Mukherjee, Manas

Machine Learning Capstone Project On Diabetes Classification

Date – June 14, 2018

Table of Contents

Introduction

A. Project Overview

Diabetes is a major chronic disease that affects more than 30 million people in the United States. The American Diabetes Association released new research on March 22, 2018 estimating the total costs of diagnosed diabetes have risen to $327 billion in 2017 from $245 billion in 2012, when the cost was last examined.

It occurs either when the pancreas does not produce enough insulin or when the body cannot effectively use the insulin it produces. Insulin is a hormone that regulates blood sugar. Hyperglycemia, or raised blood sugar, is a common effect of uncontrolled diabetes and over time leads to serious damage to many of the body's systems, especially the nerves and blood vessels.

Leading companies, as well as startups, are actively working on this specific healthcare domain and coming up with AI-powered applications on use-cases like Glucose Monitoring Systems, Nutrition Coaching, Early Diagnosis Tools etc.

Datasets and Inputs

Source :https://www.kaggle.com/uciml/pima-indians-diabetes-database
This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases.
Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years old of Pima Indian heritage.

Background details and important characteristics of Pima people

The Pima are a group of Native Americans living in an area consisting of what is now central and southern Arizona. The majority population of the surviving two bands of the Akimel O'odham is based on two reservations: the Keli Akimel O'otham on the Gila River Indian Community (GRIC) and the On'k Akimel O'odham on the Salt River Pima-Maricopa Indian Community (SRPMIC).

The Keli Akimel O'odham and the Onk Akimel O'odham have various environmentally based health issues related to the decline of their traditional economy and farming. They have the highest prevalence of type 2 diabetes in the world, much more than is observed in other U.S. populations. While they do not have a greater risk than other tribes, the Pima people have been the subject of intensive study of diabetes, in part because they form a homogeneous group.

B. Problem Statement

The goal of this project is to build a machine learning model to predict the onset of diabetes based on some diagnostic measures. The expected outcome of this project is to contribute to the clinical research study related to diabetes.

MAIN OBJECTIVE: Build a mathematical model to predict whether or not a patient has diabetes, based on certain diagnostic measurements.

SECONDARY OBJECTIVE: Identify important features or a set of clusters of features that are highly related to diabetes.

The data includes a binary prediction variable(label) outcome. In this task, it is required to build a function that will map the set of input features with the binary outcome(true/false). I would consider this as a supervised 'binary classification' task.

C. Evaluation Metrices

While evaluating the performance of the benchmark models and best solution model, I will consider the 'model accuracy' using train/test split or K-fold cross-validation. The input dataset is bit unbalanced(~35%:65%) w.r.t the prediction variable 'outcome', so I will leverage the 'recall' factor and use 'confusion matrix' to measure different types of prediction. Since this model will be used to disease diagnosis, we will try to minimize the 'false negative' type error (i.e A result that appears negative when it should not).

In this case, we would use F2 Score since it favors 'recall' over 'precision'.

Related formula

- **Fβ score = (1+β2)** * **(precision** * **recall / (β2*** **precision)+recall)**
  In this cases, β = 2.
- **Precision = [True Positives/(True Positives + False Positives)]**
- **Recall = [True Positives/(True Positives + False Negatives)]**

2. Analysis

A. Data Exploration

This dataset has 768 patients' record with 8 corresponding features. There is a label variable which indicates if the patient has diabetes.

Here are 5 sample records.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1
5	5	116	74	0	0	25.6	0.201	30	0

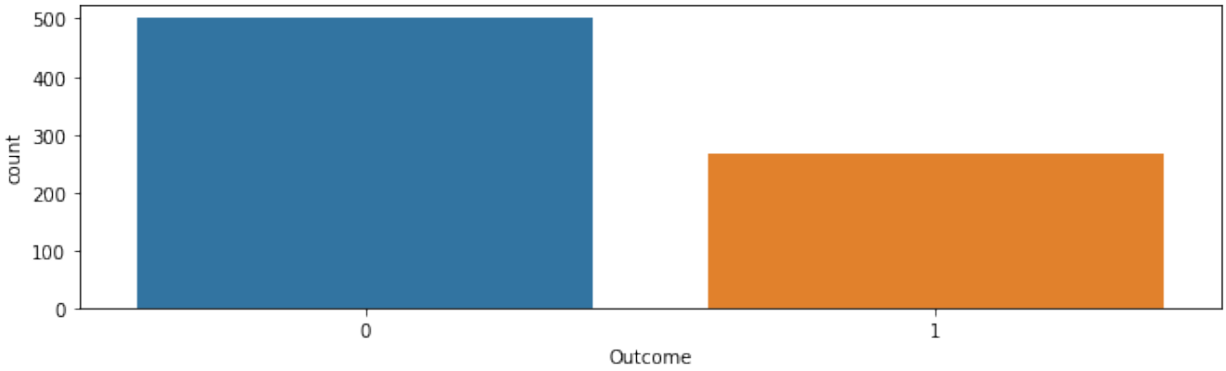
Here is the details related to each feature of this dataset.

Feature Name	Description	Datetype
Pregnancies	Number of times pregnant	Numeric
Glucose	Plasma glucose concentration a 2 hours in an oral glucose tolerance test	Numeric
BloodPressure	Diastolic blood pressure (mm Hg)	Numeric
SkinThickness	Triceps skin fold thickness (mm)	Numeric
Insulin	2-Hour serum insulin (mu U/ml)	Numeric
BMI	Body mass index (weight in kg/(height in m)^2)	Numeric
DiabetesPedigreeFunction	Diabetes pedigree function	Numeric
Age	Age (years)	Numeric
Outcome	Class variable (0 or 1)	Numeric

In this dataset, all features are of a numeric type. Here are some descriptive stats of the 8 features.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

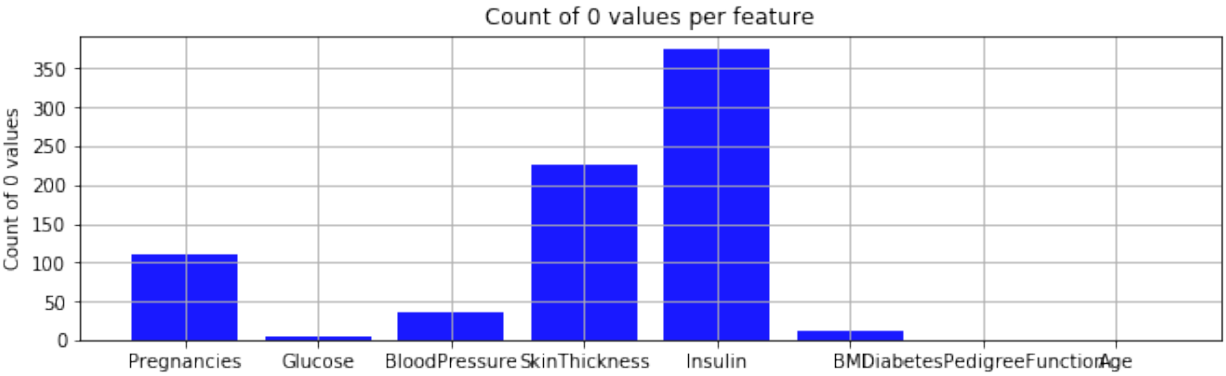
Total number of individuals: **768**
Individuals with diabetes: **268**
Individuals without diabetes: **500**
Percentage of individuals with diabetes: **34.90**%



Note - Since 500 out of 768 individuals don't have diabetes, so it would be considered as an unbalanced dataset.

B. Exploratory visualization

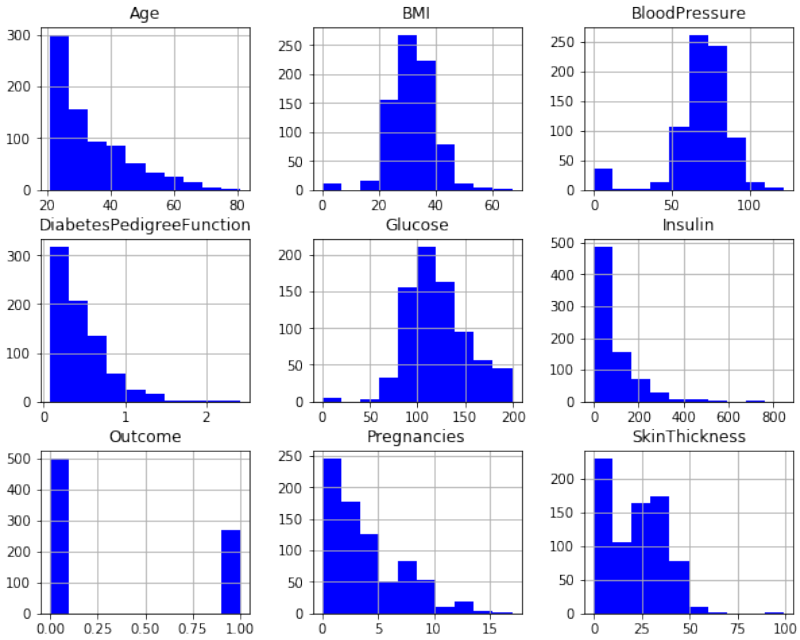
2b-1. Feature specific missing or 0 value count



Observation

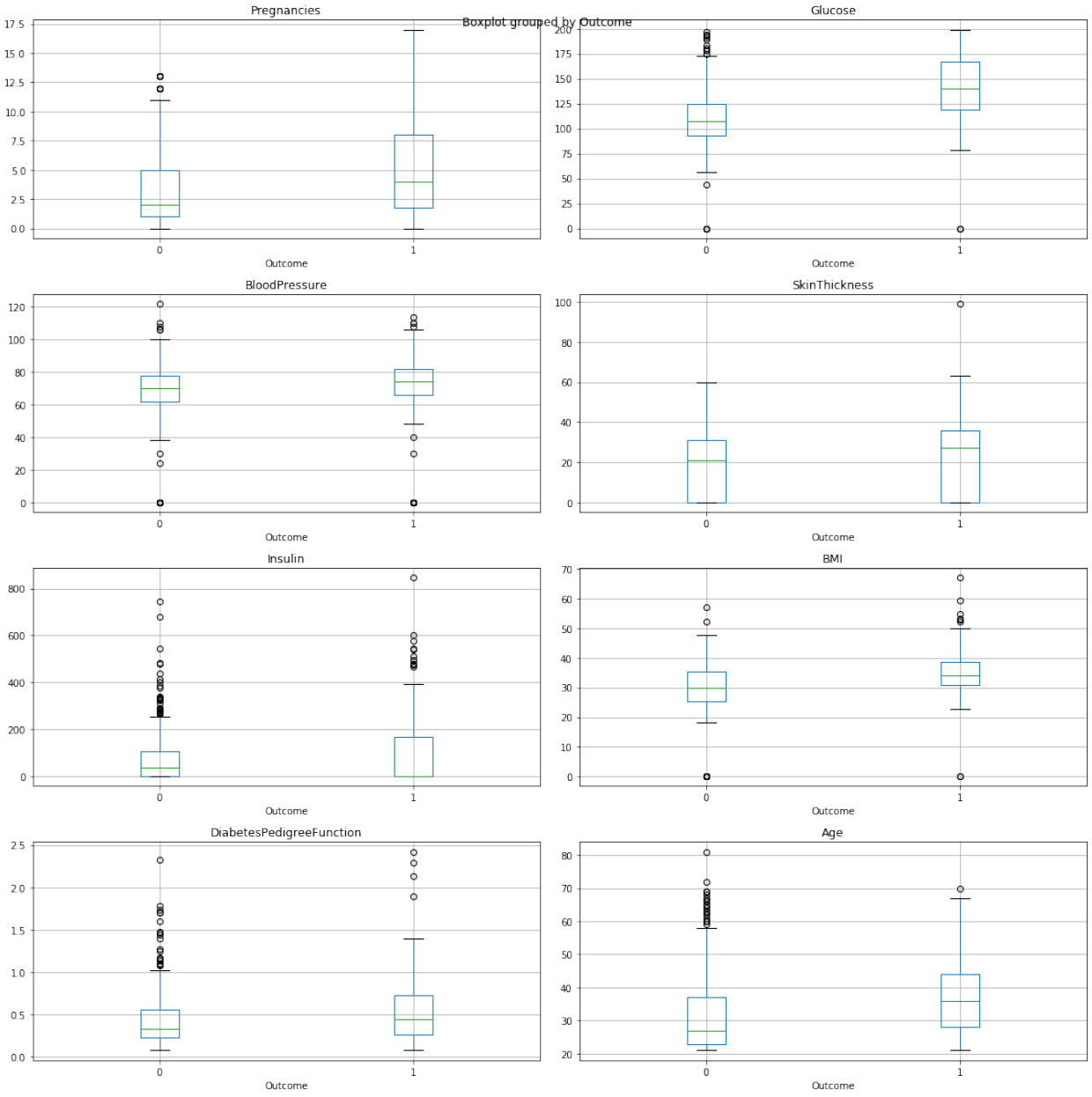
Features like 'Skin Thickness', 'Insulin' have a significant number of 0 values.

2b-2. Feature distribution visualization using histogram and boxplots



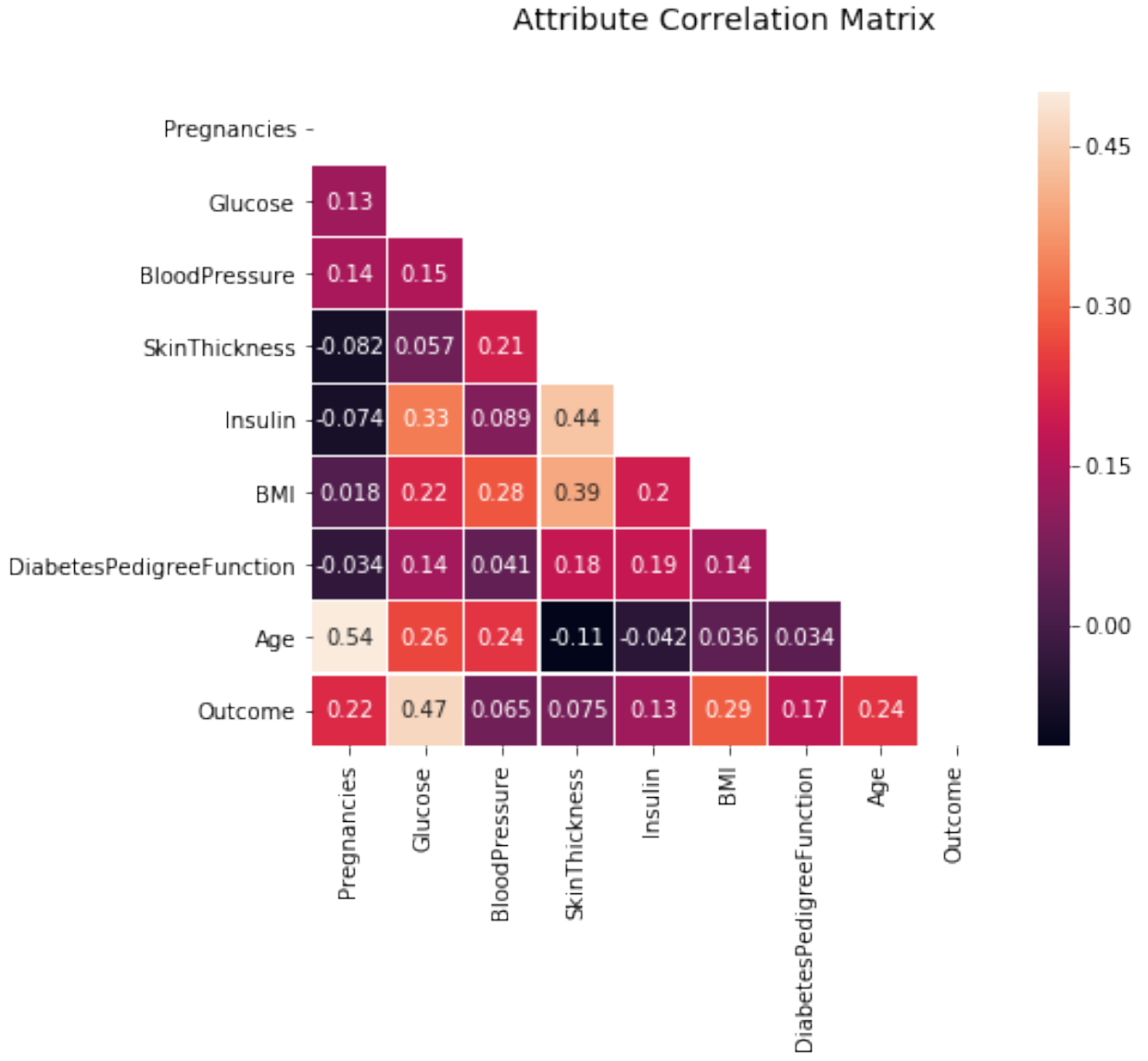
Observation

1. Most attributes like Age, Insulin etc. are highly skewed towards left.
2. A Significant number of missing or zero values in features like 'Insulin' and 'SkinThickness' have an effect on their distributions. While building the model, these should be imputed properly.



Feature comparision with Outcome(0/1)	Comments(w.r.t median and outlier)
Pregnancies	Patients with higher number of pregnancies are prone to diabetes
Glucose	High Glucose amount increases the chance of diabetes
BloodPressure	NA
SkinThickness	Higher value of 'Triceps skin fold thickness (mm)' gives an indication of diabetes
Insulin	There are too many missing or zero values. We need to impute the missing values
BMI	Higher BMI value increases the chance of diabetes
DiabetesPedigreeFunction	Patients with high DiabetesPedigreeFunction are prone to diabetes
Age	Median age of diabetes patients is ~8 yrs more than same of non-diabetic patients

2b-3. Feature correlations visualization



Observation

Following pairs have a high correlation score.

1. Age and Pregnancies (0.54)
2. Outcome and Glucose (0.47)
3. Insulin and SkinThickness (0.44)
4. BMI and SkinThickness (0.39)

C. Algorithms and Techniques

There are several out of the box classifiers available in the Scikit-learn package. For this binary classification problem, following classifiers are used.

1. Logistic Regression
2. Gaussian Naive Bayes (GaussianNB)
3. Decision Trees
4. Ensemble Methods (AdaBoost, Random Forest, Gradient Boosting)
5. Nearest Neighbors (KNeighbors)
6. Stochastic Gradient Descent Classifier (SGDC)
7. Support Vector Machines (SVM)

Here are some details on each one of the above algorithms and their appropriateness with respect to this classification problem.

1. Logistic Regression
Here the output is categorical(presence and absence of diabetes of a given patient). Since it is a classification problem, 'Logistic Regression' is one of the obvious choice to start with.

2. Gaussian Naïve Bayes
GNB works well with small dataset. Here most numerical features have almost normal distribution which is essential for GNB. It might not work if the features are inter-dependent.

3. Decision Tree
It uses tree-like model to predict the target variable(in this case – outcome). It automatically does feature selection and provides information on feature importance. This model is prone to over fitting. This dataset has multiple key attributes that can influence the prediction.

4. K-Nearest Neighbors
I doesn't assume the type/distribution of data. In general, it is memory and processing intensive but since this data set is relatively small, it should work well. It might fail because of the influence of some irrelevant features.

5. Random Forest
It is an ensemble learning method that works by constructing random decision trees at the training time and outputting the class that is the mode of the classes of the individual trees. Random decision forests correct for decision trees' habit of 'overfitting' in their training set. Since this dataset has several numeric inputs, it should work well. Though computationally intensive, that should not have much effect on this small dataset.

6. Gradient Boost Algorithm
It is one of the best algorithms from the ensemble method family. It handles outliers well and accommodates missing values which is applicable to this dataset.

7. Ada Boost Algorithm
AdaBoost is best used to boost the performance of decision trees on binary classification problems. It tries to correct the miss-classified points repeatedly, so it doesn't do a good job when data quality is bad or data has too many outliers. This dataset has a well-structured data set. Before applying this algorithm, we can easily normalize attributes which have high variance. Since this is a classification problem which can be solved using Decision Tree. AdaBoost will help to increase the model performance significantly.

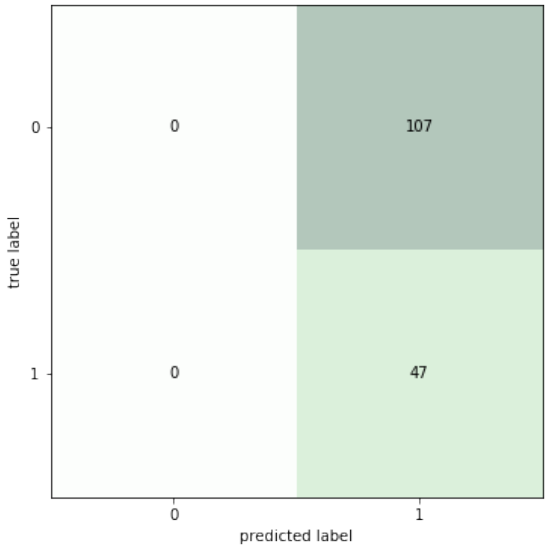
D. Benchmark

Benchmark-1 : Naive Model

Since 500 out of 768 individuals don't have diabetes, so we would consider this an unbalanced dataset. A naive benchmark model for this problem would be to predict that none of the patients have diabetes.

Naive Model KPI

- naive_accuracy_score : 0.31
- naive_recall_score : 1.0
- naive_f2_score : 0.69
- TN = 0, FP = 107, FN = 0, TP = 47



Benchmark-2 : Domain Knowledge-based Model

According to the Wikipedia "They have the highest prevalence of type 2 diabetes in the world, much more than is observed in other U.S. populations".

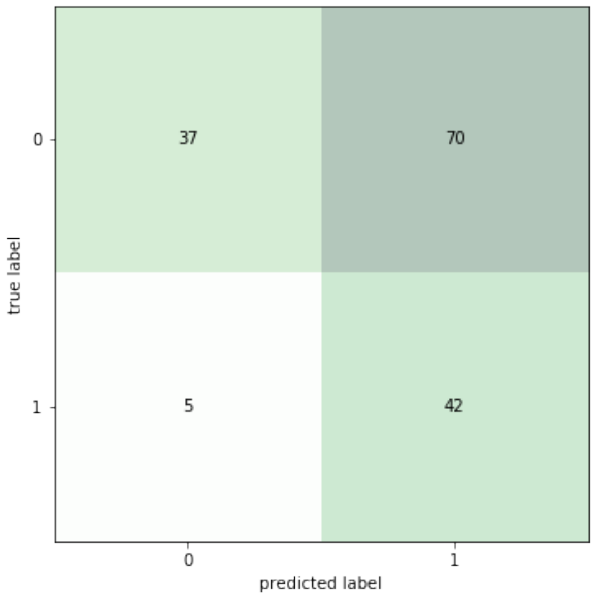
Type 2 diabetes (formerly called non-insulin-dependent, or adult-onset) results from the body's ineffective use of insulin. **Type 2 diabetes comprises the majority of people with diabetes around the world and is largely the result of excess body weight and physical inactivity.**

According to several legitimate sources, Type-2 diabetes is correlated with obesity and/or BMI. Following article suggests that **Overweight was defined as having a body mass index between 25 and 29.9**.http://www.diabetesincontrol.com/body-mass-index-and-type-2-diabetes-risk/

We'll consider the **average(27.5)** of the above BMI range(25 to 29.9) as a threshold of this domain knowledge-based model. If a patient's BMI is above 27.5 then this model will diagnosis 'diabetes'.

Domain Knoledge Based Model KPI

- naive_accuracy_score : 0.51
- naive_recall_score : 0.89
- naive_f2_score : 0.7
- TN = 37, FP = 70, FN = 5, TP = 42

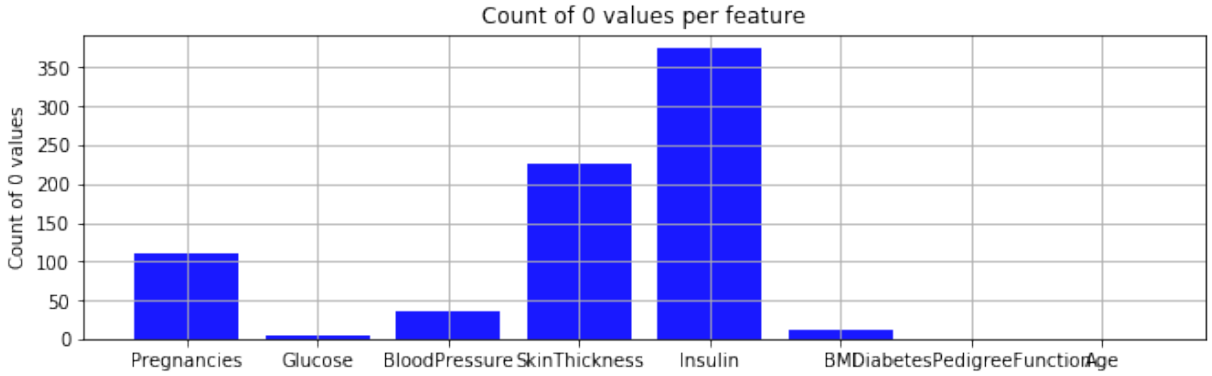


3. Methodology

A. Data Preprocessing

Before using this data as input for ML classifier, it needs to be cleaned, formatted, and restructured. This preprocessing should help with the outcome(diabetes/non-diabetes) and predictive power of the model.

3a-1. Data Cleaning (Impute missing values)

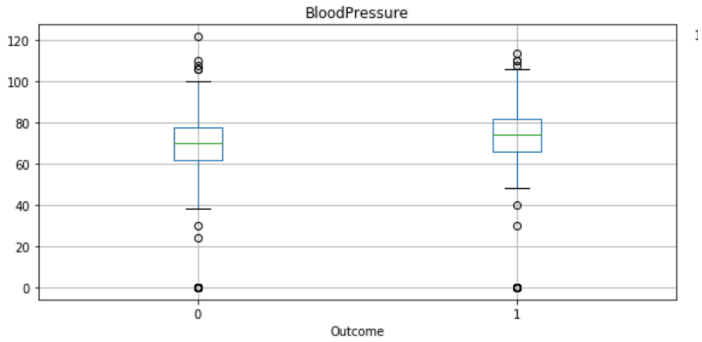


According to the above fig, following 4 features have highest number of missing values.

1. Insulin,
2. SkinThickness
3. Pregnancies
4. BloodPressure
5. BMI

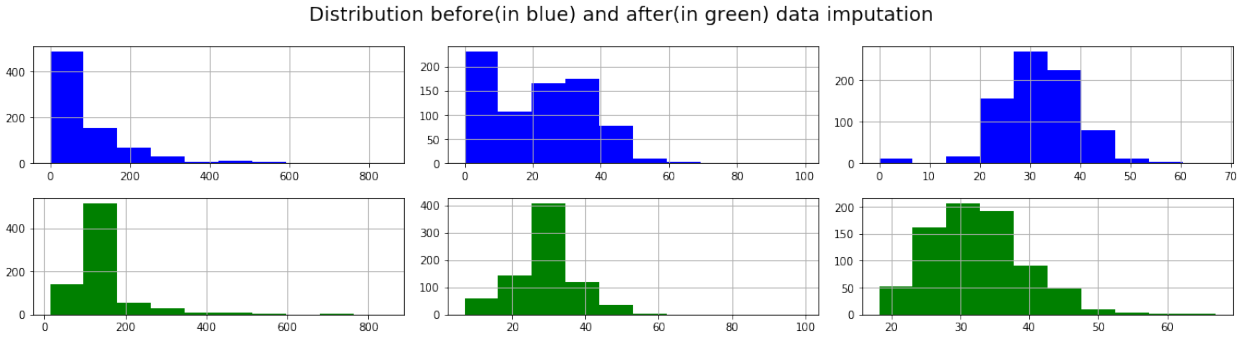
Since none other than the 'pregnancies' feature can have '0' value, all '0' values of 'Insulin', 'SkinThickness' and 'BMI are imputed with their column specific mean value.

The median blood-pressure of diabetic and non-diabetic group is relatively small. Median based imputation can cause some negative effect on the model's prediction. That's why no imputation was done for this attribute.



PN – In the mean value calculation, entries with ‘zero’ and ‘missing values’ were excluded.

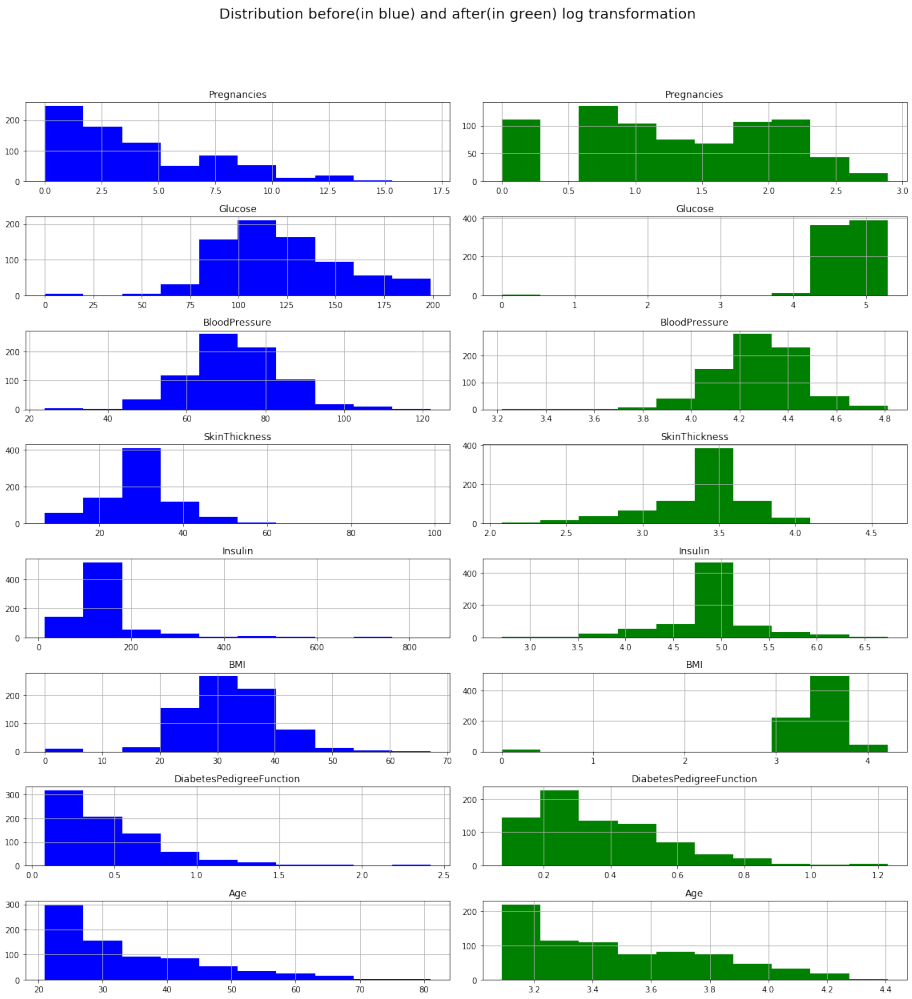
	Exclude entries with ‘zero’ or ‘NA’ values		Comment
Feature Name	**Before**	**After**	
Insulin	79.80	155.54	The median got doubled. It might have some -ve effect on the predictive model
SkinThickness	20.54	29.15	50% increase. Same as above
BMI	31.99	32.46	



PN – Above histogram shows distribution of ‘Insulin’, ‘SkinThickness’ and ‘BMI features.

3a-2. Transform numeric values in log scale (minimize the effect of outliers)

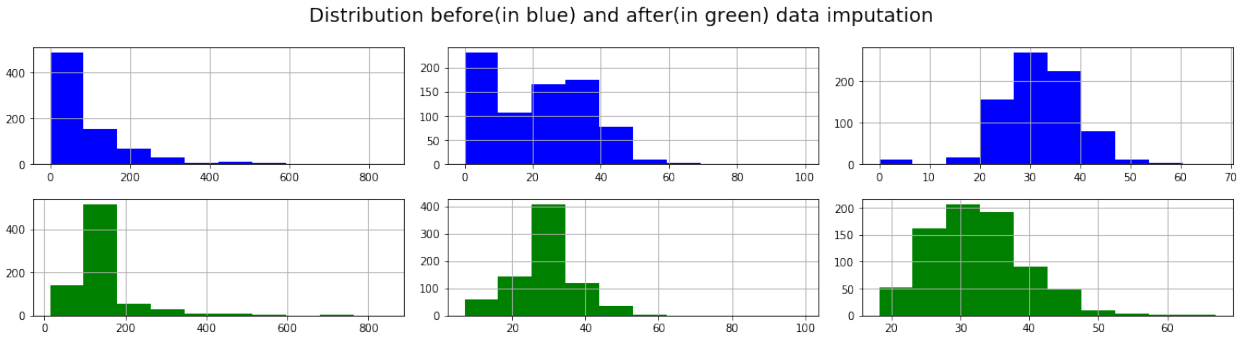
Fields like 'Age', 'Pregnancies', 'DiabetesPedigreeFunction' are highly skewed towards left. To minimize the effect of any very large and very small values, log transformations is performed.



3a-3. Scale numeric values (0 to 1 scale)

Since features are not measured in the same scale, normalization is performed to ensure that each feature is treated equally while applying a supervised classifier.

Following diagram shows the distribution(before and after scaling) of ‘Insulin’, ‘SkinThickness’ and ‘BloodPressure’.



B. Implementation

Here is the high level details of steps performed while training and evaluating different supervised algorithms on the transformed dataset.

1) Create a training and testing dataset

   Data is split in 8:2 proportion to construct the training and testing set. After splitting, **training and testing sets got 614 and 154 sample records.**

2) Create instances of appropriate classifiers available in the scikit-learn package.

   a) GaussianNB
   b) DecisionTreeClassifier
   c) KNeighborsClassifier
   d) LogisticRegression
   e) RandomForestClassifier
   f) GradientBoostingClassifier
   g) AdaBoostClassifier
   h) MLPClassifier

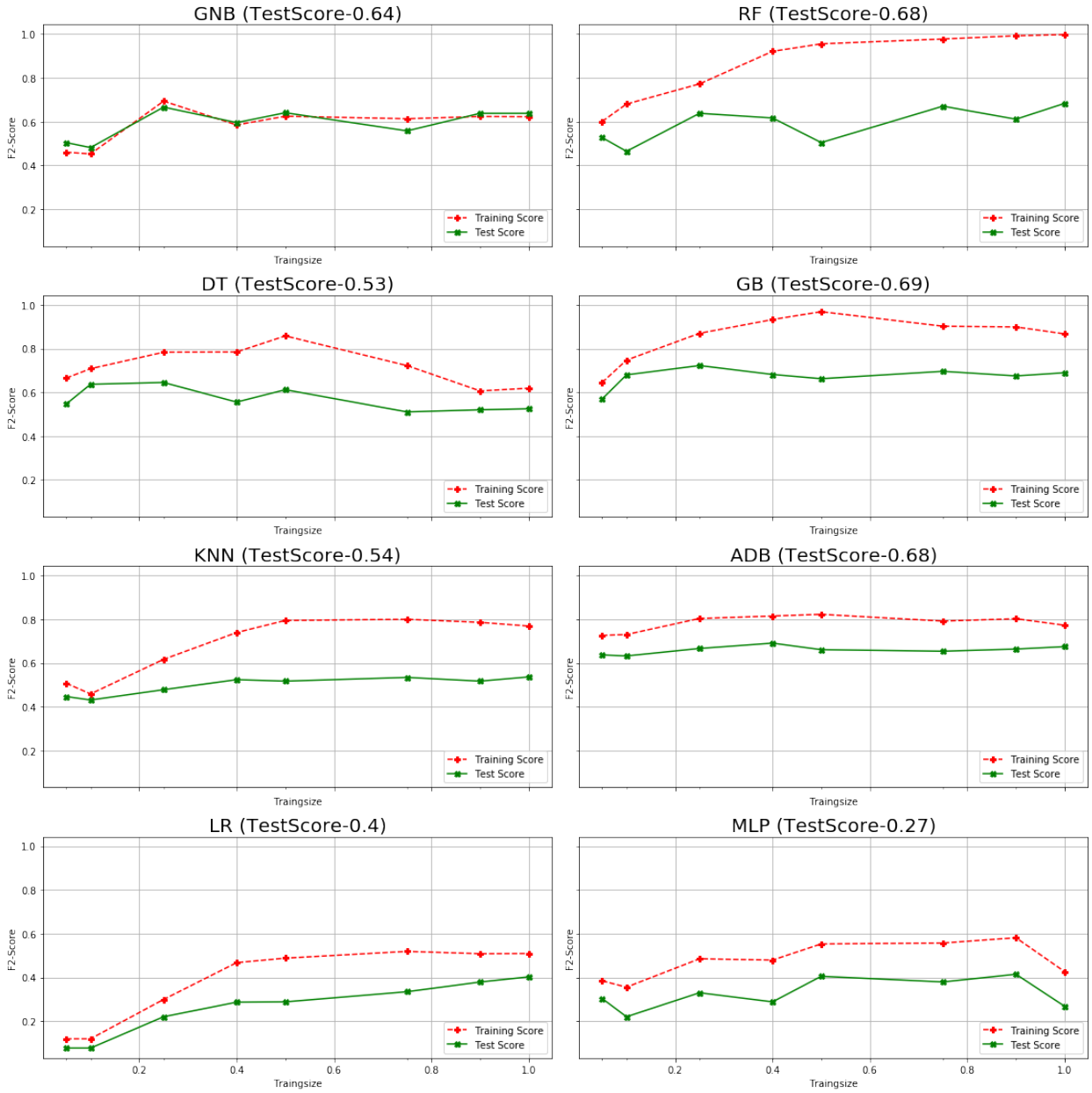
3) Create a training/testing pipeline

   a) Define a set of training samples with increasing number of entries

      Created 8 training samples with the following 8 random proportion of the training data. 5%, 10%, 25%, 40%, 50%, 75%, 90%, 100%

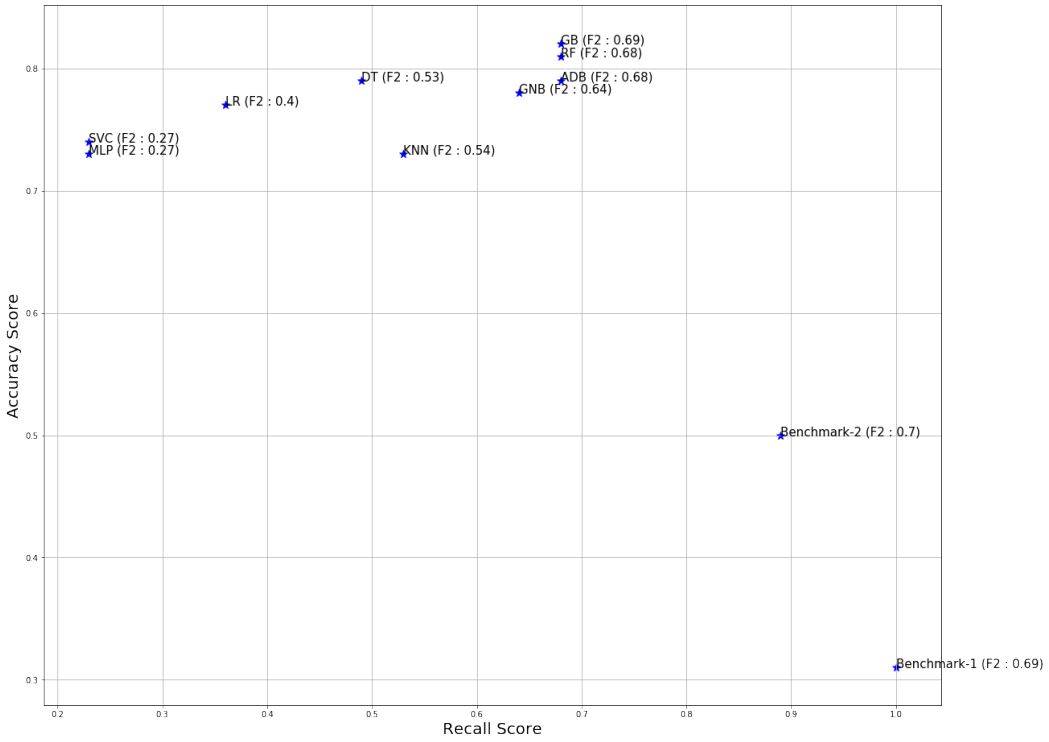
   b) For each classifier listed above,

      i) For each training sub-sets [5%, 10%, 25%, 40%, 50%, 75%, 90%, 100% of Training]
         (1) Train the classifier with sub-set of training data
         (2) Calculate Training f2-score using the first 300 training data
         (3) Calculate Testing f2-score using the full test set
         (4) Plot Training and Testing f2-scores along y-axis and the sample training size in x-axis to track the learning progress.


GNB (TestScore-0.64)
RF (TestScore-0.68)
DT (TestScore-0.53)
GB (TestScore-0.69)
KNN (TestScore-0.54)
ADB (TestScore-0.68)
LR (TestScore-0.4)
MLP (TestScore-0.27)
F2-Score
Traingsize
Training Score
Test Score

4) Model Comparison and Selection

After exploring all classifiers, results are plotted along with the baselines.



Based on the comparison, Gradient Boost and Random Forest both performed really well by maintaining a high recall, accuracy and F2 score. Since it is known that GB performs better than RF for binary classification, GB is selected as the obvious choice for further tuning.

In the following table, each model and its corresponding feature score are mentioned.

	Name	F2-Score	Recall	Precision Score	Accuracy Score	DecisionTypes	TrainingTime
9	Benchmark-2	0.70	0.89	0.38	0.50	TN=37, FP=70, FN=5, TP=42	0.000
8	Benchmark-1	0.69	1.00	0.31	0.31	TN=0, FP=107, FN=0, TP=47	0.000
5	GB	0.69	0.68	0.73	0.82	TN=95, FP=12, FN=15, TP=32	0.075
4	RF	0.68	0.68	0.70	0.81	TN=93, FP=14, FN=15, TP=32	0.019
6	ADB	0.68	0.68	0.65	0.79	TN=90, FP=17, FN=15, TP=32	0.082
0	GNB	0.64	0.64	0.64	0.78	TN=90, FP=17, FN=17, TP=30	0.002
2	KNN	0.54	0.53	0.56	0.73	TN=87, FP=20, FN=22, TP=25	0.001
1	DT	0.53	0.49	0.74	0.79	TN=99, FP=8, FN=24, TP=23	0.002
3	LR	0.40	0.36	0.74	0.77	TN=101, FP=6, FN=30, TP=17	0.001
7	MLP	0.31	0.28	0.65	0.73	TN=100, FP=7, FN=34, TP=13	0.180

PN - Gradient Boost and Random Forest both performed really well. In both cases, recall score(0.68), and number of ‘false negative’ prediction count(15) are same.

C. Model Refinement

Sklearn's GridSerachCV was used to fine tune the parameters of the 'Gradient Boost(GB)' and 'Random Forest(RF)' classifiers.

At first, RF was tuned by varying the number of n_estimator parameter but it didn't produce any noticeable improvement.

For GB, following parameter grid was used. The parameter 'n_estimator' that represents the number of trees in the model worked really well with its value as 250. Tuning on other parameters like 'learning_rate', or 'max_depth' didn't make any difference.

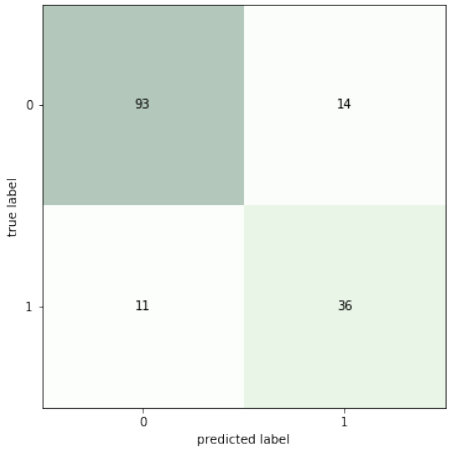
```
parameters = {
            'learning_rate': [0.1, 0.01],
            'n_estimators':[50,75,100,200,250, 260, 280],
 '          max_depth': [3,4]
}
```

	F-Score	Recall	Precision	Accuracy	Prediction Class (TN, FP, FN, TP)
Un-optimized model	0.69	0.68	0.73	0.82	95, 12, 15, 32
Optimized model	**0.76**	**0.77**	**0.72**	**0.84**	**93, 14, 11, 36**

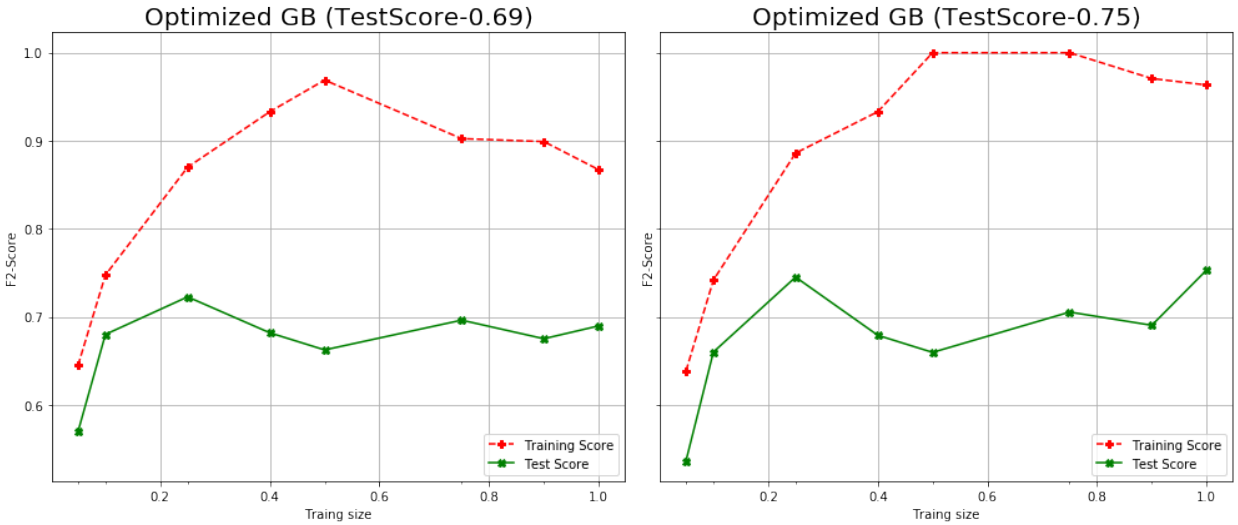
4. Results

A. Final Model Evaluation and Validation

Based on the above analysis and model comparison chart, Gradient Boosting classifier was chosen and further optimized. After parameter(n_estimator) tuning, this model out performed all other classifiers.
Here is the confusion matrix of the optimized model.



Here is the learning curve comparison between the unoptimized and optimized GB models. The optimized GB model has a much better score, recall and less number of 'false negative' prediction, but it has clearly over-fitted the data.

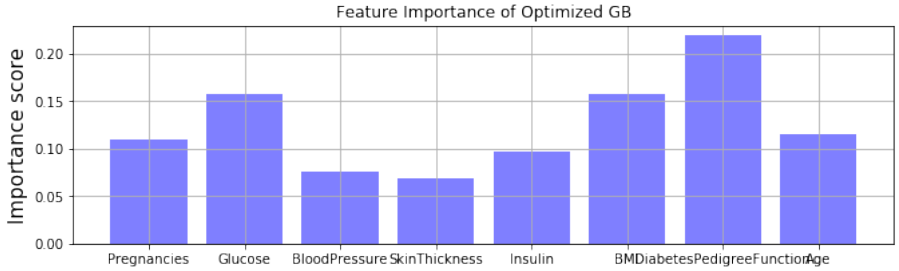


The following general purpose pruning techniques were tried to remove the over-fitting, but it neither fixed the over-fitting nor improved the overall score.

1. Reduce number of leaf nodes(ref - max_leaf_nodes)
2. Restrict the size of the sample leaf (ref - max_leaf_nodes)
3. Reduce the depth of the decision tree (ref - max_depth)

More training sample and/or less important feature removal should help to remove this over-fitting problem from this model.

The following chart shows the feature importance of the optimized GB model. According to this chart, 'DiabetesPedigreeFunction', 'BMI' and 'Glucose' are the top three important features required for predicting diabetes.

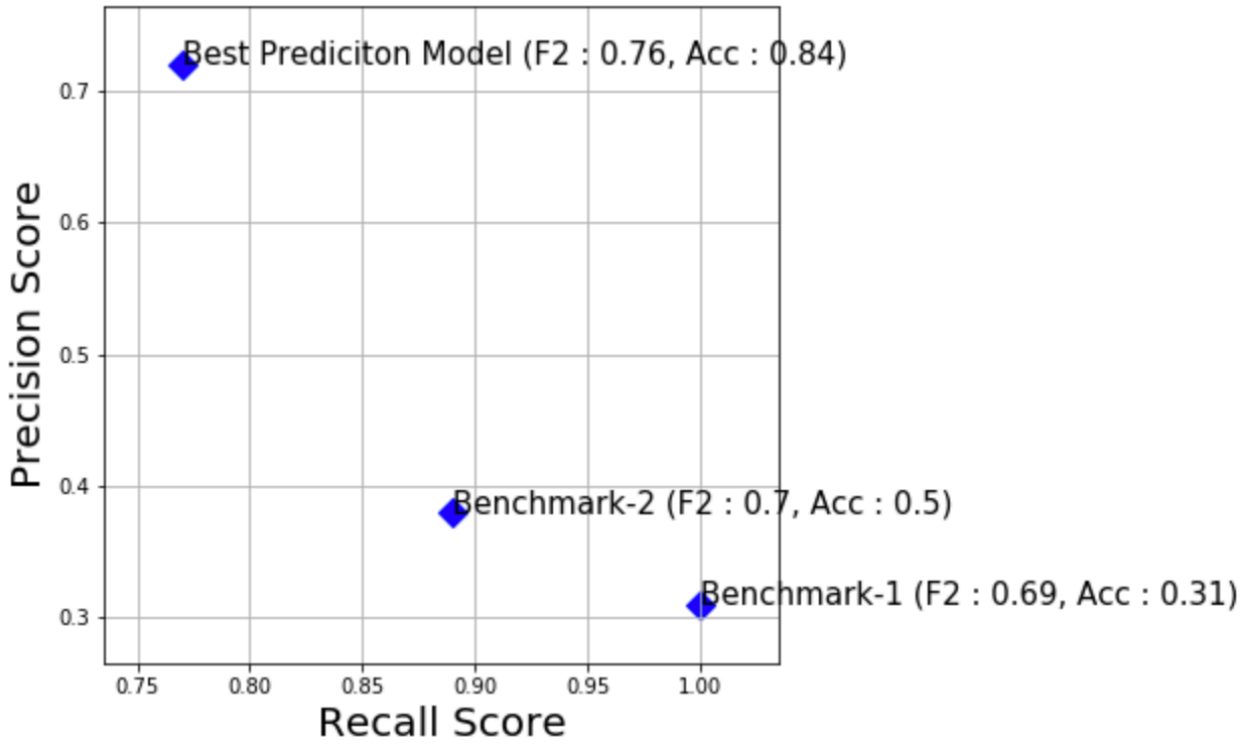


Since this dataset has too many missing values for features like 'insulin' or 'skinThickness', it would be a bit sensitive/dependent on any change(related to missing values) in the data. This model is not mature enough to be applied on any real diabetic prediction use-case yet, but the same approach can be leveraged with exhaustive training dataset to build a much more reliable and production-grade model.

B. Justification

Following chart and scatter plot shows a comparative analysis of the best prediction model with respect to the two benchmark models.
Overall, the final model produced much better F2, Precision and Accuracy score.
Recall score of the 1st benchmark model is 1 because it blindly predicts all patients have diabetic. Similarly, the recall score of the second benchmark model is entirely dependent on the single feature – 'BMI' and maintains a very poor precision score.
Though the final model couldn't exceed the recall score of the two naïve benchmark model, but overall it is much more balanced than the benchmark models w.r.t precision and recall scores.

Name	F2-Score	Recall	Precision Score	Accuracy Score	DecisionTypes
Best Prediciton Model	0.76	0.77	0.72	0.84	TN=93,FP=14,FN=11,TP=36
Benchmark-2	0.70	0.89	0.38	0.50	TN=37, FP=70, FN=5, TP=42
Benchmark-1	0.69	1.00	0.31	0.31	TN=0, FP=107, FN=0, TP=47

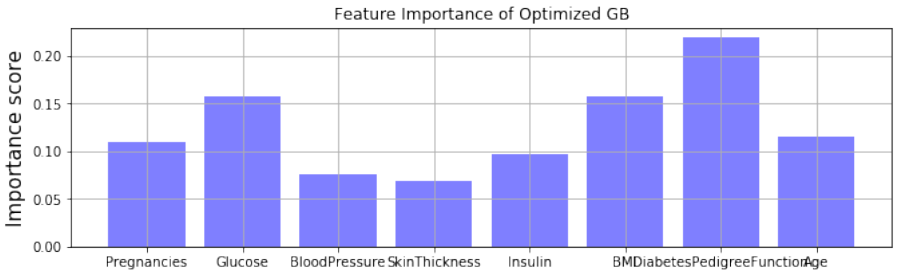


5. Conclusion

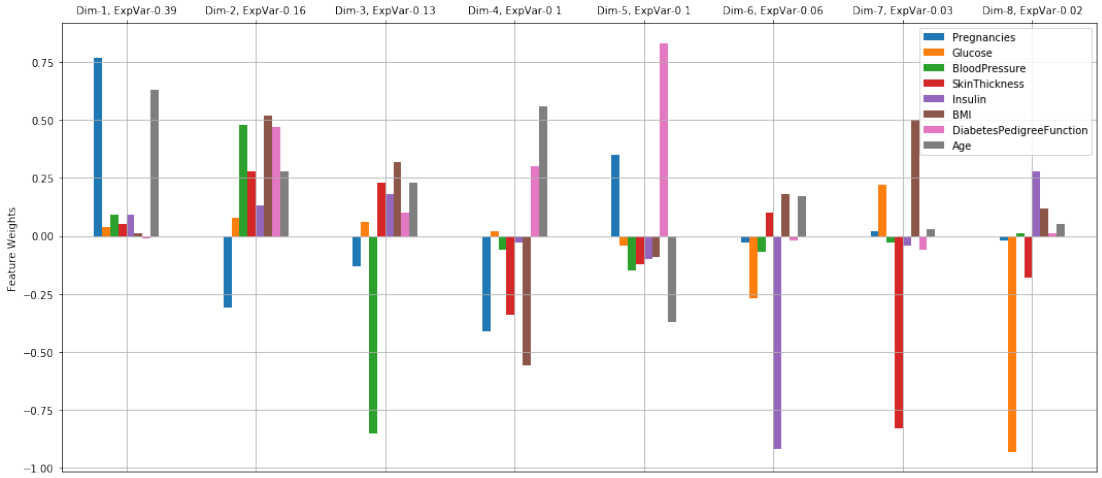
A. Free-Form Visualization

The final model considers the following three features as the most important ones required to diagnosis diabetes.

1. DiabetesPedigreeFunction
2. BMI
3. Glucose



Further investigation was done using principal component analysis to identify different feature-clusters.



The above diagram shows 8 principal components/dimensions and their dependencies on individual features like 'Glucose', 'Insulin' etc.
The first 4 dimensions explain ~78%(0.39+0.16+0.13+0.1) variance of the data
Some high level details on the first 3 dimensions are mentioned in the following table.

Dimensions	Explained Variance	Details
Dim1	0.39	This is the most prominent PC that explains ~40% of the total variance of the data. It suggests 'Pregnancies' and 'Age' are related.
Dim2	0.16	This component is dependent on all features other than 'pregnancies'.
Dim3	0.13	Dim3 is another composite feature which is negatively dependent on BloodPressure and Pregnancies.

B. Reflection

This project outlines a machine learning approach(binary classification) to build a predictive model that can diagnose diabetes based on a set of input features.

The process used for this project can be summarized using the following steps.

1. Found this dataset from the UCI archive
2. Defined the problem statement
3. Defined two naïve benchmark models and calculated its KPI
4. After doing some exploratory data analysis, a thorough data pre-processing was done
5. Feature specific missing values were imputed with the corresponding mean values.
6. The impact of outliers were reduced by doing some non-linear transformation using log function.
7. All features were normalized before feeding it into the classifiers.
8. Created an model pipeline that helped to evaluate multiple supervised classifiers using different sizes of training samples.
9. Top two classifiers were selected for further optimization using parameter turning.
10. Using 'GridSearchCV' method, multiple parameters were tested and the best model was selected based on the KPI (precision, recall and low fale negative predictions).
11. Further analysis were done to understand the feature importance and feature-clusters w.r.t variance in the data.

I found that step 5 related to 'data imputation' was a bit tricky because there were a lot of missing values marked a 0 which is also a valid value for some features. In step 8, a few classifiers were over-fitted or had a flat learning curve. I spent some significant amount of time in understanding this behavior and finding a relation between the over-fitting and the data imputation. This project has given me an opportunity to understand the domain/data in greater depth, to explore different data visualization techniques using matplotlib and of course to revise my knowledge related to ML(supervised learning classifiers).

C. Improvements

1. Instead of train/test split, I'll explore K-fold cross validation in greater detail
2. A significant number of values of the Insulin and SkinThickness features are missing. It was imputed with the corresponding mean values. A thorough study is required to analyze the effect of imputation on the classifier selection process and the overall model-performance. In future versions, will explore ways to express(math and stats based) the effect of missing values on the model's performance.
3. Will use Deep Learning (Multilayer Perceptron) to surpass the performance of the existing best model.
4. Will do an in depth analysis on why the final model has 11 false negative predictions that is lowering the F2 score. It is possible to identify the pattern/reason that caused these 11 FN predictions and use that learning for further tuning.

6. References

1) https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/
2) https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/
3) https://medium.com/@adi.bronshtein/a-quick-introduction-to-k-nearest-neighbors-algorithm-62214cea29c7
4) https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/
5) http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.4386&rep=rep1&type=pdf
6) https://stats.stackexchange.com/questions/257328/why-is-boosting-less-likely-to-overfit
7) https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d
8) https://statinfer.com/204-3-10-pruning-a-decision-tree-in-python/
9) http://htmlpreview.github.io/?https://github.com/manas-mukherjee/MLTools/blob/master/src/mlnd-projects/UnsupervisedLearning/customer_segments/report.html
10) https://github.com/udacity/machine-learning/blob/master/projects/capstone/report-example-1.pdf