Early-stage diabetes risk prediction using Logistic Regression

Raunak Nandkumar More
Department of Computer Science
University of South Dakota

ABSTRACT:

Diabetes is a chronic disease that affects millions of people worldwide. It occurs when the pancreas cannot produce enough insulin or use it effectively, resulting in high blood sugar levels. Diabetes can cause serious complications, such as heart disease, kidney failure, blindness, and amputation. It is important to diagnose and treat diabetes as early as possible. However, its diagnosis is often a challenge for people who do not have symptoms or do not seek medical attention. The tests involved to diagnose diabetes are invasive, time consuming, and expensive, thus the need of finding cheaper, non-invasive, alternative methods of detection. In this project is proposed a method of using data analysis and machine learning to detect the risk of diabetes.

INTRODUCTION:

According to the World Health Organization, diabetes was the seventh leading cause of death in 2016, and it is estimated that 422 million adults were living with diabetes in 2014. Early intervention can prevent or delay the onset of complications and improve the quality of life of patients. One of the possible methods is where we leverage the power of data and algorithms to build models that can detect the risk of diabetes based on the signs and symptoms of patients. Using a dataset of signs and symptoms of newly diabetic or would be diabetic patients, collected from Sylhet Diabetes Hospital in Sylhet, Bangladesh and applying logistic regression, a supervised learning algorithm, we train a model that can predict whether a patient has diabetes or not. The model's evaluated for performance using accuracy, confusion matrix, and classification report.The dataset is publicly available on the UCI Machine Learning Repository[1].

The dataset consists of 520 observations and 16 variables. The variables are a mix of binary, categorical, and numerical types. The target variable is class, which indicates whether the patient has diabetes (positive) or not (negative). The other 15 variables are the features that describe the signs and symptoms of the patient are:

Age: The age of the patient in years. It is a numerical variable with values ranging from 16 to 90.
Gender: The gender of the patient. It is a binary variable with two values: Male or Female.
Polyuria: Whether the patient has excessive urination. It is a binary variable with two values: Yes or No.
Polydipsia: Whether the patient has excessive thirst. It is a binary variable with two values: Yes or No.
Sudden weight loss: Whether the patient has sudden weight loss. It is a binary variable with two values: Yes or No.
Weakness: Whether the patient has weakness. It is a binary variable with two values: Yes or No.
Polyphagia: Whether the patient has excessive hunger. It is a binary variable with two values: Yes or No.
Genital thrush: Whether the patient has genital infection. It is a binary variable with two values: Yes or No.
Visual blurring: Whether the patient has visual blurring. It is a binary variable with two values: Yes or No.
Itching: Whether the patient has itching. It is a binary variable with two values: Yes or No.
Irritability: Whether the patient has irritability. It is a binary variable with two values: Yes or No.
Delayed healing: Whether the patient has delayed healing. It is a binary variable with two values: Yes or No.
Partial paresis: Whether the patient has muscle weakness. It is a binary variable with two values: Yes or No.
Muscle stiffness: Whether the patient has muscle stiffness. It is a binary variable with two values: Yes or No.
Alopecia: Whether the patient has hair loss. It is a binary variable with two values: Yes or No.
Obesity: Whether the patient is obese. It is a binary variable with two values: Yes or No.

RELATED WORK:

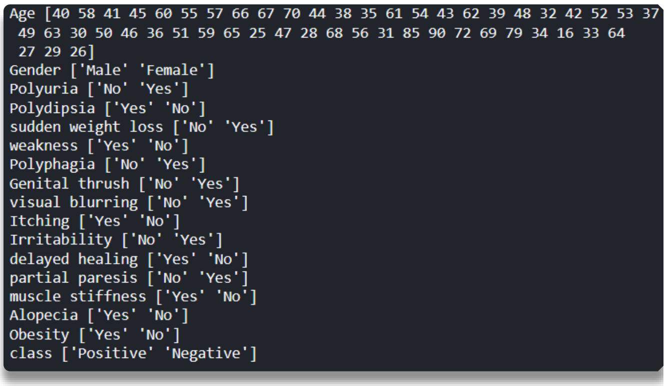
1. Wee BF, Sivakumar S, Lim KH, Wong WK, Juwono FH (2023) Diabetes detection based on machine learning and deep learning approaches. In: Multimedia Tools and Applications. Springer, p 1–36
2. Abdulhadi N, Al-Mousa A (2021) Diabetes detection using machine learning classification methods. In: 2021 International Conference on Information Technology (ICIT). IEEE, p 350–354
3. Sarker IH (2021) Machine learning: algorithms, real-world applications and research directions. In: SN Computer Science. Springer, p 1–36
4. Jurafsky, D., & Martin, J. H. (2023). Logistic regression. In Speech and language processing (3rd ed., pp. 5-25). https://web.stanford.edu/~jurafsky/slp3/5.pdf

METHODOLOGY:

We discuss the techniques used to preprocess the data, split the data into training and testing sets, train the logistic regression model, and evaluate the model performance.

Python is used for the coding part and its libraries, such as numpy, pandas, sklearn, are imported to perform the data analysis and machine learning. Jupyter Notebook used as the interactive development environment. The code and the output can be found in the ipnb file.

The first step of our methodology was to import the necessary libraries as mentioned already. We used the pandas function read_csv to read the csv file containing the dataset and store it in a data frame called df. We find unique values of each column using a for loop and the unique() method and result was as shown below:



```
Age [40 58 41 45 60 55 57 66 67 70 44 38 35 61 54 43 62 39 48 32 42 52 53 37
 49 63 30 50 46 36 51 59 65 25 47 28 68 56 31 85 90 72 69 79 34 16 33 64
 27 29 26]
Gender ['Male' 'Female']
Polyuria ['No' 'Yes']
Polydipsia ['Yes' 'No']
sudden weight loss ['No' 'Yes']
weakness ['Yes' 'No']
Polyphagia ['No' 'Yes']
Genital thrush ['No' 'Yes']
visual blurring ['No' 'Yes']
Itching ['Yes' 'No']
Irritability ['No' 'Yes']
delayed healing ['Yes' 'No']
partial paresis ['No' 'Yes']
muscle stiffness ['Yes' 'No']
Alopecia ['Yes' 'No']
Obesity ['Yes' 'No']
class ['Positive' 'Negative']
```

Using the isna().sum() method we found there were no missing values in the dataset.We noticed that some of the variables in our dataset were categorical, meaning that they had a finite number of possible values, such as gender, polyuria, polydipsia, etc. These variables were represented by strings, such as Male, Female, Yes, No, etc. However, machine learning algorithms usually require numerical inputs, so we had to encode these variables into numerical labels. We used the LabelEncoder class from sklearn and applied the fit_transform() method to each column of the data frame, converting the strings into numerical labels. For example, the gender variable was encoded as 0 for Female and 1 for Male, the polyuria variable was encoded as 0 for No and 1 for Yes, and so on. We also noticed that one of the variables in our dataset was numerical, meaning that it had a continuous range of possible values, such as age. This variable had values ranging from 16 to 90, which were much larger than the values of the binary variables, which ranged from 0 to 1. This could cause a problem for machine learning algorithms, as they might give more weight to the numerical variable and ignore the binary variables, resulting in a biased model. To avoid this problem, we had to normalize the numerical variable, meaning that we had to scale its values to a common range, such as 0 to 1. We used the MinMaxScaler class from sklearn and applied the fit_transform() method to the age column of the data frame, scaling its values to the range of 0 to 1.

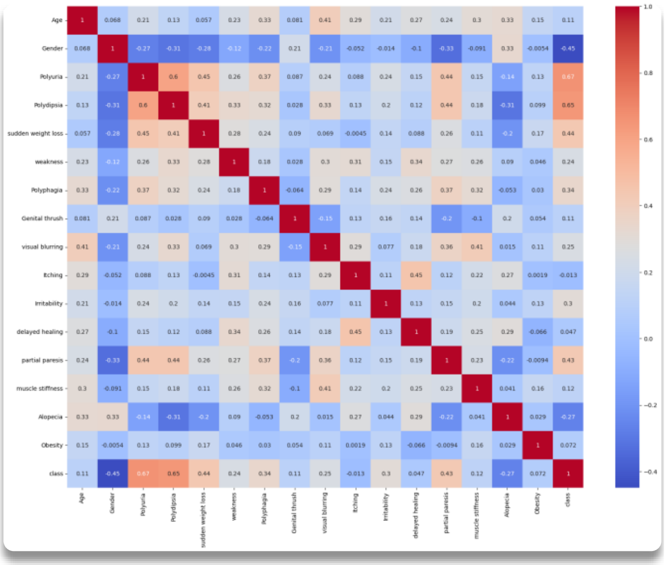
For splitting of data for getting it ready for training and testing, we used the drop() method to remove the class column from the data frame and assign it to a separate variable called y. We also assigned the remaining columns of the data frame to another variable called X. We used the train_test_split function from sklearn to split the data into training and testing sets where 80% of the data was used for training the model and 20% of the data was used for testing. We used the stratify parameter as y, meaning that the proportion of positive and negative cases of diabetes would be preserved in both sets. We also used the random_state parameter as 44552, meaning that the split would be reproducible. We assigned the resulting subsets to four variables: X_train, X_test, y_train, and y_test.

We used the LogisticRegression class from sklearn to create an instance of the logistic regression model. We assigned the instance to a variable called model.We used the fit() method to train the model on the training data, using X_train and y_train as the inputs. This method learned the coefficients and the intercept of the logistic regression equation, which defined the relationship between the features and the target variable. We used the predict() method to predict the class labels for the testing data, using X_test as the input. This method applied the logistic regression equation to the testing data and returned the predicted class labels, either 0 (negative) or 1 (positive). We assigned the predicted labels to a variable called y_pred.

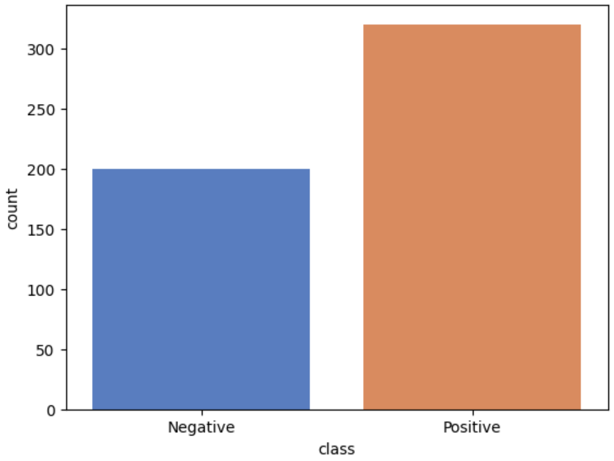
For evaluation of model we used the accuracy_score function from sklearn to calculate the accuracy of the model, which is the proportion of correct predictions out of the total number of predictions. We used y_test and y_pred as the inputs and multiplied the result by 100 to get the percentage. We assigned the accuracy to a variable called acc. We used the confusion_matrix function from sklearn to calculate the confusion matrix of the model, which is a table that shows the number of true positives, false positives, true negatives, and false negatives. We used y_test and y_pred as the inputs and used the seaborn library to plot the confusion matrix as a heatmap. We also added labels and annotations to the plot to make it more readable. We used the classification_report function from sklearn to calculate the classification report of the model, which is a summary of the precision, recall, f1-score, and support for each class. We used y_test and y_pred as the inputs, and printed the result. The precision is the proportion of positive predictions that are positive, the recall is the proportion of positive cases that are correctly predicted, the f1-score is the harmonic mean of the precision and recall, and the support is the number of observations in each class.

OBSERVATIONS:

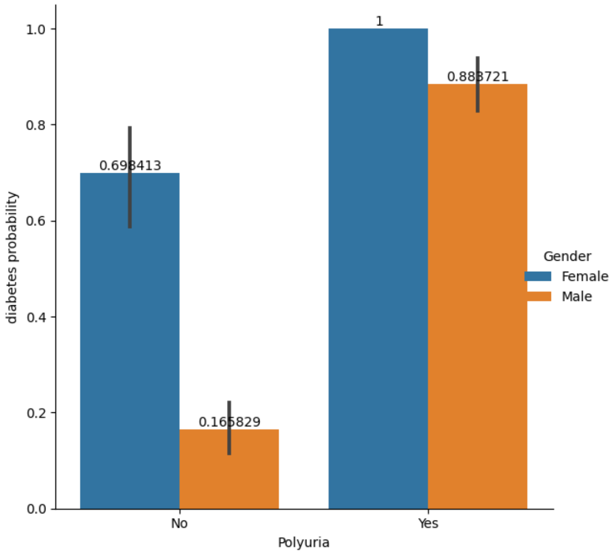
Plotting HeatMap of Correlation Matrix



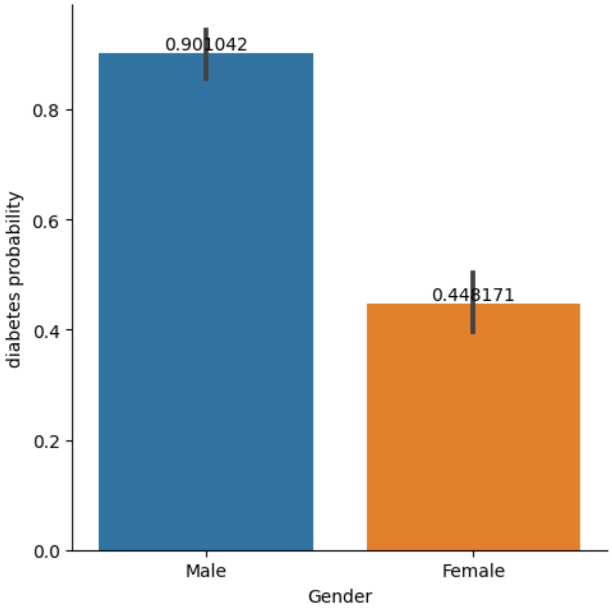
By plotting heatmap of the correlation matrix using df.corr() we find the correlation between features.



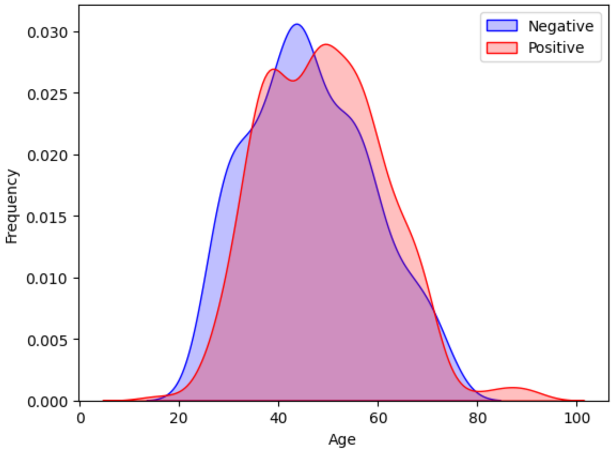
We saw that there were more positive cases than negative cases specifically, 38.46 % negative class and positive class of 61.54 %



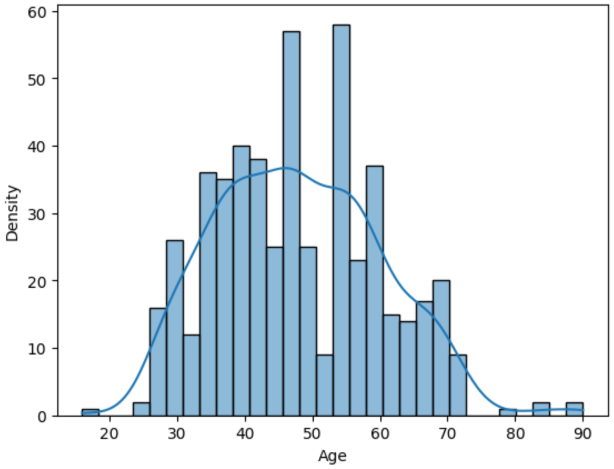
The most important variable for predicting diabetes is ‘Polyuria’,with highest correlation with the class variable (0.67), and also the highest difference in proportions between the positive and negative groups (0.88 vs 0.16). This means that people who have polyuria are much more likely to have diabetes than those who do not.



The second most important variable is ‘Gender’, which has a correlation of 0.45 with the class variable. The proportion of males who have diabetes is 0.90, while the proportion of females who have diabetes is 0.44. This means that males are more than twice as likely to have diabetes than females.



The third most important variable is ‘Age’, which has a correlation of 0.42 with the class variable. The mean age of the positive group is 49.07, while the mean age of the negative group is 46.36. This means that older people are more likely to have diabetes than younger people. The age distribution of the dataset is skewed to the right, with most of the observations between 40 and 60 years old as seen in the plot here that shows frequency of different age groups among the patients, and the peak of the curve indicates the most common age range.



Plot shows that the age distribution of the patients is skewed to the right, meaning that most of the patients are between 40 and 60 years old. The peak of the curve indicates the most common age range, which is around 50 years old. The plot also shows that there are some outliers in the data, such as the patients who are younger than 20 or older than 80 years old.

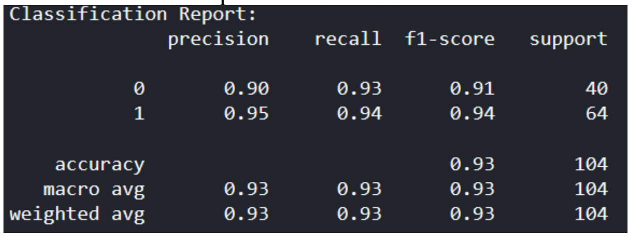
RESULTS:

The accuracy of our model is 93, indicating that our model can distinguish between positive and negative cases of diabetes with a high degree of accuracy. However, accuracy alone is not enough to evaluate the performance of a classification model, as it does not tell us how well the model performs on each class separately. Therefore, we also look at the confusion matrix and the classification report of our model.

	Predicted Negative	Predicted Positive
Actual Negative	37	3
Actual Positive	4	60

From the confusion matrix, we can see that our model has 37 true negatives, 3 false positives, 4 false negatives, and 60 true positives. This means that our model correctly identified 37 out of 40 patients who do not have diabetes, and 60 out of 64 patients who have diabetes. However, our model also misclassified 3 patients who do not have diabetes as having diabetes, and 4 patients who have diabetes as not having diabetes. These errors can have serious consequences for the patients, as they may receive inappropriate treatment or miss the opportunity to receive timely intervention.

The classification report of our model is shown below:



```
Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.93      0.91        40
           1       0.95      0.94      0.94        64

    accuracy                           0.93       104
   macro avg       0.93      0.93      0.93       104
weighted avg       0.93      0.93      0.93       104
```

From the classification report, we can see that our model has a high precision and recall for both classes, meaning that it has a low rate of false positives and false negatives. The f1-score is also high for both classes, meaning that it balances the precision and recall well. The support shows that there are more positive cases than negative cases in the testing set, reflecting the imbalance of the original dataset. The macro average and the weighted average are both 0.93, meaning that our model performs equally well on both classes.

CONCLUSION:

We conclude that while the project demonstrates the potential of using machine learning to diagnose diabetes, we also acknowledge some limitations and challenges of our project. The dataset is small and imbalanced, which may affect the reliability and generalizability of our model. We may need more data and more diverse data to improve the robustness and validity of our model. The dataset is based on self-reported signs and symptoms, which may be subjective and inaccurate. We may need more objective and reliable measures, such as blood tests, to verify the diagnosis of diabetes. The model is based on logistic regression, which is a simple and linear algorithm that may not capture the complexity and nonlinearity of the data. We may need to explore other algorithms, such as decision trees, neural networks, or ensemble methods, to improve the performance and interpretability of our model.

For further improvement following points could be addressed:

Collecting more data and more diverse data from different sources and regions, to increase the sample size and the representation of the population. Performing more data cleaning and data transformation, such as handling outliers, missing values, and skewed distributions, to improve the quality and consistency of the data. Performing more model tuning and model comparison, such as adjusting the hyperparameters, applying regularization, and comparing different algorithms, to improve the accuracy and robustness of the model.

REFERENCES:

[1]:https://archive.ics.uci.edu/dataset/529/early+stage+diabetes+risk+prediction+dataset