**Final Report**

Group5-Brenda-Jingrui Zhang

**Brief Introduction**

The title of our machine learning project is Predicting Diabetes: A Machine Learning Perspective. In this project, we design an application to predict the probability that a woman would get diabetes based on the machine learning algorithms such as Random Forest (RF), Support Vector Machine (SVM) and Naïve Bayes (NB) we learned in professor’s lecture. By inputting some of their health data such as Pregnancy frequency, age and blood pressure into the blank, women can get some graphs and their probability of getting diabetes.

**Background and Goal**

We choose this topic because the estimated number of people aged 20–79 years living with diabetes in 2021 was 537 million. These numbers are expected to increase to 643 million by 2030 and 783 million by 2045[1]. Diabetes can also cause death and complications which is hard to cure. Therefore, it is necessary for them to do some early prediction so that they would be able to take preventive measures as soon as possible.

Nowadays, several studies have explored predictive models for diabetes. Southern et al. (2010) studied the validity of using administrative databases to identify diabetes patients and discussed how linking laboratory data can improve accuracy. Jayanthi et al. (2017) reviewed various predictive models used in healthcare, specifically mentioning models for diabetes prediction. They discussed the potential of machine learning and data mining techniques to build accurate predictive models. In the French national health insurance information system (SNDS), three diabetes case definition algorithms were applied to identify diabetic patients. All of those models performed well, with specificities and negative predictive value over 99%, in the identification research conducted by S. Fuentes et al in 2019.

Our goal is to use the information uploaded by female users to aid in individuals’ early detection and preventive measures. We focus specifically on female because firstly, the hormone production would increase the likelihood of diabetes during pregnancy [2] and secondly, the patients who provide the data are all females.

**About the dataset**

This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether a patient has diabetes, based on certain diagnostic measurements included in the dataset. 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.

[3]

The diabetes datasets is 23.88kB, with 9 columns and 768 sets of data. The last column is outcome, among which 0 means without diabetes while 1 means with diabetes. There are 500 “0”and 268 “1”in the last column.

**Method**

1. Load the datasets
2. Fix the datasets
3. Visualize the datasets to see each column’s distribution
4. Splitting the data
5. Apply datasets to three different machine learning algorithms

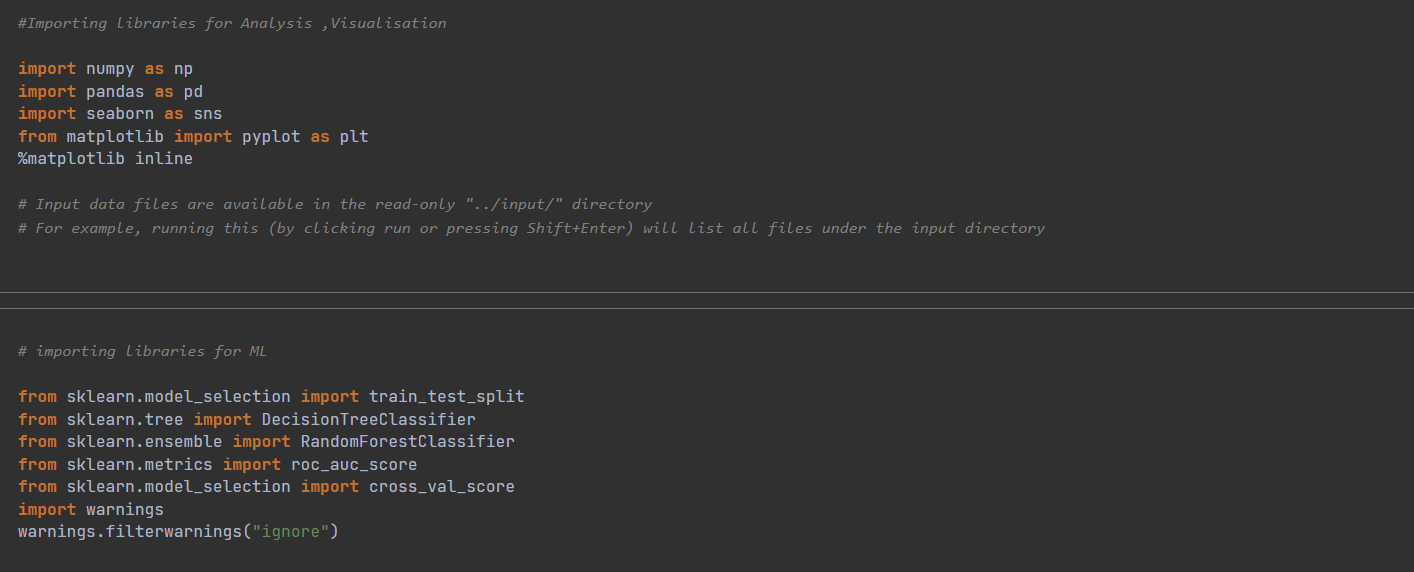
Naive Bayes

Support Vector Machine

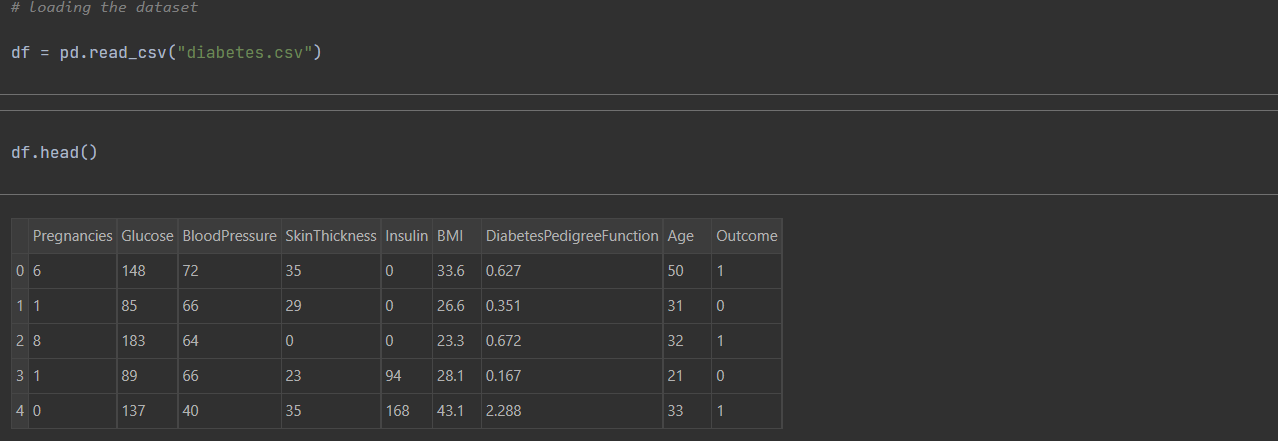
Random Forest

1. Tuning Parameters
2. Compare those models after tuned
3. Create an easy application

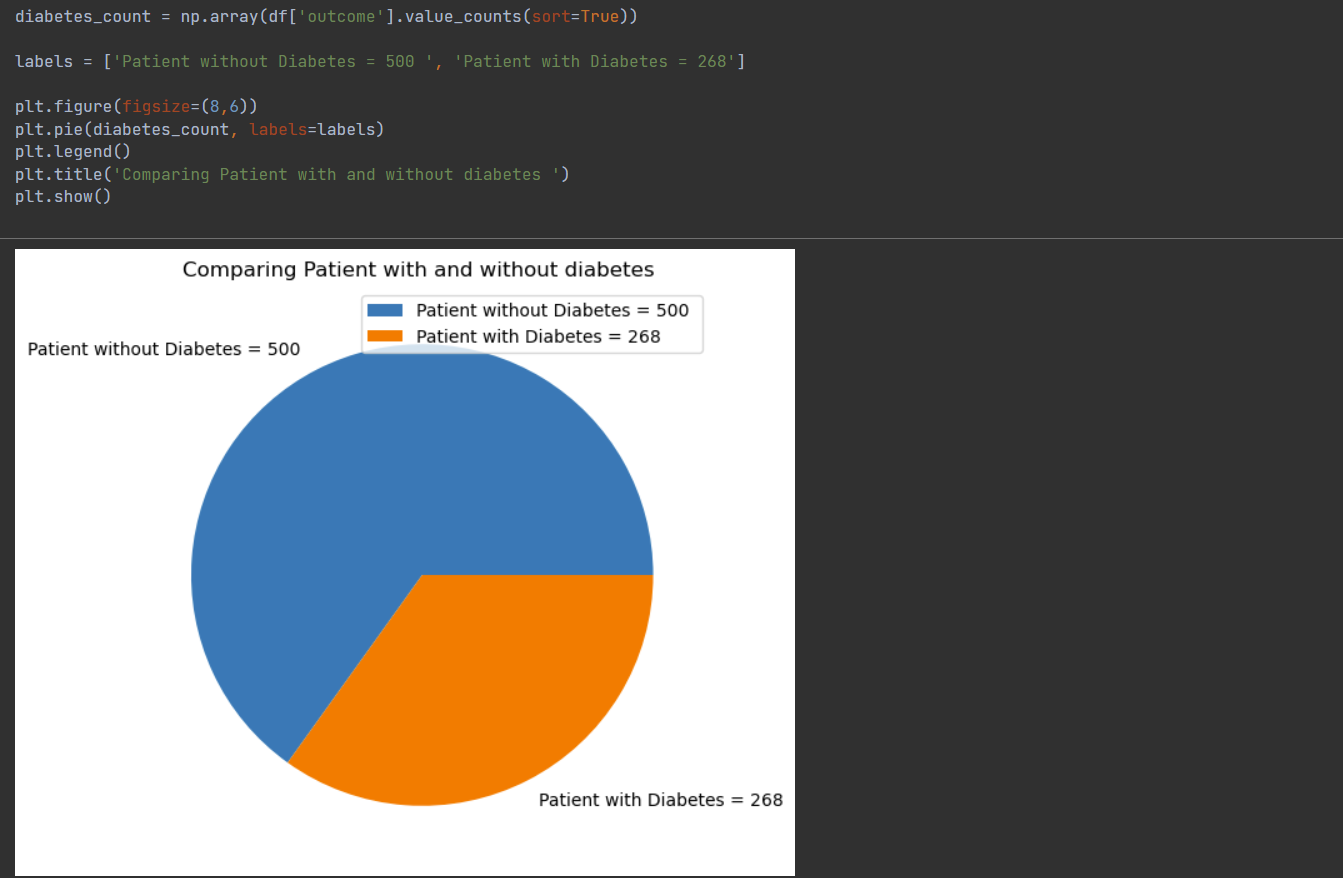
We import some necessary packages related to machine learning.



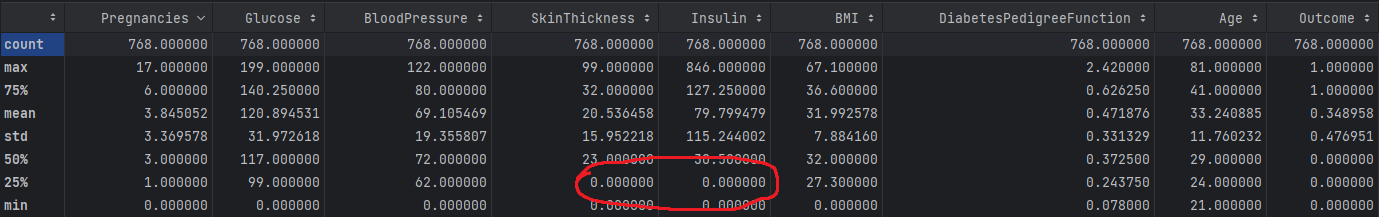
Then, we load the dataset.

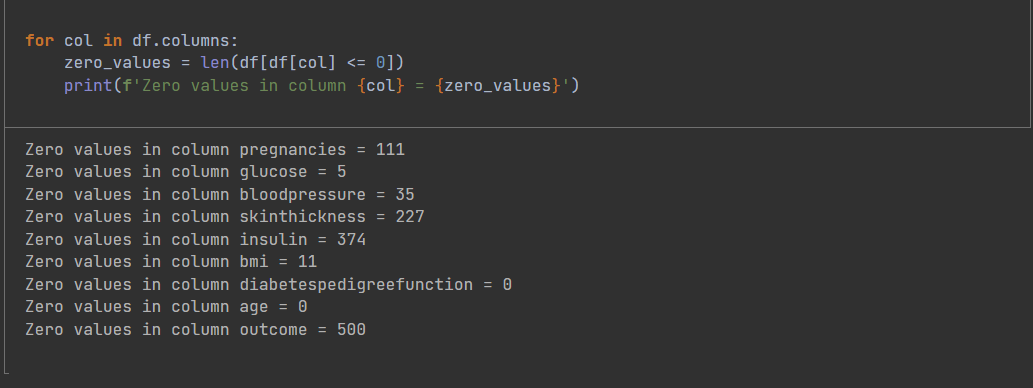


Use pie chart to visualize the distribution of datasets

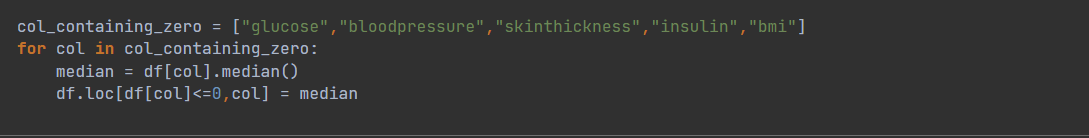


We find that there are some abnormal zeros in each column. The numbers of abnormal zeros are shown in the graph below.

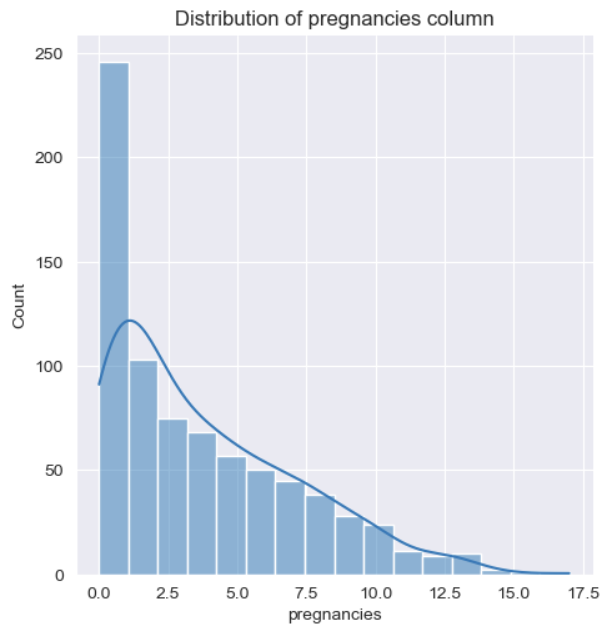
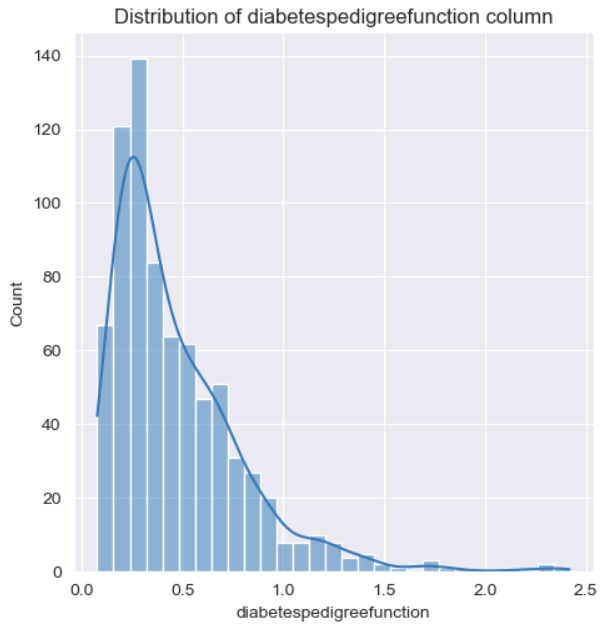
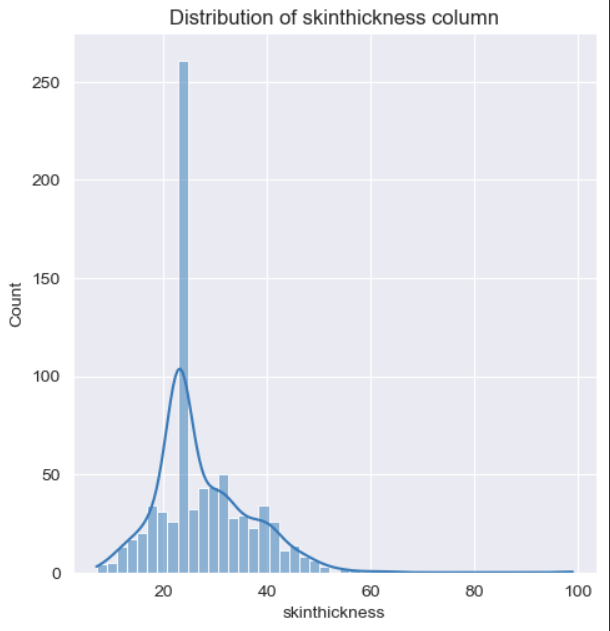
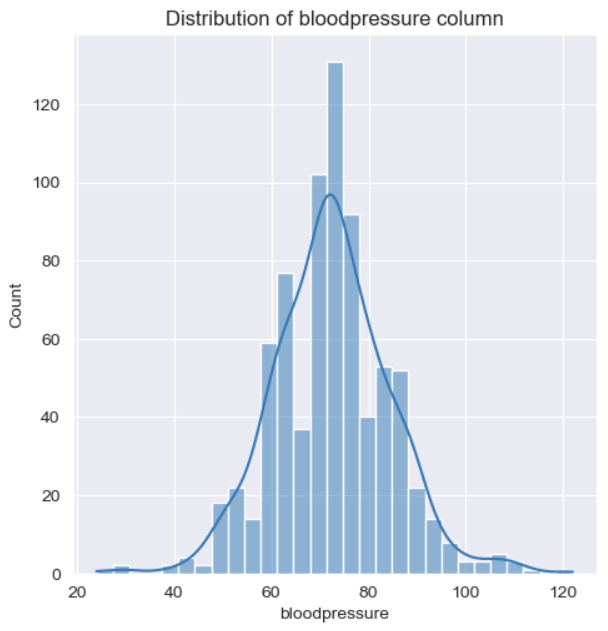




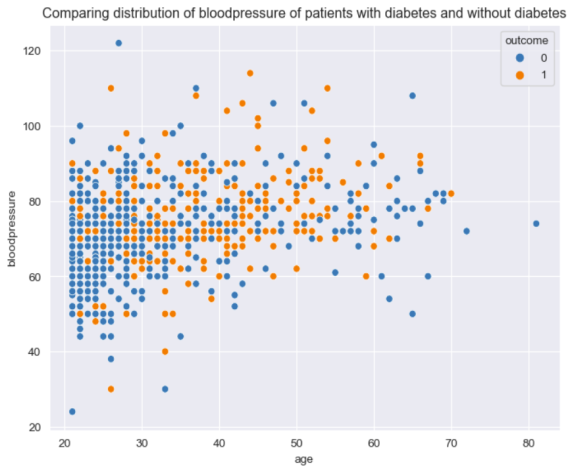
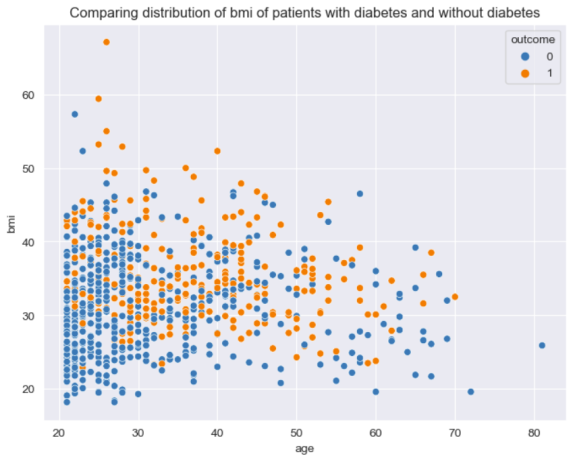
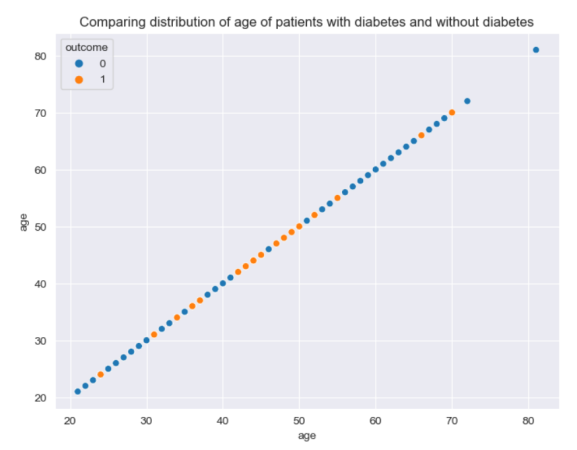
Apparently, zeros in columns “glucose”, “bloodpressure”, “skinthickness”, “insulin” and “bmi” are abnormal. We use median to replace those zeros.



Use histogram to visualize the distribution of each column

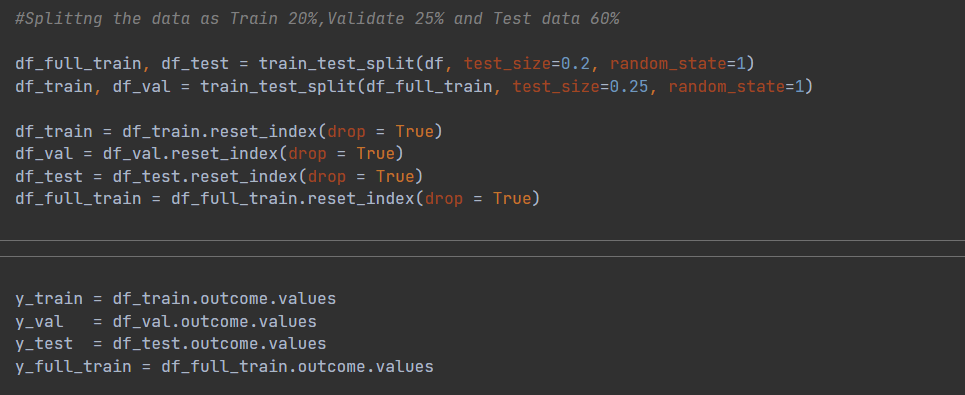


Use scatter plot to show the distribution of each index among people with and without diabetes. In the graphs below you can see the relationship between age and other index. For example the glucose and age. The yellow dots cluster between 20 and 60 on X axis while cluster between 100 and 200. Therefore, we can say that people who usually has a glucose level above 100 before eatting and aged between 20 and 60 are likely to get diabetes.

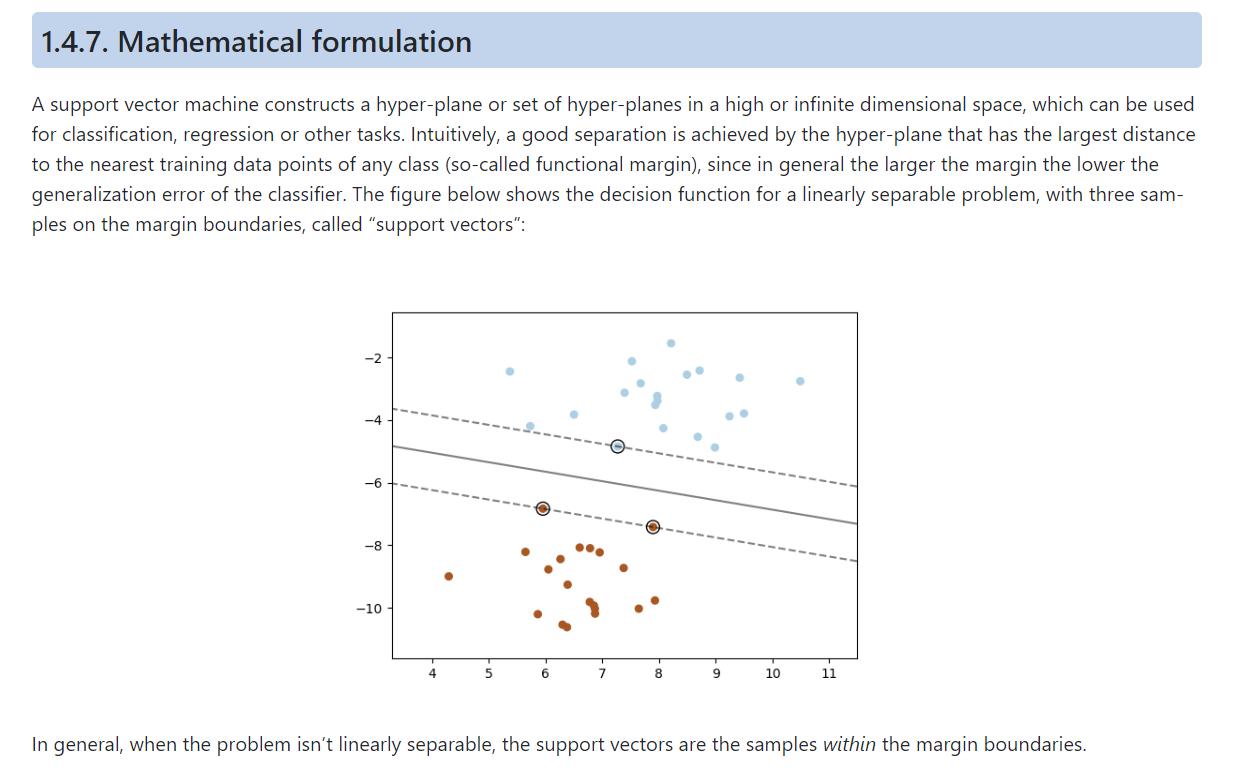
Split the data into training set and test set.

We use 20% for training, 20% for validation and 60% for test.

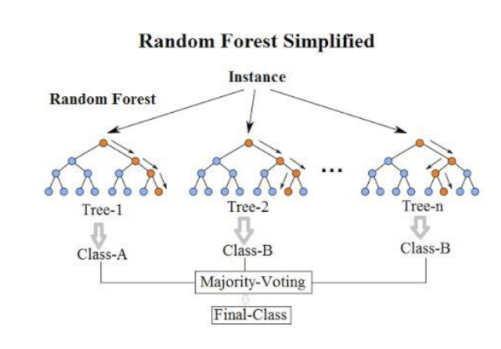


Apply datasets to three algorithms.

* Mathematical formulation of SVM

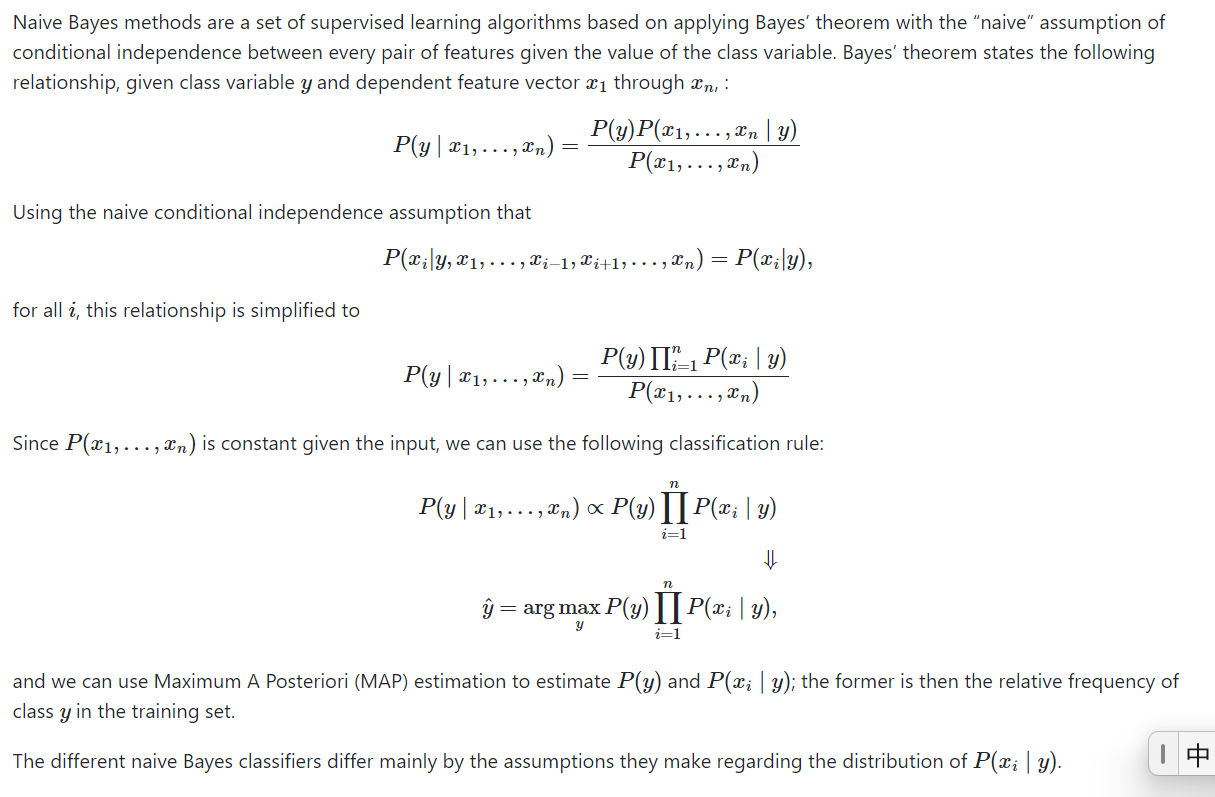
[4]

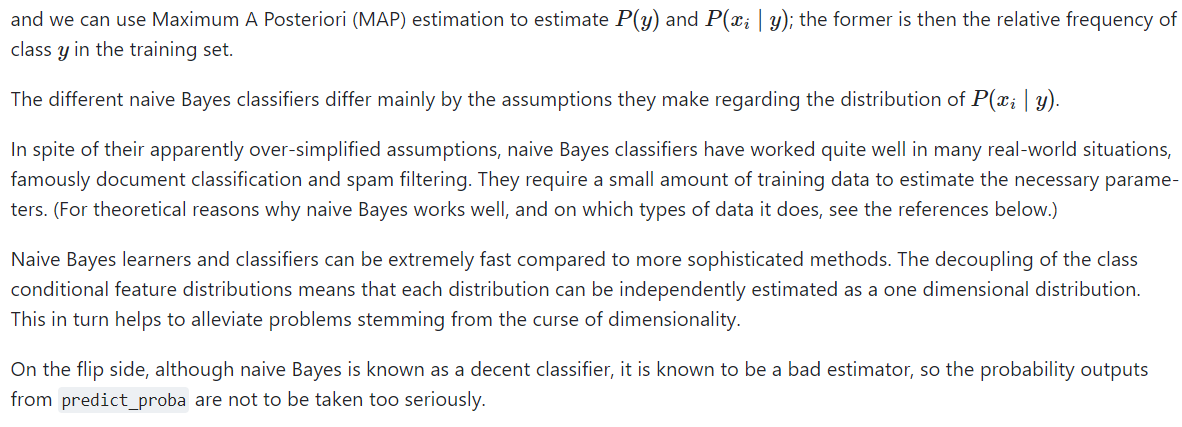
* Algorithm diagram of RF



[5]

* Mathematical formulation of NB





[6]

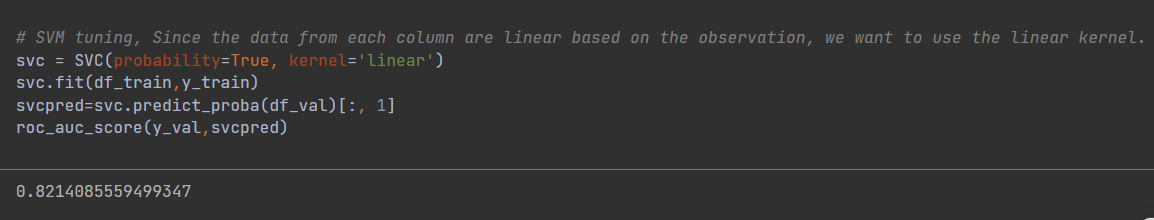
We apply these three algorithms to the dataset and get three accuracies.

**SVM: 0.7948813749299458, NB：0.7638707266953111, RF：0.781804595553895**

However, the accuracies are quite low.

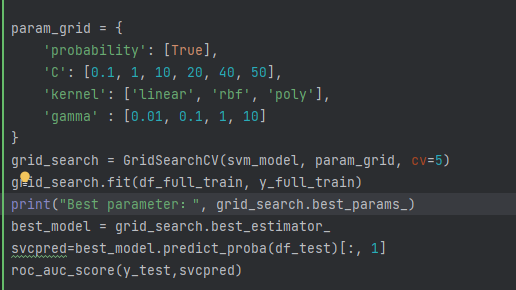
**Tuning parameters**

We specifically tune the parameter of RF and SVM model. Firstly, we change the kernel of SVM into kernel because the dataset is linear and consistent.



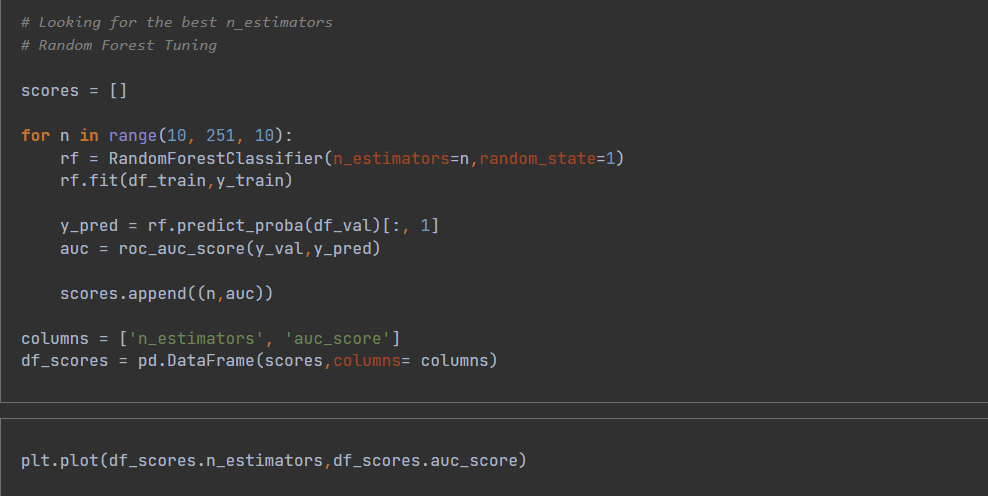
The accuracy of SVM increases to 0.8214085559499347

Then, for the further tuning, we adjust the parameter C to 40 and gamma to 0.1, the accuracy was improved to **0.8289744068746497**.



For RF algorithm, we need to look for three parameters for it.

The codes are in the figures below.



To find the best n\_estimators



To find the best max\_depth



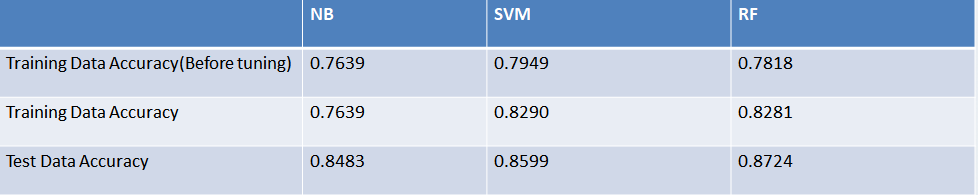
To find the best min\_samples\_leaf

We also plot the figure of each parameter. According to the figures, we finally choose 60 as n-estimator, 5 as max\_depth and 1 as min\_samples\_leaf. In this case, the accuracy of **RF** reaches **0.8281337567719036**.

The accuracies mentioned above are all about training set.

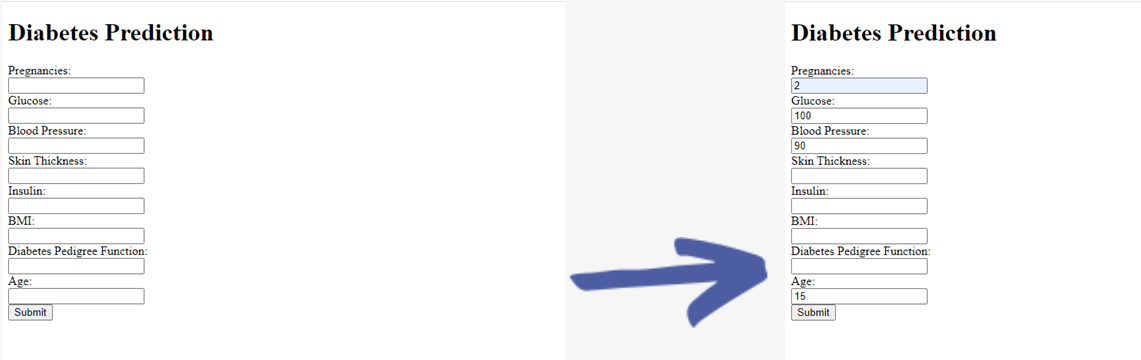
After training, we applied three algorithms to the test dataset and got the accuracies of test set.**NB:0.8483011937557392, SVM:0.8598714416896235, RF:0.8723599632690541**

**Result**

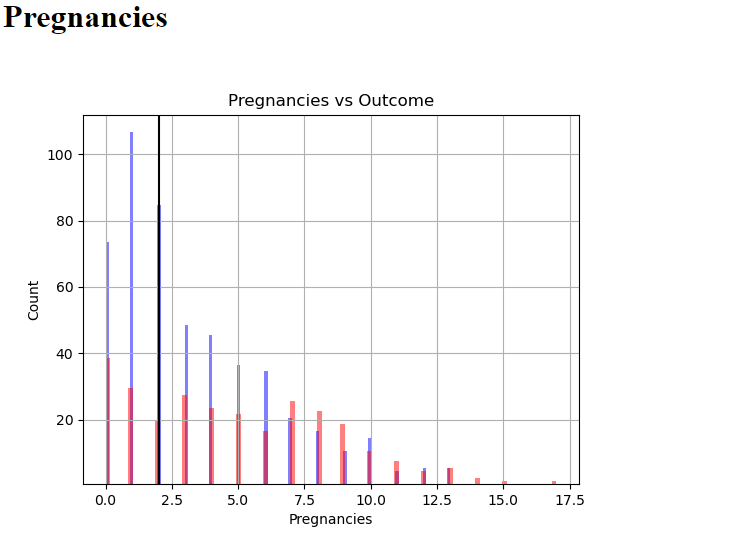


Comparing test data accuracies, we find out the RF has the highest accuracy based on the diabetes datasets.

Then we apply the RF model to our application and make a flask.



Users input some of the data about their body into to the blank and then get graphs and index to show the possibility of getting diabetes. Users do not need to fill all the blanks to get the result, but more data would lead to a more precise result.



Below the graph the probability would be shown if the program run as a page on the network.

The main code of the application is shown below.

**Application.py**

from flask import Flask, render\_template, request  
import numpy as np  
import pandas as pd  
import joblib  
import matplotlib.pyplot as plt  
import base64  
from io import BytesIO  
  
app = Flask(\_\_name\_\_)  
  
def generate\_plot(df, feature, feature\_value):  
 fig, ax = plt.subplots()  
 df[df['Outcome']==0][feature].hist(ax=ax, bins=100, bottom=0.6, color='blue', alpha=0.5)  
 df[df['Outcome']==1][feature].hist(ax=ax, bins=100, bottom=0.6, color='red', alpha=0.5)  
 plt.axvline(feature\_value, color='black') # The value for the current user  
 plt.title(f'{feature} vs Outcome')  
 plt.grid(True)  
  
 plt.xlabel(feature)  
 plt.ylabel('Count')  
  
  
 buf = BytesIO()  
 plt.savefig(buf, format="png")  
 data\_uri = base64.b64encode(buf.getvalue()).decode('ascii')  
 img\_str = "data:image/png;base64,{0}".format(data\_uri)  
  
 return img\_str  
  
def generate\_plots(df, data):  
 plots = {}  
 for feature in data.keys():  
 plots[feature] = generate\_plot(df, feature, data[feature])  
 return plots  
  
@app.route('/')  
def home():  
 return render\_template('home.html')  
  
@app.route('/predict', methods=['POST'])  
def predict():  
 model = joblib.load('best.pkl')  
 df = pd.read\_csv('diabetes.csv')  
 col\_containing\_zero = ["Glucose", "BloodPressure", "SkinThickness", "Insulin", "BMI"]  
 for col in col\_containing\_zero:  
 median = df[col].median()  
 df.loc[df[col] <= 0, col] = median  
  
 medians = {  
 'Pregnancies': df['Pregnancies'].median(),  
 'Glucose': df['Glucose'].median(),  
 'BloodPressure': df['BloodPressure'].median(),  
 'SkinThickness': df['SkinThickness'].median(),  
 'Insulin': df['Insulin'].median(),  
 'BMI': df['BMI'].median(),  
 'DiabetesPedigreeFunction': df['DiabetesPedigreeFunction'].median(),  
 'Age': df['Age'].median()  
 }  
  
 data = {  
 'Pregnancies': float(request.form.get('Pregnancies')),  
 'Glucose': float(request.form.get('Glucose')) if request.form.get('Glucose') else None,  
 'BloodPressure': float(request.form.get('BloodPressure')) if request.form.get('BloodPressure') else None,  
 'SkinThickness': float(request.form.get('SkinThickness')) if request.form.get('SkinThickness') else None,  
 'Insulin': float(request.form.get('Insulin')) if request.form.get('Insulin') else None,  
 'BMI': float(request.form.get('BMI')) if request.form.get('BMI') else None,  
 'DiabetesPedigreeFunction': float(request.form.get('DiabetesPedigreeFunction')) if request.form.get(  
 'DiabetesPedigreeFunction') else None,  
 'Age': float(request.form.get('Age'))  
 }  
 provided\_data = {k: v for k, v in data.items() if v is not None}  
 for key in data:  
 if data[key] is None:  
 data[key] = medians[key]  
  
 features = np.array([list(data.values())])  
 prediction\_proba = model.predict\_proba(features.reshape(1, -1))  
 plots = generate\_plots(df, provided\_data)  
  
 return render\_template("result.html", instruction="Under are your data fit in dataset, Blue means people do not have diabetes, Red means people have diabetes",plots=plots,probability=f"The probability of the patient having diabetes is {prediction\_proba[0][1] \* 100:.2f}%.")  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run(debug=True)

**home.html**

<!DOCTYPE html>  
<html>  
<head>  
 <title>Diabetes Prediction</title>  
</head>  
<body>  
 <h1>Diabetes Prediction</h1>  
 <form action="/predict" method="post">  
 <label for="Pregnancies">Pregnancies:</label><br>  
 <input type="text" id="Pregnancies" name="Pregnancies"><br>  
 <label for="Glucose">Glucose:</label><br>  
 <input type="text" id="Glucose" name="Glucose"><br>  
 <label for="BloodPressure">Blood Pressure:</label><br>  
 <input type="text" id="BloodPressure" name="BloodPressure"><br>  
 <label for="SkinThickness">Skin Thickness:</label><br>  
 <input type="text" id="SkinThickness" name="SkinThickness"><br>  
 <label for="Insulin">Insulin:</label><br>  
 <input type="text" id="Insulin" name="Insulin"><br>  
 <label for="BMI">BMI:</label><br>  
 <input type="text" id="BMI" name="BMI"><br>  
 <label for="DiabetesPedigreeFunction">Diabetes Pedigree Function:</label><br>  
 <input type="text" id="DiabetesPedigreeFunction" name="DiabetesPedigreeFunction"><br>  
 <label for="Age">Age:</label><br>  
 <input type="text" id="Age" name="Age"><br>  
 <input type="submit" value="Submit">  
 </form>  
</body>  
</html>

**Result.html**

<!DOCTYPE html>  
<html lang="en">  
<head>  
 <meta charset="UTF-8">  
 <title>Result</title>  
</head>  
<body>  
 {% if instruction is defined and plots is defined and probability is defined %}  
 <p>{{instruction }}</p>  
 {% for feature, plot in plots.items() %}  
 <h1>{{ feature }}</h1>  
 <img src ="{{ plot }}" alt="{{ feature }} plot">  
 {% endfor %}  
 <p>{{ probability }}</p>  
 {% endif %}  
</body>  
</html>

**Conclusion**

In the program, we use three algorithms to predict the possibility for women to get diabetes based on the diabetes dataset. We also try to make a flask to apply our model and the accuracy of this model is 87.2%. We believe that machine learning would be a valuable tool in harnessing big data sources, which may open up new avenues in diabetes research and prevention.

Personally speaking, by doing this project, I have learned a lot. Machine learning is quite difficult to me for the lack of some basic knowledge, but I enjoyed the process of learning it step by step with the help of professor and teacher assistants.

The professor’s lecture is instructive and inspiring to me because I got to know many website and resources about machine learning and also practiced those algorithms by myself. Although compared to other groups, our final presentation didn’t focus on neuron network, we all made contributions to it. Andy conceived the program, Ethan and Eric do the coding, I made the PPT and we all made the final presentation. In the process of communication with Ethan and Eric, I also picked up some knowledge and tips on machine learning. Putting the method and result on the PPT gives me a sense of achievement although it is not introduced in details on the final presentation. But generally I enjoyed this cooperative program and would continue my exploration on machine learning. Thanks for all help provided by the professor, the TA and my group members!

**Reference**

[1] Ekaterini Ioannidou, Sharmin Shabnam, Sophia Abner, Navjot Kaur, Francesco Zaccardi, Kausik K. Ray, Sam Seidu, Melanie J. Davies, Kamlesh Khunti, Clare L. Gillies, Effect of more versus less intensive blood pressure control on cardiovascular, renal and mortality outcomes in people with type 2 diabetes: A systematic review and meta-analysis, Diabetes & Metabolic Syndrome: Clinical Research & Reviews, Vol 17, 2023, pp 102782

[2] Maisa N.Feghali, Rita W. Driggers, Menachem Miodovnik, Jason G.Umans, 16-Diabetes in Pregnancy, Clinical pharmacology, 2013, Pages 257-273

[3] https://www.kaggle.com/datasets/akshaydattatraykhare/diabetes-dataset

[4] https://scikit-learn.org/stable/about.html#citing-scikit-learn

[5] https://blog.csdn.net/qq\_41185868/article/details/79325856

[6] https://scikit-learn.org/stable/modules/naive\_bayes.html#naive-bayes