type of artificial neural network capable of capturing non-linear relationships between input features and output classes.

-Random Forest (RF): An ensemble learning method that utilizes multiple decision trees to enhance prediction accuracy and reduce the risk of overfitting.

Each of these models offers distinct advantages when evaluating complex datasets, such as the one used in this study, which encompasses multiple features and diverse classifications of diabetes.

This study aims to compare SVM, MLP, and RF in predicting 13 diabetes types based on patient attributes, exploring model performance, key factor influences, and the reliability of machine learning for advanced diagnostics.

Procedure and Discussion

This section presents the proposed study, which encompasses several essential components, including data visualization and preprocessing, feature engineering, and a comparative analysis of various classification techniques. Specifically, this investigation examines the efficacy of Random Forest (RF), Support Vector Machine (SVM), and Multilayer Perceptron (MLP) algorithms applied to the Diabetes dataset.

Data Exploration and Preprocessing

The dataset comprises 70,000 rows and 34 columns, with each column representing a distinct medical attribute or feature of the patients. These features are a combination of categorical (object) and numerical (integer) data types. Importantly, the target column indicates the type of diabetes, while the other columns detail various patient attributes, including genetic markers, insulin levels, blood pressure, BMI, age, and more. Furthermore, the dataset is complete, containing no missing values or empty rows.

The frequency distribution of each categorical feature (21 categoral feature) was visualized using histograms. These plots revealed that the dataset is well-balanced, with approximately equal representation across categories for all features.

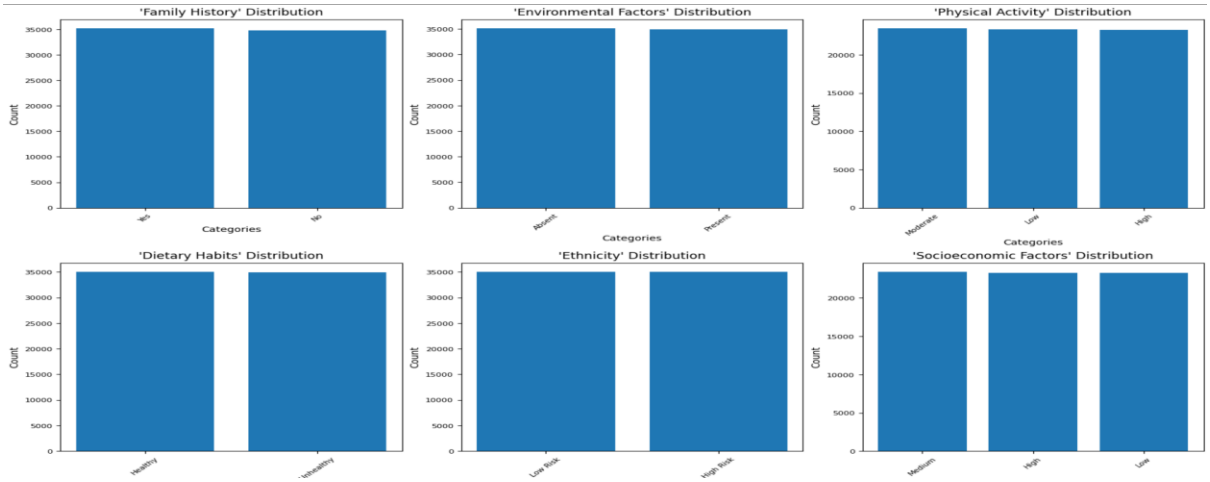

Figure 2.1: Sample of the Distribution for the Categoral Features

The target column contains 13 distinct values representing various types of diabetes. Other categorical features exhibit a smaller range of distinct values, typically between 2 and 4. For example, features like *Genetic Markers*, *Autoantibodies*, and *Family History* have two distinct values, while *Physical Activity*, *Socioeconomic Factors*, and *Alcohol Consumption* have three. The *Urine Test* feature is an exception, with four distinct values.

To prepare the data for machine learning models, the target column was encoded using label encoding, assigning each diabetes type an integer value between 0 and 12. For the remaining categorical features, one-hot encoding was applied, as these features have limited distinct values. This method transforms each distinct value into a separate binary column, effectively representing the presence or absence of that category.

After encoding, the number of columns increased from 34 to 59 due to the additional binary columns generated by one-hot encoding.

The following visualizations were created to explore the distribution of the numerical features in the dataset. Histograms with Kernel Density Estimates (KDE) were plotted for each numerical feature, providing a comprehensive view of their distribution. These plots allow us to understand the central tendency (mean, median, mode), variability (variance, standard deviation), and overall shape (skewness and kurtosis) of the data.

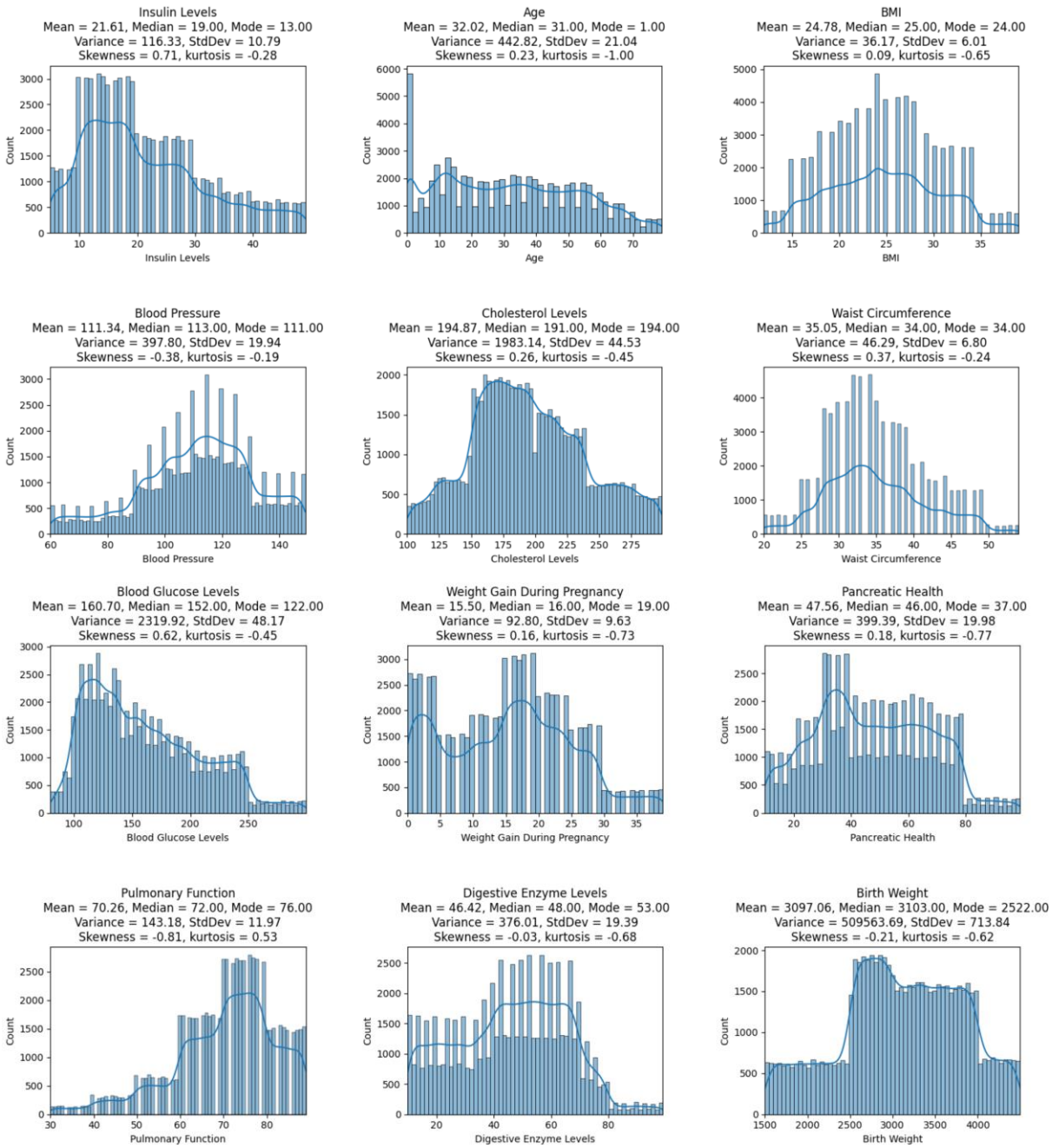

Figure 2.2: Distribution of the Numerical Features

Some features, such as BMI, Waist Circumference, Cholesterol Levels, and Blood Pressure, appear to be almost normally distributed, as indicated by the closeness of their mean, median,

and mode values. These features exhibit symmetrical distributions without significant skewness, suggesting that they are well-behaved and do not have extreme variations.
On the other hand, certain features like Pulmonary Function, Blood Glucose Levels, and Insulin Levels show evidence of skewness in their histograms. This skewness could be indicative of the presence of outliers or non-normal distributions, which may require further investigation. These skewed distributions suggest that the data may not follow a perfect bell curve and could have a higher frequency of extreme values in one tail of the distribution.

To further examine the possibility of outliers, boxplots were created for each numerical feature.

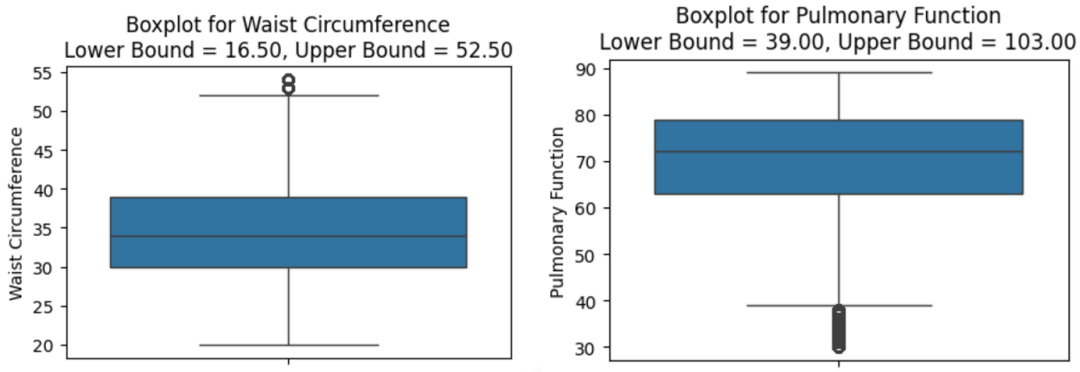

Figure 2.3: Outliers using Boxplot for Waist Circumference and Pulmonary Function

The boxplots confirm that Waist Circumference (left side of Figure 1.2) and Pulmonary (right side of Figure 1.2) Function contain potential outliers, as indicated by data points lying outside the typical range of values.

To determine whether these data points should be classified as true outliers, it is important to revisit the distribution measures, such as skewness and kurtosis. A deeper statistical analysis can help make more informed decisions about whether these points should be removed, transformed, or handled differently during the modeling process.

To address the potential outliers in the dataset, we applied the Z-score test, specifically focusing on the uniformly distributed features like Waist Circumference. The Z-score test helps in identifying data points that deviate significantly from the mean, indicating potential outliers. The threshold was set to 2.5 for higher sensitivity, and this revealed potential outliers in several features, including Insulin Level, Blood Pressure, Waist Circumference, Blood Glucose Level, Pancreatic Health, Pulmonary Function, and Digestive Enzyme Level. When the threshold was increased to 3 (lower sensitivity), only Pulmonary Function retained its outliers, suggesting that many of the previously identified outliers were not extreme enough to be classified as true outliers under a less sensitive approach.

Based on these findings, we decided to focus on the potential outliers in Waist Circumference and Pulmonary Function, while features like Pancreatic Health, Digestive Enzyme Level, Insulin Level, and Blood Pressure were excluded from further analysis. These features did not present values that were significantly far from the threshold, and they no longer showed

as outliers when the sensitivity was reduced. Blood Glucose Level, despite some extreme values, was not treated as containing outliers. From a clinical perspective, blood glucose levels higher than 126 mg/dL indicate diabetes, and in severe cases, values can go up to 400 mg/dL, so extreme values are expected within this context and are not considered anomalies.

Using the Z-score test on Waist Circumference, we identified 522 potential outliers (0.75% of the dataset). However, lowering the sensitivity (threshold set to 3) showed no outliers, suggesting the values may not be as extreme.

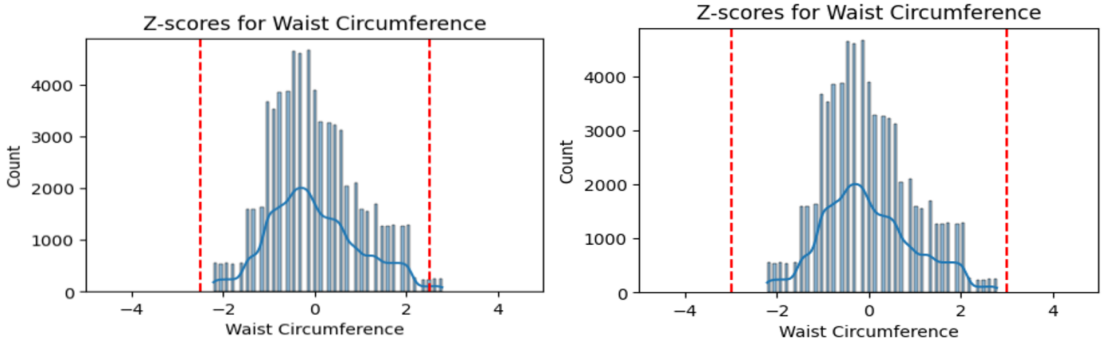

Figure 2.4: Waist Circumference when Threshold is 2.5 and 3

The left panel of the figure illustrates the Waist Circumference measurement at a threshold value of 2.5, while the right panel presents the corresponding measurement at a threshold value of 3.

The correlation between Waist Circumference and Age (0.7336) indicates that Age could be influencing the results, explaining some of the flagged values.

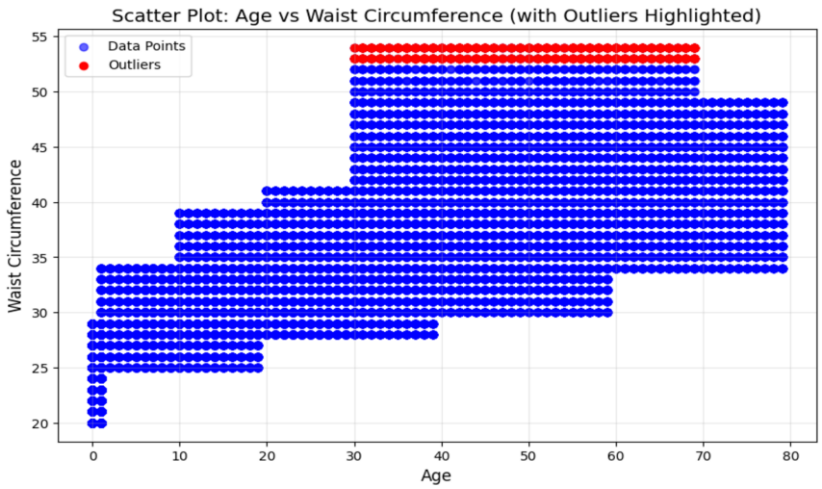

Figure 2.5: Waist Circumference vs Age

The scatter shows that outliers are part of a clear trend, indicating they might not be true outliers.

For the Pulmonary Function feature, the data is negatively skewed, and outliers were identified using the Z-score and Boxplot methods. After consulting a domain expert, we found values below 50% to be outliers. Replacing these outliers, especially those below the

5th percentile, with the median (72.00) improved the skewness from -0.81 to -0.36, ensuring a more accurate distribution. A total of 1,206 outliers (1.72%) were handled in this process.

Since the data is slightly skewed, we replaced the outliers with the median value (72.00) rather than the mean to prevent distortion of the central tendency.

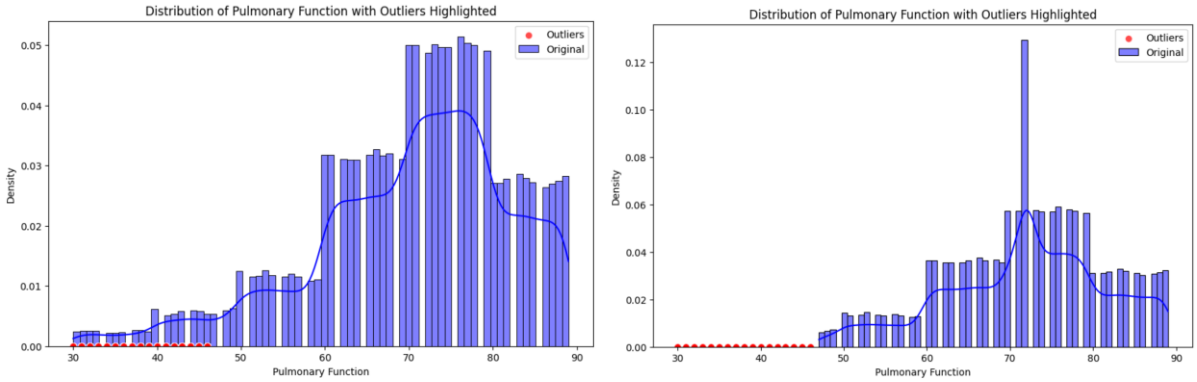

Figure 2.6: Pulmonary Function Distribution Before and After Handling Outliers

The left panel of the figure delineates the pulmonary function before handling outliers, whereas the right panel illustrates the corresponding data after handling these outliers.

The analysis reveals a clear distinction in feature relevance for the machine learning task. Features such as **Blood Glucose Levels** (0.652), **Neurological Assessments** (0.238), and **Blood Pressure** (0.050) show moderate correlations with the target variable, indicating their influence on the outcome. Other features like **Waist Circumference Z-Score** (0.046) and **Glucose Tolerance Test_Normal** (0.007) exhibit minimal correlation, suggesting that they may not significantly impact the target prediction.

When considering **Information Gain**, features such as **Age** (1.179), **Blood Pressure** (0.901), and **Blood Glucose Levels** (0.889) stand out, signifying that these attributes are most valuable in terms of the model's ability to distinguish between different classes. **Weight Gain During Pregnancy** (0.882) and **Waist Circumference Z-Score** (0.839) also provide noteworthy information gain, implying their potential relevance in the prediction process.

In terms of **Variance**, features like **Birth Weight** (509,563.69), **Blood Glucose Levels** (2,319.92), and **Cholesterol Levels** (1,983.14) exhibit substantial variance, meaning they vary significantly across instances and could be crucial for prediction. On the other hand, binary or categorical features such as **Ethnicity** and **Smoking Status** show lower variance, suggesting these features might offer limited predictive power, though their importance could depend on the specific model and task.

Before applying dimensionality reduction, we will normalize the numerical features using Z-score normalization. The numerical features to normalize are: **Insulin Levels**, **Age**, **BMI**, **Blood Pressure**, **Cholesterol Levels**, **Blood Glucose Levels**, **Weight Gain During Pregnancy**, **Pancreatic Health**, **Pulmonary Function**, **Neurological Assessments**, **Digestive Enzyme Levels**, and **Birth Weight**.

After normalization, the dataset is ready for machine learning models, and dimensionality reduction can be applied for optimization.

Training and Evaluation

Random Forest

The Random Forest model was tested with various configurations of n_estimators and max_depth. The n_estimators parameter controls the number of trees in the forest, and we observed that performance improved up to 40 trees, after which the increase in accuracy and other metrics became negligible, while training and testing time also grew significantly. Therefore, 40 trees were found to offer the best balance between performance and computational efficiency. Additionally, setting max_depth=None allowed the trees to grow to their full potential, capturing more complex patterns in the data without overfitting. This was particularly effective because a limited depth (e.g., 10 or 20) did not show significant improvement in model performance.

Table2.1: Model Performance Summary for Random Forest

n_estimators	max_depth	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Testing Time (s)
10	NaN	0.869286	0.869499	0.869286	0.868484	1.007841	0.050842
10	10	0.854619	0.868566	0.854619	0.853097	0.538933	0.041303
10	20	0.871571	0.872909	0.871571	0.870025	0.96427	0.050899
15	NaN	0.88081	0.882228	0.88081	0.879824	1.499886	0.095361
15	10	0.861952	0.87544	0.861952	0.860599	0.91202	0.074535
15	20	0.880571	0.882746	0.880571	0.879047	2.397734	0.126037
20	NaN	0.884286	0.885977	0.884286	0.883199	2.266775	0.112185
20	10	0.86481	0.87651	0.86481	0.863317	1.0439	0.052003
20	20	0.883333	0.885514	0.883333	0.881806	1.895096	0.091923
30	NaN	0.889048	0.891309	0.889048	0.887833	2.900978	0.132807
30	10	0.868286	0.878896	0.868286	0.86682	1.504421	0.071275
30	20	0.889286	0.892087	0.889286	0.887875	4.332141	0.131789
40	NaN	0.891571	0.894392	0.891571	0.890369	4.243807	0.162273
40	10	0.873619	0.882863	0.873619	0.872121	2.218915	0.117802
40	20	0.890762	0.89378	0.890762	0.889368	4.77959	0.315068
50	NaN	0.892095	0.895167	0.892095	0.890873	5.076006	0.213624
50	10	0.875095	0.884104	0.875095	0.873571	2.507278	0.112092
50	20	0.892571	0.89598	0.892571	0.891241	6.100059	0.404794

The optimal model configuration was achieved with n_estimators=40 and max_depth=None. This setup ensured high accuracy, precision, recall, and F1-score, while avoiding the excessive complexity and longer training times associated with increasing the number of trees beyond 40. Furthermore, using max_depth=None enabled the model to capture all relevant patterns without limiting tree growth, resulting in a more effective model for this dataset.

This configuration strikes a good balance between model complexity, accuracy, and training efficiency.
To reduce dimensionality and optimize model performance, the first step is to apply Principal Component Analysis (PCA) on the dataset. The goal is to identify the best number of components (n_components) that retain the most significant variance in the data while improving model performance.

Table 2.2: Summary of performance across different n_components

n_components	Accuracy	Precision	Recall	F1 Score	Training Time (seconds)
1	0.3351	0.3354	0.3351	0.3352	9.06
2	0.4657	0.4613	0.4657	0.4631	6.48
4	0.5969	0.596	0.5969	0.5958	12.23
8	0.6325	0.6311	0.6325	0.63	13.17
16	0.627	0.6252	0.627	0.6244	25.13
30	0.6931	0.6925	0.6931	0.6914	32.07
32	0.7096	0.7089	0.7096	0.7079	32.06
40	0.7447	0.7467	0.7447	0.7447	38.12

After testing different n_components, the best performance was achieved with **n_components = 40**, which resulted in an accuracy of **0.7447**, along with high precision, recall, and F1 score. This reduced dataset will serve as the input for training three different models: Random Forest (RF), Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP).

The Random Forest model is trained on a dataset reduced to 40 components using PCA, based on the results from dimensionality reduction. Hyperparameter tuning is also performed, optimizing both n_estimators and max_depth to enhance model performance. These results will then be compared with the performance of the Random Forest model trained on the original, unmodified dataset, without any dimensionality reduction. This comparison will help assess the impact of PCA on model accuracy, efficiency, and overall performance.

Table 2.3: Model Performance Summary for Random Forest with PCA

n_estimators	max_depth	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Testing Time (s)
10	NaN	0.678714	0.680807	0.678714	0.678887	11.75837	0.047348
10	10	0.692238	0.695084	0.692238	0.690959	6.841295	0.049184
10	20	0.687286	0.687664	0.687286	0.687081	9.303912	0.047718
15	NaN	0.709238	0.710898	0.709238	0.709508	14.21942	0.07879
15	10	0.708905	0.712298	0.708905	0.70805	11.20966	0.0383
15	20	0.71	0.710992	0.71	0.70976	14.04914	0.068442
20	NaN	0.722571	0.724223	0.722571	0.722756	18.17083	0.088114

20	10	0.717381	0.721548	0.717381	0.716709	13.99484	0.047697
20	20	0.72619	0.726993	0.72619	0.725653	20.12334	0.09857
30	NaN	0.736619	0.7384	0.736619	0.736738	28.06316	0.129382
30	10	0.72481	0.729223	0.72481	0.724083	20.27851	0.067878
30	20	0.739048	0.740192	0.739048	0.738362	27.62955	0.129834
40	NaN	0.744667	0.746718	0.744667	0.744702	38.63631	0.171062
40	10	0.731238	0.735378	0.731238	0.730293	28.03302	0.088634
40	20	0.748714	0.750393	0.748714	0.748065	37.16106	0.295868
50	NaN	0.749857	0.751748	0.749857	0.749796	46.78443	0.220928
50	10	0.732238	0.736877	0.732238	0.731253	34.62108	0.193129
50	20	0.751524	0.75329	0.751524	0.750753	46.41859	0.201281

Comparing the best Random Forest models trained with and without dimensionality reduction reveals that the model trained without dimensionality reduction performed significantly better across all performance metrics, including accuracy, precision, recall, and F1-score. The model without dimensionality reduction achieved an accuracy of **0.8916**, precision of **0.8944**, recall of **0.8916**, and F1-score of **0.8904**, using 40 estimators and no maximum depth constraint. In contrast, the best model with dimensionality reduction (40 components) attained an accuracy of **0.7515**, precision of **0.7533**, recall of **0.7515**, and F1-score of **0.7508**, using 50 estimators and a maximum depth of 20.

It is noteworthy that increasing the number of components in the PCA-transformed dataset improved the performance metrics. However, overall, the performance of the model without dimensionality reduction was superior. This discrepancy could be attributed to the potential loss of informative features during dimensionality reduction, which may play a significant role in evaluating and predicting the target variable.

The advantages of dimensionality reduction might become more apparent when working with datasets that have a very high number of features (e.g., in the hundreds or thousands), where PCA can help reduce computational complexity and mitigate overfitting.

SVM

The SVM model was trained using both the original dataset (without dimensionality reduction) and the reduced dataset with 40 principal components obtained through PCA. For both cases, hyperparameter tuning was conducted using a parameter grid that varied the C values between 0.1, 1, and 10, and the kernel type between linear and rbf.

Table 2.4 :Model Performance Summary for SVM

C	Kernel	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Testing Time (s)
0.1	linear	0.7633	0.7633	0.7633	0.7631	20.41	24.01
0.1	rbf	0.7796	0.7812	0.7796	0.7791	43.39	62.27
1	linear	0.7626	0.7625	0.7626	0.7623	31.94	24.24

1	rbf	0.7974	0.7987	0.7974	0.7967	26.27	47.21
10	linear	0.7629	0.7629	0.7629	0.7627	100.58	22.45
10	rbf	0.793	0.7928	0.793	0.7926	43.62	47.98

For the dataset without dimensionality reduction, the RBF kernel consistently outperformed the linear kernel in accuracy, precision, recall, and F1-score. For example, with C=1, the RBF kernel achieved 79.74% accuracy compared to 76.26% for the linear kernel. The RBF kernel also showed slight performance improvements with higher C values, whereas the linear kernel remained relatively unchanged.

However, increasing C to 10 significantly increased the training time for the linear kernel (100.58 seconds compared to 20.41 seconds at C=0.1). The RBF kernel required more testing time than the linear kernel, particularly at lower C values (e.g., 62.27 seconds vs. 24.01 seconds at C=0.1). While the RBF kernel offers better performance, it comes at a higher computational cost.

Table 2.5: Model Performance Summary for SVM with PCA

C	Kernel	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Testing Time (s)
0.1	linear	0.763381	0.763300	0.763381	0.763103	17.542277	19.967121
0.1	rbf	0.780667	0.782424	0.780667	0.780140	43.790523	53.845754
1.0	linear	0.762524	0.762454	0.762524	0.762266	29.212325	17.696251
1.0	rbf	0.798190	0.799374	0.798190	0.797456	27.740969	41.883779
10.0	linear	0.762905	0.762882	0.762905	0.762667	98.598140	16.788765
10.0	rbf	0.792190	0.791959	0.792190	0.791788	34.186793	41.995178

For the dataset with dimensionality reduction, the RBF kernel consistently outperformed the linear kernel across all performance metrics. For example, with C=1, the RBF kernel achieved 79.82% accuracy compared to 76.25% for the linear kernel. Similar trends were observed for other C values, showcasing the RBF kernel's ability to handle complex patterns even after dimensionality reduction.

However, the performance metrics for the two models—those trained with and without dimensionality reduction—are almost identical, with no noticeable improvements. Even the training times showed minimal differences, indicating that applying PCA did not offer significant benefits in this case. This could be due to the relatively small number of features, where PCA may not have had a substantial impact. Additionally, PCA might have removed some features that were important for model performance, which could explain the lack of improvement compared to the model trained without dimensionality reduction.

MLP

The Multi-Layer Perceptron (MLP) model was trained on the dataset without dimensionality reduction, and hyperparameter tuning was performed for the hidden_layer_sizes and activation functions.

Table 2.6: Model Performance Summary for MLP

Hidden Layers	Activation	Accuracy	Precision	Recall	F1 - Score	Training Time (s)	Testing Time (s)
(50,)	tanh	0.843286	0.845033	0.843286	0.842482	56.7106	0.047811
(50,)	relu	0.849952	0.851597	0.849952	0.849183	49.66036	0.026434
(100,)	tanh	0.834	0.834078	0.834	0.833539	81.62374	0.070298
(100,)	relu	0.842238	0.842937	0.842238	0.841471	62.75547	0.029732

The results revealed variations in performance metrics based on the chosen configuration. For a hidden layer size of (50,) with the Tanh activation function, the model achieved an accuracy of 84.33%, with a training time of 56.71 seconds. In contrast, the same hidden layer size combined with the ReLU activation function delivered the best overall performance, achieving an accuracy of 84.99% while reducing the training time to 49.66 seconds.

When using a larger hidden layer size of (100,) with the Tanh activation function, the model's accuracy dropped slightly to 83.40%, and the training time increased significantly to 81.62 seconds, indicating reduced efficiency. Similarly, with a hidden layer size of (100,) and the ReLU activation function, the model achieved an accuracy of 84.22%, slightly lower than its smaller counterpart, with a training time of 62.76 seconds.

Overall, the configuration with a smaller hidden layer size of (50,) and ReLU activation provided the best balance between accuracy and computational efficiency.

Table 2.7: Model Performance Summary for MPL with PCA

Hidden Layer	Activation	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Testing Time (s)
(50,)	tanh	0.841	0.8425	0.841	0.8401	105.55	0.154
(50,)	relu	0.8528	0.8539	0.8528	0.8517	62.59	0.024
(100,)	tanh	0.8269	0.8271	0.8269	0.8263	146.74	0.19
(100,)	relu	0.843	0.8442	0.843	0.8421	80.96	0.061

After applying dimensionality reduction using PCA, the **MLP model** with different configurations of hidden layer sizes and activation functions was evaluated. The best performance was observed with the **ReLU activation function** and a hidden layer size of 50, achieving an accuracy of 85.28% and an F1-score of 85.17%. This configuration also demonstrated the shortest training time at 62.59 seconds and a minimal testing time of 0.02 seconds, making it both the most efficient and effective model.

The MLP with dimensionality reduction outperformed MLP without dimensionality reduction.

Summary of the RF, SVM, and MLP

Table 2.8: Summary of RF, SVM and MLP

Model	PCA	Hyper-parameters	Accuracy	Precision	Recall	F1-Score	Training Time (s)	Testing Time (s)
RF	With	50,20	0.7515	0.7533	0.7515	0.7508	46.42	0.201
RF	Without	40	0.8916	0.8944	0.8916	0.8904	4.24	0.162
SVM	With	C =1.0,kernel =rbf	0.7982	0.7994	0.7982	0.7975	27.74	41.88
SVM	Without	C =1.0,kernel =rbf	0.7974	0.7987	0.7974	0.7967	26.27	47.21
MLP	With	(50,),relu	0.8528	0.8539	0.8528	0.8517	62.59	0.024
MLP	Without	(50,),relu	0.85	0.8516	0.85	0.8492	49.66	0.026

The performance of different models with and without dimensionality reduction (PCA) has been evaluated, with a focus on key metrics like accuracy, precision, recall, F1-score, training time, and testing time.

For **Random Forest (RF)**, the model without PCA showed significantly better performance with an accuracy of 89.16%, precision of 89.44%, and a F1-score of 89.04%. It also had the shortest training time of 4.24 seconds and testing time of 0.16 seconds, making it the most efficient option in terms of computational resources. In contrast, the RF model with PCA had lower accuracy (75.15%), precision (75.33%), and F1-score (75.08%), with a longer training time of 46.42 seconds and testing time of 0.201 seconds.

For the **Support Vector Machine (SVM)**, both models with and without PCA performed similarly. The SVM with PCA had an accuracy of 79.82%, while the model without PCA had an accuracy of 79.74%. While both configurations had similar precision, recall, and F1-score, the model with PCA had a slightly higher testing time (47.21 seconds) compared to the one without PCA (41.88 seconds).

The **Multi-Layer Perceptron (MLP)** models showed the best accuracy among all models. The MLP with PCA had an accuracy of 85.28% and an F1-score of 85.17%, while the model without PCA had a slightly lower accuracy of 85.0%. However, the MLP with PCA took longer to train (62.59 seconds) than the one without PCA (49.66 seconds), with both models having similar testing times.

Conclusion

In conclusion, this analysis compared the performance of three machine learning models—Random Forest, Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP)—on both dimensionality-reduced and non-reduced datasets. The models were assessed across multiple classification performance metrics, including accuracy, precision, recall, and F1-score, while also considering their computational efficiency in terms of training and testing time.

The results showed that the Random Forest model without dimensionality reduction consistently outperformed other models in terms of both classification performance and computational efficiency. It achieved the highest accuracy and F1-score while requiring significantly less training and testing time compared to other models. On the other hand, dimensionality reduction (PCA) did not provide significant improvements in performance and, in some cases, led to a drop in accuracy and increased computational costs.
Based on these findings, the Random Forest model without PCA offers the best balance between prediction accuracy and computational efficiency for this dataset, making it the most suitable choice for the given task.

Appendix

For Further Details about the code, you can visit this site:
https://colab.research.google.com/drive/1QAfrSC4HSPYxfFWrg8IKTs_kZGp7gLhV?usp=sharing