UNIVERSITÄT BERN

Bern University of Applied Sciences

- Project 1: Diabetes prediction -

Author	Daniel Monti & Daniel Bürgler & Camille Serquet
Date	December 13, 2021
Course	D101722-HS2021-0: Technology and Diabetes Management
Professor	Stavroula Mougiakakou, PhD
Assistant	Matthias Fontanellaz, MSc, PhD Fellow

Contents

1 Introduction **1**

2 Related work **1**

3 Methods **1**

3.1 Programming Language and used packages . . . 1
3.2 Decision Tree . . . 2
3.3 K-Nearest Neighbours . . . 2
3.4 Logistic Regression . . . 2
3.5 Naïve Bayes . . . 3
3.6 Support Vector Machines . . . 3
3.7 Stochastic Gradient Descent . . . 4
3.8 Cross Validation . . . 4
3.9 Tuning Algorithms . . . 5
3.10 Model Comparisons . . . 5

4 Data and Experiment setup **6**

4.1 Dataset . . . 6
4.2 Data preparation . . . 7
4.2.1 Data Scaling / Feature Scaling . . . 8
4.2.2 Imbalanced Data . . . 8

5 Results **9**

6 Discussion **10**

1 Introduction

Diabetes is a chronic metabolic disease associated with hyperglycemia. The chronic hyperglycemia results from defects in insulin secretion, insulin action, or both. Over time, diabetes leads to serious damage to the heart, eyes, kidney, nerves, and heart vessels. The prediction at an early stage of such a disease leads to improved treatment and mitigate the risk of long-term complications. Prediction Analysis uses statistics to create models that examine current and past data and predict potential future outcomes. When applied in health care, Prediction Analysis can help determine the patient's risk of developing certain disease such as diabetes. It is therefore used as a tool to support medical decision-making and eliminate human efforts by supporting automation with minimum flaws [reference]. In current medical diagnosis method, three types of errors are still to be deplored: false-negative, false-positive, and unclassified type.

Classification algorithms need to be evaluated with evaluation metrics. The resulting accuracy gives a good appreciation of the algorithm and helps to detect which one suits best for a medical application. In this paper, the Pima Indian Diabetes Database [10] is prepared and applied to classification algorithms such as Logistic Regression, Naïve Bayes, Stochastic Gradient Descent, K-Nearest Neighbors, Decision Tree, and Support Vector Machine. All algorithms will be reviewed. Finally, a comparison between all algorithms' evaluation metrics will be evaluated.

2 Related work

There is currently considerably research being done on diabetes prediction with machine learning algorithms. This paper compares the six most common and traditional machine learning methods. However, more algorithms and methods are being developed to help predict diabetes. For instance, Polat and Güneş [11] use a principal component analysis (PCA) and neuro fuzzy inference. Other studies use ensemble methods to improve the accuracy of their results. Ozcift and Gulten [7] proposed an ensemble approach called rotation forest. It combines 30 machine learning methods. A quantum particle swarm optimization (QPSO) algorithm and weighted least squares support vector machine (WLS-SVM) was introduced by Chen et al. to predict diabetes type 2. Çalişir and Doğantekin [1] proposed a new system called LDA-MWSVM. This system used Linear Discriminant Analysis (LDA) to reduce the dimensions and extract the features. To deal with the high dimensional datasets, Razavian et al. [12] built prediction models based on logistic regression for different onsets of type 2 diabetes prediction. Georga et al. [3] focused on glucose and used support vector regression (SVR) to predict diabetes, which is a multivariate regression problem.

3 Methods

3.1 Programming Language and used packages

This project is coded with Python version 3.9.9. the used packages can be seen in table 1.

Python Package	Version	Description
NumPy [6]	1.21.4	NumPy is the fundamental package for array computing with Python.
pandas [8]	1.3.4	pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool
matplotlib [4]	3.5.0	Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python
seaborn [14]	0.11.2	Seaborn is a library for making statistical graphics in Python
scikit-learn [13]	1.0.1	scikit-learn is a Python module for machine learning built on top of SciPy

Table 1: Used Python libraries

The source code for this project is hosted on GitHub and is openly available on the following webpage: `https://github.com/d-monti/diabetesPrediction`.

3.2 Decision Tree

Decision Tree is a tree-like model that contains conditional control statements. Each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf represents the outcome (a final decision). The splitting criterion at a decision node is generated by Attribute Selection Measure (ASM).

Decision Tree algorithm has multiple advantages as it does not require the normalization of data nor the scaling of tha data, and the missing values do not affect the building process of the decision tree. Some drawbacks are the long time needed for training the model and the instability caused by a small change in the data.

The chosen model is the DecisionTreeClassifier. All default parameters are kept as so. The generation of the evaluation metrics gives an accuracy of [put result].
Best Parameters are: criterion: gini, splitter: best, max_feature: log2, class weight balanced

3.3 K-Nearest Neighbours

K-Nearest Neighbors (KNN) based classification is a type of instance-based learning or non-generalizing learning. The algorithm does not attempt to construct an internal model but instead stores training data instances. It then classifies the new point by a majority vote of the k-nearest neighbours. The optimal choice for the k-factor depends on the data itself. In general, a larger k suppresses noise but makes the classification boundaries less distinct. Since KNN relies on the distance to classify, the data was normalized beforehand. Normalization was chosen over standardization since the algorithm parameters p were left at its default = 2 and the metric, which is 'minkowski'. This is equivalent to the standard Euclidean metric. The ideal parameters were found to achieve the highest accuracy by using the GridSearch() function and 5-fold cross-validation. A Grid from 1-30 for the k-value was created, and different leaf_sizes, algorithms and weights were checked. The highest precision was found with a k-neighbor size of 21, algorithm = 'auto', leaf_size = 10, weights = 'distance'. The leaf_size is passed to the algorithms BallTree or KDTree. This can affect the speed of the construction and the memory, which is required to store the tree. The weights were selected as 'distance', which means that closer neighbours of a query point will have a greater influence than further away neighbours.

3.4 Logistic Regression

Logistic regression (LR) is one of the most used machine learning algorithms for binary classification problems. LR is a linear method, but the predictions are transformed using the logistic(sigmoid) function. An optional L1, L2 or Elastic-Net regularization can be applied. It is also commonly referred to as a penalized logistic regression. Using a penalty is favourable as they solve optimization problems to achieve a good result and improve numerical stability. The Elastic-Net is a combination of both L1 and L2. No regularization amounts to setting C to a very high value. C is the inverse of regularization strength. Hence smaller values specify stronger regularization. To achieve the highest accuracy and find the optimal parameters/hyperparameters for the LR algorithm, a randomized search approach with 5-fold cross-validation was used. The randomized search approach was used. Furthermore, a comparison between standardized scaled and unscaled data was made. The scaled data consistently achieved a higher accuracy of roughly 1% compared to the unscaled data. The highest accuracy was found using the L1 penalty with the 'saga' solver. The found C value with random search lies at 0.33. This indicates a strong regularization. Since all classes should have equal weight, the weight_class parameter was kept at the default value 'balanced'. It is essential to state that the retrieved parameters can change with each code compilation since a random search was used.

3.5 Naïve Bayes

Naive Bayes (NB) is a probabilistic classification technique based on Bayes' theorem with the assumption of independence between all features in a class. Bayes' theorem gives the probability that an event happens with the knowledge of another event with the condition that they are not interdependent. In a Gaussian Naive Bayes classifier, each values are continuous and assumed to be normally distributed.

NB is a good classifier when it comes to multi-class prediction dataset and when the independence between all classes remains true. One big advantage of NB is that it requires a small amount of training data to estimate the test data which leads to a short training period. A drawback of this implementation is the assumption of independent predictors which, in real life, is almost impossible to achieve.

The chosen model is the GaussianNB(). Both parameters *priors* and *var_smoothing* are kept as default. The generation of the evaluation metrics gives an accuracy of [put result].

3.6 Support Vector Machines

Support vector machines (SVMs) are supervised learning methods used for classification, regression, and outliners detection.

An advantage of SVMs is that it is effective in high dimensional spaces, even in cases where the number of dimensions is greater than the number of samples. SVMs are also memory efficient because they use a subset of training points in the decision function (called support vectors). Another advantage is that SVMs is versatile in which different Kernel functions can be specified for the decision function. One disadvantage of SVMs is that if the number of features is much greater than the number of samples, the choice of Kernel functions and the regularization term become crucial to avoid overfitting. Another drawback is that SVMs do not directly provide probability estimates; these are calculated using time expensive k-fold cross-validation.

For this model the dataset will be tested scaled and non scaled, this means training an SVMs over the scaled and non-scaled data leads to the generation of different models. To tune the SVMs model the following parameters where tested with a grid search and cross validation with 5 time folding. The underlined parameters are the best-performing ones, and we used them in this project. The scaled test dataset was used, because it scored a higher accuracy. The best parameters can vary for every run of the source code.

Kernel	C	gamma	class_weight	coef0	degree
Linear	0.9, 1, 2, 10		balanced		
Rbf	0.9, 1, 2, 10	scale, auto	balanced		
Polynomial	0.9, 1, 2, 10	scale	balanced	0.0	2, 3, 4, 5, 6, 7 ,8
Sigmoid	0.9, 1, 2, 10	scale, auto	balanced		

Table 2: Support Vector Machine grid search used test parameters

3.7 Stochastic Gradient Descent

Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to fit linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

The advantages of Stochastic Gradient Descent are its efficiency and the ease of the implementation with a lots of opportunities for code tuning. One drawback of SGD is that its requires a number of hyperparameters such as the regularization parameter and the number of iterations. SGD is also sensitive to feature scaling, and therefore this model was tested with the scaled and non scaled dataset. This model has a lot of parameter to test, following a list of all tested parameters. The underlined parameters are the best-performing ones, and we used them in this project. The scaled test dataset was used, because it scored a higher accuracy. The best parameters can vary for every run of the source code.

Parameters	Values
loss	hinge, log, modified_huber, squared_hinge, perceptron, squared_error, huber, epsilon_insensitive, squared_epsilon_insensitive
penalty	l1, l2, elasticnet
class_weight	balanced
l1_ratio	$0 - 1$ in 10 steps (best: 0.0)
alpha	$0.0001 - 0.1$ (best: 0.01)

Table 3: Stochastic Gradient Descent grid search used test parameters

3.8 Cross Validation

The dataset needs to be split into a test dataset and a training dataset to train and test the machine learning models. In our case, the test dataset is 25% of the entire dataset. We use the training dataset to fit the machine learning model and the test dataset to evaluate the machine learning model. We applied the cross-validation technique to resample the data accordingly. This means that different portions of the data are used to train and then evaluate the models with each iteration. We used fivefold cross-validation for our algorithms. The principle of data resampling can be seen in Figure 1. We chose five as more iterations only slow down the overall code speed and does not significantly impact the results.

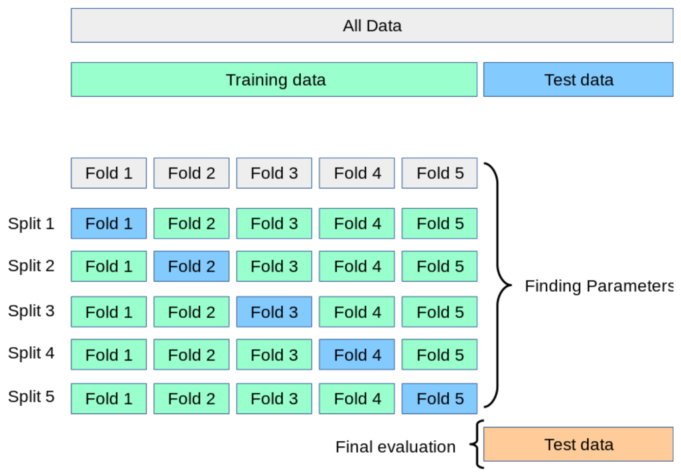


Figure 1: Fivefold Cross Validation [13]

3.9 Tuning Algorithms

The final accuracy score of the algorithms can be fined tuned by changing and adapting its parameters. Some parameters can be learned automatically, but hyper-parameters must be set manually. Setting the hyper-parameters is usually not clear and has to be searched with different methods. The algorithms SGD, SVMs, Decision Tree and KNN use the Grid Search, and the Random Search method is used for Logistic Regression. In Figure 2 the hyper-parameters and parameters are represented as dots, and the green curve above displays its respective model accuracy. The grid search is labelled as (a) and the Random Search as (b). The Random Search defines a search space as a bounded domain of hyperparameter values and randomly samples points in that domain. It has greater variability, and therefore changes in the hyperparameter can have a greater influence on the model tuning. On the other hand, the grid search defines a search space as a grid of hyperparameter values and evaluates every position in the grid.

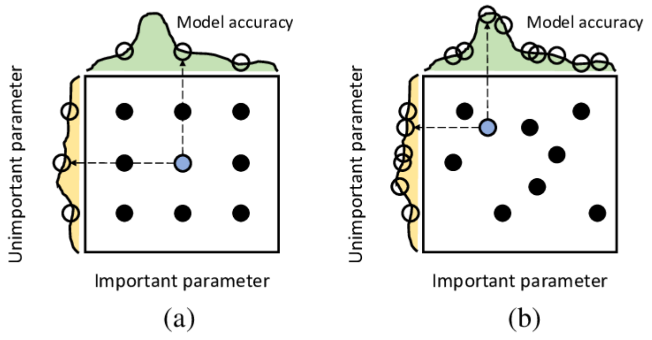


Figure 2: Comparison between (a) Grid Search and (b) Random Search. Changing hyper-parameters has an influence on the model accuracy [9]

3.10 Model Comparisons

To compare the models against each other, the best parameters are automatically chosen, and with them, another cross-validation with 5 times folding is done. The accuracy of all models is then used to create a boxplot, which allows us to compare the model against each other quickly. The accuracy only demonstrates to us a small part of the performance of a model. Therefore also the Specificity, Sensitivity and the F1 value are evaluated. Following the equations to get these values.

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{False Positve}}$$

$$\text{Sensitivity} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

$$F_1 = \frac{\text{True Positive}}{\text{True Positive} + \frac{1}{2}(\text{False Positve} + \text{False Negative})}$$

4 Data and Experiment setup

4.1 Dataset

The Pima Indians Diabetes Database [10] is used to build a model that accurately predict whether or not the patients in the dataset have diabetes or not. The datasets contains several medical predictor variables and one target variable; the output. All patients registered in this dataset are females, take part of the Pima Indian heritage, and are at least 21 years. The dataset contains 768 patients.

Predictor variables and target variable of the dataset

- **Pregnancies** : number of times pregnant
- **Glucose** : plasma glucose concentration a 2 hours in an oral glucose tolerance test
- **BloodPressure** : diastolic blood pressure (mmHg)
- **SkinThickness** : triceps skin fold thickness (mm)
- **Insulin** : 2-Hour serum insulin (muU/ml)
- **BMI** : Body mass index (weight in kg/(height in m)^2)
- **DiabetesPedigree** : diabetes pedigree function (A function that scores the likelihood of diabetes based on family history)
- **Age** : age (years)
- **Outcome** : class variables(0 or 1)

Multiple statistical tools allows a clear overview of the dataset. As some algorthm require the independancy of all variabels, the coorelation between them are calculated and shown in figure 3. The figure 4 show the histogram of the dataset.

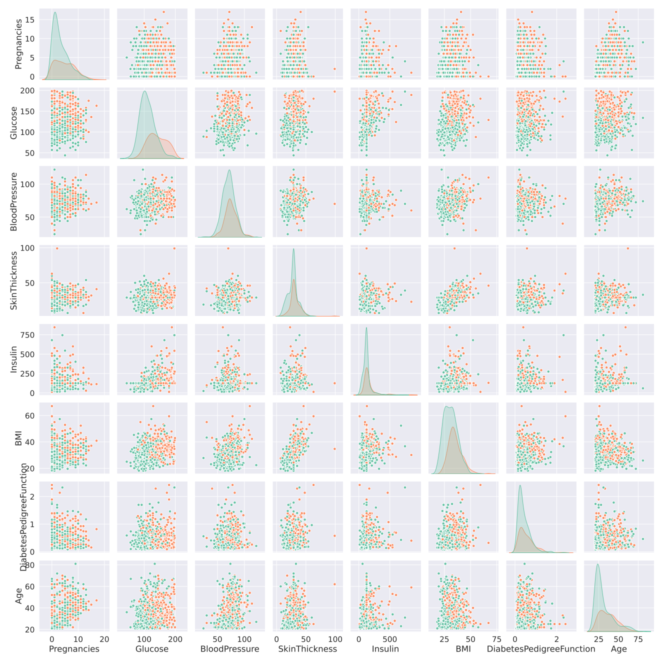


Figure 3: Correlation of the Dataset

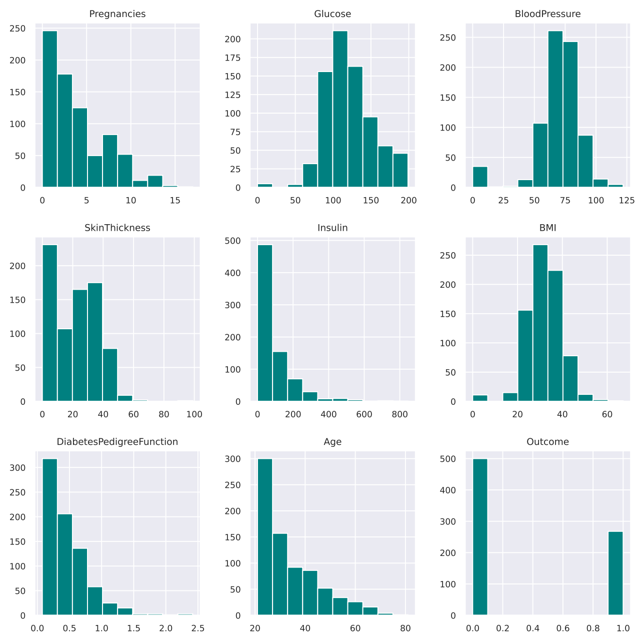
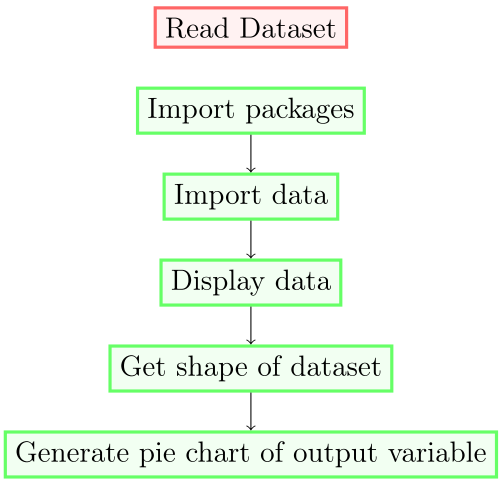


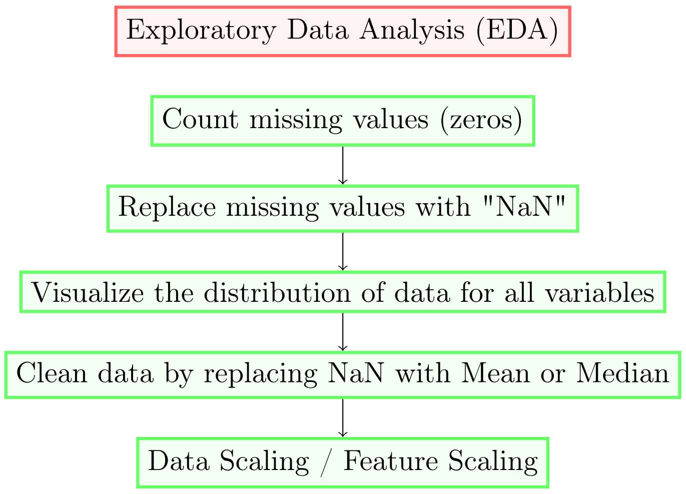
Figure 4: Histogram of the Dataset

4.2 Data preparation

The reading of the dataset is realised with multiples steps as shown in figure [5a]. The raw data needs to be prepared in order to be applicable for all classification algorithm used in this project. All missing values, denoted with the value "0" needs to be replaced and the imbalanced of the dataset needs to be checked. To realized these steps, an Exploratory Data Analysis (EDA), shown in figure [5b] needs to be followed.



(a) Reading dataset.


Exploratory Data Analysis (EDA)
Count missing values (zeros)
Replace missing values with "NaN"
Visualize the distribution of data for all variables
Clean data by replacing NaN with Mean or Median
Data Scaling / Feature Scaling

(b) Exploratory Data Analysis (EDA)

Figure 5: Dataset import and preparation

First we have a look at the dataset to have an overview, shown in figure 4. The summary of all variables is generated and gives useful information such as the number of entries, the mean, the standard deviation, the minimum value, the quartiles, and the maximum value. shown in figure 6.

The output variable is shown in figure 7. This variable is assumed to be exact in the sense that their is no false positive nor false negative. From this statement, the goal is to reach the exact same output using all the predictors given in the dataset.

By looking at the data in table 4, it is easy to see that some "0" values do not make sense. For example, a value of zero in the BMI or in the Blood pressure variable would indicate that the patient is not existing or dead. The replacement of the zero values is made by first analyzing the distribution of the data. If the distribution is symmetric, the missing values are replaced with the mean value. If the distribution shows a lot of outliners, the missing values are replaced by the median value. The distribution and box plots are generated (see figure 6) to permit the decision. Once the missing values are replaces, those figures are re-generated to assure a similar distribution as before the cleaning. Table 4 shows the amount of the "0" values and how they were handled.

Data column	Amount of "0" values	Chosen replacement method
Pregnancies	0	None
Glucose	5	Mean
Blood Pressure	35	Mean
Skin Thickness	227	Median
Insulin	374	Median
BMI (Body Mass Index)	11	Median
Diabetes Pedigree Function	0	None
Age	0	None

Table 4: Amount of zero values in dataset

4.2.1 Data Scaling / Feature Scaling

Data scaling or feature scaling is used in some machine learning algorithms. If there is a vast difference between numbers sizes, the model could assume that bigger numbers are more significant than smaller ones. Therefore all values are normalized or standardized.

In our case, data scaling was used for the LR, SGD, SVMs and KNN algorithms, in all cases, the scaled data had better accuracy than the non-scaled data. Data scaling also helps some models (for example, SGD) with the training time, significantly improving the converge time.

4.2.2 Imbalanced Data

The Pima Indians Diabetes Database [10] is quite imbalanced. This can be seen in figure 7, the outcome has way more non-diabetes patients (diabetes: 34.9%, No diabetes: 65.1%) than patients with diabetes. Some algorithms have problems with imbalanced data, for example, the SVMs algorithm. Suppose the imbalanced data are not handled in the SVMs algorithm. It will predict every patient as a non-diabetes patient and, therefore, have a pretty good accuracy because 65.1% of the patients are non-diabetic. This gives us a specificity of 100% and a sensitivity of 0%. Therefore, the model is useless. The imbalanced data need to be checked and handled if the algorithm requires it.

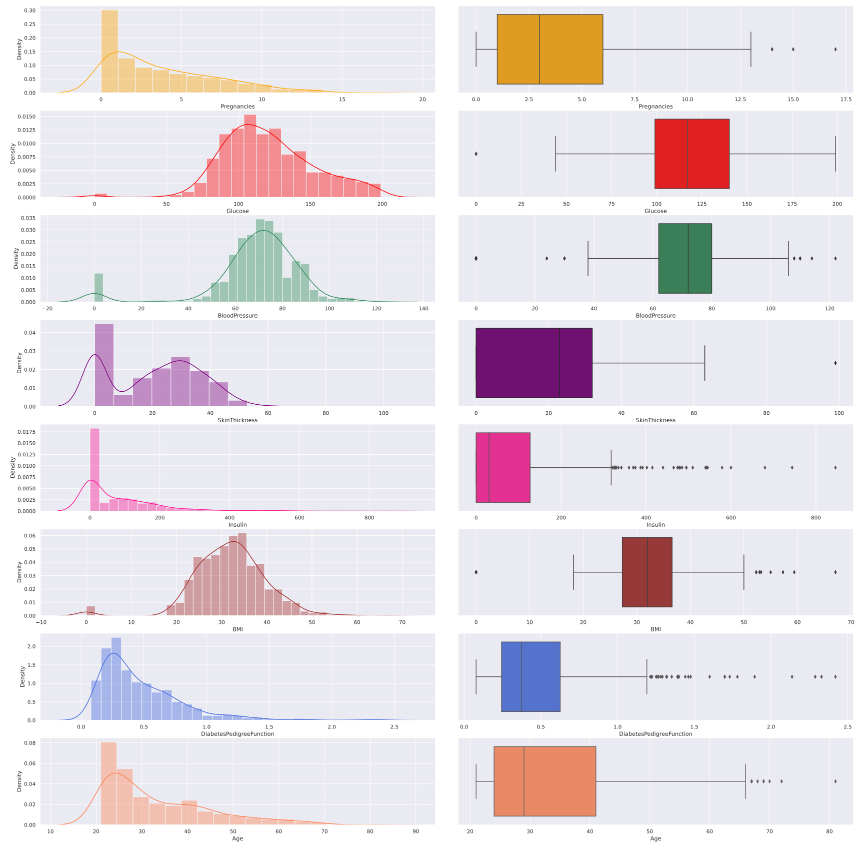


Figure 6: Dataset distribution and boxplots

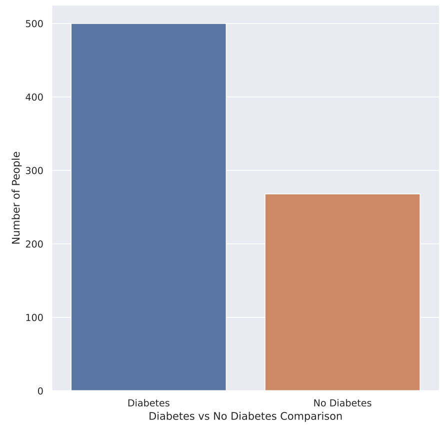


Figure 7: Dataset Outcome

5 Results

We applied the mentioned machine learning algorithms and got the performance results in the table 5, and for the accuracy and f1 score, boxplots were made, see figure 8.

Model	Accuracy	Specificity	Sensitivity	F1
Decision Trees	0.708502	0.715447	0.608696	0.570121
K Nearest Neighbors	0.740216	0.861789	0.565217	0.619996
Logistic Regression	0.765992	0.853659	0.565217	0.614503
Naive Bayes	0.729555	0.772358	0.666667	0.619294
Support Vector Machines	0.718489	0.617886	0.724638	0.625714
Stochastic Gradient Descent	0.745344	0.569106	0.869565	0.493682

Table 5: Model Performance Results

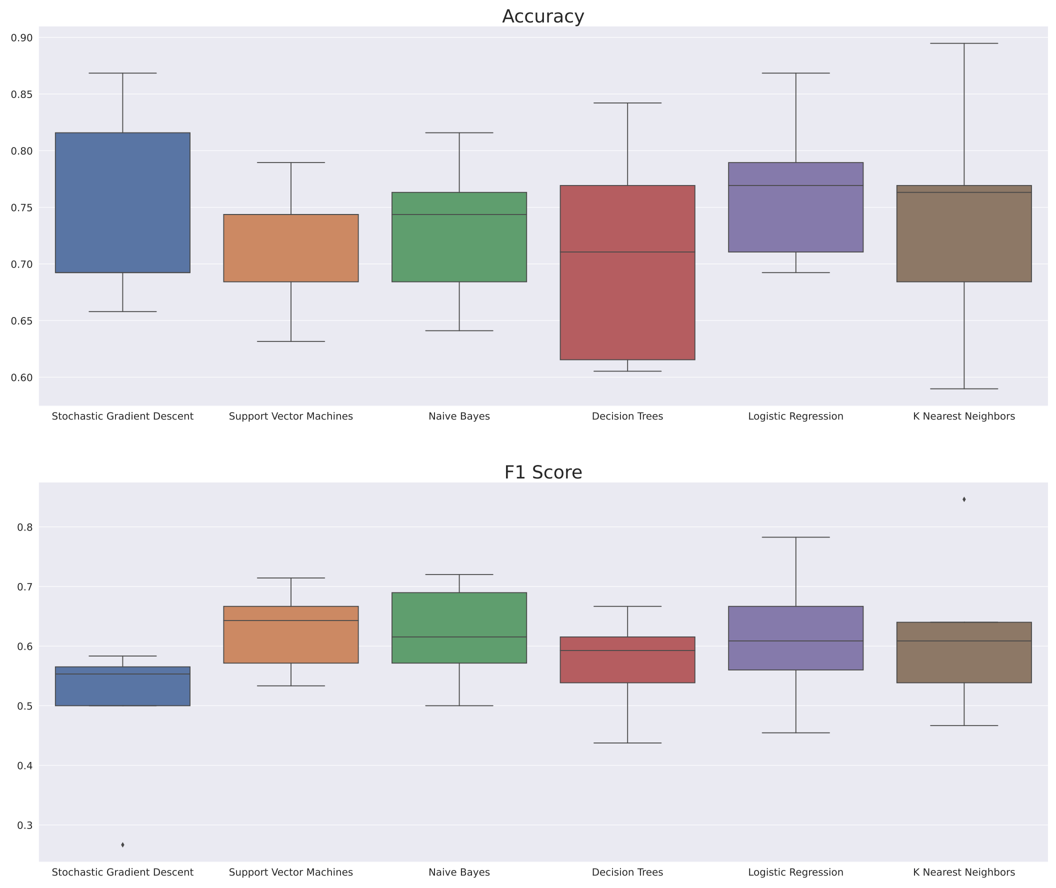


Figure 8: Model Comparison

The overall best model is the one with the Logistic Regression algorithm.

6 Discussion

Diabetes is a well-knows chronic metabolic disease that results from defects in insulin secretion, insulin action, or both. The long-term complications related to diabetes could be critical. To avoid dramatic situations, the diagnosis of diabetes should be made as early as possible. To this day, Prediction Analysis as been used to assist medical decision-making. What differs between each algorithm used to predict disease such as diabetes is the accuracy find in the evaluation metrics. By evaluating the accuracy of multiple algorithms, one can find the one that suits the most in a medical application such as diabetes diagnosis. In order to do this comparison, the Pima Indian Diabetes Database is prepared and applied to classification algorithms such as Logistic Regression, Naïve Bayes, Stochastic Gradient Descent, K-Nearest Neighbors, Decision Tree, and Support Vector Machine. The result shows that the best accuracy is reached with [???????] with an accuracy of [???????]. The result is explained by a good ratio of number of correct predictions to the total number of input samples. The database includes only woman and is therefore limited to female diagnosis only. Future work could include a study of male-diagnosed patient to develop a more inclusive diagnosis method.

TODO -> a ccuracy of arounf 77% is really bad not? its a bit better than guessing, shall we write something about that? WHO criteria of ≥80% sensitivity and ≥97% specificity -> example requirments who coivd tests

References

[1] Duygu Çalişir and Esin Doğantekin. "An automatic diabetes diagnosis system based on LDA-Wavelet Support Vector Machine Classifier". In: Expert Systems with Applications 38 (July 2011), pp. 8311–8315. DOI: https://doi.org/10.1016/j.eswa.2011.01.017.

[2] Yuna Chen et al. "Identifying Type 2 Diabetic Brains by Investigating Disease-Related Structural Changes in Magnetic Resonance Imaging". In: Frontiers in Neuroscience 15 (2021), p. 1418. ISSN: 1662-453X. DOI: https://doi.org/10.3389/fnins.2021.728874.

[3] E. I. Georga et al. "Multivariate prediction of subcutaneous glucose concentration in type 1 diabetes patients based on support vector regression". In: IEEE journal of biomedical and health informatics 17 (July 2013), pp. 71–81. DOI: https://doi.org/10.1109/TITB.2012.2219876.

[4] Matplotlib: Visualization with Python. https://matplotlib.org/. Accessed: 09.12.2021.

[5] Aishwarya Mujumdar and V Vaidehi. "Diabetes Prediction using Machine Learning Algorithms". In: Procedia Computer Science 165 (2019), pp. 292–299. ISSN: 1877-0509. DOI: https://doi.org/10.1016/j.procs.2020.01.047.

[6] Numpy: The fundamental package for scientific computing with Python. https://numpy.org/. Accessed: 09.12.2021.

[7] Akin Ozcift and Arif Gulten. "Classifier ensemble construction with rotation forest to improve medical diagnosis performance of machine learning algorithms". In: Computer Methods and Programs in Biomedicine 104.3 (2011), pp. 443–451. ISSN: 0169-2607. DOI: https://doi.org/10.1016/j.cmpb.2011.03.018.

[8] Panda: open source data analysis and manipulation tool. https://pandas.pydata.org/. Accessed: 09.12.2021.

[9] Karl Ezra Salgado Pilario, Yi Cao, and Mahmood Shafiee. "A Kernel Design Approach to Improve Kernel Subspace Identification". In: IEEE Transactions on Industrial Electronics 68.7 (2021), pp. 6171–6180. DOI: https://doi.org/10.1109/TIE.2020.2996142.

[10] Pima Indians Diabetes Database. https://www.kaggle.com/uciml/pima-indians-diabetes-database. Accessed: 09.12.2021.

[11] Kemal Polat and Salih Güneş. "An expert system approach based on principal component analysis and adaptive neuro-fuzzy inference system to diagnosis of diabetes disease". In: Digital Signal Processing 17.4 (2007), pp. 702–710. ISSN: 1051-2004. DOI: https://doi.org/10.1016/j.dsp.2006.09.005.

[12] N. Razavian et al. "Population-Level Prediction of Type 2 Diabetes From Claims Data and Analysis of Risk Factors". In: Big data 3 (July 2015), pp. 277–287. DOI: https://doi.org/10.1089/big.2015.0020.

[13] scikit-learn: Machine Learning in Python. https://scikit-learn.org/. Accessed: 09.12.2021.

[14] Seaborn: Statistical data visualization. https://seaborn.pydata.org/. Accessed: 09.12.2021.