Diabetes

Subria Islam

Classification

—

Logistic Regression and Decision Tree

Description of the work

Description of Dataset:

This diabetes dataset is collected from the National Institute of Diabetes and Digestive and Kidney Diseases. It has 768 instances. It has 9 attributes(columns) and all are in numerical value.

All attributes (columns) are:
Pregnancies: Number of times a person has been pregnant.
Glucose: Blood sugar level measured 2 hours after consuming a glucose drink.
BloodPressure: Diastolic blood pressure (mm Hg)
SkinThickness: Thickness of a fold of skin on the triceps (mm).
Insulin: Blood insulin level measured 2 hours after consuming glucose (mu U/ml).
BMI: Body mass index, a measure of body weight relative to height (weight in kg/(height in m)^2).
DiabetesPedigreeFunction: A function that predicts the likelihood of diabetes based on family history.
Age: Age (years).
Outcome: A binary class variable, where 0 typically means no diabetes, and 1 means diabetes is present.

Index	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	79.7995	33.6	0.627	50	1
1	1	85	66	29	79.7995	26.6	0.351	31	0
2	8	183	64	20.5365	79.7995	23.3	0.672	32	1

Goal:

Based on the diagnostic measurements, make predictions about whether a patient is likely to have diabetes or not.

Data Preparation for the training

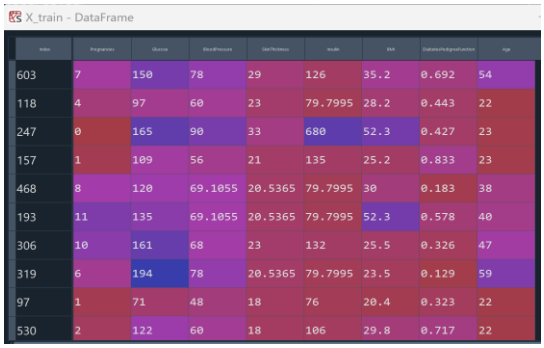
Dataset columns:

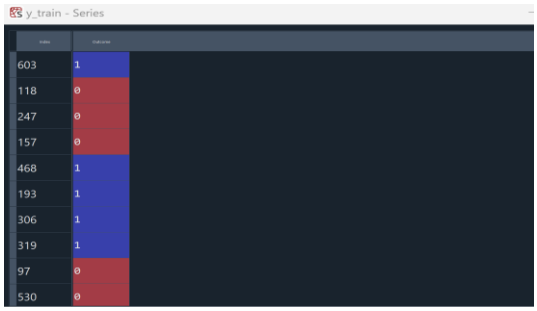
In the dataset, all columns don't have any null values. But some columns have a missing value which is mentioned as zero. But Glucose, Blood Pressure, Skin Thickness, Insulin, and BMI values don't be zero.

Data preparation:

For both machine learning algorithms (logistic regression and decision tree) zero values were replaced with the mean value of the column. Then input data (X) and output data (y) were created from the dataset. For training and testing purposes all data were split. 80% for training and 20% for tasting.

Screenshot of training data:





Standardization() which is an instance of the scikit-learn StandardScaler class, was used to scale the data.

Relevant metrics for the cases

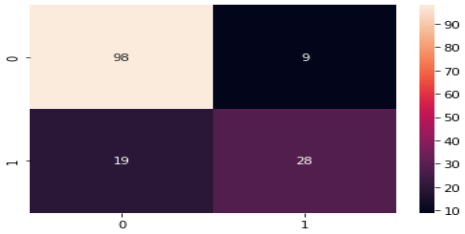
For both Logistic regression and decision tree, a confusion matrix was used to estimate the result. Also, values were calculated for accuracy, precision, and recall.

Logistic regression:

By using Logistic Regression, the model was trained. And finally, Predicting outputs with X_test as inputs.

Confusion matrix for logistic regression:

The result was Estimated by a confusion matrix.



Here,
TN(True Negative): In the confusion matrix, there are 98 instances where the actual class was negative and the model correctly predicted them as negative.

TP(True Positive): In the confusion matrix, there are 28 instances where the actual class was positive and the model correctly predicted them as positive.

FP(False Positive): In the confusion matrix, there are 9 instances where the actual class was negative and the model incorrectly predicted them as positive.

FN(False Negative): In the confusion matrix, there are 19 instances where the actual class was positive and the model incorrectly predicted them as negative.

Accuracy score for logistic regression model:

The accuracy score is 0.82 means that the model's correctness is 82%.

Precision score for logistic regression model:

The precision score is 0.76 means that out of all the positive predictions made by this model, approximately 76% of them were correct.

Recall score for logistic regression model:

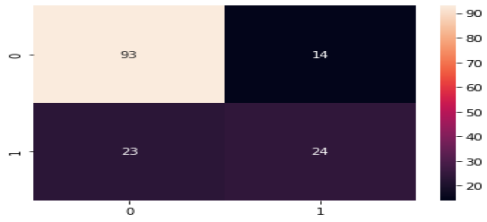
The recall score is 0.60 means that this model correctly identified approximately 60% of the actual positive cases in the dataset.

Decision tree:

The model was trained by using Decision Tree. And finally, Predicting outputs with X_test as inputs.

Confusion matrix for Decision tree:

The result was Estimated by a confusion matrix.



Here,
TN(True Negative): In the confusion matrix, there are 93 instances where the actual class was negative and the model correctly predicted them as negative.

TP(True Positive): In the confusion matrix, there are 24 instances where the actual class was positive and the model correctly predicted them as positive.

FP(False Positive): In the confusion matrix, there are 14 instances where the actual class was negative and the model incorrectly predicted them as positive.

FN(False Negative): In the confusion matrix, there are 23 instances where the actual class was positive and the model incorrectly predicted them as negative.

Accuracy score for Decision tree model:

The accuracy score is 0.76 means that the model's correctness is 76%.

Precision score for Decision tree model:

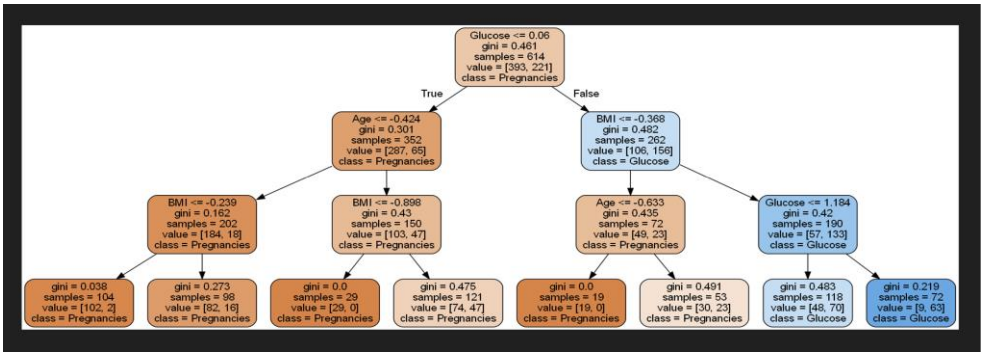
The precision score is 0.63 means that out of all the positive predictions made by this model, approximately 63% of them were correct.

Recall score for Decision tree model:

The recall score is 0.51 means that this model correctly identified approximately 51% of the actual positive cases in the dataset.

Decision tree model:

export_graphviz and graphviz library were used.



Conclusions of the results

The logistic regression model has a higher accuracy rate than the decision tree model. Higher accuracy generally is a positive indicator.

Precision measures the proportion of true positive predictions among all positive predictions made by the model. A higher precision means fewer false positives. In this case, the logistic regression model also outperforms the decision tree in terms of precision.

Recall measures the proportion of true positive predictions among all actual positives. It indicates how well the model captures positive instances. Again, the logistic regression model has a higher recall compared to the decision tree.

Based on these metrics, logistic regression appears to perform better than the decision tree for this dataset.

New Data:

Two new patient data were made for each model and each patient's diabetes output was predicted by using each trained model. The prediction output was good enough for getting the information about whether the patient has diabetes or not.

Both for Logistic regression and Decision tree:

Get a better model:

To enhance model performance, need to focus on data preprocessing, hyperparameter tuning, regularization techniques to reduce overfitting, etc.