Diabetes Patient Risk Healthcare Dashboard

Saturday, 06/11/2022

Thesis Submitted to -

Dr. Sonali Agarwal
(Exploratory Data Analysis)

Report by

Abhishek Karmakar (ids2022007) && Sharik Gazi (ids2022009)
M.Tech - Indian Institute of Information Technology, Allahabad

Abstract

Diabetes is a group of metabolic disorders in which there are high blood sugar levels over a prolonged period. Symptoms of high blood sugar include frequent urination, increased thirst, and increased hunger. If left untreated, diabetes can cause many complications. Acute complications can include diabetic ketoacidosis, hyperosmolar hyperglycemic state, or death. Serious long-term complications include cardiovascular disease, stroke, chronic kidney disease, foot ulcers, and damage to the eyes.

Diabetes is a type of chronic disease which is more common among the people of all age groups. Predicting this disease at an early stage can help a person to take the necessary precautions and change his/her lifestyle accordingly to either prevent the occurrence of this disease or control the disease.

If we can have a Dashboard for all Visualization purposes then we can form the statistical point of view into our mind about how the health related issues are going on now-a-days. We have used Tableau for this purpose as Tableau is one of the best Toolbox for Visualization purposes.

In addition to Visualization, the Model training proves to be a boon for model comparison. And then we have attached the images for model performance into our Dashboard. For Naive Bayes Model, the accuracy is 79.22%, Decision Tree is 72.07%, Random Forest is 82.46%, XgBoost is 75.97%, SVM is 75.97%, Logistic Regression is 82.46%, KNN is 81.16% and Neural Network is 75.32%.

Table of Contents

1. Introduction

Diabetes mellitus is a group of metabolic disorders due to less insulin in the blood and it can also create several kinds of different diseases as well. Warning signs of this diabetes result in more hunger, feeling thirsty , weight loss and if not medicated it will lead to death sometimes . There are three types of diabetes Type1, Type2 and Type3. Type1 is due to the failure of pancreas that produces little or no insulin, Type2 when the whole body either doesn't produce enough insulin or it resists insulin and Type3 occurs when neurons in the brain are unable to respond to insulin. In these type2 is the most common type , nearly 90% cases of total diabetes patients.

Diabetes mellitus is a group of metabolic disorders due to less insulin in the blood. Diabetes can create several kinds of different diseases as well. Diabetes can cause a worldwide health care crisis because according to some research around 600 million people will be affected by this disease by the end of 2035. It can lead to several disorders for example urinary organ diseases ,blindness etc. So patients need to go to the hospital to get reports after consulting with the doctor and sometimes it will be time consuming so by using machine learning and deep learning methods we can have an answer for this issue. The aim of this project is to predict type2 diabetes at an early stage so that it can lead to improved treatment for a patient.

We have created a Tableau Dashboard for the purpose of Statistical Visualization as well to understand how things vary in real life applications. We can have a Dashboard for all Visualization purposes then we can form the statistical point of view into our mind about how the health related issues are going on now-a-days. In addition to Visualization, the Model training proves to be a boon for model comparison.

Also in this study we have trained models - Naive Bayes, Decision Tree, Random Forest, XgBoost, Support Vector Machine (SVM), Logistic regression , K-Nearest Neighbors (KNN) and Neural Network to predict diabetes. For Naive Bayes Model, the accuracy is 79.22%, Decision Tree is 72.07%, Random Forest is 82.46%, XgBoost is 75.97%, SVM is 75.97%, Logistic Regression is 82.46%, KNN is 81.16% and Neural Network is 75.32%.

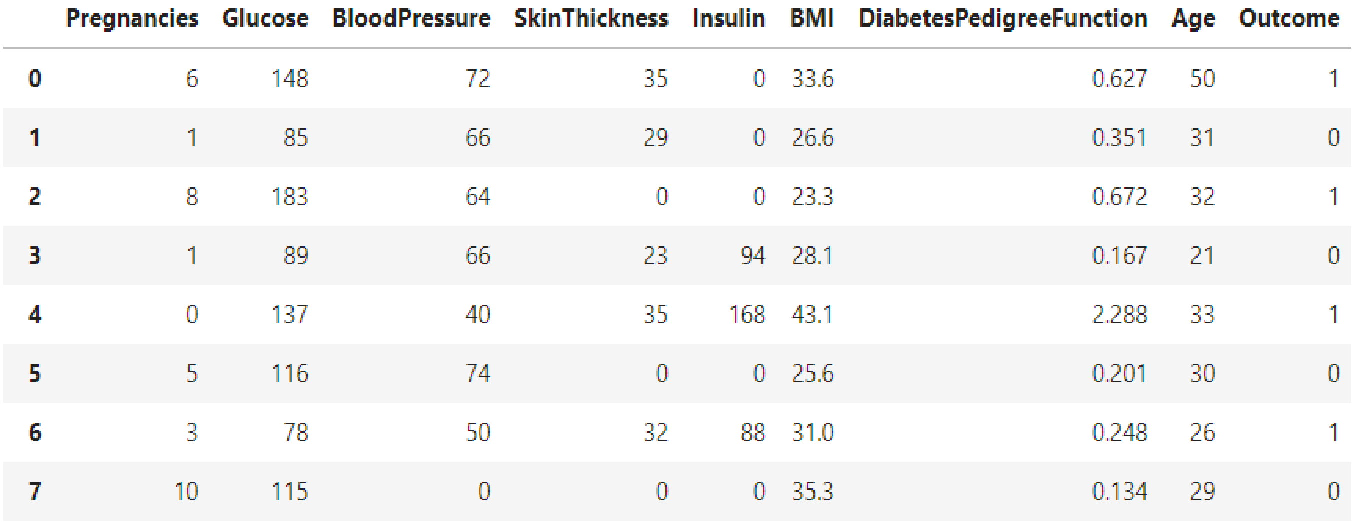
2. Dataset Description

The datasets consist of several medical predictor variables and one target variable, Outcome. Predictor variables include the number of pregnancies the patient has had, their BMI, insulin level, age, and so on.

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- BloodPressure: Diastolic 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)
- DiabetesPedigreeFunction: Diabetes pedigree function
- Age: Age (years)
- Outcome: Class variable (0 or 1)

Number of Observation Units: 768

Variable Number: 9



	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
6	3	78	50	32	88	31.0	0.248	26	1
7	10	115	0	0	0	35.3	0.134	29	0

3. Exploratory Data Analysis - Data Preprocessing

Preprocessing technique is used so that the raw data can be converted into an understandable format which is used for training and testing. We will use various efficient preprocessing techniques like data imputation, handling null values etc.

We have used the Pre-processed data for feeding purposes into Tableau. Exploratory Data Analysis (EDA) is an approach to analyze the data using visual techniques. It is used to discover trends, patterns, or to check assumptions with the help of statistical summary and graphical representations. With Exploratory Data Analysis; The data set's structural data were checked. The types of variables in the dataset were examined. Size information of the dataset was accessed. The 0 values in the data set are missing values. Primarily these 0 values were replaced with NaN values. The reason behind this was because there were attributes like insulin, glucose, skin thickness, blood pressure, BMI cannot be zero. There has to be some value present; for a human to remain alive. Descriptive statistics of the data set were examined. An additional field has also been added which is age wise distribution of the data - '21 to 31 years', '32 to 41 years', '42 to 51 years', '52 to 61 years', '62 to 71 years' , '72 to 81 years'.

Additionally we replace the NaN values to the mean and median values accordingly. Data Preprocessing section, the NaN values missing observations were filled with the median values of whether each variable was sick or not. The X variables were standardized with the Scalarization method for the model training part displayed later in Tableau Story.

4. Tableau

Its extract based data abstraction in Tableau for reading the diabetes dataset. The Age_Label, Chances_of_Diabetes, Age_String, Outcome_String is a calculation and field in Tableau whereas the Parameters are 'Top N Parameter', 'Frome Age', 'To Age'.

Chances_of_Diabetes is a calculated field which stores 3 types of categories:-

1. Diabetic - Person who is having Diabetes. Mainly old age people of people having bad health.
2. Maybe Diabetic - Person who has a chance of getting affected by Diabetes.
3. Non - Diabetic - Mainly the young people.

In total there are 35 sheets and 9 Dashboards and 1 story. The sheets are merged to Dashboard and all the Dashboards are merged to Story.

In many cases the Aggregations are not used to perform visualization in this Tableau report. Mainly in the story there is a GUI resembling buttons. Clicking each button would demonstrate each single Dashboard.

4.1. Age Group Variation - Part 1

On clicking the first button, we shall get the first Dashboard which is the Age Variation with respect to different parameters. The Calculated Field - Age label has been used as a filter here. All the multiple select age bands - '21 to 31 years', '32 to 41 years', '42 to 51 years', '52 to 61 years', '62 to 71 years' , '72 to 81 years' are used as a filter. Also, selecting as many bands would reflect on all other Sheets related to the same datasource.

The attributes taken into consideration are Age, skin thickness, glucose, insulin and pregnancies. The plotting has been to do a comparison between age with respect to skin thickness, glucose, insulin and pregnancies.

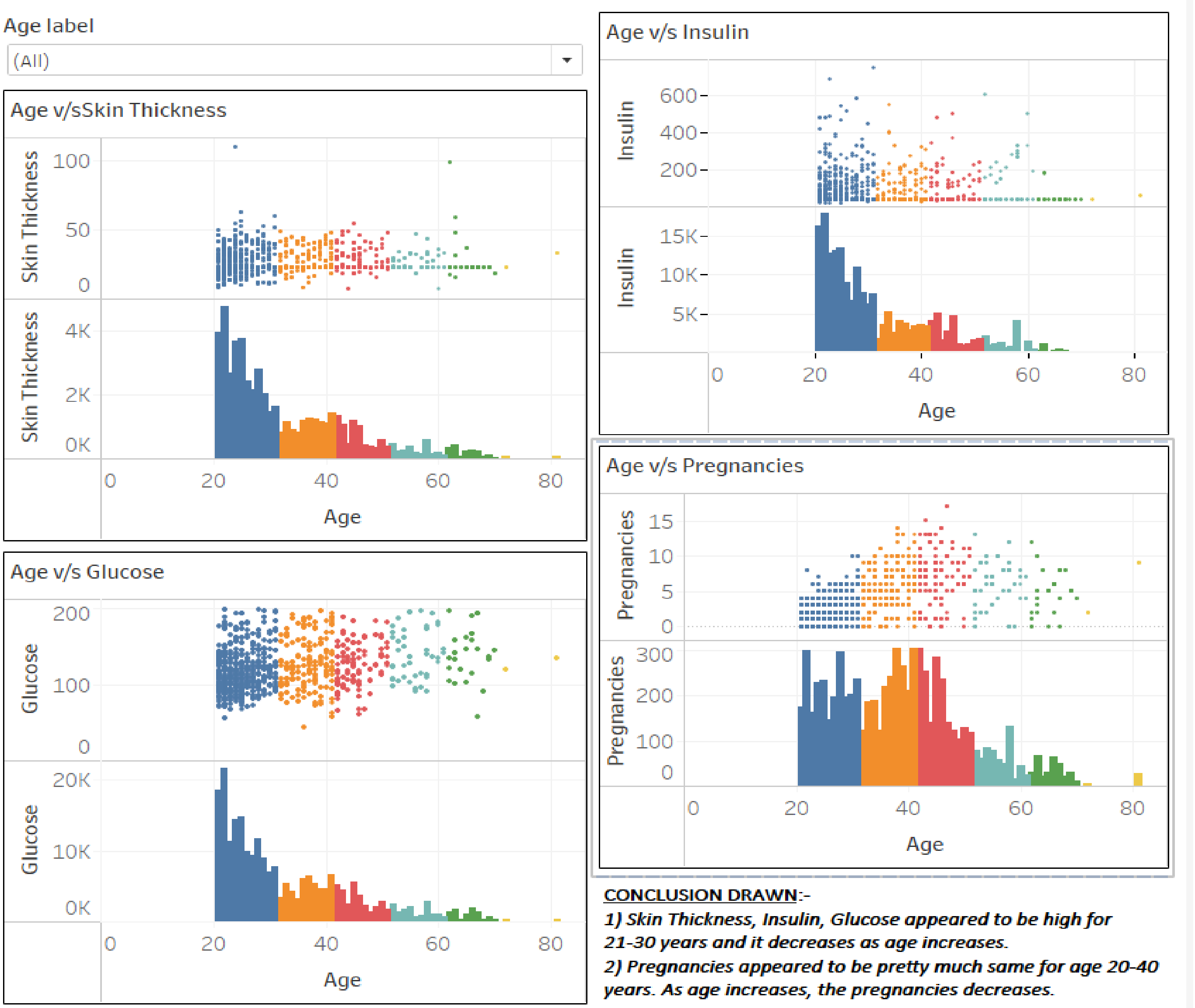
The data visualized is a right skewed data mostly, highly influenced by 21-31 years of age group of people for the Skin Thickness, Glucose and Insulin. But for pregnancies, the frequency of distribution is similar.

To the histogram plotted we can see:-

1) Skin Thickness, Insulin, Glucose appeared to be high for 21-30 years and it decreases as age increases.

2) Pregnancies appeared to be pretty much the same for ages 20-40 years. As age increases, the pregnancy decreases.

This clearly shows that this dataset is a biased dataset in terms of frequency distribution of age bands. Clearly it can be seen that the age band 60-80 years, the data frequency decreases. Hence less influence on the human population for drawing conclusions or for model training purposes. If we have a look at the frequency distribution, we can find that the frequency decreases as the age increases.



CONCLUSION DRAWN:-

1) Skin Thickness, Insulin, Glucose appeared to be high for 21-30 years and it decreases as age increases.

2) Pregnancies appeared to be pretty much same for age 20-40 years. As age increases, the pregnancies decreases.

4.2. Age Group Variation - Part 2

The worksheet is the continuation of the previous worksheet. The attributes taken into consideration are Age, skin thickness, glucose, insulin and pregnancies. The plotting has been done to do a comparison between age with respect to BMI, Diabetes Pedigree Function and Blood Pressure.

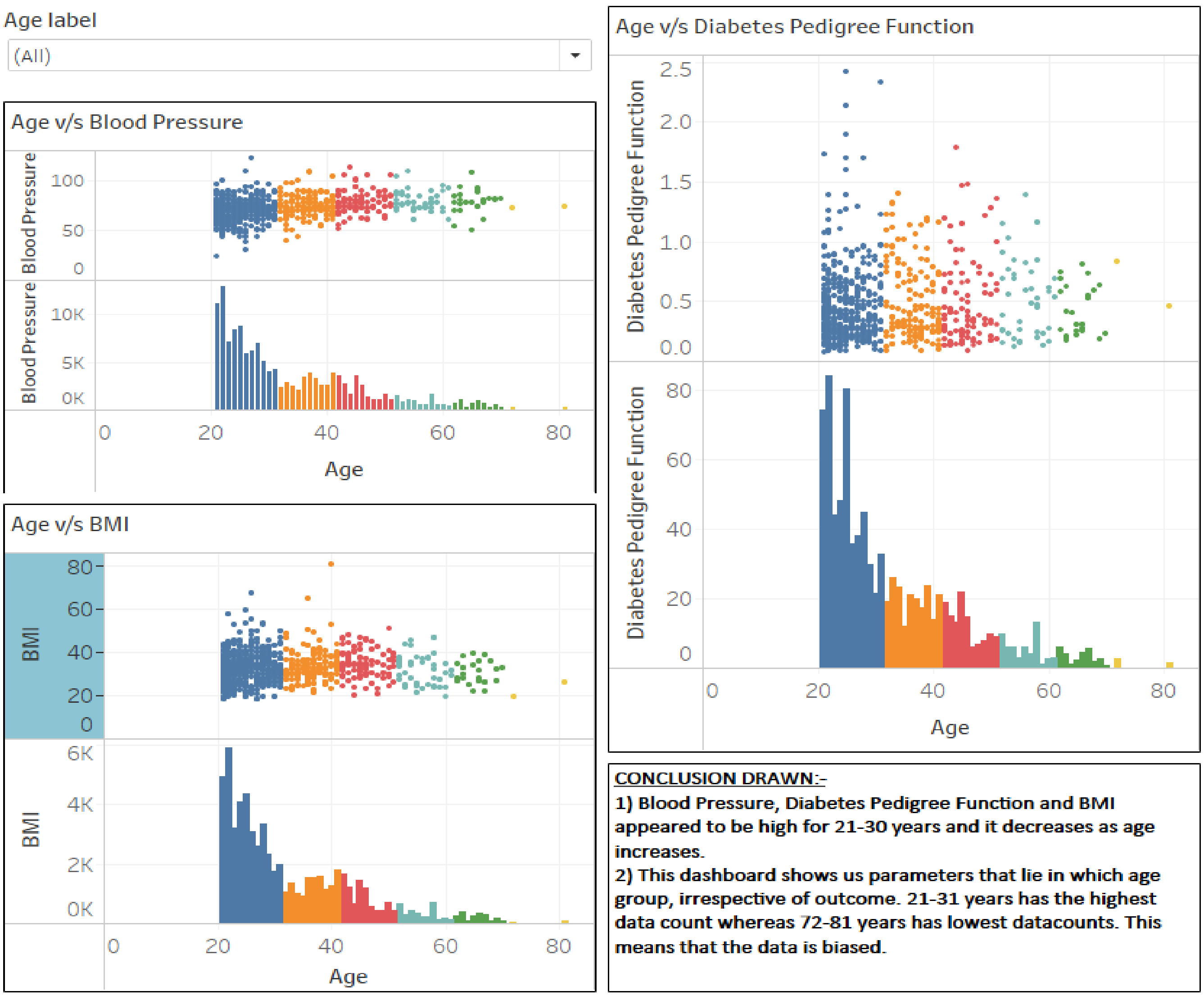
The Calculated Field - Age label has been used as a filter here. All the multiple select age bands - '21 to 31 years', '32 to 41 years', '42 to 51 years', '52 to 61 years', '62 to 71 years' , '72 to 81 years' are used as a filter. Also, selecting as many bands would reflect on all other Sheets related to the same datasource.

The data visualized is a right skewed data mostly, highly influenced by 21-31 years of age group of people for the attributes BMI, Blood Pressure and Diabetes Pedigree Function.

The following conclusions could be drawn from this dashboard :-

1) Blood Pressure, Diabetes Pedigree Function and BMI appeared to be high for 21-30 years and it decreases as age increases.

2) This dashboard shows us parameters that lie in which age group, irrespective of outcome. 21-31 years has the highest data count whereas 72-81 years has lowest datacounts. This means that the data is biased.



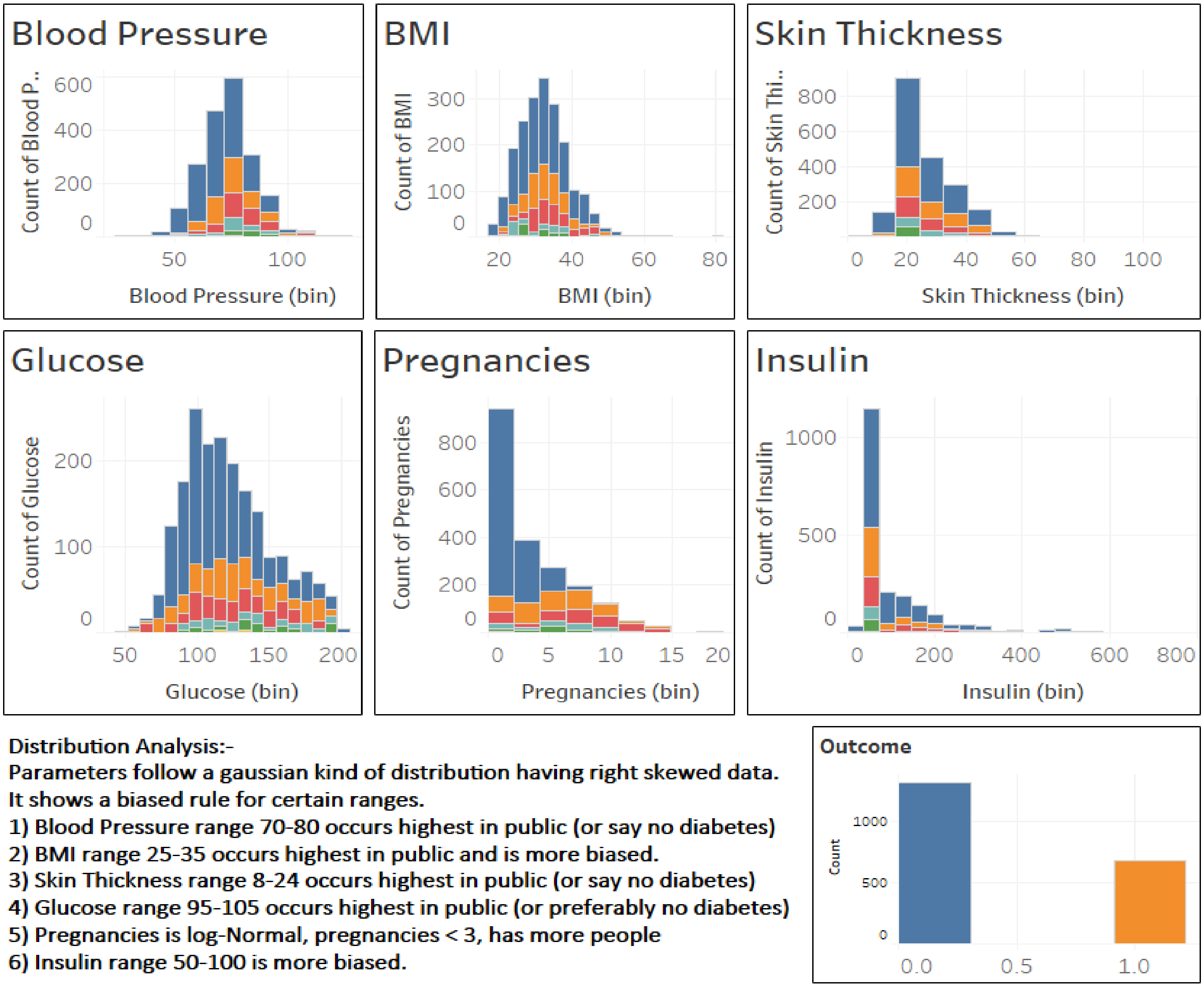
CONCLUSION DRAWN:-

1) Blood Pressure, Diabetes Pedigree Function and BMI appeared to be high for 21-30 years and it decreases as age increases.

2) This dashboard shows us parameters that lie in which age group, irrespective of outcome. 21-31 years has the highest data count whereas 72-81 years has lowest datacounts. This means that the data is biased.

4.3. Distributions of all Parameters

We have shown here the stacked bar plot influenced by all age bands. In the lower right we have the outcome count values. The filter has been added for the Age Label as well as for outcome count values as well. On selecting either of the stacked values in the bar plot, it reflects on all other sheets. The same goes for the outcome chat.



Distribution Analysis:-
Parameters follow a gaussian kind of distribution having right skewed data.
It shows a biased rule for certain ranges.
1) Blood Pressure range 70-80 occurs highest in public (or say no diabetes)
2) BMI range 25-35 occurs highest in public and is more biased.
3) Skin Thickness range 8-24 occurs highest in public (or say no diabetes)
4) Glucose range 95-105 occurs highest in public (or preferably no diabetes)
5) Pregnancies is log-Normal, pregnancies < 3, has more people
6) Insulin range 50-100 is more biased.

<u>Distribution Analysis from the Graphs Shown</u>:-

Parameters follow a gaussian kind of distribution having right skewed data. It shows a biased rule for certain ranges.

1) Blood Pressure range 70-80 occurs highest in public (or say no diabetes). This deduction has been examined by selecting the mode values and verifying it from the scatter analysis.

2) BMI range 25-35 occurs highest in public and is more biased.

3) Skin Thickness range 8-24 occurs highest in public (or say no diabetes)

4) Glucose range 95-105 occurs highest in public (or preferably no diabetes)

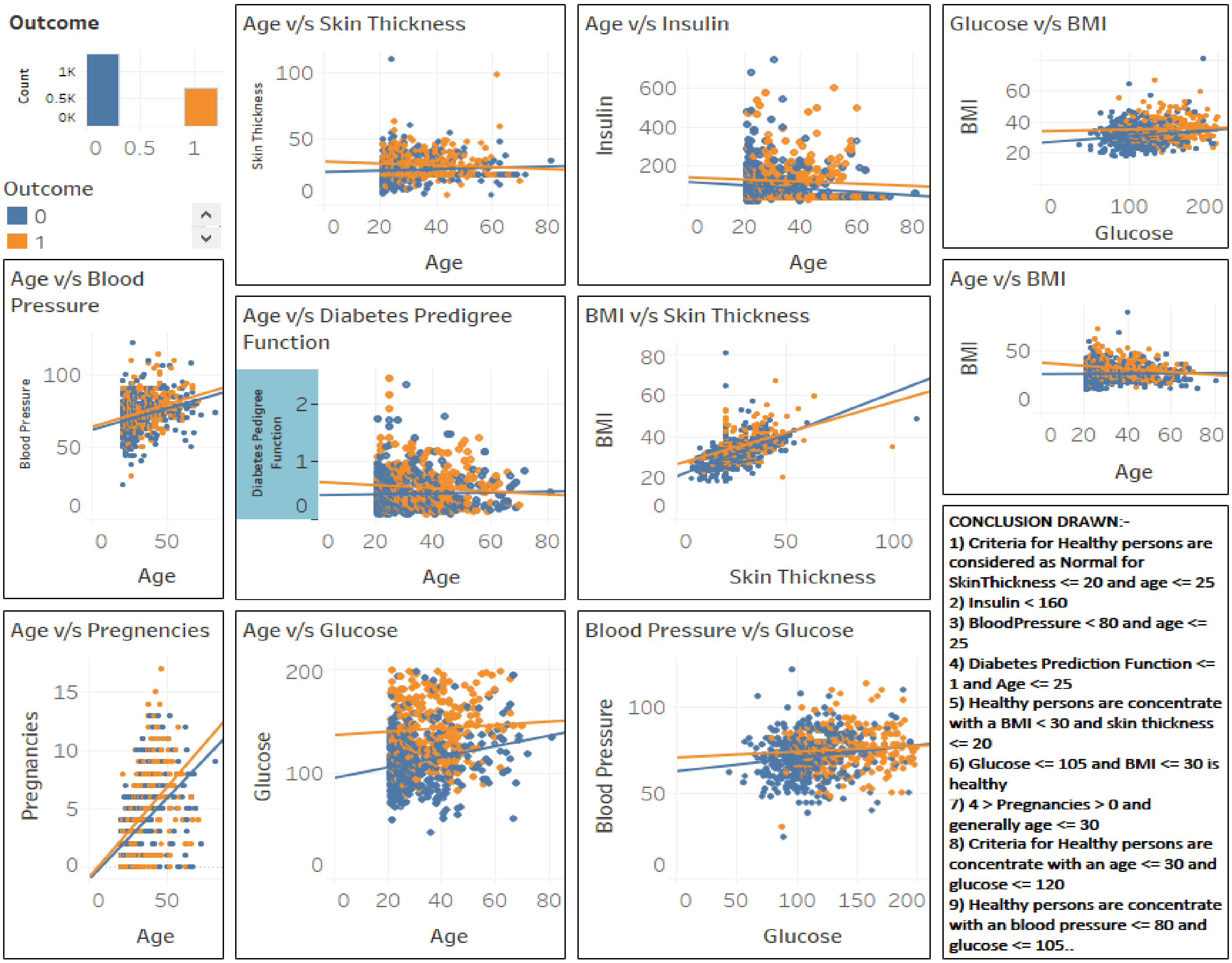
5) Pregnancies is log-Normal, pregnancies < 3, has more people

6) Insulin range 50-100 is more biased.

4.4. Scatter Analysis to draw Conclusions

We here did the scatter analysis of different variables. Also we did the Linear Regression or trend line analysis with the following so that we can get an influence in what range the value line would be if the age is so and so.

1. Age v/s Skin Thickness : Here we can find a somewhat straight line trend.
2. Age v/s Insulin : Here we can see a decreasing trend. As the age increases the insulin decreases. Although people have a decreasing trend, some have diabetes and some don't.
3. Age v/s Blood Pressure : Here people show an increasing trend. As the age increases, people are influenced by high blood pressure.
4. Age v/s Diabetes Pedigree Function: Here people having diabetes are showing a decreasing trend and non-diabetic people are showing an increasing trend. The correlation value is approx plus minus 0.5 so the slope is not so steep.
5. BMI v/s Skin Thickness : Here both the categories of people have an increasing trend.
6. Glucose v/s BMI : Here both the categories of people have an increasing trend.
7. Age v/s BMI : Here both the categories of people have a decreasing trend.
8. Age v/s Pregnancies : Here both the categories of people have an increasing trend.
9. Age v/s Glucose : Here both the categories of people have an increasing trend.
10. Blood Pressure v/s Glucose : Here both the categories of people have an increasing trend.



CONCLUSION DRAWN:-

1) Criteria for Healthy persons are considered as Normal for SkinThickness <= 20 and age <= 25
2) Insulin < 160
3) BloodPressure < 80 and age <= 25
4) Diabetes Prediction Function <= 1 and Age <= 25
5) Healthy persons are concentrate with a BMI < 30 and skin thickness <= 20
6) Glucose <= 105 and BMI <= 30 is healthy
7) 4 > Pregnancies > 0 and generally age <= 30
8) Criteria for Healthy persons are concentrate with an age <= 30 and glucose <= 120
9) Healthy persons are concentrate with an blood pressure <= 80 and glucose <= 105..

Conclusion drawn from Regression Analysis:-

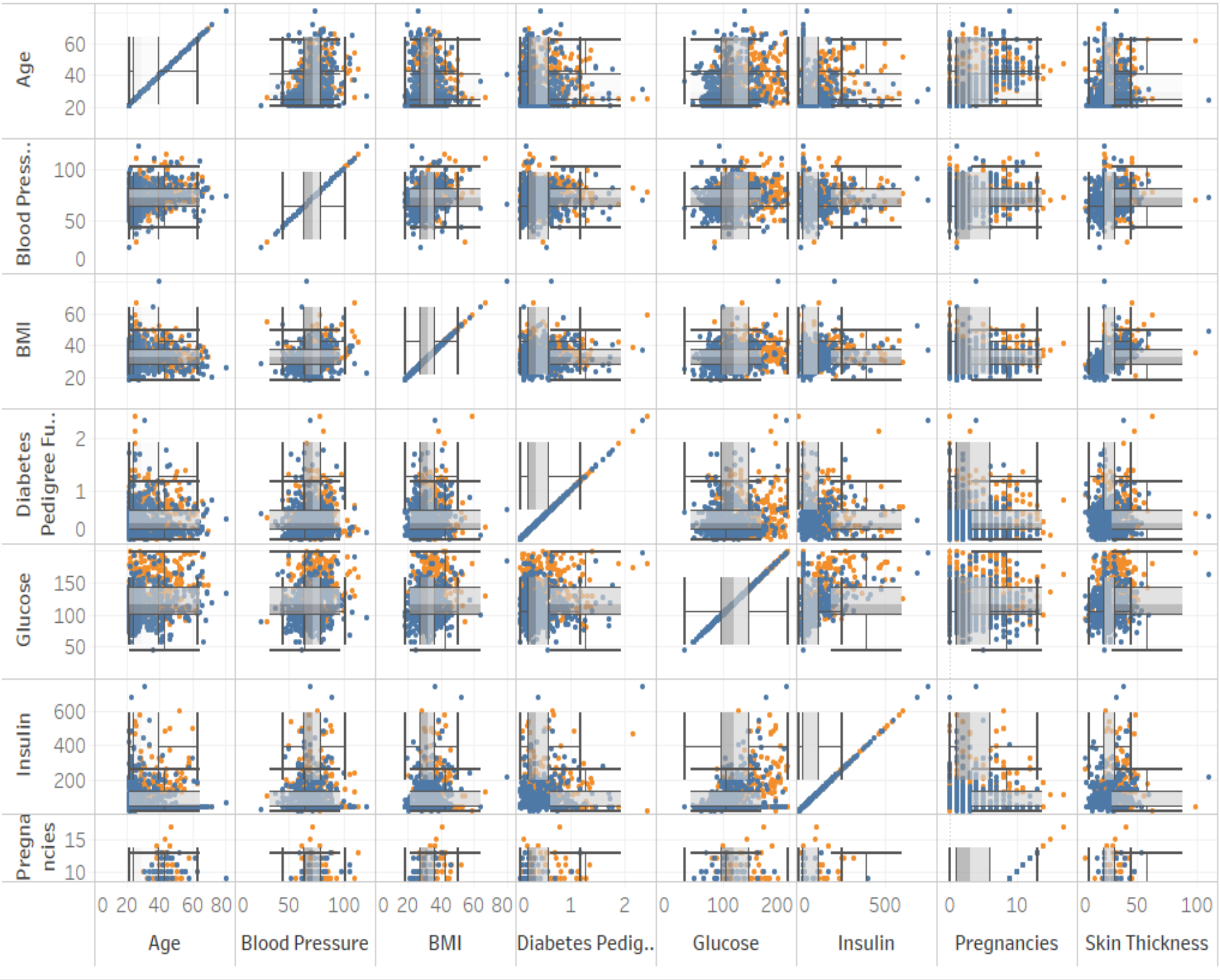
1. Criteria for Healthy persons are considered as Normal for SkinThickness <= 20 and age <= 25
2. Insulin < 160
3. BloodPressure < 80 and age <= 25
4. Diabetes Prediction Function <= 1 and Age <= 25
5. Healthy persons are concentrate with a BMI < 30 and skin thickness <= 20
6. Glucose <= 105 and BMI <= 30 is healthy
7. 4 > Pregnancies > 0 and generally age <= 30

8. Criteria for Healthy persons are concentrate with an age <= 30 and glucose <= 120
9. Healthy persons are concentrate with an blood pressure <= 80 and glucose <= 105
10. BMI <= 30 and age <= 30 is considered to be healthy

4.5. Outlier Visualization

We have used a box plot to see the outliers. The data which lie outside the InterQuartile Range are called outliers.

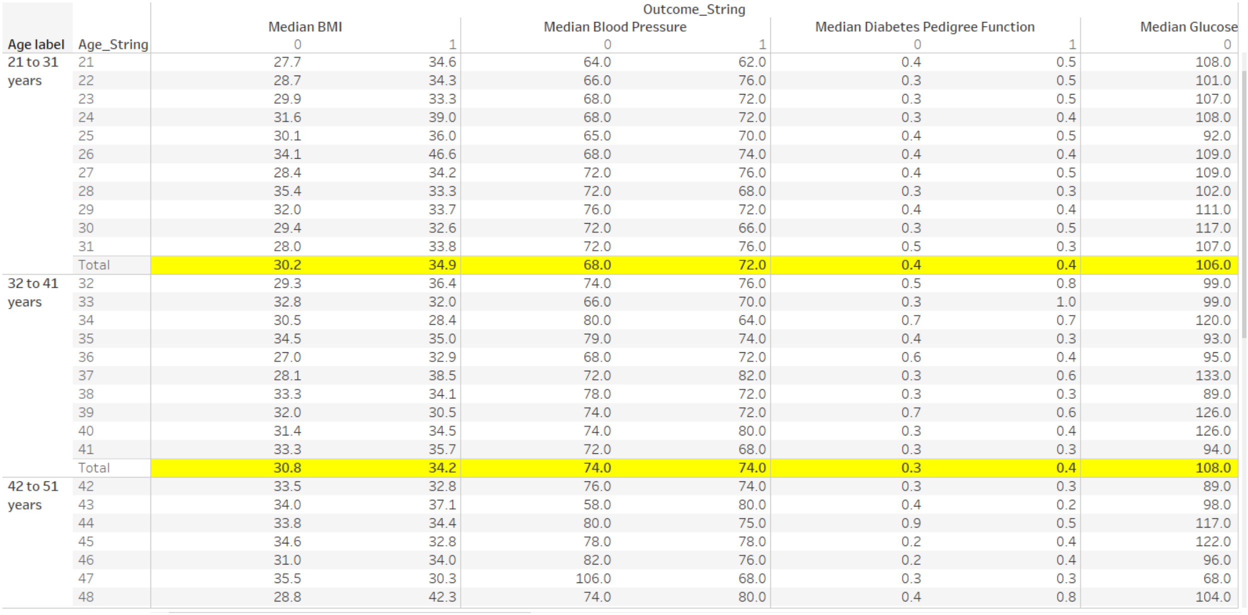
We have done a pair plot in tableau for different variables and then added a box plot to it. Box plot is useful for outliers finding. Each box has two box plots, one horizontally for denoting the outliers in row attributes and vertically for denoting the outlier in the column attributes.



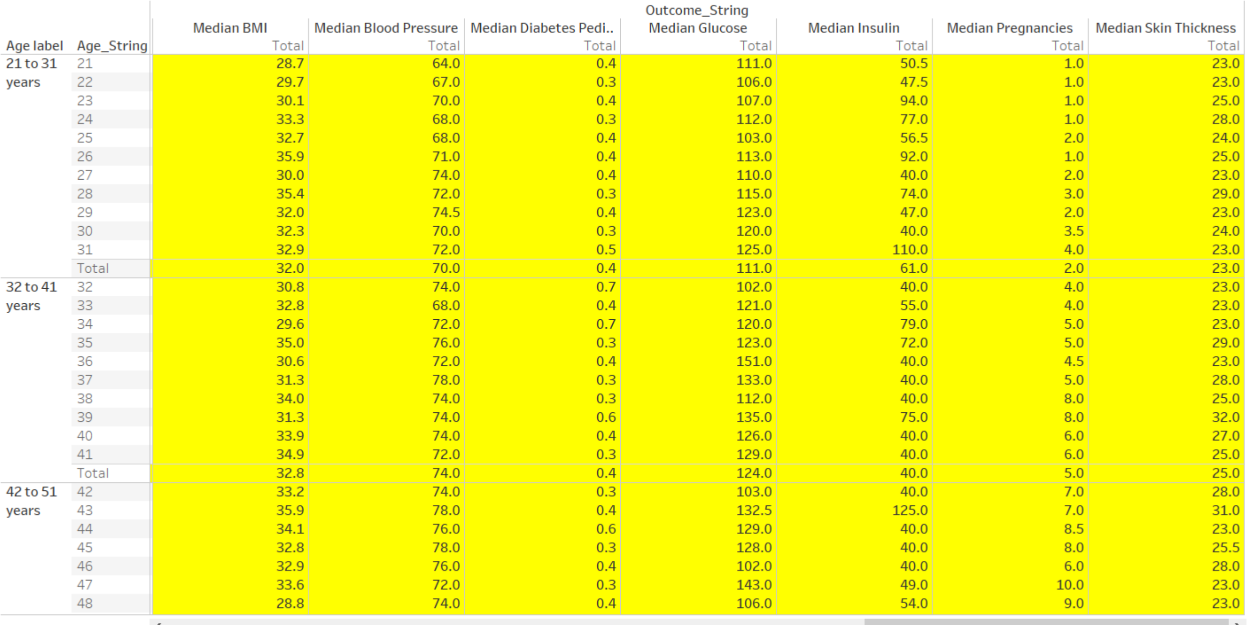
4.6. Table Calculations for Median

This is a calculated table. Each attribute has been on the basis of outcome - 0 or 1. The age label bands have used the Age String and outcome String variable. For each label, the median has been calculated marked in yellow. As we scroll right side we can find that for each age wise, the median calculation has been marked with yellow. At the lower bottom we can see the median values for that column.

The whole table is the median based calculation to find out the median values for that particular band or for that particular age.



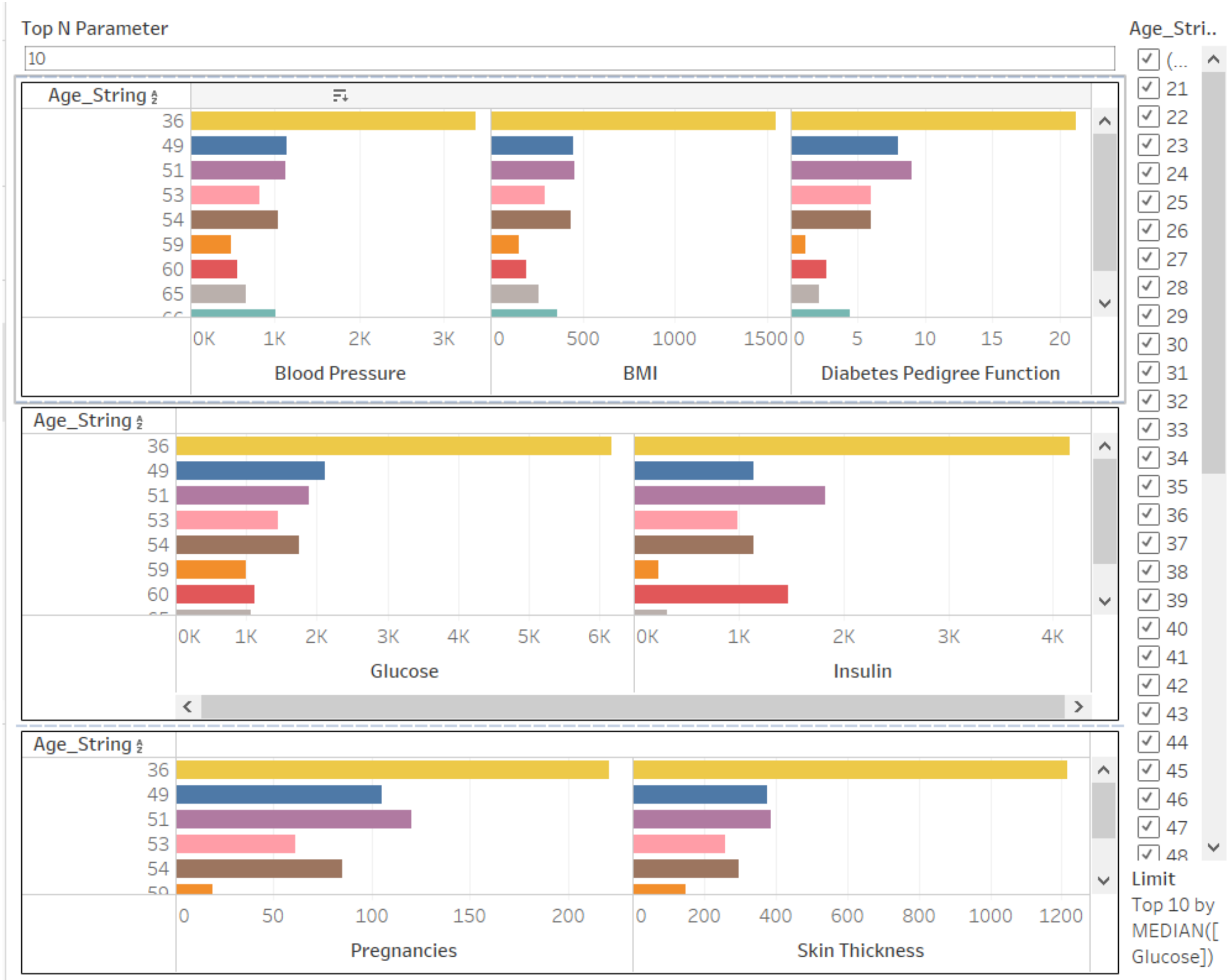
		Outcome_String						
		Median BMI		Median Blood Pressure		Median Diabetes Pedigree Function		Median Glucose
Age label	Age_String	0	1	0	1	0	1	0
21 to 31 years	21	27.7	34.6	64.0	62.0	0.4	0.5	108.0
	22	28.7	34.3	66.0	76.0	0.3	0.5	101.0
	23	29.9	33.3	68.0	72.0	0.3	0.5	107.0
	24	31.6	39.0	68.0	72.0	0.3	0.4	108.0
	25	30.1	36.0	65.0	70.0	0.4	0.5	92.0
	26	34.1	46.6	68.0	74.0	0.4	0.4	109.0
	27	28.4	34.2	72.0	76.0	0.4	0.5	109.0
	28	35.4	33.3	72.0	68.0	0.3	0.3	102.0
	29	32.0	33.7	76.0	72.0	0.4	0.4	111.0
	30	29.4	32.6	72.0	66.0	0.3	0.5	117.0
	31	28.0	33.8	72.0	76.0	0.5	0.3	107.0
	Total	30.2	34.9	68.0	72.0	0.4	0.4	106.0
32 to 41 years	32	29.3	36.4	74.0	76.0	0.5	0.8	99.0
	33	32.8	32.0	66.0	70.0	0.3	1.0	99.0
	34	30.5	28.4	80.0	64.0	0.7	0.7	120.0
	35	34.5	35.0	79.0	74.0	0.4	0.3	93.0
	36	27.0	32.9	68.0	72.0	0.6	0.4	95.0
	37	28.1	38.5	72.0	82.0	0.3	0.6	133.0
	38	33.3	34.1	78.0	72.0	0.3	0.3	89.0
	39	32.0	30.5	74.0	72.0	0.7	0.6	126.0
	40	31.4	34.5	74.0	80.0	0.3	0.4	126.0
	41	33.3	35.7	72.0	68.0	0.3	0.3	94.0
	Total	30.8	34.2	74.0	74.0	0.3	0.4	108.0
42 to 51 years	42	33.5	32.8	76.0	74.0	0.3	0.3	89.0
	43	34.0	37.1	58.0	80.0	0.4	0.2	98.0
	44	33.8	34.4	80.0	75.0	0.9	0.5	117.0
	45	34.6	32.8	78.0	78.0	0.2	0.4	122.0
	46	31.0	34.0	82.0	76.0	0.2	0.4	96.0
	47	35.5	30.3	106.0	68.0	0.3	0.3	68.0
	48	28.8	42.3	74.0	80.0	0.4	0.8	104.0



		Outcome_String						
		Median BMI	Median Blood Pressure	Median Diabetes Pedi..	Median Glucose	Median Insulin	Median Pregnancies	Median Skin Thickness
Age label	Age_String	Total	Total	Total	Total	Total	Total	Total
21 to 31 years	21	28.7	64.0	0.4	111.0	50.5	1.0	23.0
	22	29.7	67.0	0.3	106.0	47.5	1.0	23.0
	23	30.1	70.0	0.4	107.0	94.0	1.0	25.0
	24	33.3	68.0	0.3	112.0	77.0	1.0	28.0
	25	32.7	68.0	0.4	103.0	56.5	2.0	24.0
	26	35.9	71.0	0.4	113.0	92.0	1.0	25.0
	27	30.0	74.0	0.4	110.0	40.0	2.0	23.0
	28	35.4	72.0	0.3	115.0	74.0	3.0	29.0
	29	32.0	74.5	0.4	123.0	47.0	2.0	23.0
	30	32.3	70.0	0.3	120.0	40.0	3.5	24.0
	31	32.9	72.0	0.5	125.0	110.0	4.0	23.0
	Total	32.0	70.0	0.4	111.0	61.0	2.0	23.0
32 to 41 years	32	30.8	74.0	0.7	102.0	40.0	4.0	23.0
	33	32.8	68.0	0.4	121.0	55.0	4.0	23.0
	34	29.6	72.0	0.7	120.0	79.0	5.0	23.0
	35	35.0	76.0	0.3	123.0	72.0	5.0	29.0
	36	30.6	72.0	0.4	151.0	40.0	4.5	23.0
	37	31.3	78.0	0.3	133.0	40.0	5.0	28.0
	38	34.0	74.0	0.3	112.0	40.0	8.0	25.0
	39	31.3	74.0	0.6	135.0	75.0	8.0	32.0
	40	33.9	74.0	0.4	126.0	40.0	6.0	27.0
	41	34.9	72.0	0.3	129.0	40.0	6.0	25.0
	Total	32.8	74.0	0.4	124.0	40.0	5.0	25.0
42 to 51 years	42	33.2	74.0	0.3	103.0	40.0	7.0	28.0
	43	35.9	78.0	0.4	132.5	125.0	7.0	31.0
	44	34.1	76.0	0.6	129.0	40.0	8.5	23.0
	45	32.8	78.0	0.3	128.0	40.0	8.0	25.5
	46	32.9	76.0	0.4	102.0	40.0	6.0	28.0
	47	33.6	72.0	0.3	143.0	49.0	10.0	23.0
	48	28.8	74.0	0.4	106.0	54.0	9.0	23.0

4.7. Top N Attributes

The number of parameters are to be chosen from the dropdown. This is a parameter created in tableau which takes in the value N and it displays the top N values for the respective attributes. The attributes which get proclaimed are Blood Pressure, BMI, Diabetes Pedigree Function, Glucose, Insulin, Pregnancies and Skin Thickness.



Top N Parameter

10

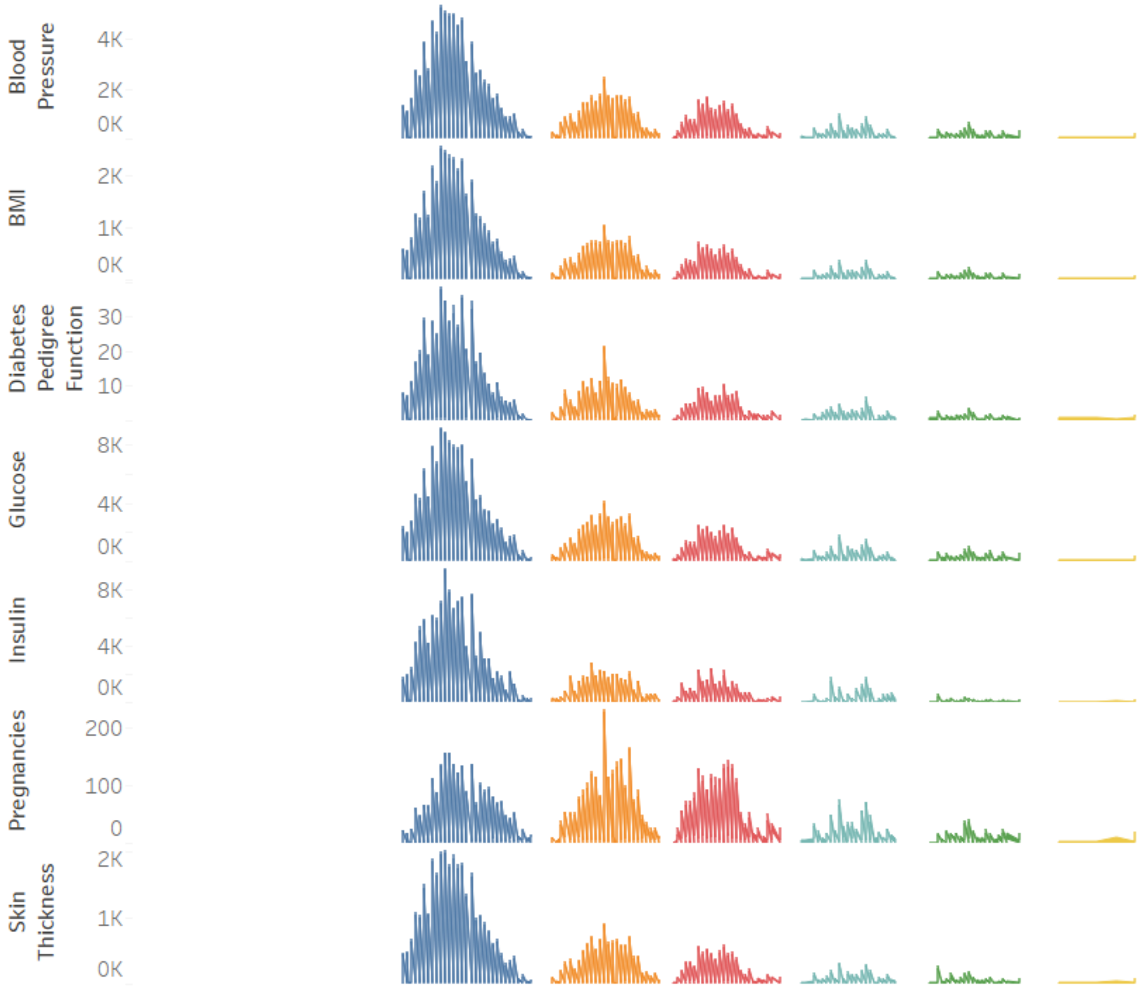
Limit
Top 10 by
MEDIAN([
Glucose])

4.8. KPIs in Age Label for median of Attributes

A Key Performance Indicator is a measurable value that shows how effectively a company is achieving key business objectives. For each attribute, the KPI is shown altogether in a single Dashboard. Differentiation of the KPI has been done on the basis of Age Label, hence different color is reflected on it.

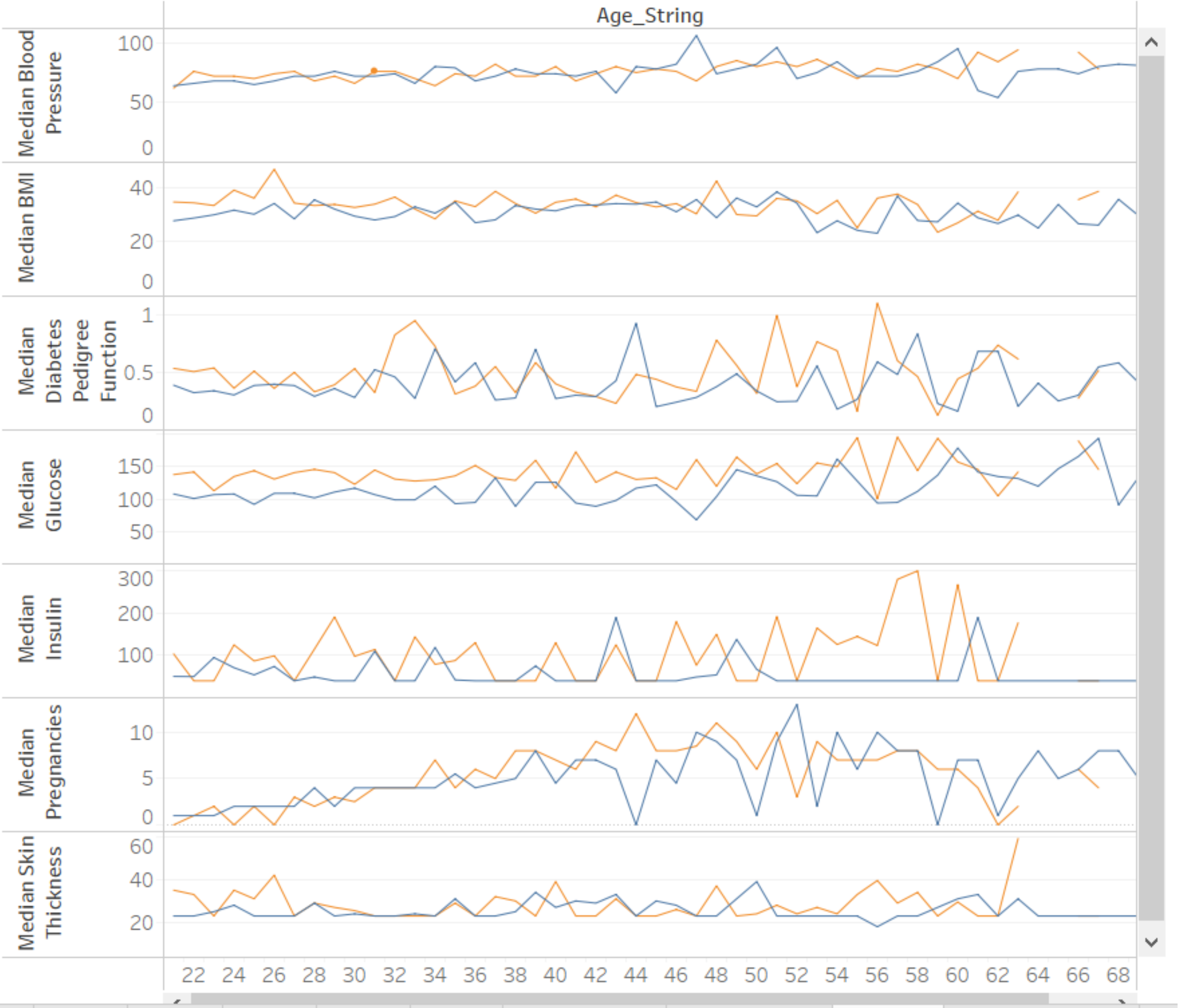
Here for the KPI, the median values have been selected for the Age group.

On clicking the particular Age Label, it gets reflected on all other attributes. Example if we select the blue color, which means '21-30 years', then it gets reflected on all other attributes. Similarly, it gets reflected on the rest of the other as per the selected Indicator.



4.9. Diabetes Variation wrt Median of All Attributes

Here in this Dashboard we have shown the line plot for the median values for two outcomes - having or not having diabetes. Clicking on a point of the line would show all attribute values for that age.



4.10. Possibility of Diabetes

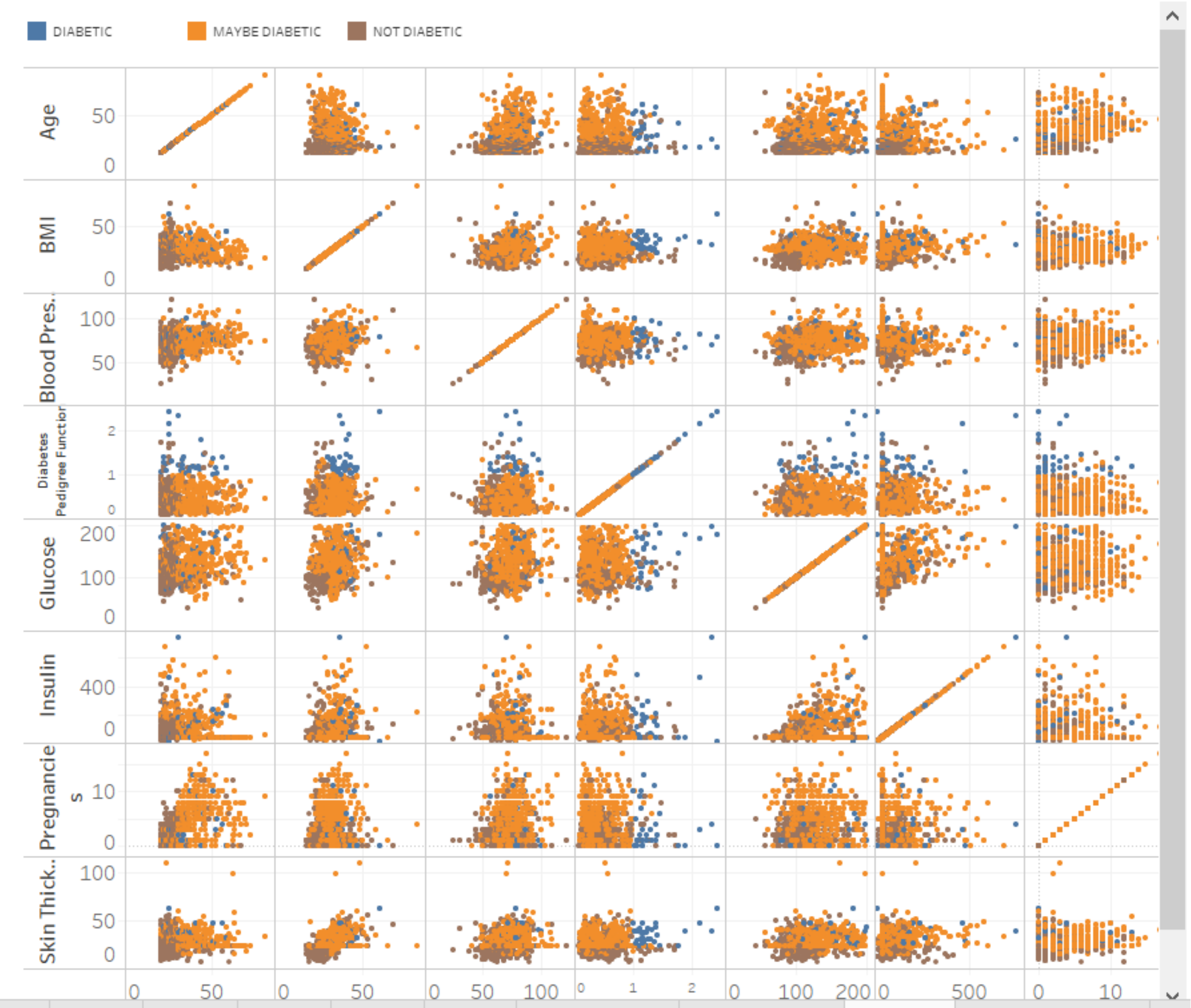
The Chances_of_Diabetes has been used here in this Dashboard. Blue Colour is for Diabetic People, Orange is for Maybe Diabetic People and Brown is for Non-Diabetic People.

Chances_of_Diabetes is a calculated field which stores 3 types of categories:-

1. Diabetic - Person who is having Diabetes. Mainly old age people of people having bad health.
2. Maybe Diabetic - Person who has a chance of getting affected by Diabetes.

3. Non - Diabetic - Mainly the young people.

The influence of these categories on diabetes dataset has been demonstrated.



4.11. Model Training Part1

In our dataset we divide the train and test dataset (train=80%,test=20%) and then train our model. Here are the following models that we used and its image has been attached here on this Dashboard. The models in this Dashboard are - Naive Bayes, Decision Tree and Random Forest.

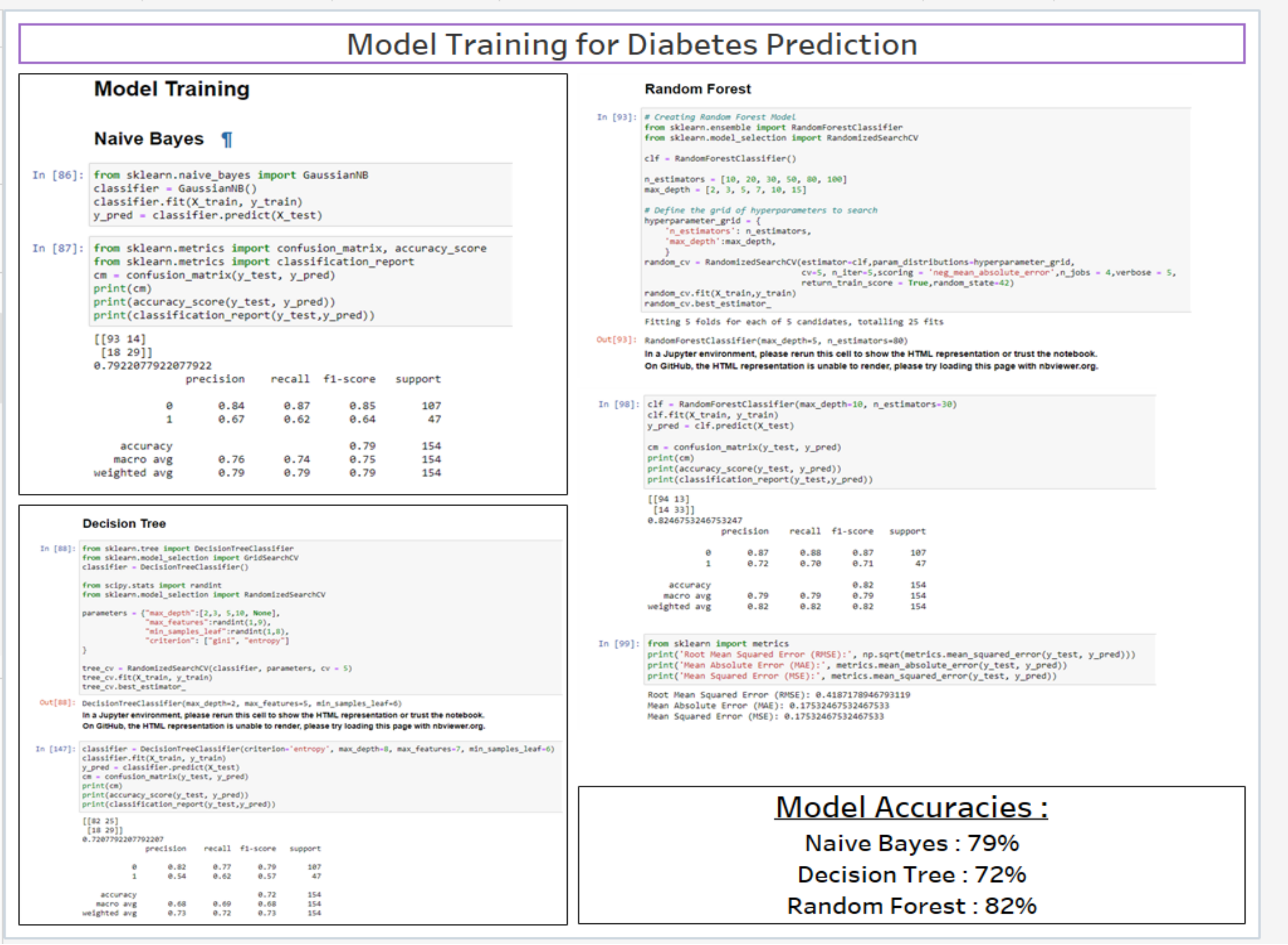
Whose accuracies are -

1. Naive Bayes : 79%

2.Decision Tree:72%

3.Random Forest:82%

This Dashboard consists of the Images of our model performance.



Model Training for Diabetes Prediction

Model Training

Naive Bayes ¶

```
In [86]: from sklearn.naive_bayes import GaussianNB
         classifier = GaussianNB()
         classifier.fit(X_train, y_train)
         y_pred = classifier.predict(X_test)

In [87]: from sklearn.metrics import confusion_matrix, accuracy_score
         from sklearn.metrics import classification_report
         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         print(accuracy_score(y_test, y_pred))
         print(classification_report(y_test,y_pred))

         [[93 14]
          [18 29]]
         0.7922077922077922
                       precision    recall  f1-score   support

                    0       0.84      0.87      0.85       107
                    1       0.67      0.62      0.64        47

             accuracy                           0.79       154
            macro avg       0.76      0.74      0.75       154
         weighted avg       0.79      0.79      0.79       154
```

Decision Tree

```
In [88]: from sklearn.tree import DecisionTreeClassifier
         from sklearn.model_selection import GridSearchCV
         classifier = DecisionTreeClassifier()

         from scipy.stats import randint
         from sklearn.model_selection import RandomizedSearchCV

         parameters = {"max_depth":[2,3, 5,10, None],
                       "max_features":randint(1,9),
                       "min_samples_leaf":randint(1,8),
                       "criterion": ["gini", "entropy"]
         }

         tree_cv = RandomizedSearchCV(classifier, parameters, cv = 5)
         tree_cv.fit(X_train, y_train)
         tree_cv.best_estimator_

Out[88]: DecisionTreeClassifier(max_depth=2, max_features=5, min_samples_leaf=6)
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [147]: classifier = DecisionTreeClassifier(criterion='entropy', max_depth=8, max_features=7, min_samples_leaf=6)
          classifier.fit(X_train, y_train)
          y_pred = classifier.predict(X_test)
          cm = confusion_matrix(y_test, y_pred)
          print(cm)
          print(accuracy_score(y_test, y_pred))
          print(classification_report(y_test,y_pred))

          [[82 25]
           [18 29]]
          0.7207792207792207
                        precision    recall  f1-score   support

                     0       0.82      0.77      0.79       107
                     1       0.54      0.62      0.57        47

              accuracy                           0.72       154
             macro avg       0.68      0.69      0.68       154
          weighted avg       0.73      0.72      0.73       154
```

Random Forest

```
In [93]: # Creating Random Forest Model
         from sklearn.ensemble import RandomForestClassifier
         from sklearn.model_selection import RandomizedSearchCV

         clf = RandomForestClassifier()

         n_estimators = [10, 20, 30, 50, 80, 100]
         max_depth = [2, 3, 5, 7, 10, 15]

         # Define the grid of hyperparameters to search
         hyperparameter_grid = {
             'n_estimators': n_estimators,
             'max_depth':max_depth,
             }
         random_cv = RandomizedSearchCV(estimator=clf,param_distributions=hyperparameter_grid,
                                        cv=5, n_iter=5,scoring = 'neg_mean_absolute_error',n_jobs = 4,verbose = 5,
                                        return_train_score = True,random_state=42)
         random_cv.fit(X_train,y_train)
         random_cv.best_estimator_

         Fitting 5 folds for each of 5 candidates, totalling 25 fits

Out[93]: RandomForestClassifier(max_depth=5, n_estimators=80)
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [98]: clf = RandomForestClassifier(max_depth=10, n_estimators=30)
         clf.fit(X_train, y_train)
         y_pred = clf.predict(X_test)

         cm = confusion_matrix(y_test, y_pred)
         print(cm)
         print(accuracy_score(y_test, y_pred))
         print(classification_report(y_test,y_pred))

         [[94 13]
          [14 33]]
         0.8246753246753247
                       precision    recall  f1-score   support

                    0       0.87      0.88      0.87       107
                    1       0.72      0.70      0.71        47

             accuracy                           0.82       154
            macro avg       0.79      0.79      0.79       154
         weighted avg       0.82      0.82      0.82       154

In [99]: from sklearn import metrics
         print('Root Mean Squared Error (RMSE):', np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
         print('Mean Absolute Error (MAE):', metrics.mean_absolute_error(y_test, y_pred))
         print('Mean Squared Error (MSE):', metrics.mean_squared_error(y_test, y_pred))

         Root Mean Squared Error (RMSE): 0.4187178946793119
         Mean Absolute Error (MAE): 0.17532467532467533
         Mean Squared Error (MSE): 0.17532467532467533
```

Model Accuracies :

Naive Bayes : 79%

Decision Tree : 72%

Random Forest : 82%

4.12. Model Training Part2

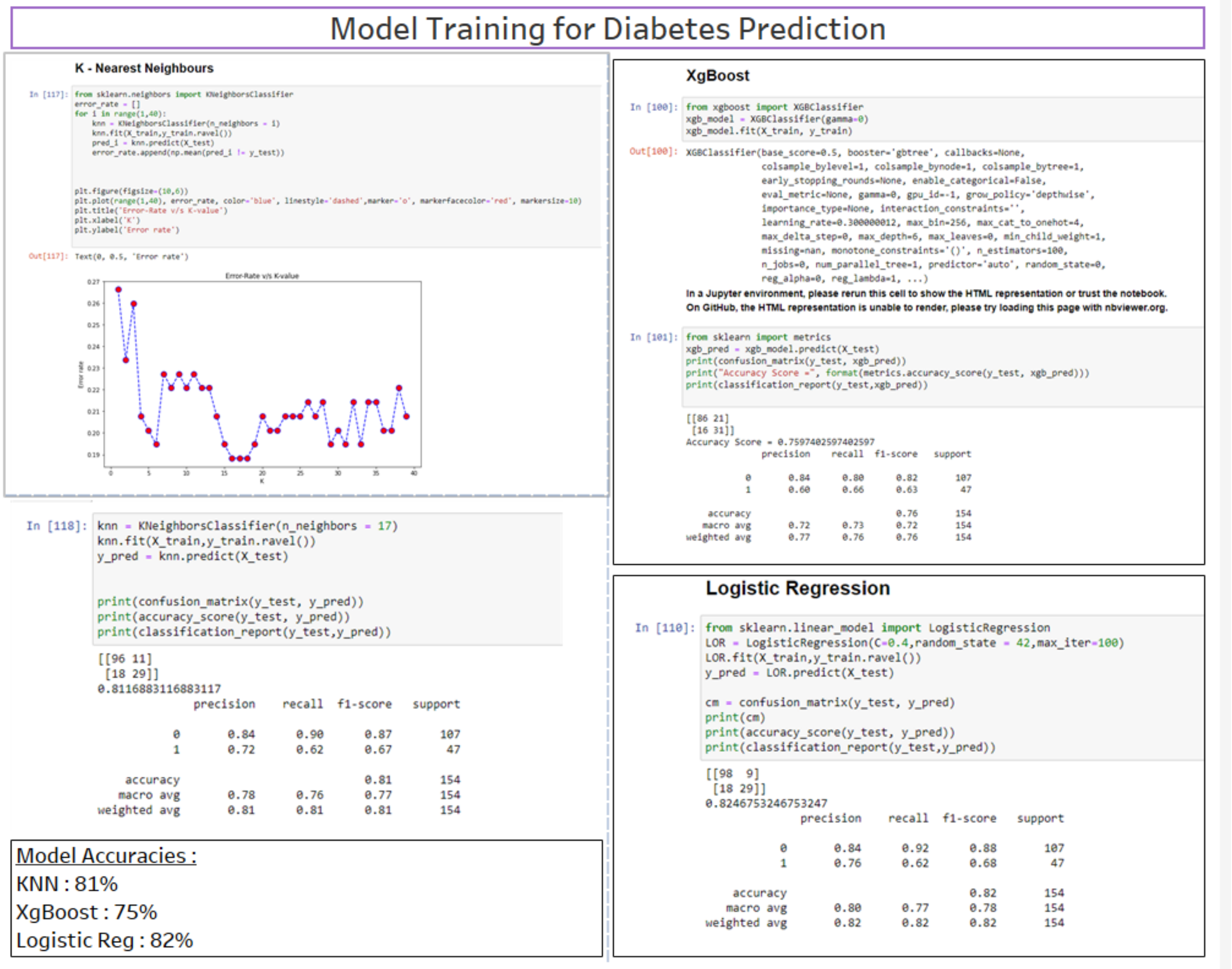
The Following model performance has been reflected in this

1. KNN - 81%

2. Xgboost - 75%

3. Logistic Regression - 82%

The Image has been attached.



Model Training for Diabetes Prediction

K - Nearest Neighbours

```
In [117]: from sklearn.neighbors import KNeighborsClassifier
          error_rate = []
          for i in range(1,40):
              knn = KNeighborsClassifier(n_neighbors = i)
              knn.fit(X_train,y_train.ravel())
              pred_i = knn.predict(X_test)
              error_rate.append(np.mean(pred_i != y_test))



          plt.figure(figsize=(10,6))
          plt.plot(range(1,40), error_rate, color='blue', linestyle='dashed',marker='o', markerfacecolor='red', markersize=10)
          plt.title('Error-Rate v/s K-value')
          plt.xlabel('K')
          plt.ylabel('Error rate')

Out[117]: Text(0, 0.5, 'Error rate')
```

```
In [118]: knn = KNeighborsClassifier(n_neighbors = 17)
          knn.fit(X_train,y_train.ravel())
          y_pred = knn.predict(X_test)


          print(confusion_matrix(y_test, y_pred))
          print(accuracy_score(y_test, y_pred))
          print(classification_report(y_test,y_pred))

          [[96 11]
           [18 29]]
          0.8116883116883117
                        precision    recall  f1-score   support

                     0       0.84      0.90      0.87       107
                     1       0.72      0.62      0.67        47

              accuracy                           0.81       154
             macro avg       0.78      0.76      0.77       154
          weighted avg       0.81      0.81      0.81       154
```

Model Accuracies :
KNN : 81%
XgBoost : 75%
Logistic Reg : 82%

XgBoost

```
In [100]: from xgboost import XGBClassifier
          xgb_model = XGBClassifier(gamma=0)
          xgb_model.fit(X_train, y_train)

Out[100]: XGBClassifier(base_score=0.5, booster='gbtree', callbacks=None,
                        colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
                        early_stopping_rounds=None, enable_categorical=False,
                        eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
                        importance_type=None, interaction_constraints='',
                        learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
                        max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
                        missing=nan, monotone_constraints='()', n_estimators=100,
                        n_jobs=0, num_parallel_tree=1, predictor='auto', random_state=0,
                        reg_alpha=0, reg_lambda=1, ...)
          In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
          On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.
```

```
In [101]: from sklearn import metrics
          xgb_pred = xgb_model.predict(X_test)
          print(confusion_matrix(y_test, xgb_pred))
          print("Accuracy Score =", format(metrics.accuracy_score(y_test, xgb_pred)))
          print(classification_report(y_test,xgb_pred))

          [[86 21]
           [16 31]]
          Accuracy Score = 0.7597402597402597
                        precision    recall  f1-score   support

                     0       0.84      0.80      0.82       107
                     1       0.60      0.66      0.63        47

              accuracy                           0.76       154
             macro avg       0.72      0.73      0.72       154
          weighted avg       0.77      0.76      0.76       154
```

Logistic Regression

```
In [110]: from sklearn.linear_model import LogisticRegression
          LOR = LogisticRegression(C=0.4,random_state = 42,max_iter=100)
          LOR.fit(X_train,y_train.ravel())
          y_pred = LOR.predict(X_test)

          cm = confusion_matrix(y_test, y_pred)
          print(cm)
          print(accuracy_score(y_test, y_pred))
          print(classification_report(y_test,y_pred))

          [[98  9]
           [18 29]]
          0.8246753246753247
                        precision    recall  f1-score   support

                     0       0.84      0.92      0.88       107
                     1       0.76      0.62      0.68        47

              accuracy                           0.82       154
             macro avg       0.80      0.77      0.78       154
          weighted avg       0.82      0.82      0.82       154
```

4.13. Model Training Part3

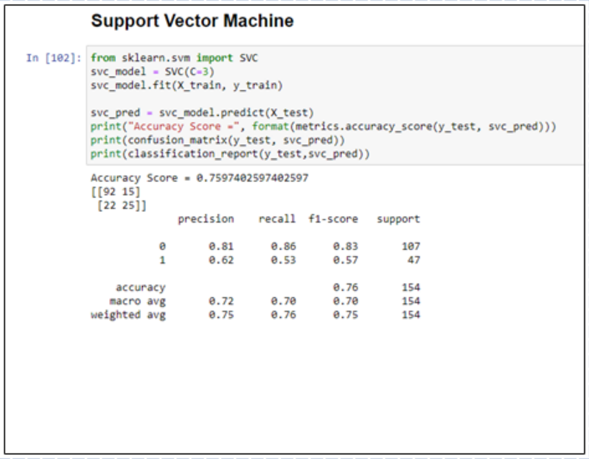
The Following model performance has been reflected in this

1. Support Vector Machine - 76%

2. Neural Network - 75%

The Image has been attached in the Dashboard.

Model Training for Diabetes Prediction



Support Vector Machine

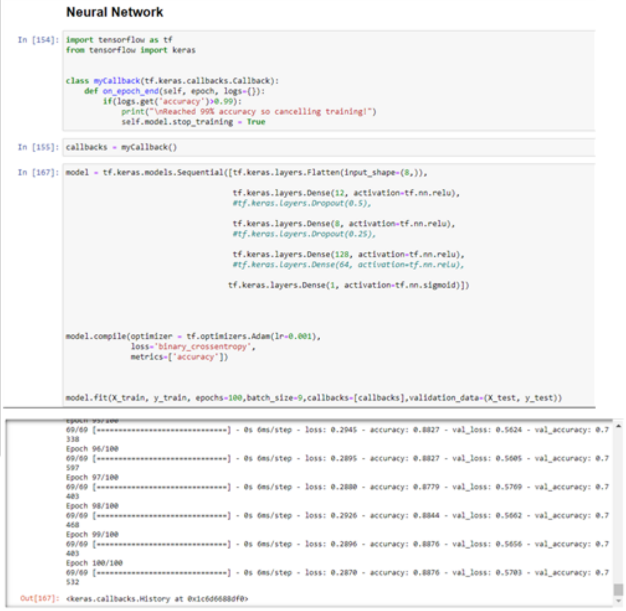
```
In [102]: from sklearn.svm import SVC
          svc_model = SVC(C=3)
          svc_model.fit(X_train, y_train)

          svc_pred = svc_model.predict(X_test)
          print("Accuracy Score =", format(metrics.accuracy_score(y_test, svc_pred)))
          print(confusion_matrix(y_test, svc_pred))
          print(classification_report(y_test,svc_pred))

          Accuracy Score = 0.7597402597402597
          [[92 15]
           [22 25]]
                        precision    recall  f1-score   support

                     0       0.81      0.86      0.83       107
                     1       0.62      0.53      0.57        47

              accuracy                           0.76       154
             macro avg       0.72      0.70      0.70       154
          weighted avg       0.75      0.76      0.75       154
```



Neural Network

```
In [154]: import tensorflow as tf
          from tensorflow import keras


          class myCallback(tf.keras.callbacks.Callback):
              def on_epoch_end(self, epoch, logs={}):
                  if(logs.get('accuracy')>0.99):
                      print("\nReached 99% accuracy so cancelling training!")
                      self.model.stop_training = True

In [155]: callbacks = myCallback()

In [167]: model = tf.keras.models.Sequential([tf.keras.layers.Flatten(input_shape=(8,)),

                                              tf.keras.layers.Dense(12, activation=tf.nn.relu),
                                              #tf.keras.layers.Dropout(0.5),

                                              tf.keras.layers.Dense(8, activation=tf.nn.relu),
                                              #tf.keras.layers.Dropout(0.25),

                                              tf.keras.layers.Dense(128, activation=tf.nn.relu),
                                              #tf.keras.layers.Dense(64, activation=tf.nn.relu),

                                             tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)])



          model.compile(optimizer = tf.optimizers.Adam(lr=0.001),
                        loss='binary_crossentropy',
                        metrics=['accuracy'])



          model.fit(X_train, y_train, epochs=100,batch_size=9,callbacks=[callbacks],validation_data=(X_test, y_test))

          Epoch 95/100
          69/69 [==============================] - 0s 6ms/step - loss: 0.2945 - accuracy: 0.8827 - val_loss: 0.5624 - val_accuracy: 0.7
          338
          Epoch 96/100
          69/69 [==============================] - 0s 6ms/step - loss: 0.2895 - accuracy: 0.8827 - val_loss: 0.5605 - val_accuracy: 0.7
          597
          Epoch 97/100
          69/69 [==============================] - 0s 6ms/step - loss: 0.2880 - accuracy: 0.8779 - val_loss: 0.5769 - val_accuracy: 0.7
          403
          Epoch 98/100
          69/69 [==============================] - 0s 6ms/step - loss: 0.2926 - accuracy: 0.8844 - val_loss: 0.5662 - val_accuracy: 0.7
          468
          Epoch 99/100
          69/69 [==============================] - 0s 6ms/step - loss: 0.2896 - accuracy: 0.8876 - val_loss: 0.5656 - val_accuracy: 0.7
          403
          Epoch 100/100
          69/69 [==============================] - 0s 6ms/step - loss: 0.2870 - accuracy: 0.8876 - val_loss: 0.5703 - val_accuracy: 0.7
          532

Out[167]: <keras.callbacks.History at 0x1c6d6688df0>
```

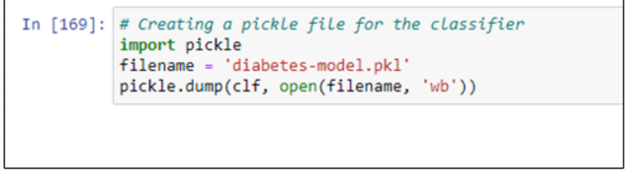
Model Accuracies :

Support Vector Machine : 76%

Neural Network : 75%

Random Forest and Logistics Regression proves to be the best Solution for this dataset.

Saving the Model as pickle file



```
In [169]: # Creating a pickle file for the classifier
          import pickle
          filename = 'diabetes-model.pkl'
          pickle.dump(clf, open(filename, 'wb'))
```

5. Future Work

- The combination of Sheets, Dashboards and Stories could be made more attractive.
- More calculations related to the medical field and be formulated.
- More the dataset size, higher the influence for real life intuitions can be made.
- If the kind-of-attributes remains of any other medical related problems, this dashboard can be of use for point of reference.