Diabetes Prediction

What is Diabetes?

Diabetes is a chronic disease that occurs when the pancreas is no longer able to make insulin, or when the body cannot make good use of the insulin it produces. Learning how to use Machine Learning can help us predict Diabetes.

Motivation:

- I wanted to do my analysis on some real-time problem like diseases, which is when diabetes struck my mind as a problem. As it doesn't look like very dangerous but when we see the number of deaths every year it becomes very scary.
- However, my initial plan was to collect data, but that cannot be done without specific collection method or some government papier-works.
- So, instead of making one I turned towards using one.

Source of the data:

- I found this dataset from UCI (machine Learning Repository)
- Original owners: National Institute of Diabetes and Digestive and Kidney Diseases.
- Donor of database: Vincent Sigillito (vgs@aplcen.apl.jhu.edu) Research Center, RMI Group Leader Applied Physics Laboratory the Johns Hopkins University Johns Hopkins Road Laurel, Date received: 9 May 1990
- 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.

About this project: -

- The objective of this project is to classify whether someone has diabetes or not by modeling some everyday biological features.
- Dataset consists of several Medical Variables (Independent) and one Outcome Variable (Dependent)
- The independent variables in this data set are: -'Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin','BMI', 'DiabetesPedigreeFunction', 'Age'
- The outcome variable value is either 1 or 0 indicating whether a person has diabetes (1) or not (0).

About the Dataset

- Pregnancies: - Number of times a woman has been pregnant
- Glucose: - Plasma Glucose concentration of 2 hours in an oral glucose tolerance test
- BloodPressure: - Diastollic Blood Pressure (mm hg)
- SkinThickness: - Triceps skin fold thickness(mm)
- Insulin: - 2-hour serum insulin (mu U/ml)
- BMI: - Body Mass Index ((weight in kg/height in m) ^2)

- Age: - Age(years)
- DiabetesPedigreeFunction: -scores likelihood of diabetes based on family history)
- Outcome: - 0(doesn't have diabetes) or 1 (has diabetes)

A Look at the Data Set:

Here's what my first 5 lines of data looks like:

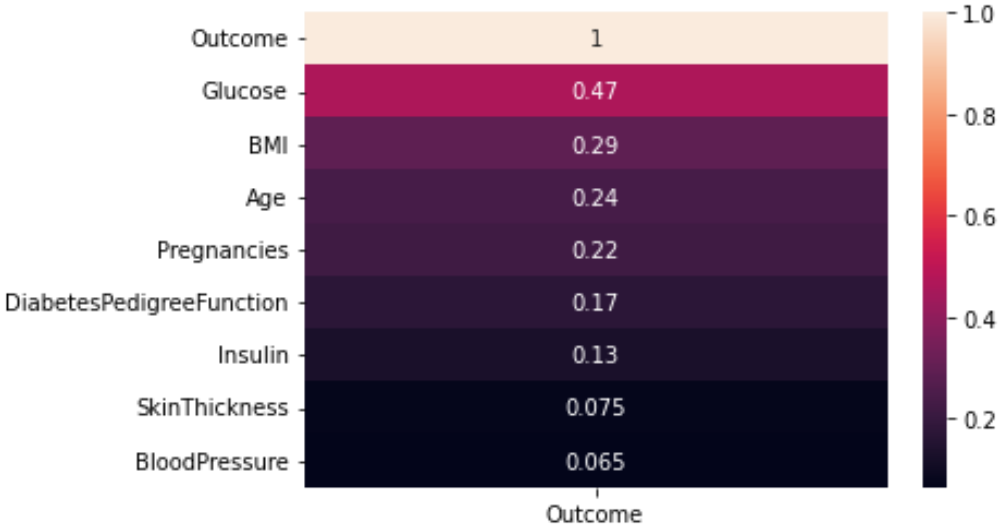
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

Describing it little more:

	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

Looking at the data set we can see many features like, Glucose, Blood Pressure, Skin Thickness, Insulin, and BMI, has 0 values as their minimum which cannot be possible, as a person has a non-zero value since the time of his birth. However, further studying data, it occurred to me that these values just represent the missing or not collected data. Also, zero values would give us wrong prediction, so we must work on them. Before going to data cleaning lets take a look at how the data features are correlated with each other.

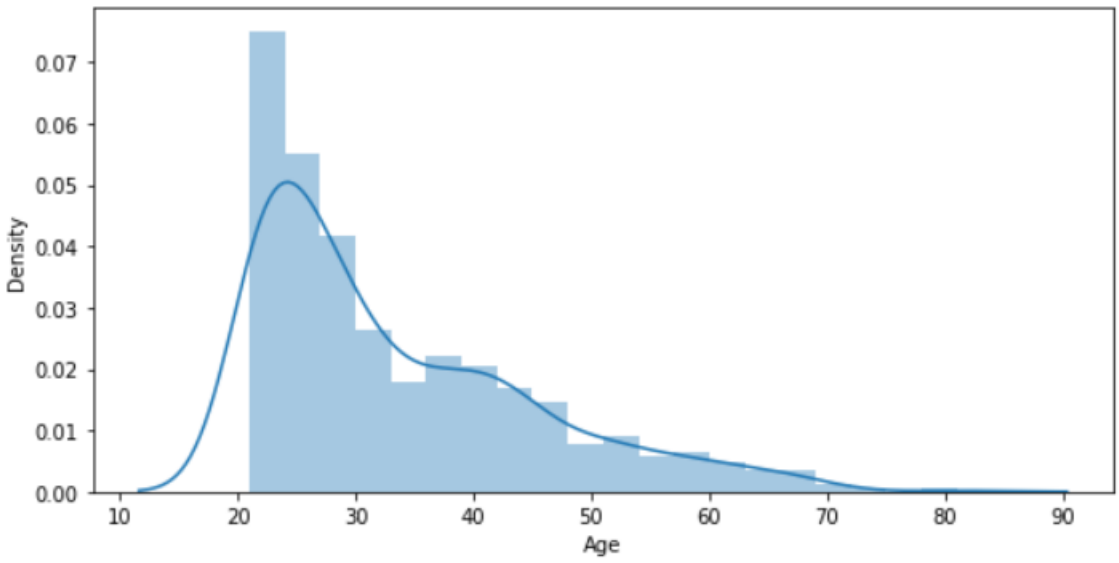
Initial Correlation between Features:



From above correlation we can say that Glucose, BMI, and Age are some of the must required features. We can also conclude that we can remove Skin-Thickness as feature from our data set as it is very less correlated. Also, some features like Blood-Pressure and Insulin looks like very redundant even though from biological perspective, they should be the ones identifying diabetes. This shows that data is in bad state and needs cleaning for further analysis, and we cannot reach any conclusion from here.

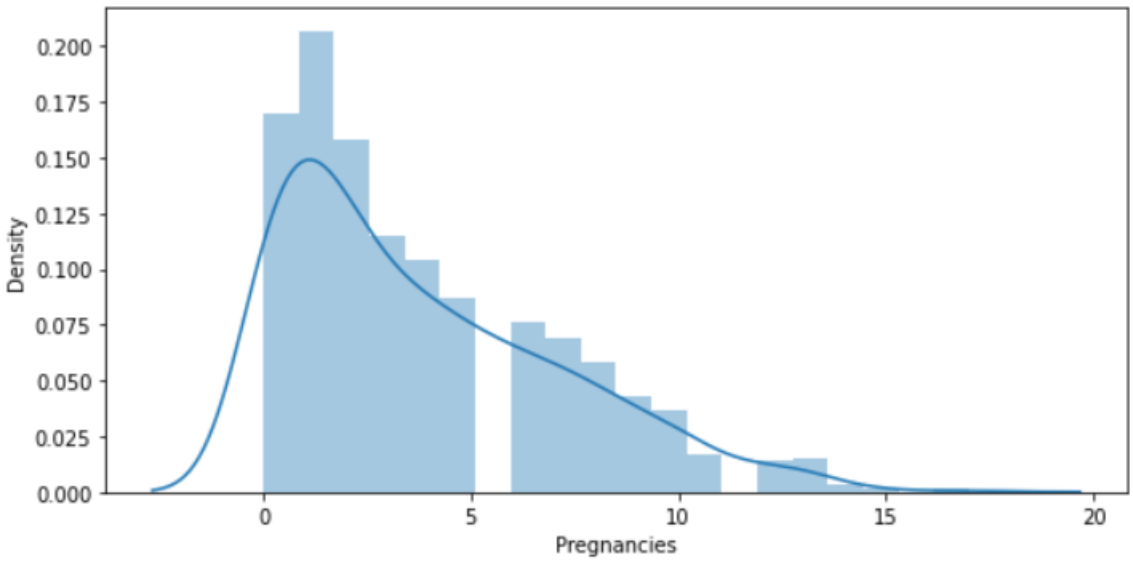
Let's Visualize Some Features:

1.



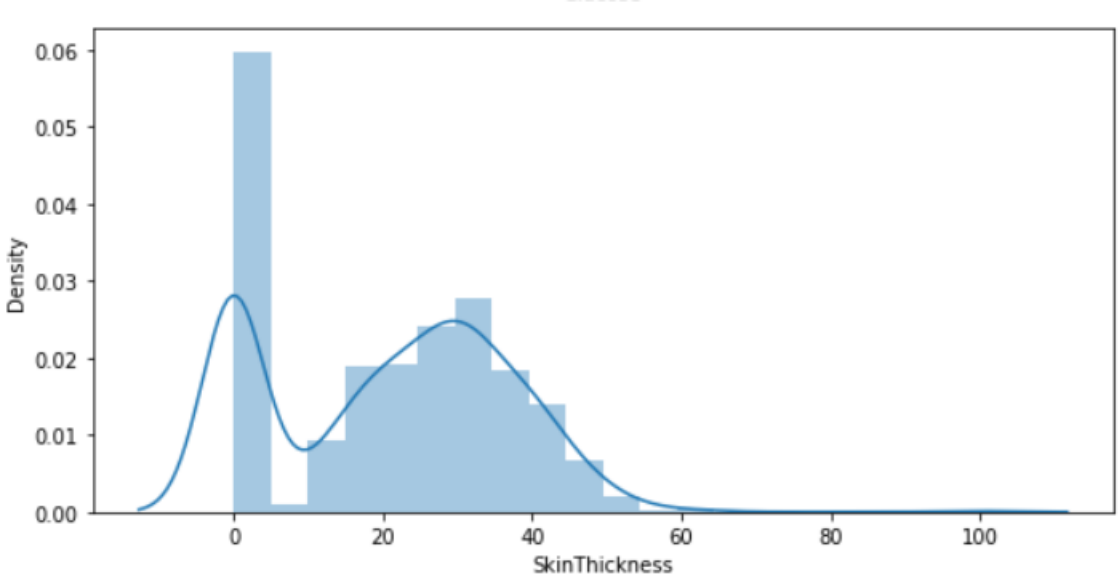
From above graph, we can conclude that, there are many females of age between 21-27, however the participants decreases as the age increase, and it can be because the data was collected in 1990 and people might be scared, as the PIMA Indian community is very backward.

2.



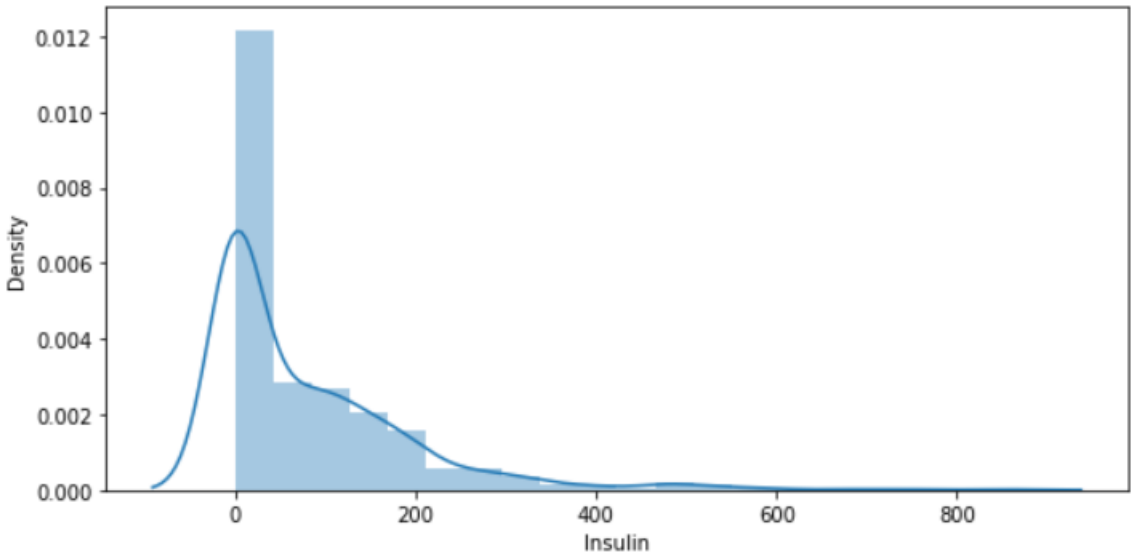
Because from graph 1, there were many young females participating, it is safe to say that they might not have been pregnant yet, which is why there are many participants at 0 pregnancies. However, we also see pregnancy value for more than 10, which ideally cannot be right, but because the community is vey backward, they might be following this tradition.

3.



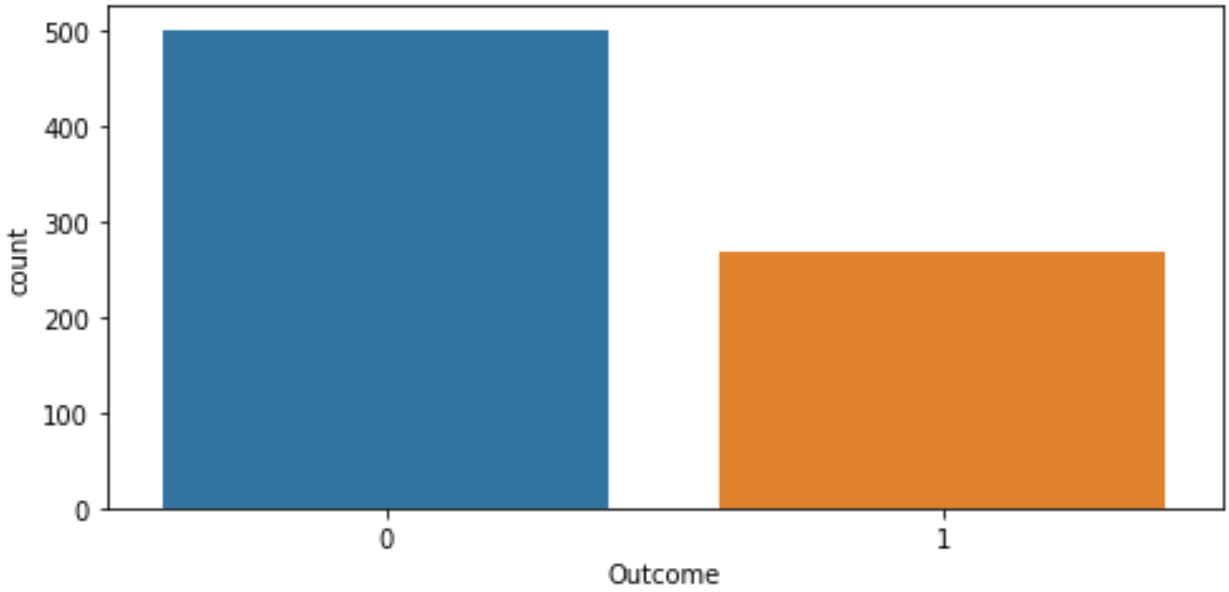
From above graph we can see that there are too many of the zero values for Skin-Thickness, which by default cannot be the case, so we will work on that.

4.



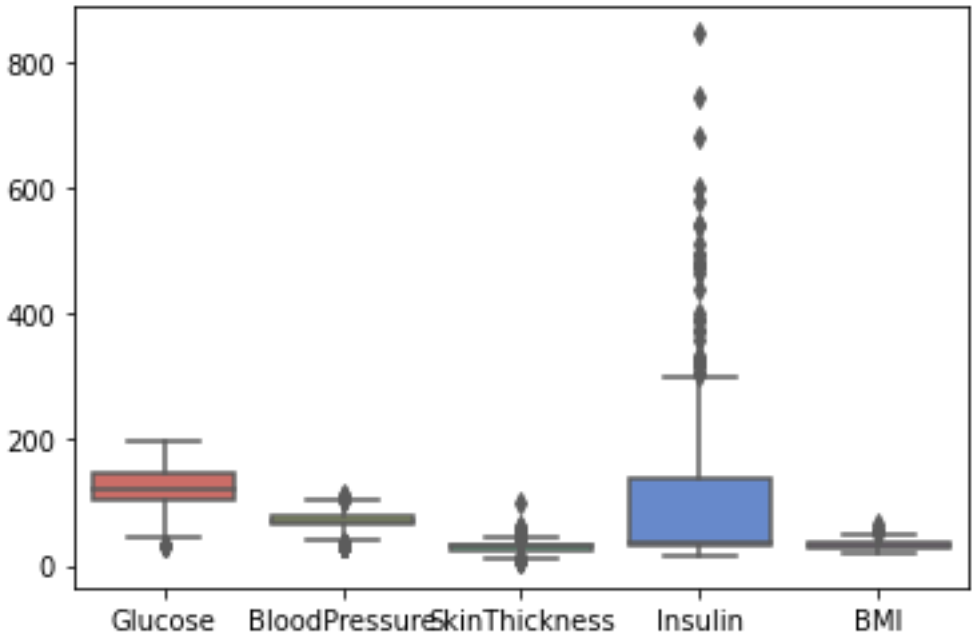
Just what we saw in the graph 3, we have the same problem for Insulin.

5.



From above graph we can see that, our data is very imbalanced as we have more training examples of people not having diabetes than people having diabetes. This will result in biased model, so we need to take care of it.

6.



The above graph shows, that the data for Insulin is very farfetched, and biologically the value of Insulin can never be more than 600. So, these are unwanted outliers we need to remove, or else they will give out very high cost and the performance of the model will decrease.

Data Cleaning:

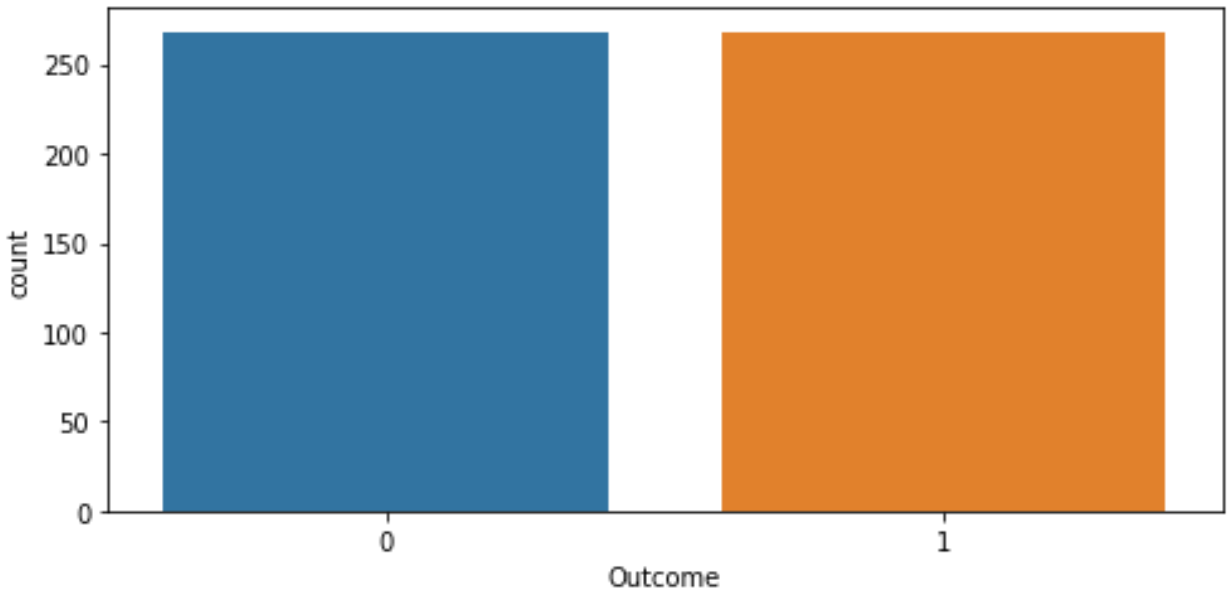
1. First of all, I got rid of the zero values by replacing them with the mean of that particular feature.
   Before:

```
zero values of Blood Pressure =  35
zero values of Glucose =  5
zero values of Skin Thickness =  227
zero values of Insulin =  374
zero values of BMI =  11
```

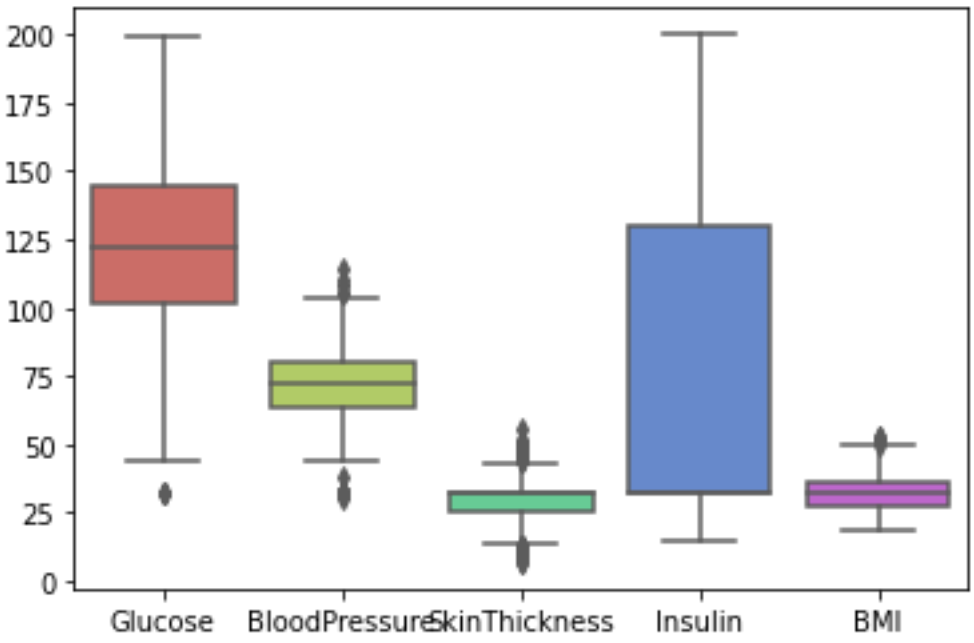
   After:

```
zero values of Blood Pressure =  0
zero values of Glucose =  0
zero values of Skin Thickness =  0
zero values of Insulin =  0
zero values of BMI =  0
```

2. Now we make our data set balance, I did this using a random sampler which randomly selected 268 Zero-values to balance them with 1-outcome.
   So, it looks like this,



3. For Insulin problem, we used the training examples that were in the quantile range of 10-90% of it's range and ignored the other examples, in order to remove the outliers.
   So, this is what our range looks like now,



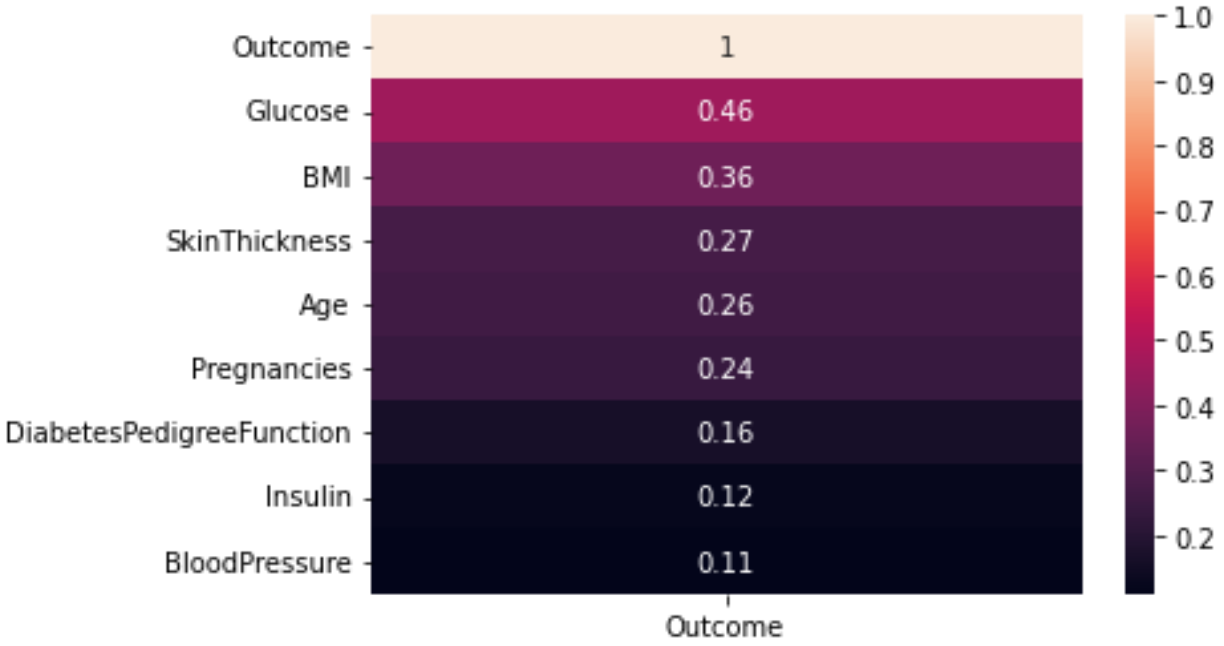
Cleaned Data:

After doing the necessary cleaning, this is what our final data looks like:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	505.000000	505.000000	505.000000	505.000000	505.000000	505.000000	505.000000	505.000000	505.000000
mean	4.087129	125.172220	71.273926	29.993792	81.979523	32.540324	0.463768	33.845545	0.489109
std	3.415118	31.494847	14.402432	8.192781	63.405133	6.616879	0.293587	11.415837	0.500377
min	0.000000	32.492724	30.000000	7.000000	15.000000	18.200000	0.078000	21.000000	0.000000
25%	1.000000	102.000000	64.000000	25.000000	32.492724	27.700000	0.246000	24.000000	0.000000
50%	3.000000	122.000000	72.000000	32.492724	32.492724	32.300000	0.370000	31.000000	0.000000
75%	6.000000	145.000000	80.000000	32.492724	130.000000	36.500000	0.652000	41.000000	1.000000
max	14.000000	199.000000	114.000000	56.000000	200.000000	53.200000	1.441000	69.000000	1.000000

There are no more zero-values in require columns and the maximum value of Insulin is now 200. And we are left with 505 training examples of completely balanced 0 and 1 outcomes.

This is what our final correlation looks like:



Important features like Skin-Thickness are slightly up, Insulin and Blood Pressure are also much better than our past correlation. Though Insulin has lost a decimal as we got rid of the outliers.

Train-Test Split and Feature Scaling:

I split my data in 80% for training and 20% for test set. I am not using validation set as I will be using cross-validation method to tune my hyperparameters in further report.

Training Shape: `(404, 8)`
Testing Shape: `(101, 8)`

Now, all of our data features are in different range, we can use the data set as it is to predict the model, however, it will take more time to predict, and it will also increase the chance of overfitting. So, I feature scaled my features, to make prediction fast. I used the standard scaling method, which taking each value subtracting the mean and dividing it by the standard deviation of the feature.
After applying this, this what our final data looks like before starting prediction,

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	-1.173748	-0.249955	0.896214	2.014498	1.808547	2.013977	0.278731	-0.220125
1	-0.879401	-1.469435	-0.734534	1.420777	-0.557792	1.664383	0.713046	-0.918991
2	-0.879401	-1.031673	-0.055056	0.114592	-0.799209	-0.326784	-0.528343	-0.918991
3	1.769730	0.094001	0.352631	-0.241640	0.594242	0.509202	-0.648036	0.478740
4	1.181034	-0.812792	0.216736	1.183289	1.808547	1.041194	0.654909	0.828173
5	0.592339	-0.062342	0.080840	0.291844	-0.799209	-0.752376	-0.347093	-0.394842
6	-0.585053	-0.812792	-0.326847	-1.191593	0.096065	0.053210	1.359389	-0.482200
7	-0.585053	0.469226	-0.870430	0.470824	0.687650	-1.086771	0.784862	-0.831632
8	1.181034	0.875719	0.488527	0.233336	-0.799209	-0.022789	-0.090608	1.002889
9	1.475382	0.938257	1.032109	0.291844	-0.799209	-1.177969	-0.819026	1.701754

We can see now each feature is now in the same range, allowing the model to predict faster and increase performance.

Fitting the Logistic Regression Model:

Just to get a glance, I tried to fit the model using random initial parameters, where parameters look like,
`(random_state=0, n_jobs=1, max_iter=100000, C = 1, penalty='l2',solver='liblinear'`
`)`

Using this model, we achieved the Accuracy score of,
`Accuracy: 73.26732673267327`

The above score is low and can be increased, to do so I tuned my hyperparameters using GridSearchCV, which takes 10 folds of validation test and try different values of hyperparameters and give out the best parameters to achieve the most accurate results.
So after checking for all these different C values,

```
[1.00000000e-05 8.48342898e-05 7.19685673e-04 6.10540230e-03
 5.17947468e-02 4.39397056e-01 3.72759372e+00 3.16227766e+01
 2.68269580e+02 2.27584593e+03 1.93069773e+04 1.63789371e+05
  1.38949549e+06 1.17876863e+07 1.00000000e+08]
```

We got the best score of `0.7447560975609757` for C = `8.48342898e-05`

Now, we fit our logistic model using the above parameters,

```
LogisticRegression(random_state=0, n_jobs=-1, max_iter=100000, C = 8.48342898e-
05, penalty='l2', solver='liblinear')
```

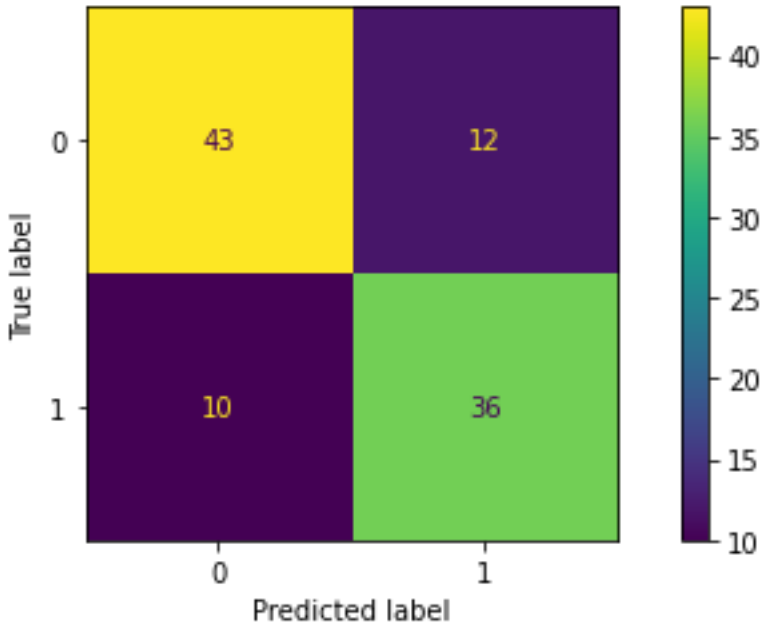
To achieve the best accuracy of, `Accuracy: 78.21782178217822`

This is what my model scores for Logistic Regression looks like,

	precision	recall	f1-score	support
0	0.81	0.78	0.80	55
1	0.75	0.78	0.77	46
accuracy			0.78	101

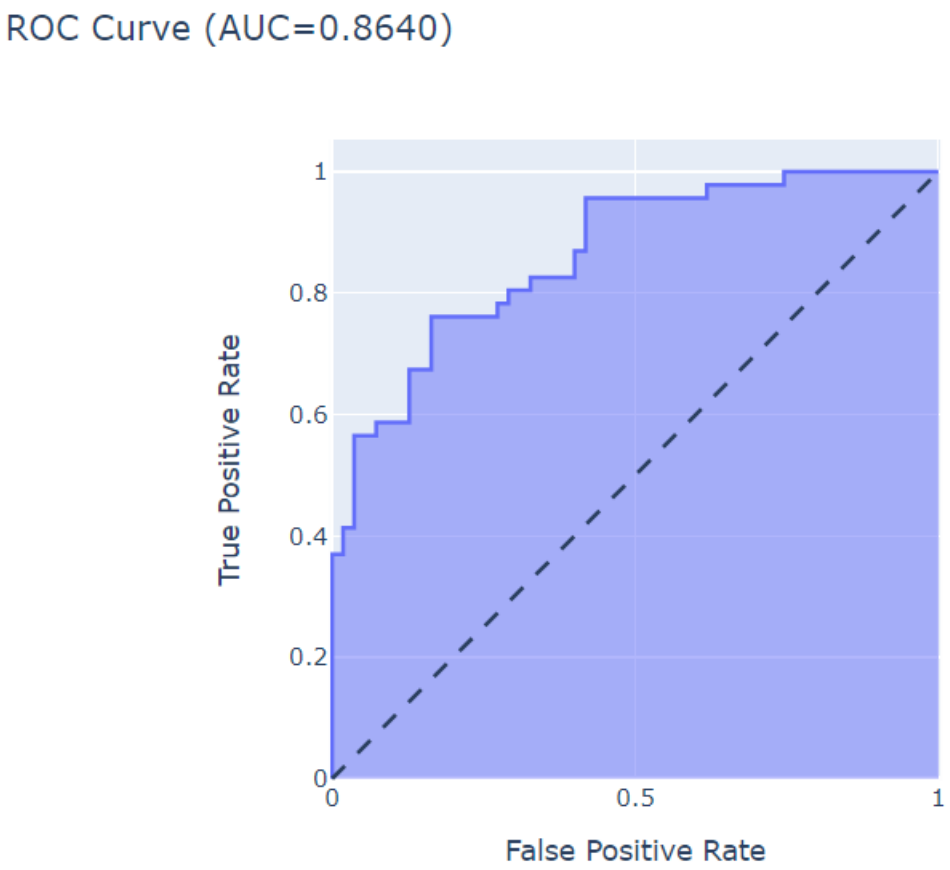
```
Accuracy_score is: 0.7821782178217822
Precision_score is: 0.75
Recall_score is: 0.782608695652174
F1_score is: 0.7659574468085107
```

Confusion Matrix:

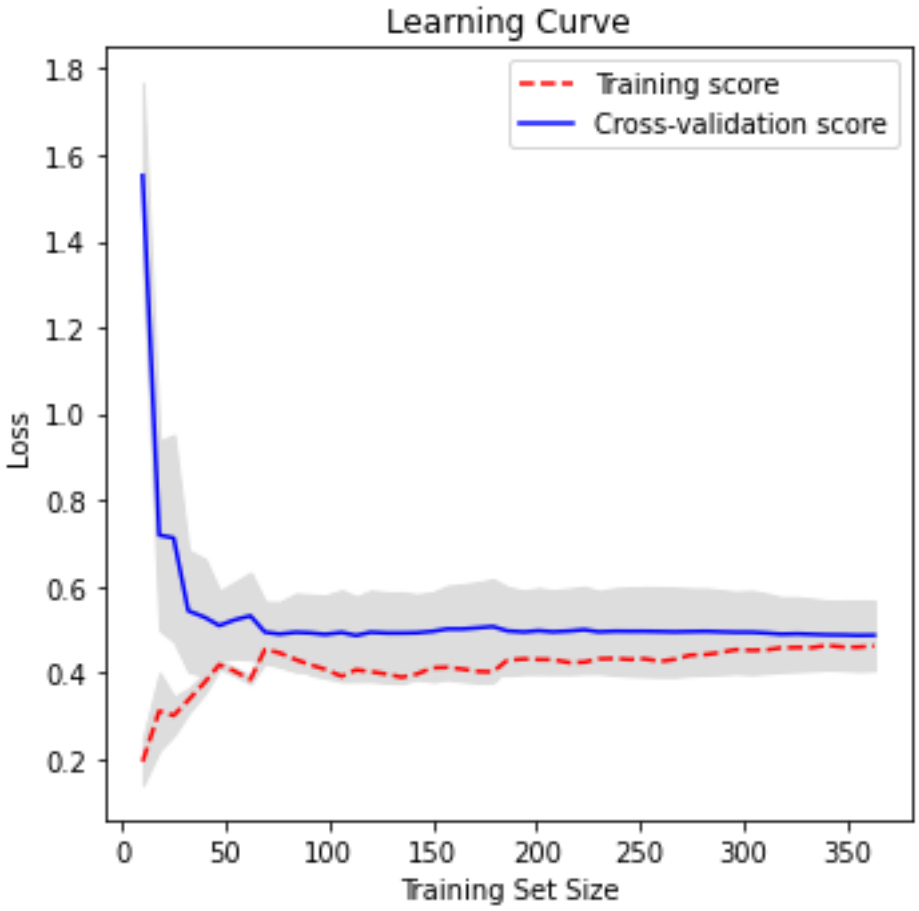


(It predicts both label very accurately, but we can do better)

ROC Curve with AUC score for Logistic Regression:

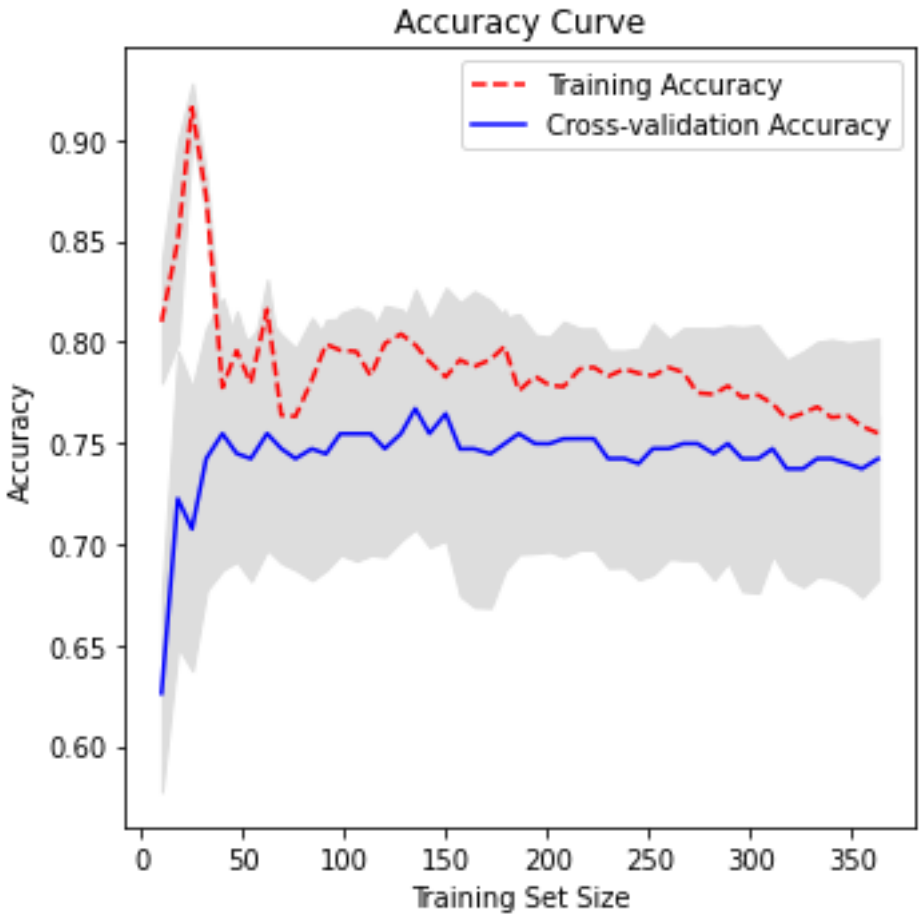


Learning Curve for Logistic Regression:



(Looks like a case of underfitting (high bias)

Accuracy Curve for Logistic Regression:



(Accuracy is being consistent as the training examples increases.)

Analysis:

From above observation we conclude that Logistic Regression can successfully predict 81% correctly for people not having diabetes while only 75% for people to have diabetes. We also achieved the AUC score of 0.864 which shows that model performs well, but it can be improved by using other models. However, coming back to learning curve, we can see that both training score and validation score, converge almost until they meet, this shows the case of underfitting and high bias. This high bias can be due to more mean values for Skin-Thickness as we replaced all zeroes by mean of respective features. To get rid of this we can predict model by using polynomial features and try to increase features. Although decreasing Regularization will not work as we hyper tuned the parameter with very small value.

Fitting the Support Vector Machine (SVM) Model:

For this model, from the start I applied GridSearchCV, in order to find the best (tuning) parameters, which came out to be,

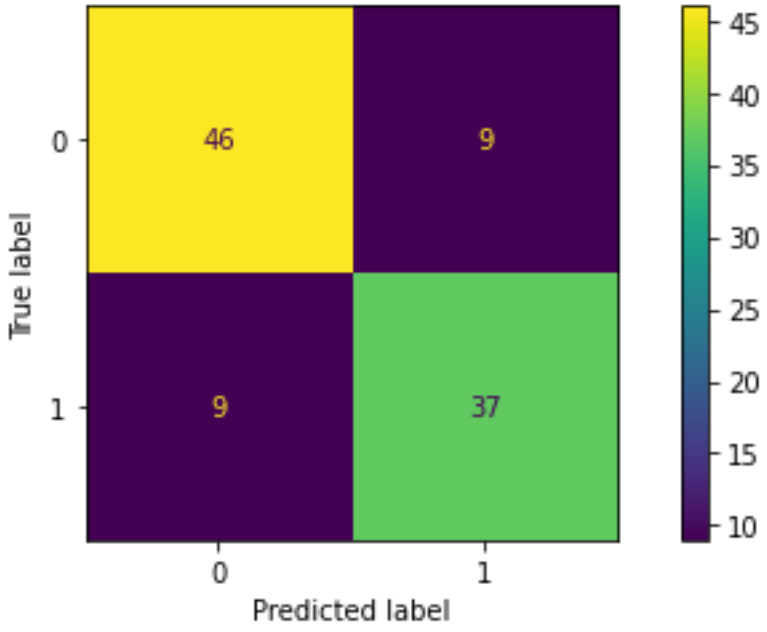
```
{'C': 10, 'gamma': 0.01}
```

After using above parameters, this is what my model scores look like,

	precision	recall	f1-score	support
0	0.84	0.84	0.84	55
1	0.80	0.80	0.80	46
accuracy			0.82	101

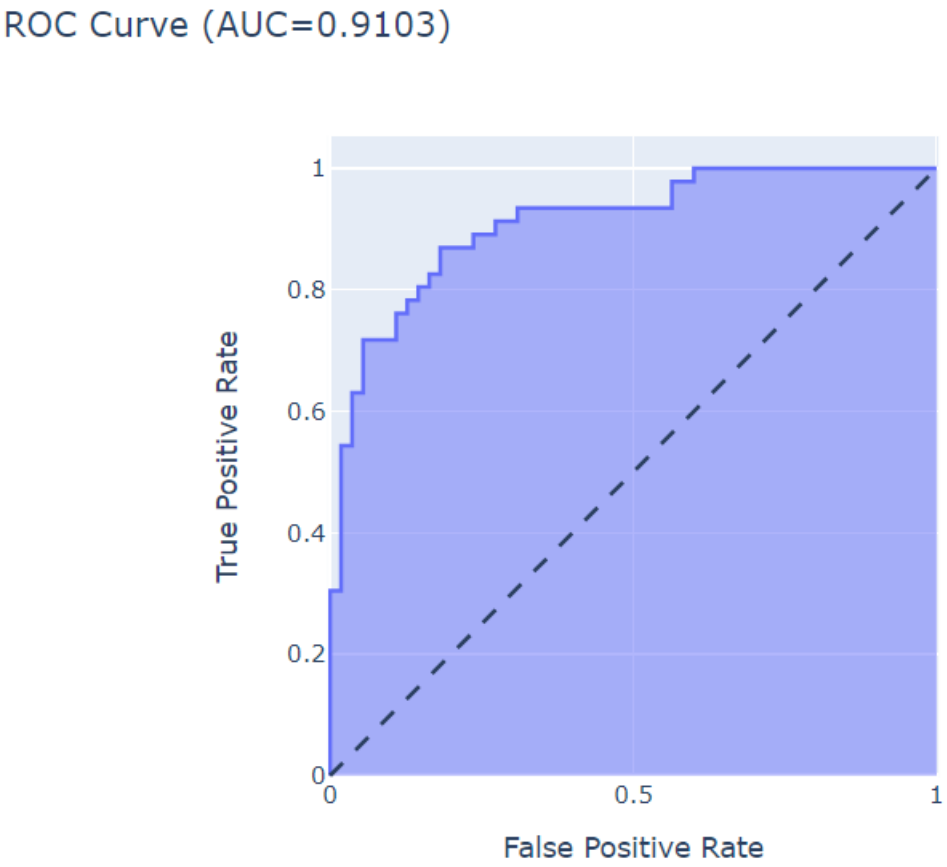
```
Accuracy_score is: 0.8217821782178217
Precision_score is: 0.8043478260869565
Recall_score is: 0.8043478260869565
F1_score is: 0.8043478260869565
```

Confusion Matrix:

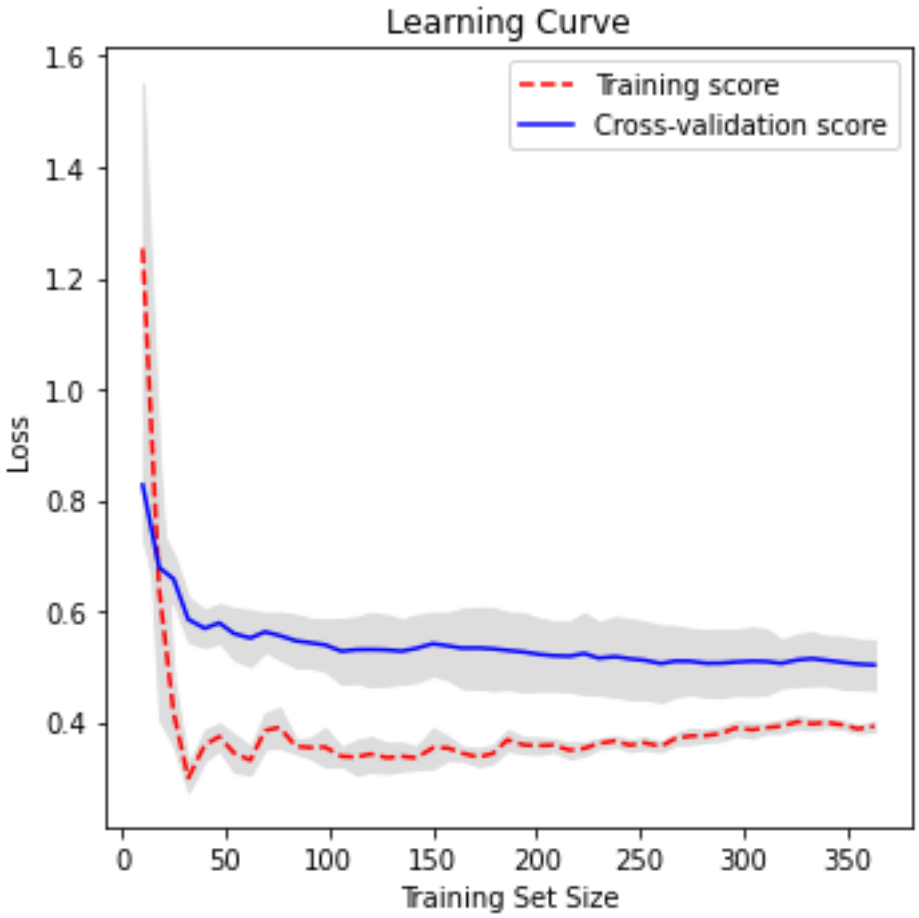


(It predicts both label more accurately than previous model, as number of 0-labels predicted right is increasing)

ROC Curve with AUC score for Logistic Regression:

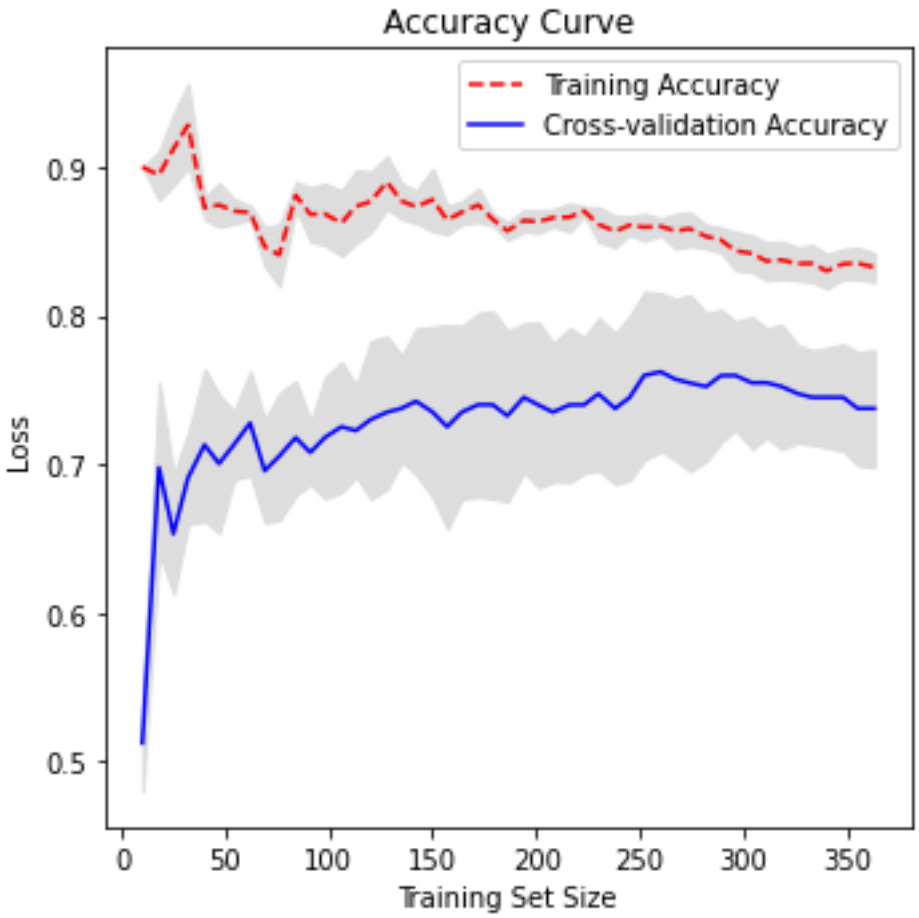


Learning Curve for Logistic Regression:



(Looks like a case of overfitting (high variance))

Accuracy Curve for Logistic Regression:



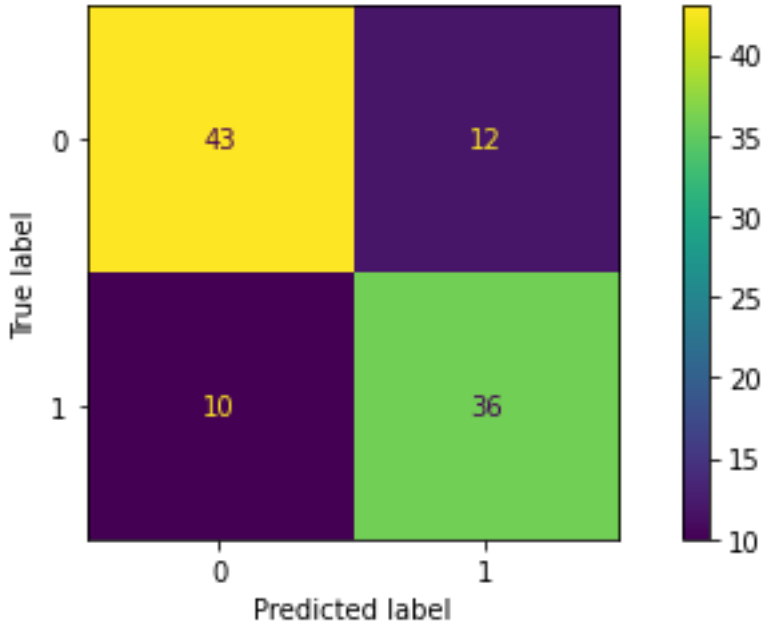
(Accuracies are getting parallel as training examples increases.)

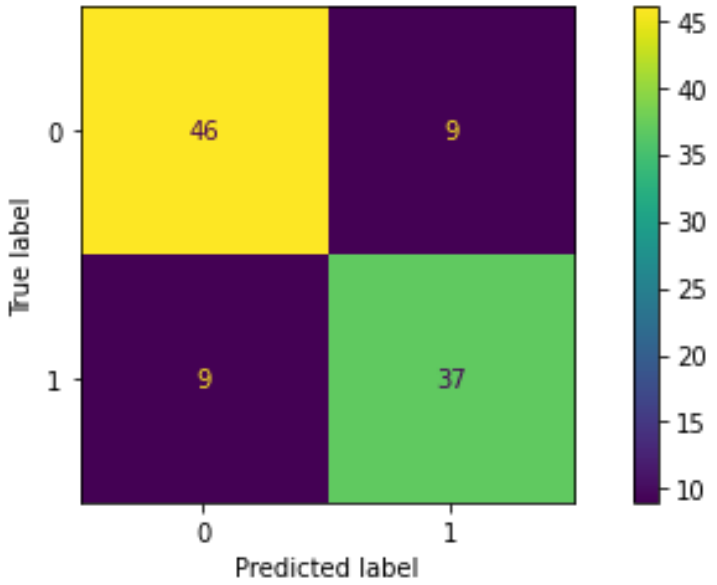
Analysis:

From above observation we conclude that SVM can successfully predict 83% of times for people not having diabetes while 80% for people to have diabetes. We also achieved the AUC score of 0.91 which shows that model performs very well on test set. However, coming back to learning curve, we can see that both training score and validation score, converge until certain point, but after that they become parallel, which shows that the model has high variance, and it is overfitting. This high variance can be because we replace the zero values with the mean of respective features, it may have triggered unwanted correlation in-between features. To get rid of this we can predict model with less features and only selecting those features that were directly related at the start without replacing 0-values.

Comparison between Logistic Regression and SVM models:

On the basis of Confusion Matrix:

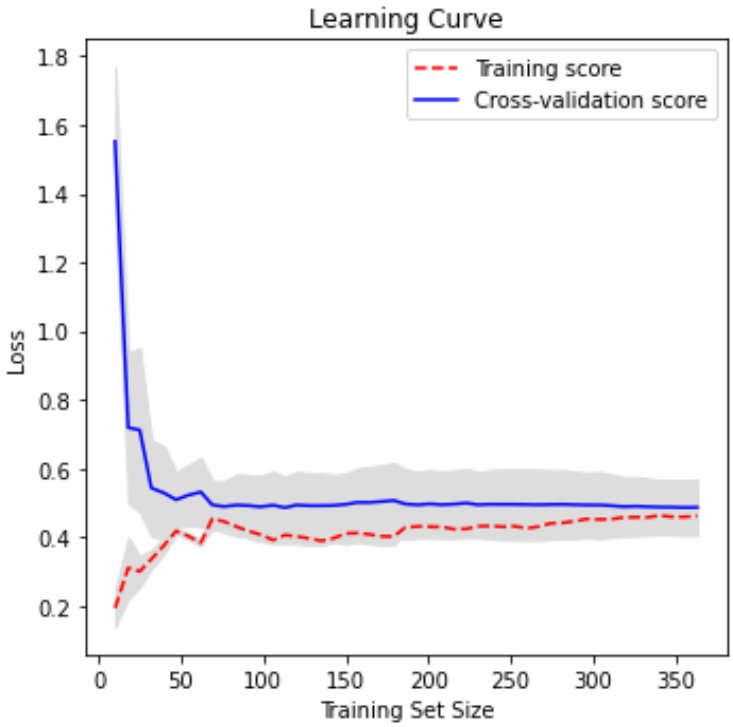


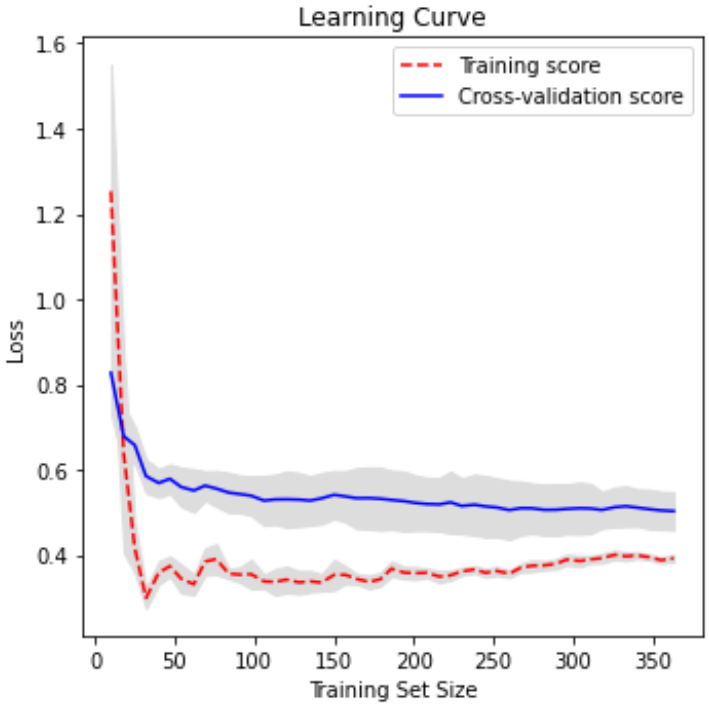


Logistic Regression

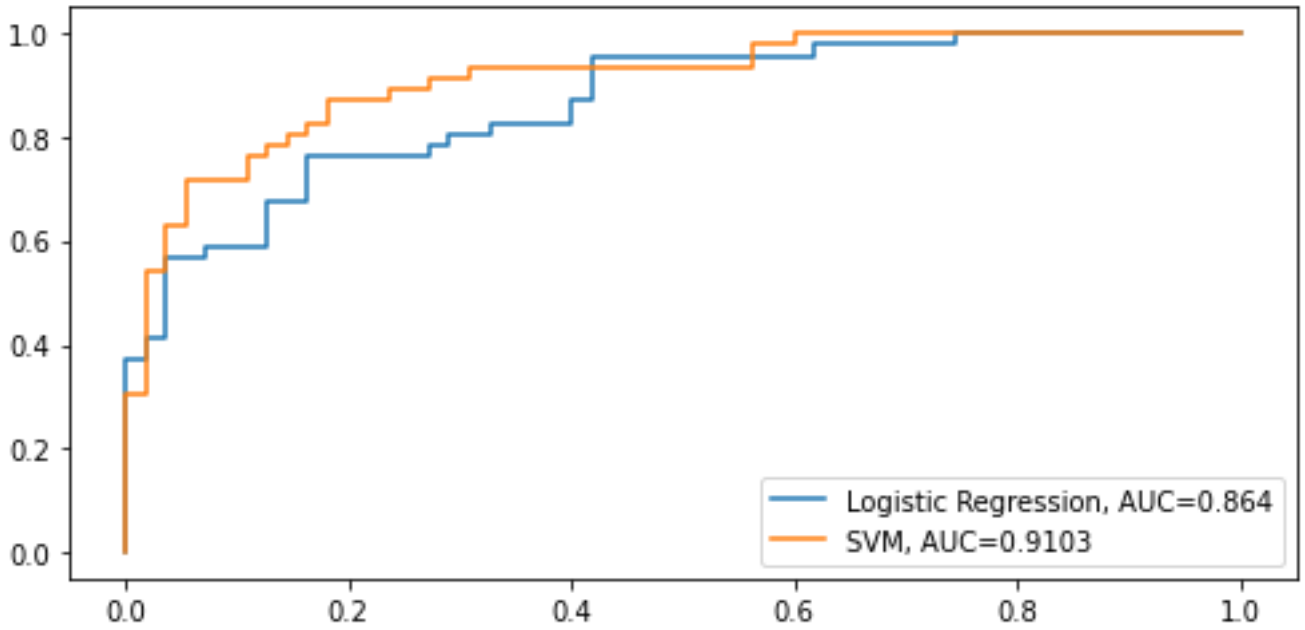
SVM Model

Bothe the models are predicting 0 and 1 label correctly, however the SVM gives a more positive response towards predicting 0-label, but both model has low accuracy when predicting 1-labesl.





The learning curve of logistic regression looks much better compared to SVM model, as it constantly converges towards same point. Also, the training score of SVM starts from top, which cannot be possible, it was starting correctly initially, but due to some changes it changed and never came back on track.



In comparison of ROC curve, SVM clearly beats Logistic Regression by almost 0.5 points more on AUC-score, which shows that SVM has higher performance compared to Logistic regression. Also, the model scores of SVM are better than Logistic regression, even though not by much, but still better. So, in my eyes SVM predicts the person having or not having diabetes more than the Logistic regression.