Predicting Diabetes

in the

Pima Indians Diabetes Dataset

Springboard Capstone Project

Chen-Yu Wang

Background & Objective

- The Pima Native American people in Arizona are observed to have a very high prevalence of Type II Diabetes while the genetically identical Pima people living in Mexico do not.
- It is hypothesized that it is a combination of genetic and environmental factors that causes the high incidence of diabetes in the Pima people in Arizona.
- They are the subject of many studies on Type II Diabetes. The Pima Indians Diabetes dataset originated from one of such studies.
- The objective of this project is to build a machine learning model that predicts if individuals in this population have diabetes outcomes based on their diagnostic measurements.

The Pima Indians Diabetes Dataset

- The dataset was sampled from a population of Pima Native American women of at least 21 years of age.
    - CSV file
    - 768 entries
- There are eight different diagnostic measurements as independent variables:
    - Pregnancies
    - Glucose
    - Blood pressure
    - Skin Thickness
    - Insulin
    - BMI
    - Diabetes Pedigree Function
    - Age
- A binary outcome variable
    - 0 - no diabetes
    - 1 - diabetes

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Pregnancies               768 non-null    int64
 1   Glucose                   768 non-null    int64
 2   BloodPressure             768 non-null    int64
 3   SkinThickness             768 non-null    int64
 4   Insulin                   768 non-null    int64
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64
 8   Outcome                   768 non-null    int64
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

Data Wrangling - Zeros and missing values

- Zeros in the ‘Pregnancies’ column could denote ‘never pregnant’
- Zeros in other columns make little sense and are likely missing values.
- Zeros are converted to NaN values in all columns except for ‘Pregnancies’.
- There are a lot of data missing in ‘SkinThickness’ and ‘Insulin’.

Number of missing values in all columns except for ‘Pregnancies’

```
Glucose                     5
BloodPressure              35
SkinThickness             227
Insulin                   374
BMI                        11
DiabetesPedigreeFunction    0
Age                         0
dtype: int64
```

Percentage of missing values in all columns except for ‘Pregnancies’

```
Glucose                    0.65
BloodPressure              4.56
SkinThickness             29.56
Insulin                   48.70
BMI                        1.43
DiabetesPedigreeFunction   0.00
Age                        0.00
dtype: float64
```

EDA - Data distribution of each feature by ‘Outcome’

Boxplots (Figure 1.) were generated for each of the features grouped by ‘Outcome’.

In all 8 features, the average values in individuals with diabetes are slightly higher than those without.

However, in ‘Glucose’ there is significant separation between the ranges of the two different ‘Outcome’ groups. Therefore ‘Glucose’ should be a feature of high importance in our model.

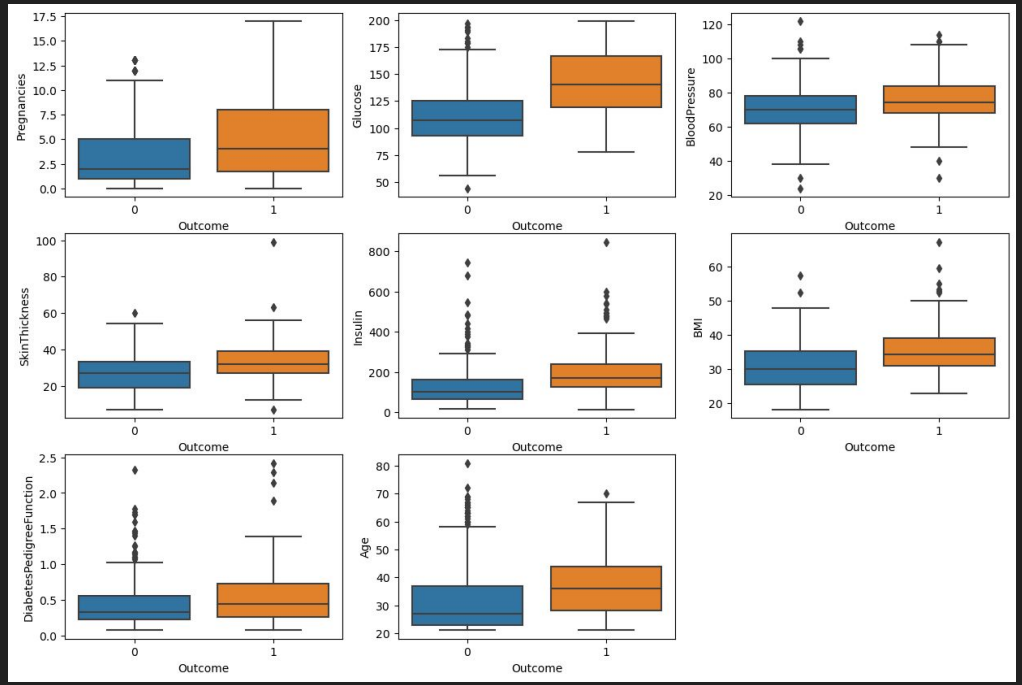


Figure 1. Boxplots of values in each feature grouped by ‘Outcome’.

Pre-processing - Imputation of missing values

The values in each feature were plotted as histograms (Figure 2).

The distribution for ‘BloodPressure’ appears to be close to Normal.

Distributions of the other 7 features display skewness of varying degree.

In order to minimize the effect of outliers, for imputation of missing data, medians were used for all the features except for ‘BloodPressure’ due to its fairly Normal distribution.

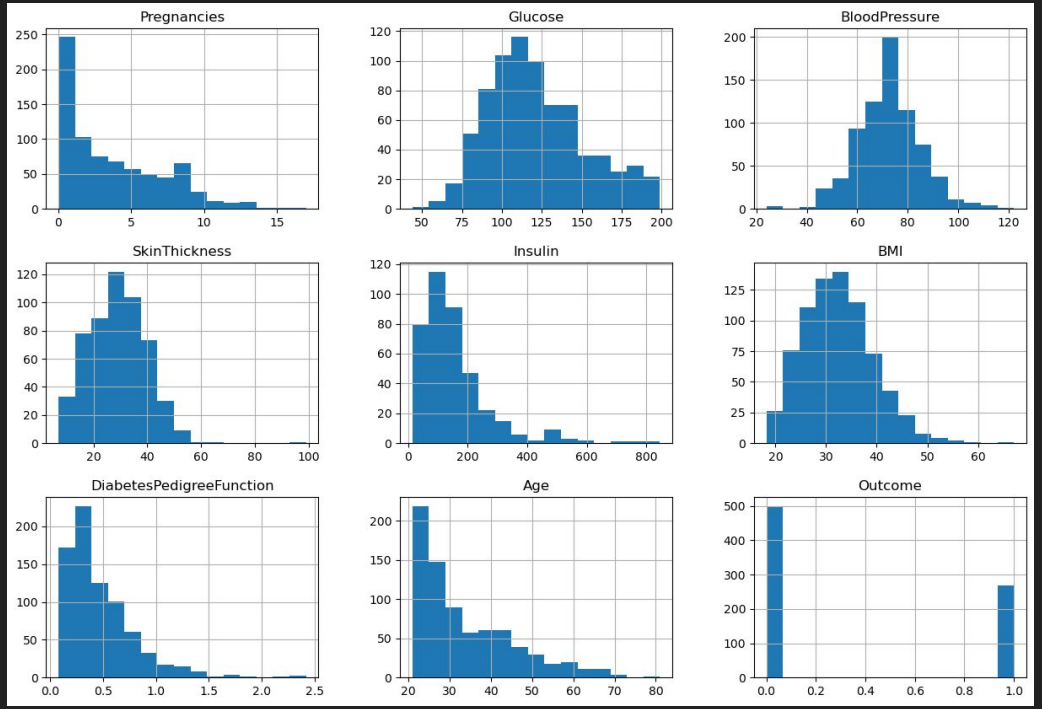


Figure 2. Distribution of values in each feature before imputation of missing values.

Modeling 1 - No hyperparameter tuning

A simple Logistic Regression model, a Decision Tree model and a Support Vector Machine model were fitted without any hyperparameter tuning. Models were evaluated by comparing training and testing accuracy. F-1 score for all of the models were also evaluated because we want to select a model that minimizes false positive and false negative rates.

Table 1: Comparing accuracy scores and F-1 scores for 3 models with no hyperparameter tuning:

	Logistic Regresson	Decision Tree	Support Vector Machine
Training Accuracy	0.78	0.85	0.85
Testing Accuracy	0.73	0.71	0.71
F-1 score	0.69	0.66	0.65

Both the Decision Tree and SVM models show signs of overfitting, while the Logistic Regression model has the best testing accuracy and F-1 score.

Modeling 2: GridSearchCV

GridSearchCV was used to tune hyperparameters of a Logistic Regression model, a Random Forest model and a Support Vector Machine model. Models were evaluated by comparing training and testing accuracy as well as F-1 score.

Table 2: Comparing accuracy scores and F-1 scores for 3 models with hyperparameter tuning using GridSearchCV:

	Logistic Regresson	Random Forest	Support Vector Machine
Training Accuracy	0.78	0.88	0.77
Testing Accuracy	0.75	0.73	0.75
F-1 score	0.70	0.69	0.70

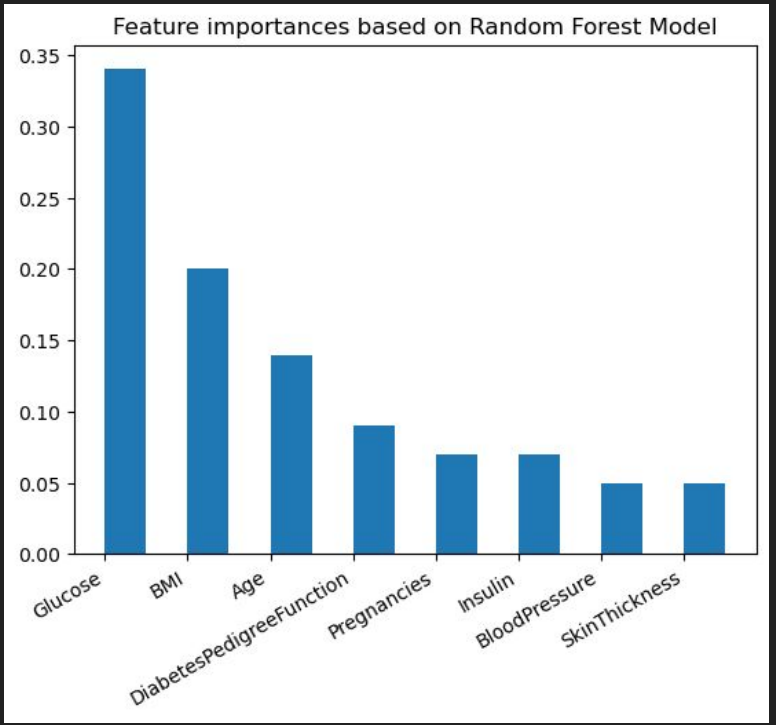
There is overfitting in the Random Forest model. The Logistic Regression model and SVM model performed similarly.

Modeling 3: Dropping some features.

We looked at feature importances based on the Random Forest model and refitted models with the 2 least important features (‘BloodPressure’ and ‘SkinThickness’) dropped.

Coincidentally, ‘SkinThickness’ is the column with the second highest missing values.

Best estimators for Logistic Regression, Random Forest and Support Vector Machine after GridSearchCV were refitted to see if the models have improved.



Modeling 3: Dropping some features

Table 3: Comparing accuracy scores and F-1 scores for 3 models after GridSearchCV and dropping 2 features with least importance as determined by the Random Forest model:

	Logistic Regresson	Random Forest	Support Vector Machine
Training Accuracy	0.77	0.88	0.77
Testing Accuracy	0.78	0.79	0.77
F-1 score	0.75	0.76	0.73

Dropping the features ‘BloodPressure’ and ‘SkinThickness’ improves the test metrics in all 3 models.

Although the Random Forest model has the highest accuracy score and F-1 score, its gap between training and testing accuracy indicates that it is potentially over-fitting.

The Logistic Regression model has nearly as good of testing accuracy score and F-1 as Random Forest, and it might predict more accurately with new data.

Conclusion

A Logistic Regression model trained with the following hyperparameters and features gave the best metrics:

hyperparameters:

{C=10.0, penalty='l1', solver='liblinear'}

Features used:

{'Glucose', 'BMI', 'Age', 'DiabetesPedigreeFunction', 'Pregnancies', 'Insulin'}

The model can achieve an accuracy score of 0.78 and F-1 score of 0.75 while not overfitting to the training data.