**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **CHAPTERS** | **PAGE NO** |
| ABSTRACT | 4 |
| Chapter 1: Introduction |  |
| 1.1 Introduction | 5 |
| 1.2 Problem definition | 6 |
| 1.3 SCOPE | 7 |
| 1.4 PURPOSE | 8 |
| 1.5 PROBLEMAND EXISTINGTECHNOLOGY | 9 |
| 1.6 PROPOSEDSYSTEM | 10 |
| CHAPTER 2: REQIUREMENTS& ANALYSIS |  |
| 2.1 PLATFORMREQUIREMENTS | 11 |
| 2.2 MODULEDESCRIPTION | 12 |
| CHAPTER 3: DESIGN & IMPLEMENTATION |  |
| 3.1 ALGORITHMS | 13 |
| 3.2 PSEUDOCODE | 17 |
| CHAPTER4: SCREENSHOTS | 23 |
| CHAPTER5: CONCLUSION | 32 |
| CHAPTER6: REFERENCES | 33 |

**ABSTRACT**

Nowadays, Diabetes is one of the most common and severe diseases in India as well as all over the world. With type 2 diabetes, the body either does not produce enough insulin, or it resists insulin. It is not only harmful to the blood but also causes different kinds of diseases like blindness, renal disease, kidney problem, heart diseases etc. that causes a lot of death per year. So, it badly needs to develop a system that can effectively diagnose or predict the diabetes patients using electronic medical records (datasets). In this project we predict whether the patient is diagnosed with type2 diabetes or not using the deep learning algorithm ANN. The results show that our new proposed survival analysis approach consistently outperforms traditional survival models and demonstrate the effectiveness of the multi-task framework over modeling each complication independently.

**CHAPTER** **1.INTRODUCTION**

**1.1 INTRODUCTION**

Type 2 diabetes mellitus (T2DM) is a chronic disease that affects almost half a billion people around the globe (World Health Organization 2016). It is characterized by hyperglycemia— abnormally elevated blood glucose (blood sugar) levels, and is almost always associated with a number of complications (Forbes and Cooper 2013). Over time, the chronic elevation of blood glucose levels caused by T2DM leads to blood vessel damage which in turn leads to associated complications, including kidney failure, blindness, stroke, heart attack, and in severe cases even death. T2DM management requires continuous medical care with multifactorial risk-reduction strategies beyond glycemic control (American Diabetes Association and others 2013). Early prediction of T2DM complications is critical for healthcare professionals to appropriately adapt treatment plans for patients.

The recent abundance of the electronic health records (EHRs) has provided dunun precedented opportunity to apply predictive analytics to improve T2DM management. In this When will a patient develop complications after the initial T2DM diagnosis? Given the EHR records of two patients, which patient is more likely to develop complications? In the literature, EHRs have been applied to disease onset prediction, patient stratiﬁcation, readmission prediction and mortality prediction. However, there are unique characteristics and challenges to the problem of T2DM complications time-to-event prediction.

Although roughly 25% of those with type 2 diabetes are undiagnosed in the United States, population-wide screening for diabetes currently is not cost-effective because of the additional time and laboratory testing required1-3. Intervention studies have shown that diabetes can be prevented in high-risk individuals3-5, while weight loss and lifestyle changes can revert the recently diagnosed patients (<4 years) to pre-diabetes status6. This makes population-wide screening not just an issue of prevention, but also one of treatment.

**1.2 PROBLEM DEFINITION**

Current risk scores pose an economic and logistical challenge for population-wide screening and are less sensitive to detecting diabetes in non-white populations who are more likely to have diabetes16,17. More generally, automatically predicting chronic disease using electronic health records has numerous applications outside of clinical practice and would open the risk assessment doorway to those who ultimately bear the financial burden incurred by the disease: the insurers. A better estimate of risk on the part of insurers could encourage targeted patient education and incentive programs to reduce financial liability.

EHR models extend screening, conventionally framed between the doctor and the patient, to the payer and the patient. Data mining methods are powerful, but wild-type electronic records frequently are messy18; these tools should be validated against real-world data if realistic results are desired. We examine whether augmenting risk scores using EHR-derived phenotypes would increase sensitivity in the general population for detecting patients who should be screened further using laboratory testing, even when records are incomplete, and are not recorded systematically across health professionals and/or practice locations. When implemented on a population, this step-wise screening process would decrease the public health cost of more expensive testing, while simultaneously identifying previously overlooked at-risk patients.

Many current diabetes risk models are not generalizable to the disproportionate number of ethnic/racial minorities with Type 2 diabetes in the U.S. 12. For example, the FINDRISC score was developed on Finnish patients13. Although it was found to be the best risk assessment tool in Caucasian patients8, it was suboptimal for use on Arab and Filipino populations. Therefore, wide-scale screening methods that assume a “one-size-fits all” approach is simply not feasible among heterogeneous populations. Considering that African Americans, Hispanic/Latino Americans, American Indians, Asian Americans, and Pacific Islander Americans are at particularly high risk for type 2 diabetes1, it is important to understand the unique risk factors that continue to drive this growing health disparity among these groups, and to assess risk accurately in the subpopulations most likely to have the disease.

**1.3 SCOPE**

This work leverages the large amounts of medical data routinely collected and stored in electronic form by health providers in most developed countries. This is a rich data source which contains a variety of information about each patient including the patient’s age and sex, mother tongue, religion, marital status, profession, etc. In the context of the present work, of main interest is the information collected each time a patient is admitted to the hospital (including out-patient visits to general practitioners or specialists). The format of this data is explained next. Each time a patient is admitted to the hospital the reason for the admission, as determined by the medical practitioner in primary charge during the admission, is recorded in the patient’s medical history.

This is performed using a standardized coding schema such as that provided by the International Statistical Classi- 6 Vasiljeva & Arandjelovic´ fication of Diseases and Related Health Problems (ICD-10) (World Health Organization, 2004) and the related Australian Refined Diagnosis-Related Groups (AR-DRGs). These have hierarchical structures (Arandjelovic, 2016). ICD-10, for exam- ´ ple, contains 22 chapters, each chapter encompassing a spectrum of related health issues (usually symptomatically rather than etiologically related). For example, ICD-10 Chapter 4 which includes codes E00-E90, covers “Endocrine, nutritional and metabolic diseases”. At each subsequent depth level of the tree the grouping is refined and the scope of conditions narrowed down. In this paper we use the classification attained at the depth of two of ICD-10, which achieves a good compromise between specificity and frequency of occurrence. This results in each diagnosis being given a three character code which comprises a leading capital letter (A-Z, first grouping level), followed by a two digit number (further refinement). For example, E66 codes for “Obesity” within the broader range of “Endocrine, nutritional and metabolic diseases”

**1.4 PURPOSE**

The primary purpose of the Markovian assumption is to constrain the mechanism underlying a specific process and thus formulate it in a manner which leads to a tractable learning problem. Although it is seldom strictly true, that it is often a reasonable approximation to make is witnessed by its successful application across a diverse range of disciplines; examples of modelled phenomena include meteorological events (Gabriel and Neumann, 1962), software usage patterns (Whittaker and Thomason, 1994), breast cancer screening (Duffy and Yau, 1995), human motion and behaviour (Lee et al., 2005; Arandjelovic,´ 2011), and many others. Nonetheless, the key premise motivating the model in this paper is that the Markovian assumption is in fact not appropriate for the high-level modelling of disease progression (note that this does not reject its possible applicability in disease progression modelling on different levels of abstraction). Indeed, we will demonstrate this empirically. The aforementioned premise is readily substantiated using a theoretical argument as well. Consider a patient who is admitted for what is diagnosed as a serious chronic illness. If the 10 Vasiljeva & Arandjelovic´ same patient is subsequently admitted for an unrelated ailment, possibly a trivial one, the knowledge of the serious underlying problem is lost and the power to predict the next related diagnosis lost. The model proposed in the section which follows solves this problem, while at the same retaining the tractability of Markov process based approaches.

**1.5 PROBLEM AND EXISTING TECHNOLOGY**

This dataset was rich in the breadth of information it contained: lab results; medication dosage, and history; basic patient demographic information (age, gender, state); smoking status; transcripts (BMI, systolic/diastolic blood pressure (BP), height, weight); allergies; and diagnosed conditions for each visit as ICD-9 codes. We identified and condensed redundant features manually (e.g. Warfarin and Coumadin). When BMI values were over 70, or below 10, we re-coded height and weight as not available (“-NA-”). Moreover, the “Healthy Controls” in this sample, on average, had more prescribed medications and higher smoking rates than patients with Type 1 or Type 2 Diabetes. This natural sample provided a more realistic control group to identify which factors in the EHR were predictive of a Type 2 diabetes diagnosis.

Missing data were common in this sample: less than 1% of the patients had a recorded family history of diabetes (ICD9 V18.0), despite a prevalence of 11.8% in the US population. This posed a “worst case" scenario for prediction: given missing, unsystematic and incomplete information from a patient’s medical history, could residual information still augment current diabetes risk scores in a way that improves the accuracy and efficiency of type 2 diabetes screening in the general population?

We assessed whether Type 2 diabetes risk scores could be improved with EHR phenotypes, created using the additional medical and diagnostic information contained in the EHR. Because the visit dates were removed to protect patient privacy, longitudinal data were not available. It is therefore unknown whether patients developed diabetes during their time of service, or whether it preceded their entry into this study. It is similarly unknown whether patients identified as “healthy” had undiagnosed diabetes. Similarly, the ordering of medications, nondiabetes diagnoses, and the diabetes diagnosis are similarly unknown. Using real-world clinical data, these models then assess the current likelihood of a patient having a current diagnosis of Type 2 Diabetes, rather than the future likelihood of developing

diabetes. The value of including EHR information was computed by comparing models using a chi-square test.

**1.6 PROPOSED SYSTEM**

In this paper our aim is to predict the probability of a specific diagnosis a following the patient history H: p(H → a|H). The difficulty of formulating this as a tractable learning problem lies in the fact that the space of possible histories is infinite as H can be of an arbitrary length. Even if the length l(H) is limited, the number of possible histories is extremely large: [l(H)]na where na is the number of different diagnosis codes. Therefore it is necessary to make an approximation which constrains and simplifies the task. We already argued why the Markovian assumption on the level of diagnosis codes is inappropriate. In its stead we propose a different representation of a patient’s state, particularly suitable for the modelling of disease progression.

Consider a particular diagnosis history ´ H = d1 → . . . → dn.

The proposed method makes use of the well known observation that when it comes to chronic diseases, the very presence of past complications strongly predicts future complications. Towards Sophisticated Learning from EHR 11 Thus, a history H is represented using a history vector v = v(H) which is a fixed length vector with binary values .Each vector element corresponds to a specific diagnosis code (except for one special element explained shortly) and its value is 1 if and only if the corresponding diagnosis is present in the history: ∀d ∈ D.

otherwise where D is the set of diagnosis codes, i(d) indexes the diagnosis code d in a history vector, and H1,2 may take on degenerate forms of empty histories. By collapsing an arbitrary length history of diagnoses onto a fixed length vector, the space of possible states over which learning is performed is dramatically reduced and the problem immediately made far more tractable. Notice the importance of the observation that it is the presence of past complications which most strongly predicts future ailments, given that under this representation any information on the ordering of diagnoses is discarded. The binary nature of the representation also has the effect of reducing the size of the space over which inference is performed.

**CHAPTER 2: REQIUREMENTS& ANALYSIS**

**2.1 PLATFORM REQUIREMENTS**

* **SOFTWARE REQUIREMENTS:**

The major software requirements of the project are as follows:

Language : Python

Operating system : Windows

We have done this project in i5 processor and operating system used is windows 10

* **HARDWARE REQUIREMENTS:**

The hardware requirements that map towards the software are as follows:

RAM : 4 GB

Processor : Intel i3 processor

**2.2:MODULEDESCRIPTION**

**Unsupervised Learning**

An unsupervised clustering algorithm was applied to provide insight into the distribution of positive DMT2 cases in the feature space. The gap statistic, as deﬁned in Hastie, et al. , was computed for the data using k-means clustering with k = 2 to k = 7. This was implemented using the fpc package for R . Local maxima in the gap statistic were interpreted as the optimal numbers of clusters.

**Supervised Learning**

Methodology The EMR dataset was used to train several supervised learning algorithms implemented in R . In order to evaluate algorithm eﬀectiveness, 10-fold cross-validation was applied to compute the test error, precision, and recall for each of the models. The train error was also computed to provide insight into the variance and bias of the models. Learning curves were obtained by varying the size of the training set and computing the test error, precision, and recall.

**Logistic Regression**

Following on the results of the naive Bayes model, logistic regression was investigated since it make weaker assumptions concerning the conditional probability distribution of features. Speciﬁcally, as a generalized linear model based upon the conditional mean p(y|x) subject to the Bernoulli distribution, it does not require features to be conditionally independent nor multivariate normal. Further, the data provides a suﬃcient number of samples (almost 10,000) for the eﬀective use of logistic regression using the glm package for R .

**Decision Trees**

Finally, in order to better communicate the hierarchy of indicators of DMT2, this paper explores a single white box approach using decision trees. By applying this non-parametric greedy algorithm whose objective it is to maximize information gain in a top-down search of features, this paper is able to provide visualization of some of the decision boundaries with respect to individual features.

**CHAPTER 3: DESIGN & IMPLEMENTATION**

**3.1 ALGORITHM**

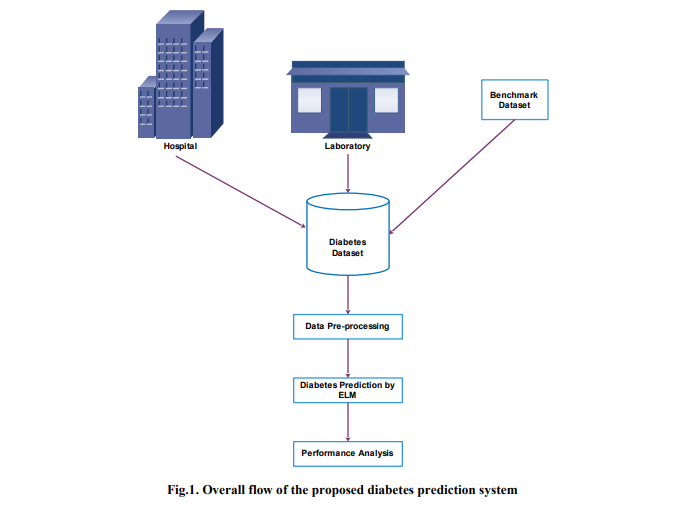
**Method Evaluation**

We see that SVMs and logistic regression are the methods that yield the smallest generalization error, approximately 18%. The SVM is particularly interesting because changing the cost function parameter for `1 regularization causes a tradeoﬀ between precision and recall. Speciﬁcally, increasing the cost function parameter decreases the precision and increases the recall. This suggests that the cost function parameter can be adjusted to tune the precision and recall to match the needs of doctors.

**Bias and Variance**

One can estimate the relative contributions of bias (model limitations) and variance (overﬁtting) to the error of a model by comparing the test error and train error. From Table 3, we see that the test error and train error are very close for naive Bayes and logistic regression, suggesting that the bulk of the error is due to bias. This is corroborated by their ﬂat learning curves, which indicates that there is some inherent error in the models even when the training set is large. On the other hand, the train errors for knearest neighbors and decision trees are fairly small compared to the test error, implying that the error is largely due to variance. Finally, the SVM has a train error that is moderately smaller than the test error, indicating that both bias and variance contribute to error in the SVM.

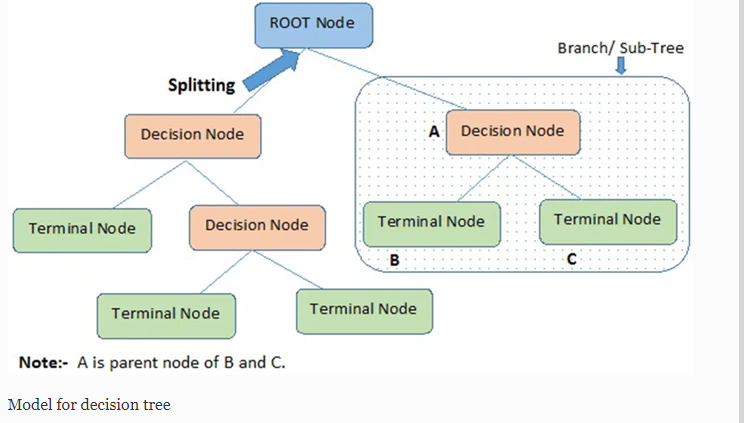
**FLOW CHART**



Input: training data set

Output: decision model (tree structure).

The decision model is a tree structure, where a structure includes the collection of nodes. It includes the decision nodes (split node with the condition) and leaf nodes. The representation of the decision tree is shown in Fig. [3](https://link.springer.com/article/10.1186/s40537-019-0175-6#Fig3). Among the various attributes in the dataset, choosing the right attributes-root node to start the split is a difficult task. The decision node can have 2 or more branches. To start with the first node called root node. The model predicts the best attribute as the root node or best predictor node from the set of nodes available. There are many ways to choose the best attribute to be as the root node, based on the degree of impurity of the child nodes. The Performance measures [[39](https://link.springer.com/article/10.1186/s40537-019-0175-6#ref-CR39)] are Entropy, Giniindex, classification error. These measures are done for all attributes and comparison is done, to select the best spilt.

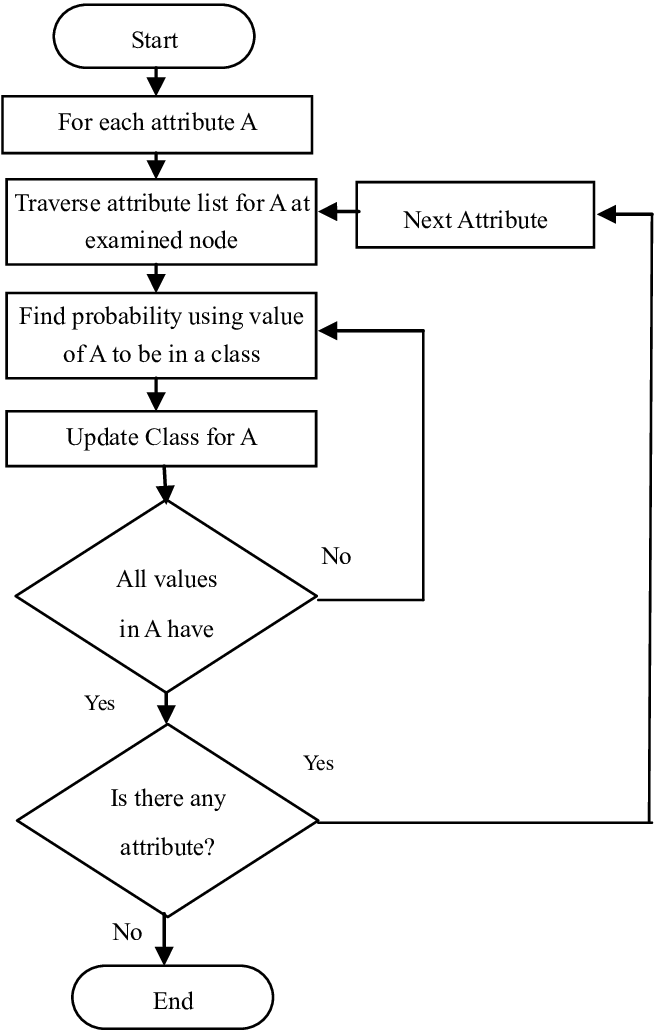


### Naive Bayesian

A classification algorithm , a probabilistic classifier which is based on Bayes theorem with the independence assumption between the predictors. Naïve Bayesian method takes the dataset as input, performs analysis and predicts the class label using Bayes’ Theorem. It calculates a probability of class in input data and helps to predict the class of the unknown data sample.It is a powerful classification technique suitable for large datasets. The Bayes Theorem formula calculates the posterior probability for each class using below formula. The Flowchart for Naïve Bayesian is shown in Fig. [4](https://link.springer.com/article/10.1186/s40537-019-0175-6#Fig4).

P(c|x)=P(x|c)P(c)P(x)P(c|x)=P(x|c)P(c)P(x)

P(c|X)=P(x1|c)×P(x2|c)×⋯×P(xn|c)×P(c)



* 1. **PSEUDOCODE:**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

df=pd.read\_csv('diabetes.csv')

df.head()

#lets describe the data

df.describe()

#infromation of dataset

df.info()

#any null values

#not neccessary in above information we can see

df.isnull().values.any()

#histogram

df.hist(bins=10,figsize=(10,10))

plt.show()

#correlation

sns.heatmap(df.corr())

# we can see skin thickness,insulin,pregnencies and age are full independent to each other

#age and pregencies has negative correlation

#lets count total outcome in each target 0 1

#0 means no diabeted

#1 means patient with diabtes

sns.countplot(y=df['Outcome'],palette='Set1')

sns.set(style="ticks")

sns.pairplot(df, hue="Outcome")

#box plot for outlier visualization

sns.set(style="whitegrid")

df.boxplot(figsize=(15,6))

#box plot

sns.set(style="whitegrid")

sns.set(rc={'figure.figsize':(4,2)})

sns.boxplot(x=df['Insulin'])

plt.show()

sns.boxplot(x=df['BloodPressure'])

plt.show()

sns.boxplot(x=df['DiabetesPedigreeFunction'])

plt.show()

#outlier remove

Q1=df.quantile(0.25)

Q3=df.quantile(0.75)

IQR=Q3-Q1

print("---Q1--- \n",Q1)

print("\n---Q3--- \n",Q3)

print("\n---IQR---\n",IQR)

#print((df < (Q1 - 1.5 \* IQR))|(df > (Q3 + 1.5 \* IQR)))

#outlier remove

df\_out = df[~((df < (Q1 - 1.5 \* IQR)) |(df > (Q3 + 1.5 \* IQR))).any(axis=1)]

df.shape,df\_out.shape

#more than 80 records deleted

#Scatter matrix after removing outlier

sns.set(style="ticks")

sns.pairplot(df\_out, hue="Outcome")

plt.show()

#lets extract features and targets

X=df\_out.drop(columns=['Outcome'])

y=df\_out['Outcome']

#Splitting train test data 80 20 ratio

from sklearn.model\_selection import train\_test\_split

train\_X,test\_X,train\_y,test\_y=train\_test\_split(X,y,test\_size=0.2)

train\_X.shape,test\_X.shape,train\_y.shape,test\_y.shape

from sklearn.metrics import confusion\_matrix,accuracy\_score,make\_scorer

from sklearn.model\_selection import cross\_validate

def tn(y\_true, y\_pred): return confusion\_matrix(y\_true, y\_pred)[0, 0]

def fp(y\_true, y\_pred): return confusion\_matrix(y\_true, y\_pred)[0, 1]

def fn(y\_true, y\_pred): return confusion\_matrix(y\_true, y\_pred)[1, 0]

def tp(y\_true, y\_pred): return confusion\_matrix(y\_true, y\_pred)[1, 1]

#cross validation purpose

scoring = {'accuracy': make\_scorer(accuracy\_score),'prec': 'precision'}

scoring = {'tp': make\_scorer(tp), 'tn': make\_scorer(tn),

'fp': make\_scorer(fp), 'fn': make\_scorer(fn)}

def display\_result(result):

print("TP: ",result['test\_tp'])

print("TN: ",result['test\_tn'])

print("FN: ",result['test\_fn'])

print("FP: ",result['test\_fp'])

#Lets build the model

#Logistic Regression

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import roc\_auc\_score

acc=[]

roc=[]

clf=LogisticRegression()

clf.fit(train\_X,train\_y)

y\_pred=clf.predict(test\_X)

#find accuracy

ac=accuracy\_score(test\_y,y\_pred)

acc.append(ac)

#find the ROC\_AOC curve

rc=roc\_auc\_score(test\_y,y\_pred)

roc.append(rc)

print("\nAccuracy {0} ROC {1}".format(ac,rc))

#cross val score

result=cross\_validate(clf,train\_X,train\_y,scoring=scoring,cv=10)

display\_result(result)

#display predicted values uncomment below line

#pd.DataFrame(data={'Actual':test\_y,'Predicted':y\_pred}).head()

#Support Vector Machine

from sklearn.svm import SVC

clf=SVC(kernel='linear')

clf.fit(train\_X,train\_y)

y\_pred=clf.predict(test\_X)

#find accuracy

ac=accuracy\_score(test\_y,y\_pred)

acc.append(ac)

#find the ROC\_AOC curve

rc=roc\_auc\_score(test\_y,y\_pred)

roc.append(rc)

print("\nAccuracy {0} ROC {1}".format(ac,rc))

#cross val score

result=cross\_validate(clf,train\_X,train\_y,scoring=scoring,cv=10)

display\_result(result)

#display predicted values uncomment below line

#pd.DataFrame(data={'Actual':test\_y,'Predicted':y\_pred}).head()

#KNN

from sklearn.neighbors import KNeighborsClassifier

clf=KNeighborsClassifier(n\_neighbors=3)

clf.fit(train\_X,train\_y)

y\_pred=clf.predict(test\_X)

#find accuracy

ac=accuracy\_score(test\_y,y\_pred)

acc.append(ac)

#find the ROC\_AOC curve

rc=roc\_auc\_score(test\_y,y\_pred)

roc.append(rc)

print("\nAccuracy {0} ROC {1}".format(ac,rc))

#cross val score

result=cross\_validate(clf,train\_X,train\_y,scoring=scoring,cv=10)

display\_result(result)

#display predicted values uncomment below line

#pd.DataFrame(data={'Actual':test\_y,'Predicted':y\_pred}).head()

#Random forest

from sklearn.ensemble import RandomForestClassifier

clf=RandomForestClassifier()

clf.fit(train\_X,train\_y)

y\_pred=clf.predict(test\_X)

#find accuracy

ac=accuracy\_score(test\_y,y\_pred)

acc.append(ac)

#find the ROC\_AOC curve

rc=roc\_auc\_score(test\_y,y\_pred)

roc.append(rc)

print("\nAccuracy {0} ROC {1}".format(ac,rc))

#cross val score

result=cross\_validate(clf,train\_X,train\_y,scoring=scoring,cv=10)

display\_result(result)

#display predicted values uncomment below line

#pd.DataFrame(data={'Actual':test\_y,'Predicted':y\_pred}).head()

#Naive Bayes Theorem

#import library

from sklearn.naive\_bayes import GaussianNB

clf=GaussianNB()

clf.fit(train\_X,train\_y)

y\_pred=clf.predict(test\_X)

#find accuracy

ac=accuracy\_score(test\_y,y\_pred)

acc.append(ac)

#find the ROC\_AOC curve

rc=roc\_auc\_score(test\_y,y\_pred)

roc.append(rc)

print("\nAccuracy {0} ROC {1}".format(ac,rc))

#cross val score

result=cross\_validate(clf,train\_X,train\_y,scoring=scoring,cv=10)

display\_result(result)

#display predicted values uncomment below line

#pd.DataFrame(data={'Actual':test\_y,'Predicted':y\_pred}).head()

#Gradient Boosting Classifier

from sklearn.ensemble import GradientBoostingClassifier

clf=GradientBoostingClassifier(n\_estimators=50,learning\_rate=0.2)

clf.fit(train\_X,train\_y)

y\_pred=clf.predict(test\_X)

#find accuracy

ac=accuracy\_score(test\_y,y\_pred)

acc.append(ac)

#find the ROC\_AOC curve

rc=roc\_auc\_score(test\_y,y\_pred)

roc.append(rc)

print("\nAccuracy {0} ROC {1}".format(ac,rc))

#cross val score

result=cross\_validate(clf,train\_X,train\_y,scoring=scoring,cv=10)

display\_result(result)

#display predicted values uncomment below line

#pd.DataFrame(data={'Actual':test\_y,'Predicted':y\_pred}).head()

#lets plot the bar graph

ax=plt.figure(figsize=(9,4))

plt.bar(['Logistic Regression','SVM','KNN','Random Forest','Naivye Bayes','Gradient Boosting'],acc,label='Accuracy')

plt.ylabel('Accuracy Score')

plt.xlabel('Algortihms')

plt.show()

ax=plt.figure(figsize=(9,4))

plt.bar(['Logistic Regression','SVM','KNN','Random Forest','Naivye Bayes','Gradient Boosting'],roc,label='ROC AUC')

plt.ylabel('ROC AUC')

plt.xlabel('Algortihms')

plt.show()

#Great....

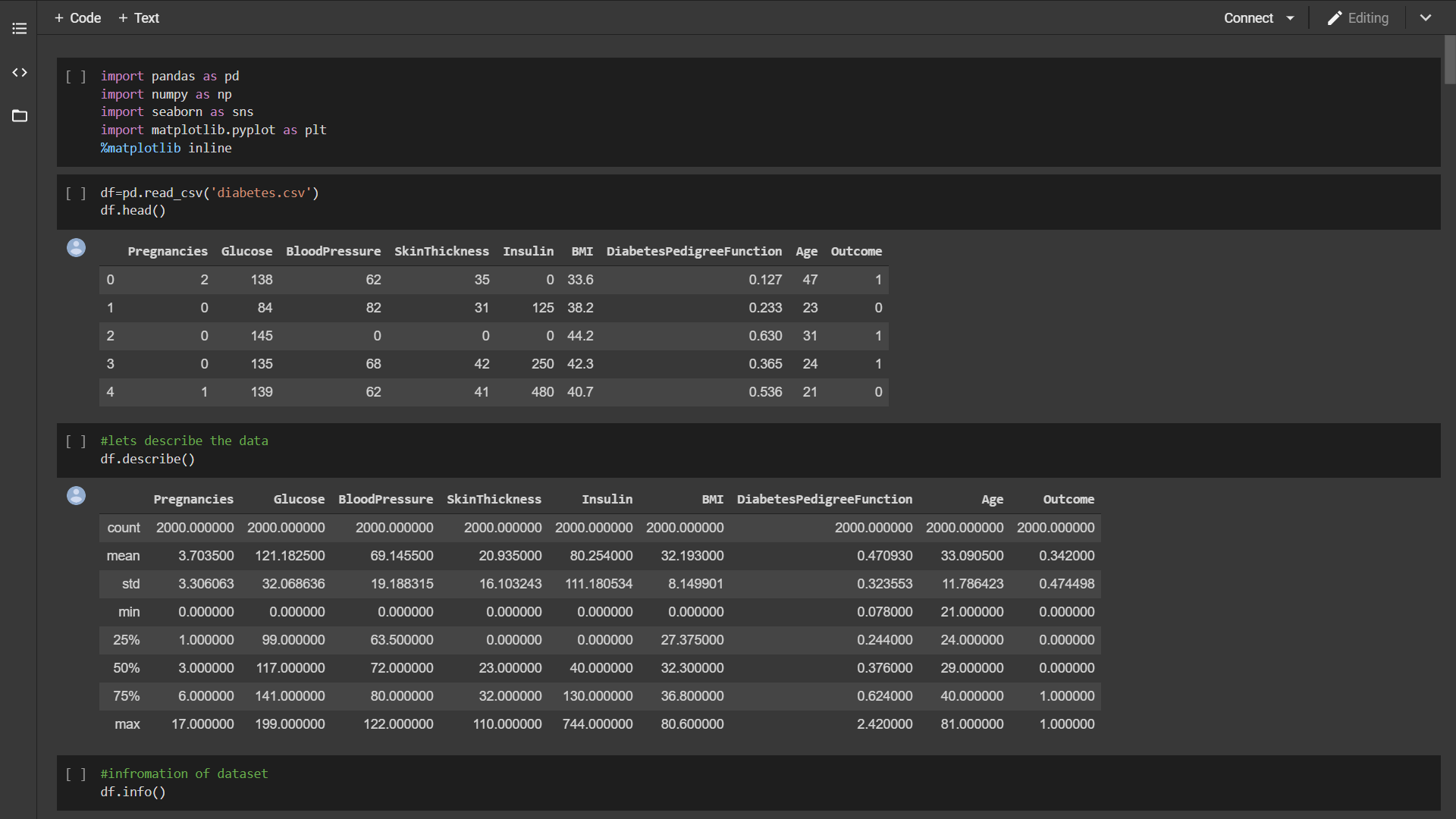
#Random forest has highest accuracy 98% and ROC\_AUC curve 97%

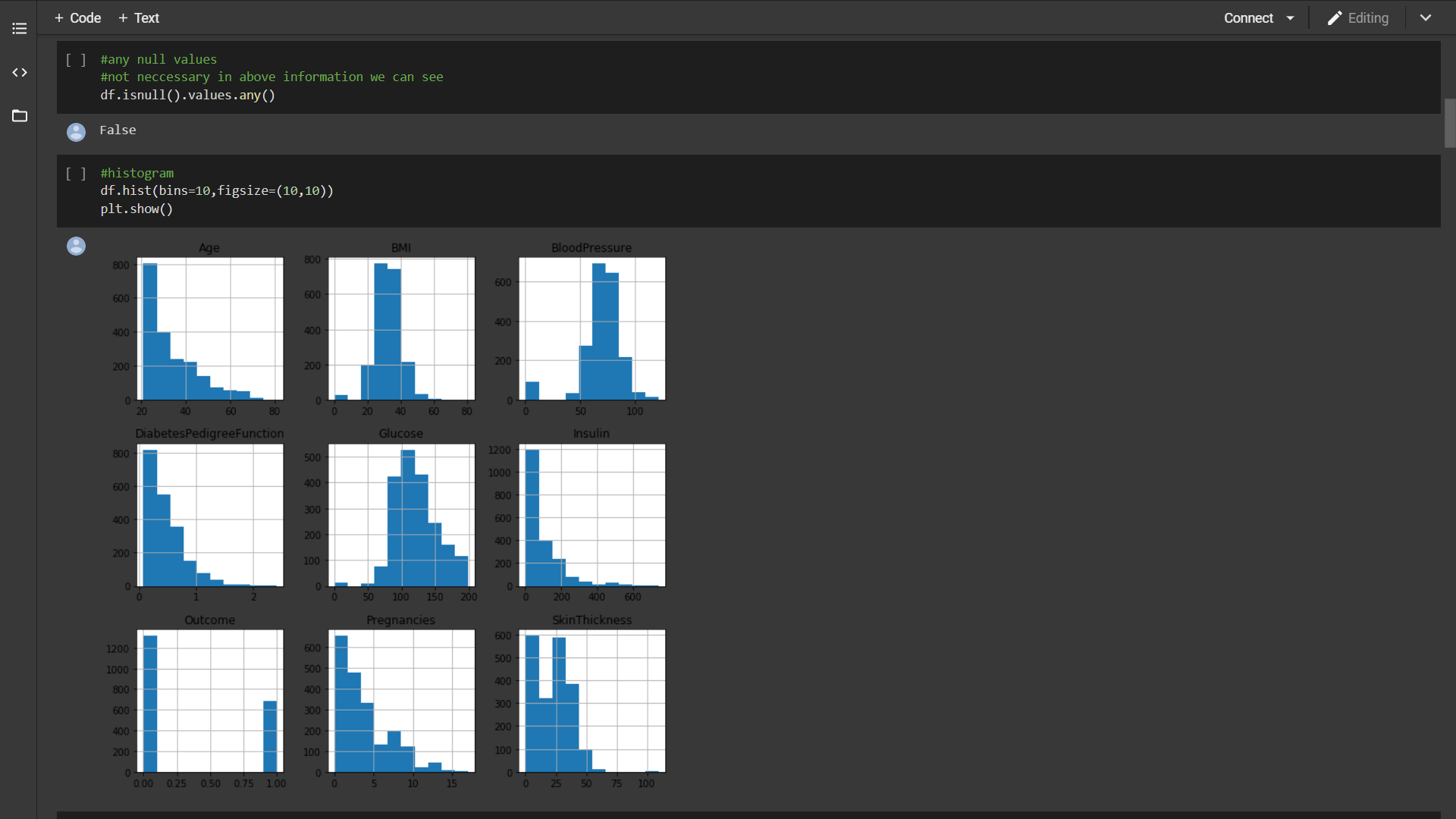
#model can be improve more if we take same count of labels

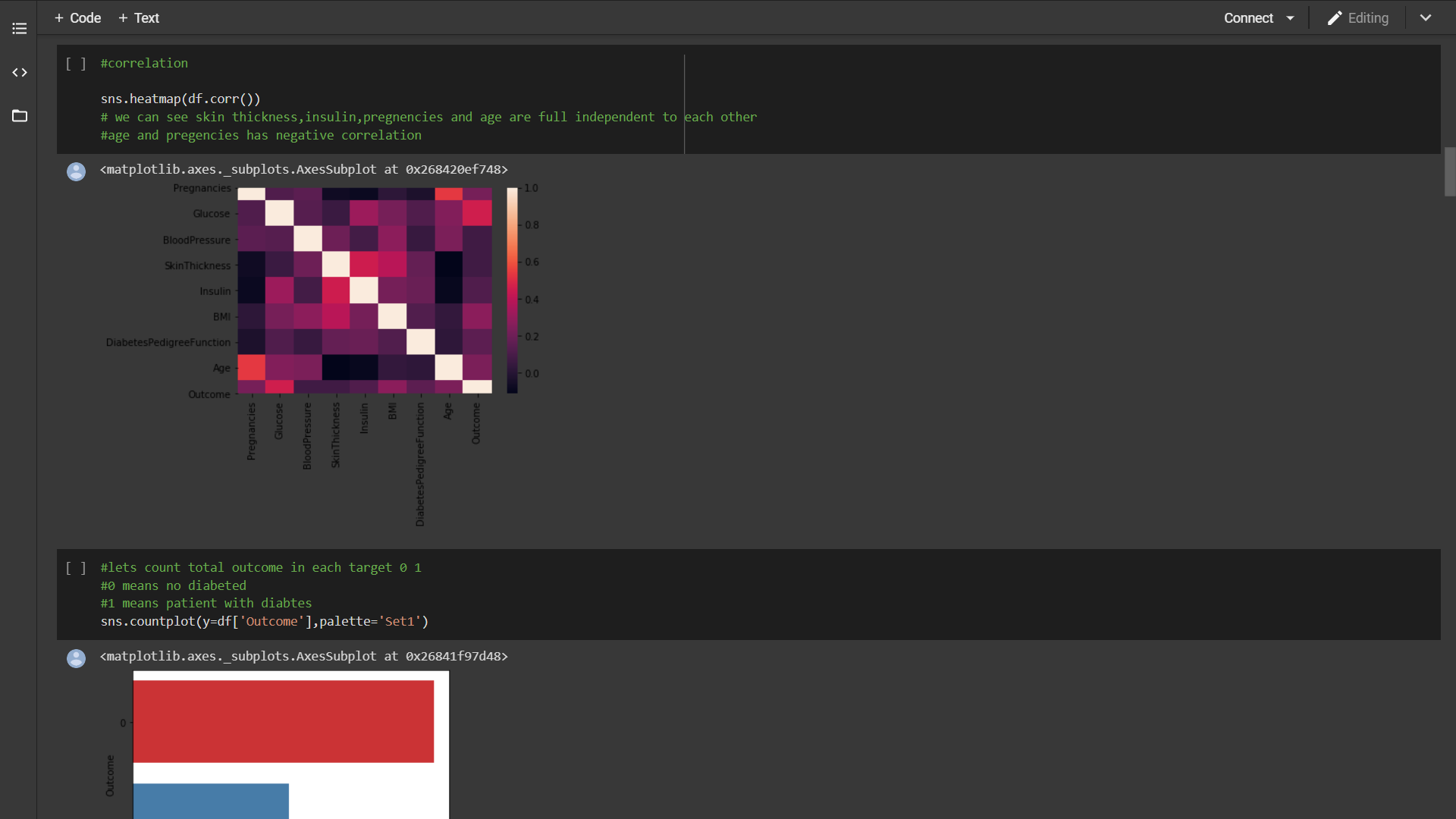
#in our model 30% is diabetic and 70% no diabetic patient

#model can be improve with fine tunning

**CHAPTER4: SCREENSHOTS**

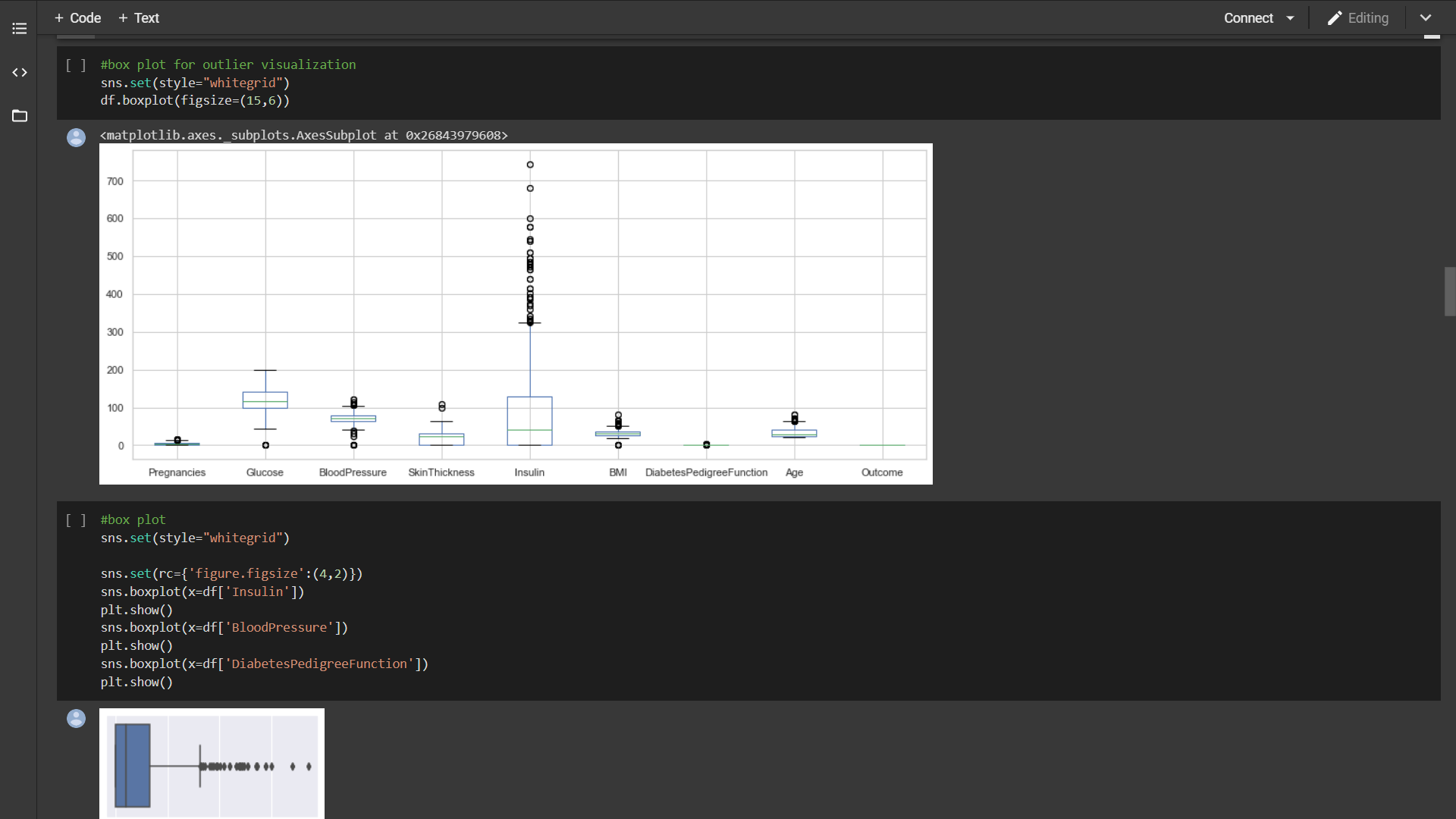


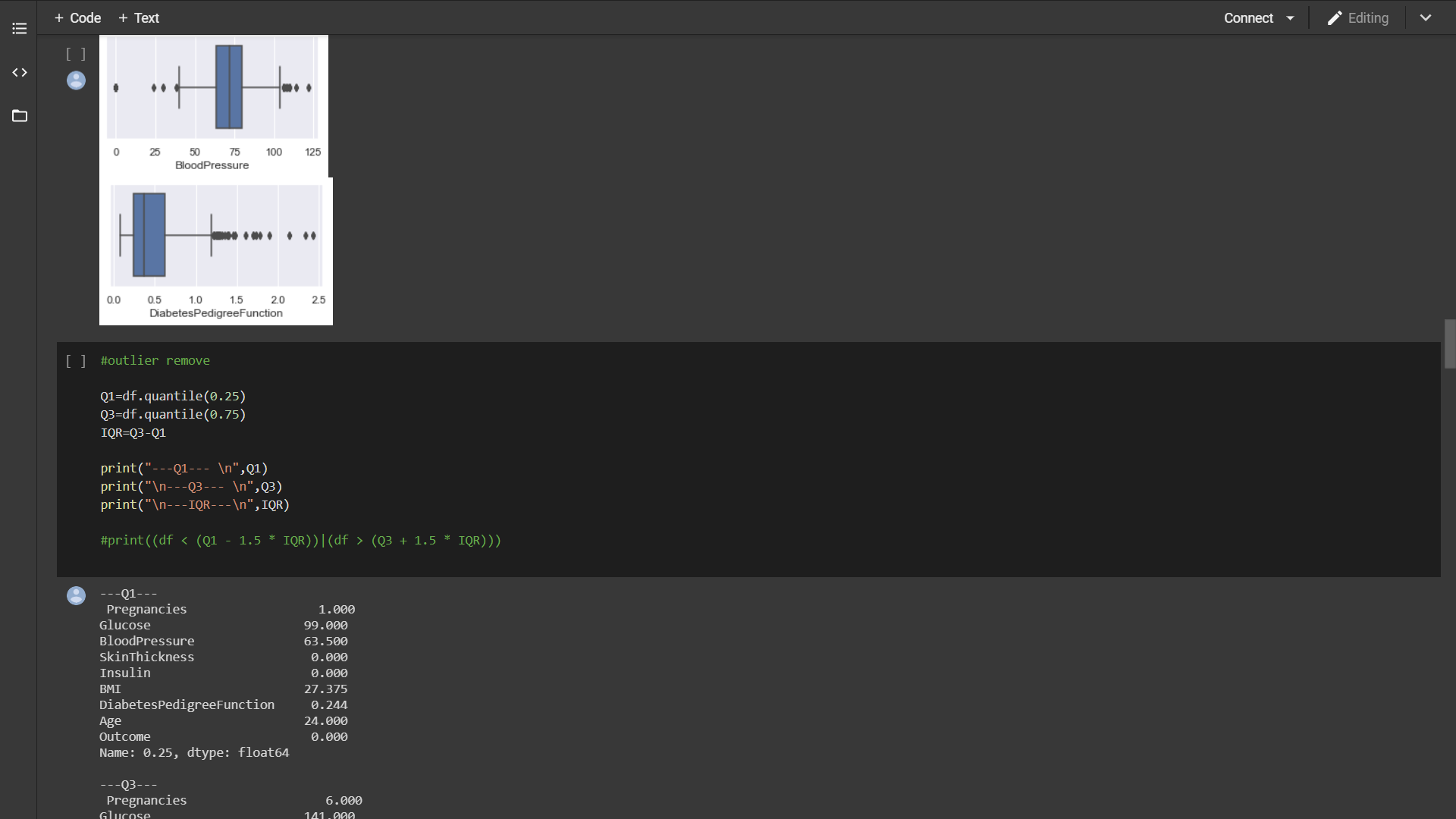


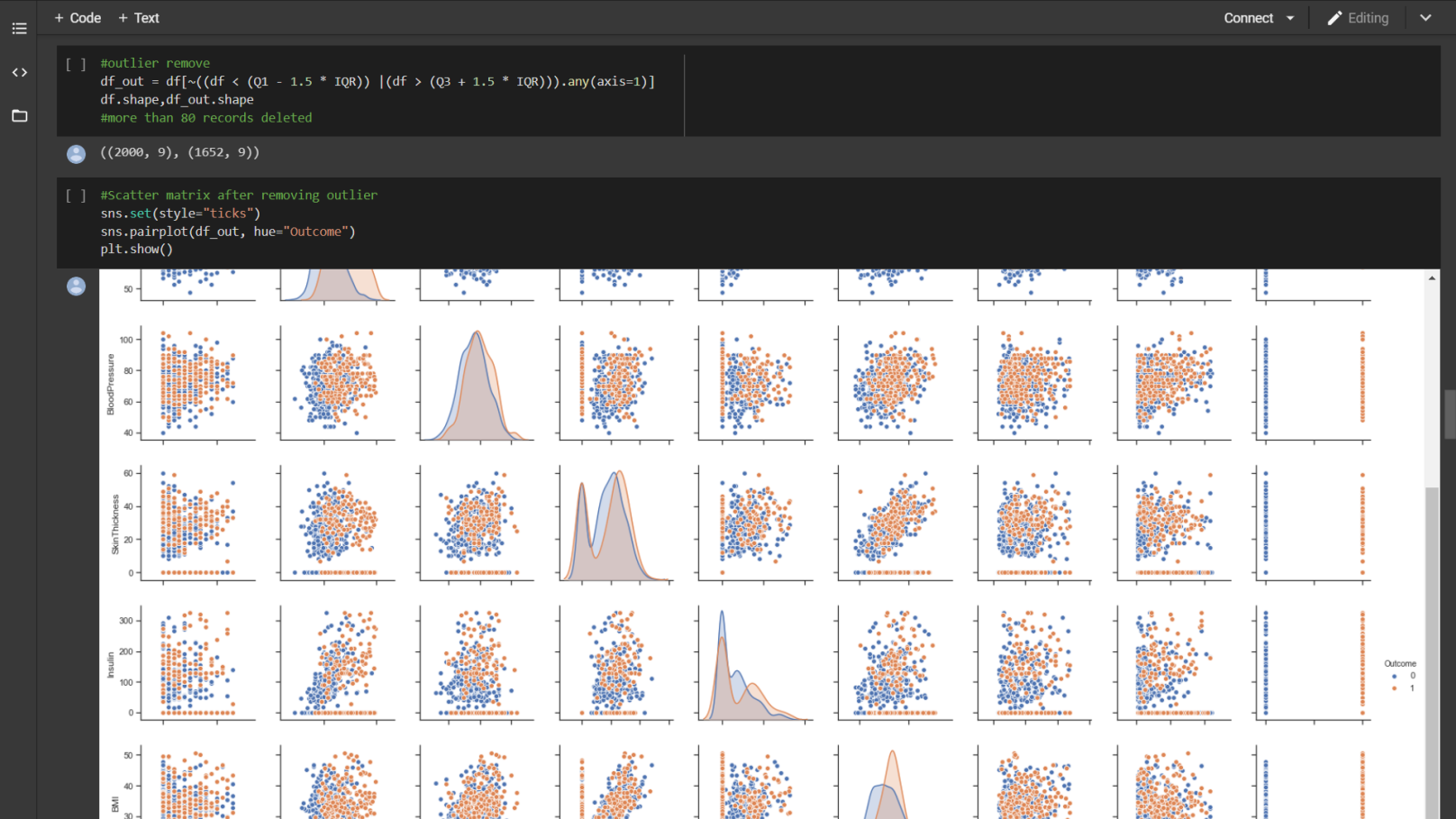




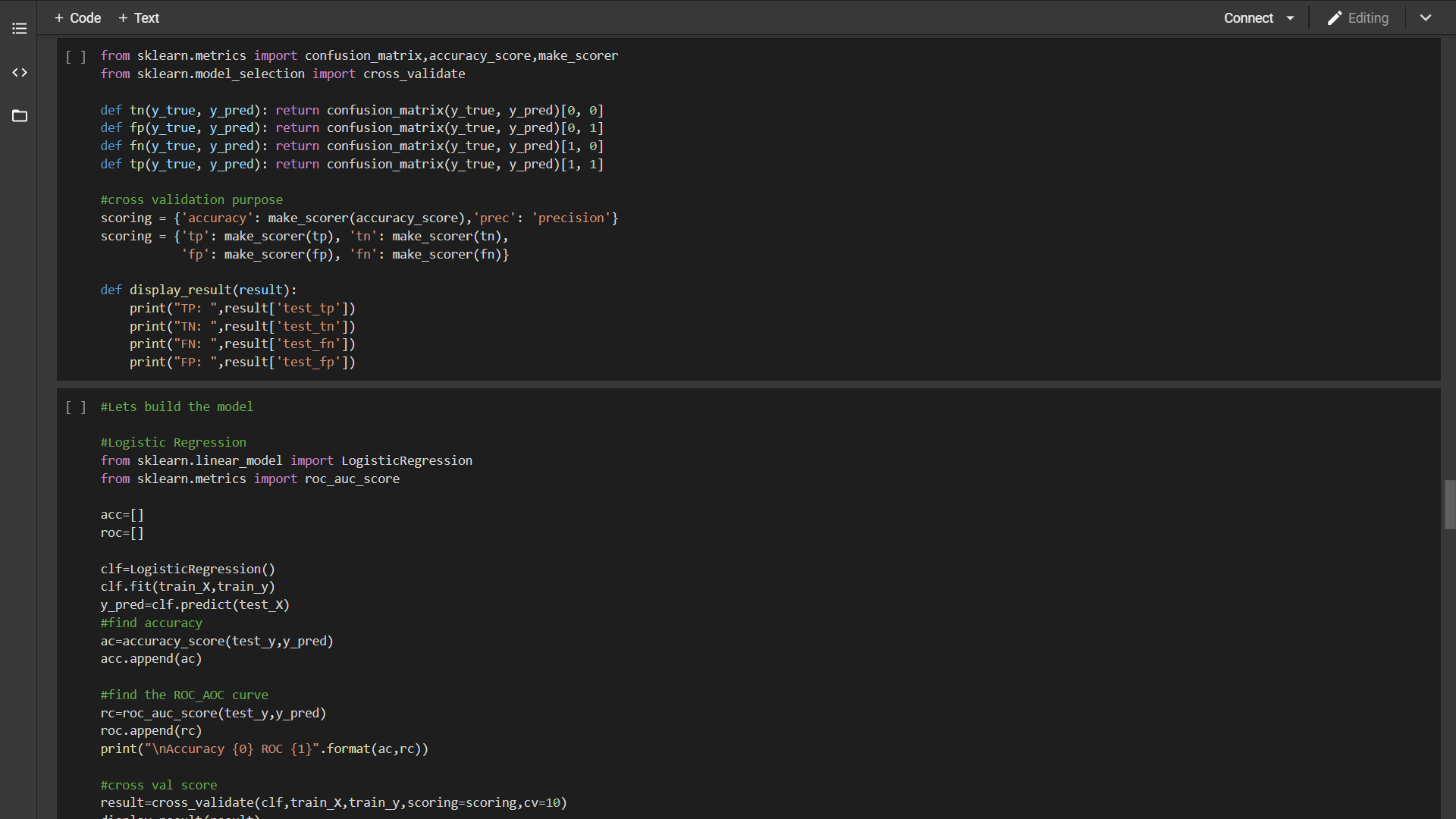


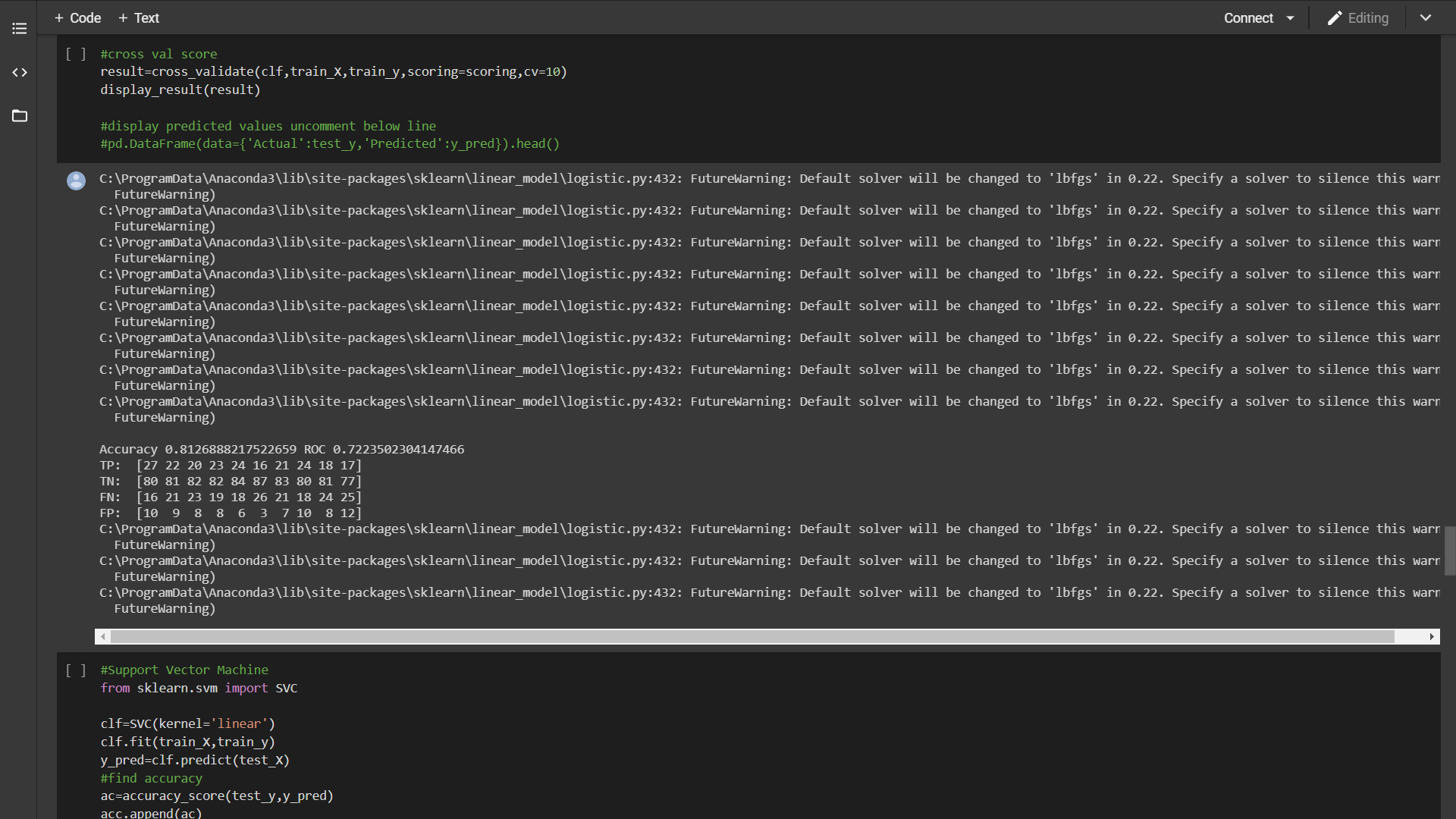


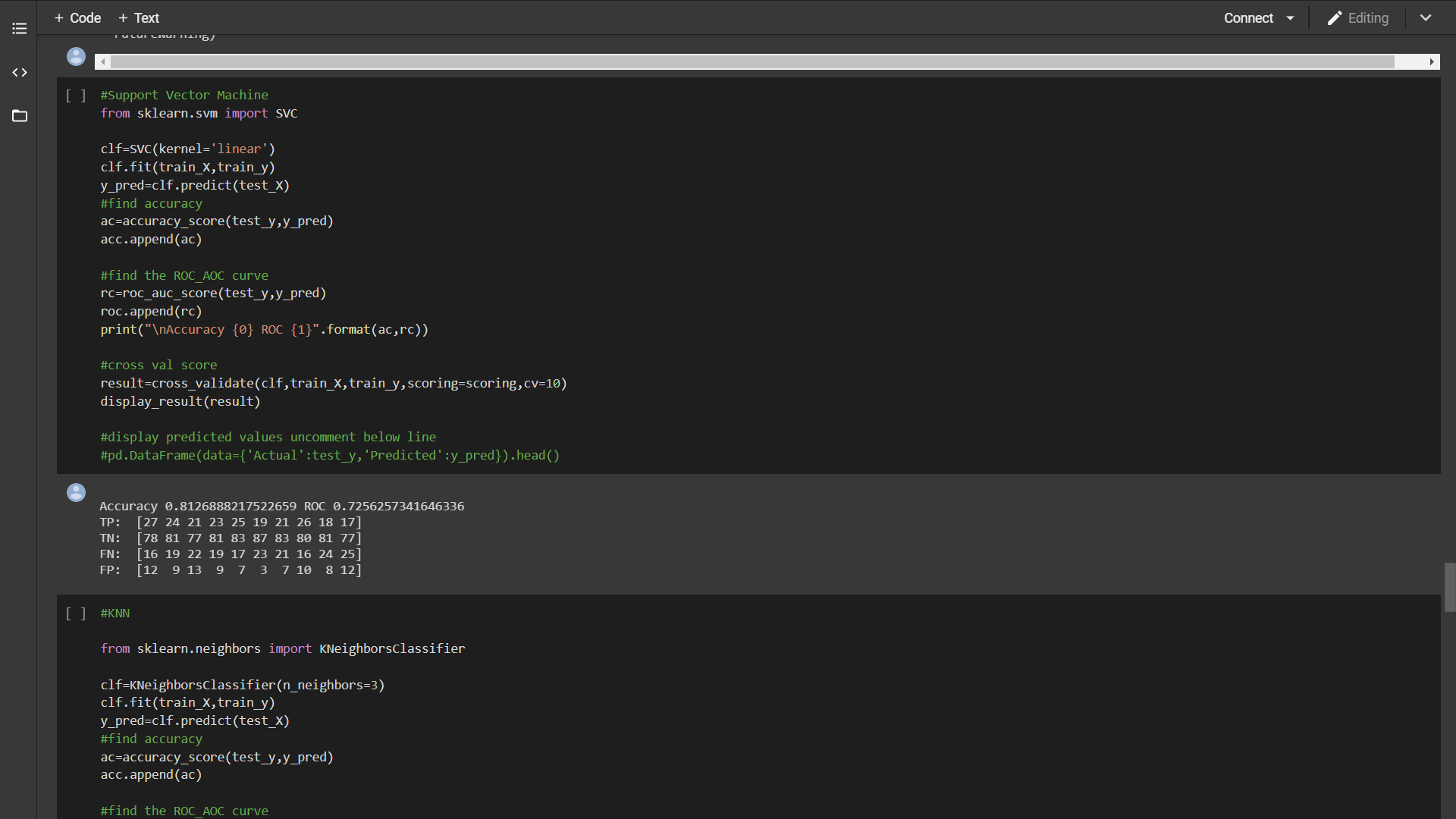


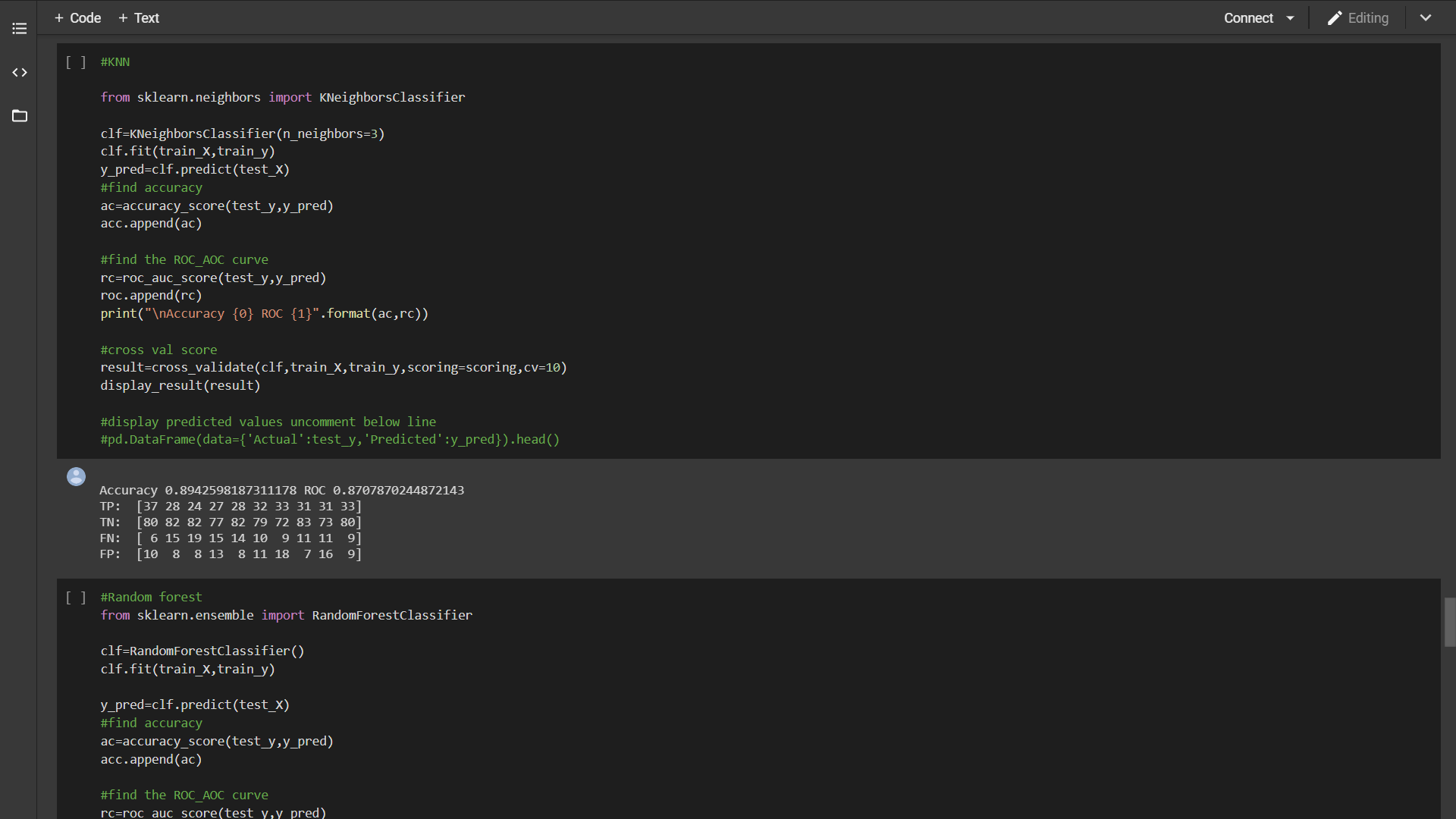


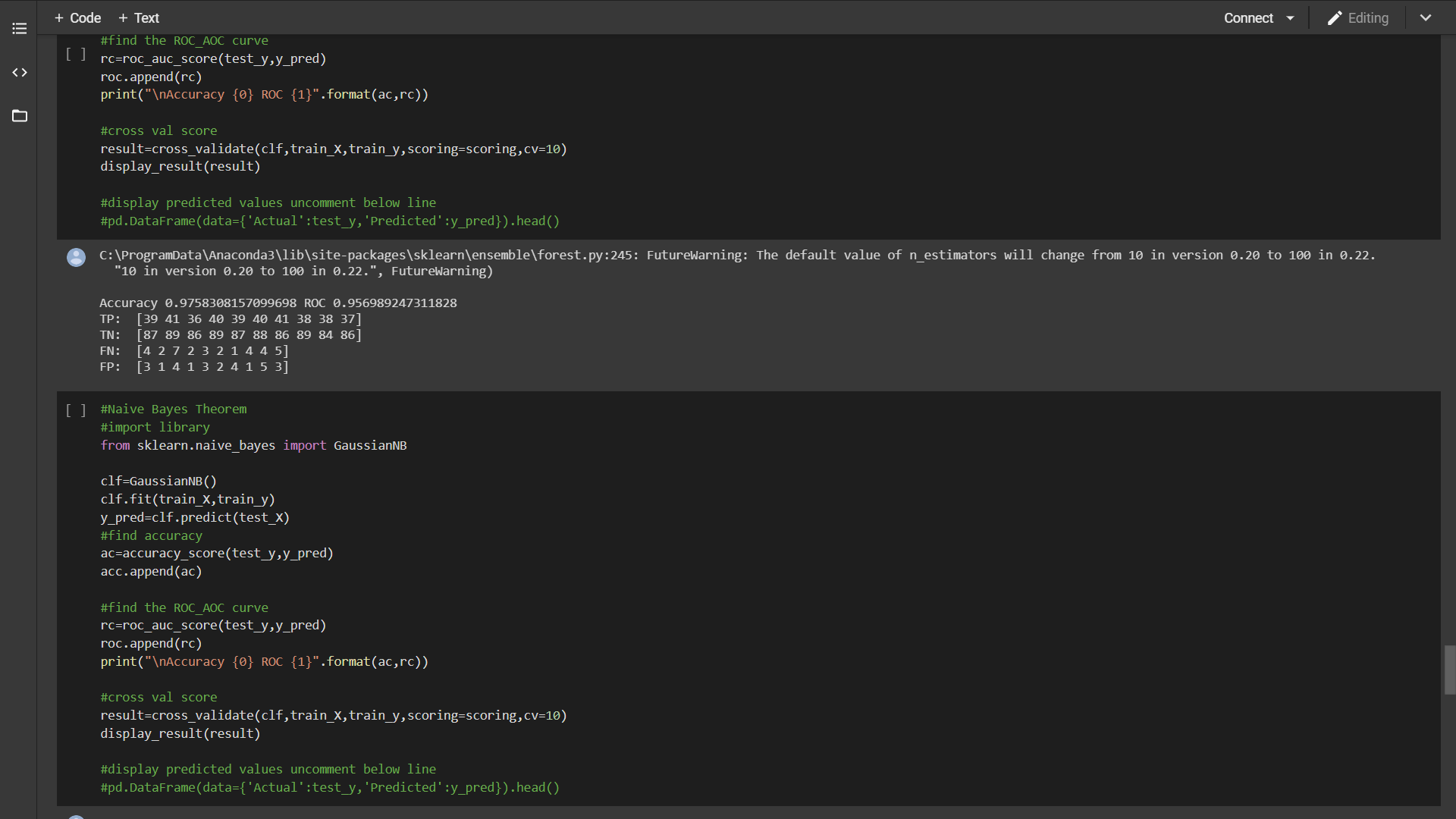


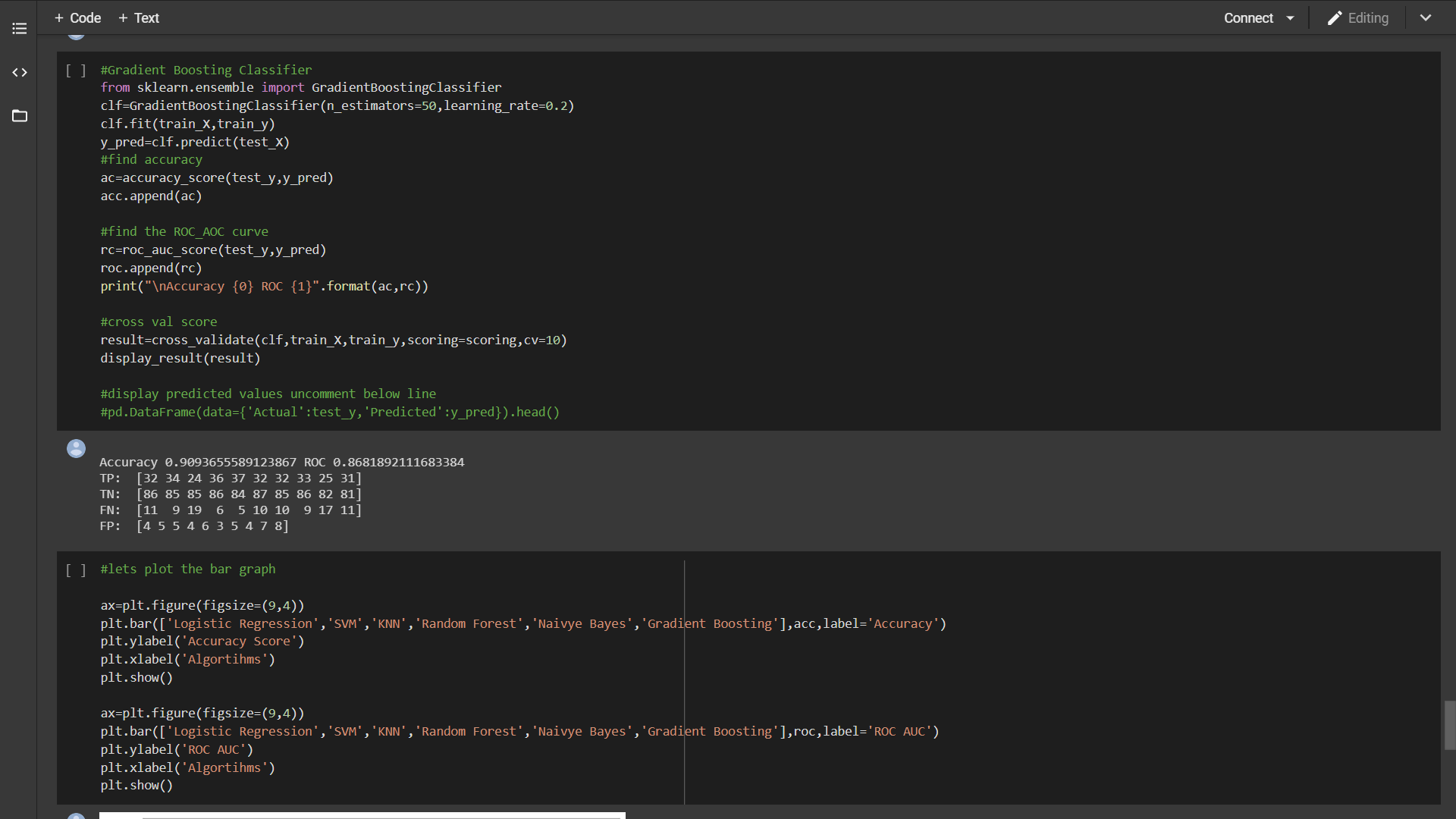


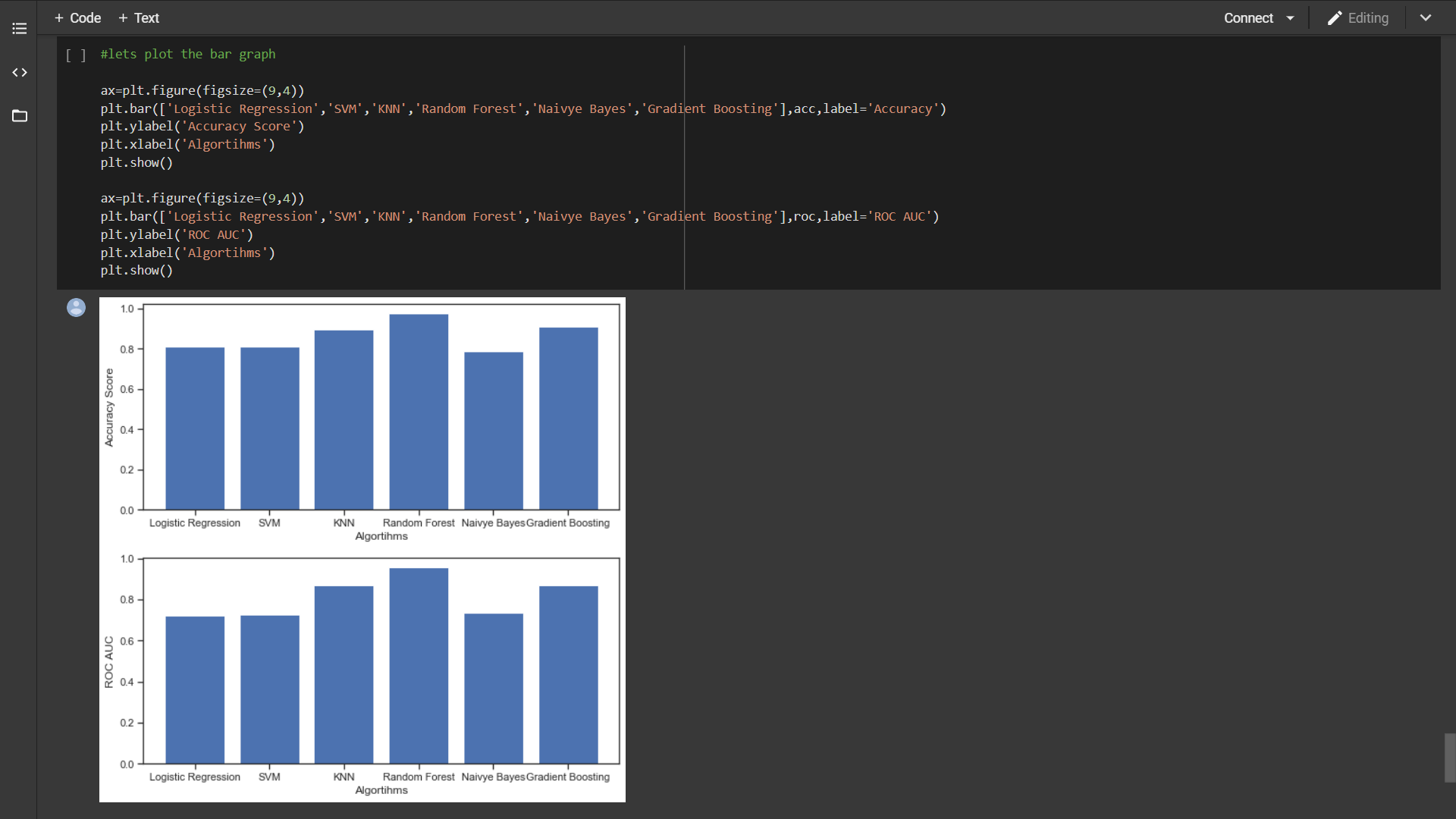












**CHAPTER5: CONCLUSION**

Electronic medical records (EMR) were used to train learning algorithms for DMT2 diagnosis. A variety of supervised learning algorithms were evaluated and it was found that SVMs and logistic regression produced the smallest error. SVMs are especially promising since one can adjust their behavior with diﬀerent choices of kernel or cost function parameter to suit the needs of medical practitioners trading oﬀ false negatives and false positives. Based upon the results, future studies should attempt to reduce the bias present in the logistic regression and SVM models. Speciﬁcally, new features such as genetic markers, lifestyle factors and more relevant lab tests (e.g. glucose, which was crucially missing) would provide additional dimensions along which to separate classes. Furthermore, future work should incorporate time series data, which is crucial for identifying the onset of DMT2 in a particular year. This will account for the possibility of internal structure that is currently not captured.

**CHAPTER6: REFERENCES**

1. <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16199/15676>
2. [https://towardsdatascience.com/machine-learning-for-diabetes-562dd7df4d42](https://www.google.com/url?q=https://towardsdatascience.com/machine-learning-for-diabetes-562dd7df4d42&sa=D&source=hangouts&ust=1587653047815000&usg=AFQjCNF5Ptcu8WihFNINDMLohbq1f9K30Q)
3. [https://www.sciencedirect.com/science/article/pii/S1877050918307737](https://www.google.com/url?q=https://www.sciencedirect.com/science/article/pii/S1877050918307737&sa=D&source=hangouts&ust=1587653047815000&usg=AFQjCNEdSab_ULxaWujCr9OSZ0pHkgtaDQ)
4. [https://www.kaggle.com/meenavyas/diabetic-retinopathy-detection](https://www.google.com/url?q=https://www.kaggle.com/meenavyas/diabetic-retinopathy-detection&sa=D&source=hangouts&ust=1587653047815000&usg=AFQjCNGZdLPiE_Wsgqte5ohCmsk-VKgaww)
5. [https://towardsdatascience.com/build-the-artificial-intelligence-for-detecting-diabetes-using-neural-networks-and-keras-89962097f0b0](https://www.google.com/url?q=https://towardsdatascience.com/build-the-artificial-intelligence-for-detecting-diabetes-using-neural-networks-and-keras-89962097f0b0&sa=D&source=hangouts&ust=1587653047815000&usg=AFQjCNHxhMjgVI_INyI2-AI6gbmtqqIr8A)