






ML – MINOR PROJECT- MAY

Problem Statement:

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 or not 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.

Packages and Libraries Used:

I've created this model using **Google Colaboratory**. To create this model, I've imported some of the python packages and libraries. Python packages and libraries which I've used are:

-  Pandas
-  Numpy
-  Matplotlib.pyplot
-  Seaborn
-  Scikit-learn
 - train_test_split from sklearn.model_selection
 - StandardScaler from sklearn.preprocessing
 - LogisticRegression from sklearn.linear_model
 - Accuracy_score from sklearn.metrics
 - Confusion_Matrix from sklearn.metrics

Algorithm:

I've solved this problem using **Logistic Regression**.

Logistic Regression:

Logistic Regression is a Machine Learning **classification** algorithm that is used to predict the probability of a categorical dependent variable. In Logistic regression, the dependent variable is a binary variable the contains data codes as **1 (yes, positive, etc.)** or **0 (no, negative, etc.)**.

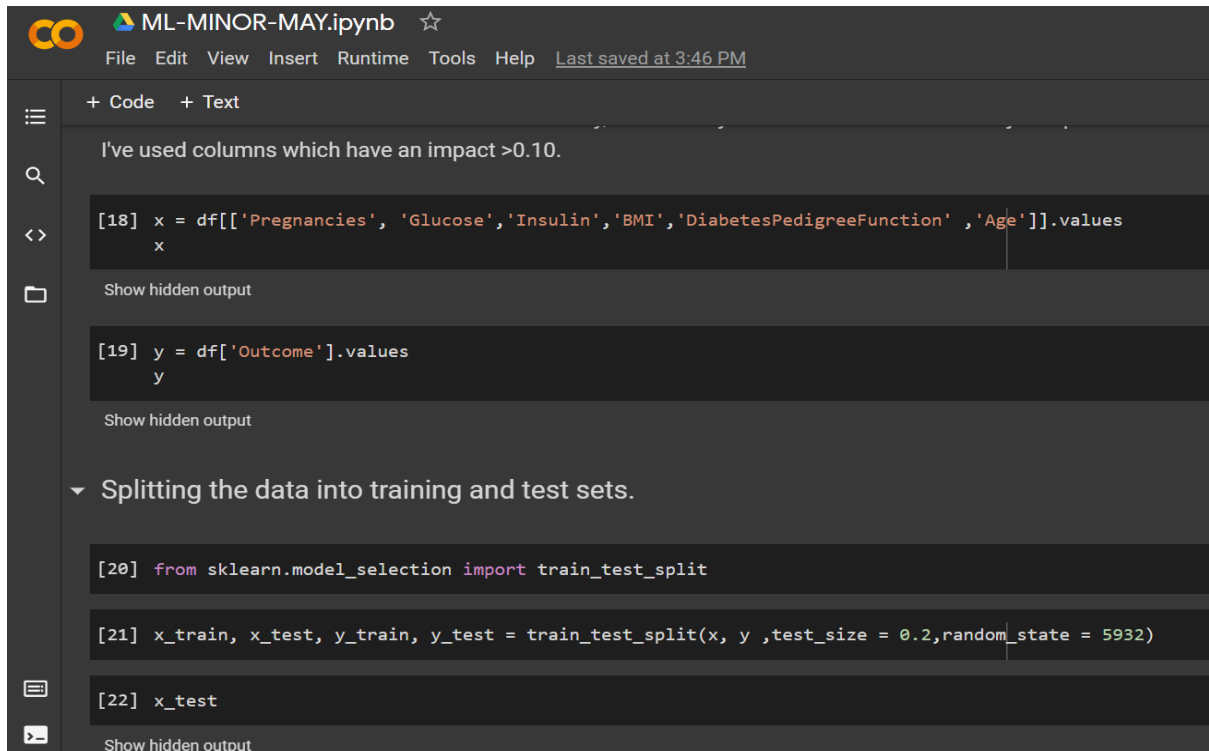
Logistic Regression Assumptions:

- Binary logistic regression requires the dependent variable to be binary.
- For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
- Only the meaningful variables should be included.
- The independent variables should be independent of each other. That is, the model should have little or no multicollinearity.
- The independent variables are linearly related to the log odds.
- Logistic regression requires quite large sample size.

ML – MINOR PROJECT- MAY

In this model, first I've imported some necessary python libraries. Now, in the data exploration part, I used pandas to read the data of the dataset and some other methods to display or explore the data. After the data exploration, I've visualized the data using libraries seaborn and matplotlib.pyplot. In the data preprocessing part, I used sklearn library to split the data into training and test sets and to scale the data(Feature Scaling). After predicting the values, I've calculated the accuracy_score and accuracy using Confusion Matrix by importing the sklearn library.

Here are the screenshots of some of the code and the entire code is in the ipynb notebook which is attached along with this document.



The screenshot shows a Jupyter Notebook interface with the title 'ML-MINOR-MAY.ipynb'. The menu bar includes File, Edit, View, Insert, Runtime, Tools, and Help. The left sidebar has icons for a list, search, expand/collapse, and a file explorer. The main area contains the following code cells:

```
+ Code + Text

I've used columns which have an impact >0.10.

[18] x = df[['Pregnancies', 'Glucose', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']].values
      x
Show hidden output

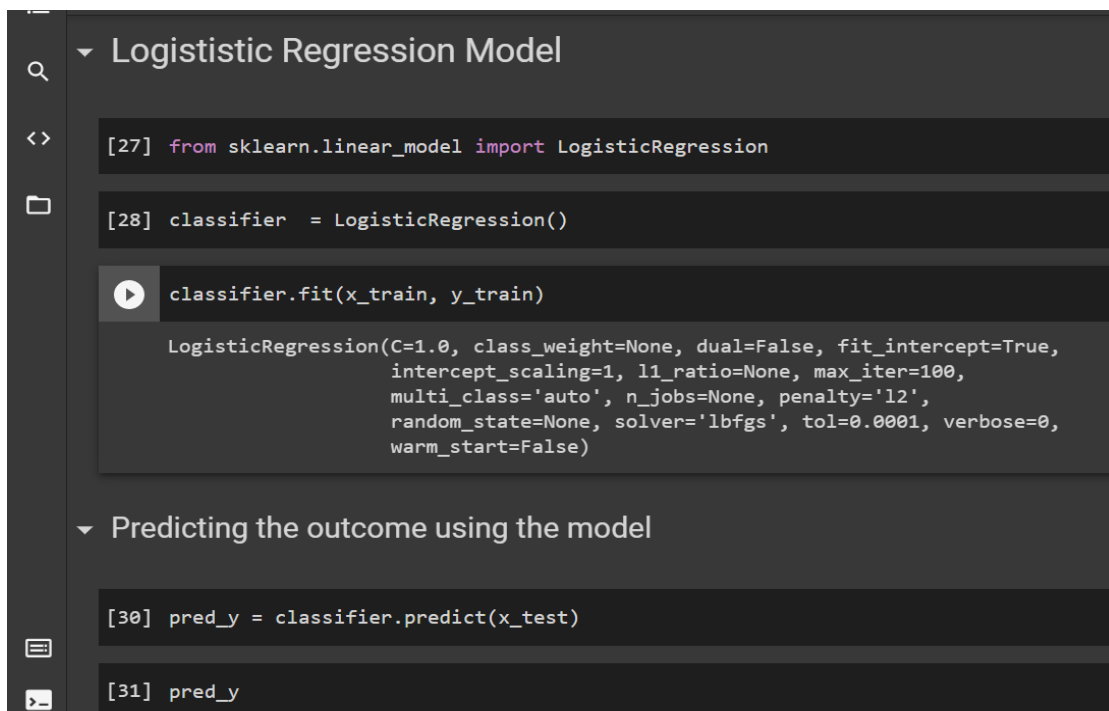
[19] y = df['Outcome'].values
      y
Show hidden output

▼ Splitting the data into training and test sets.

[20] from sklearn.model_selection import train_test_split

[21] x_train, x_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 5932)

[22] x_test
Show hidden output
```



The screenshot shows a Jupyter Notebook interface with the title 'Logistic Regression Model'. The left sidebar has icons for a list, search, expand/collapse, and a file explorer. The main area contains the following code cells:

```
▼ Logistic Regression Model

[27] from sklearn.linear_model import LogisticRegression

[28] classifier = LogisticRegression()

[29] classifier.fit(x_train, y_train)

LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
intercept_scaling=1, l1_ratio=None, max_iter=100,
multi_class='auto', n_jobs=None, penalty='l2',
random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
warm_start=False)

▼ Predicting the outcome using the model

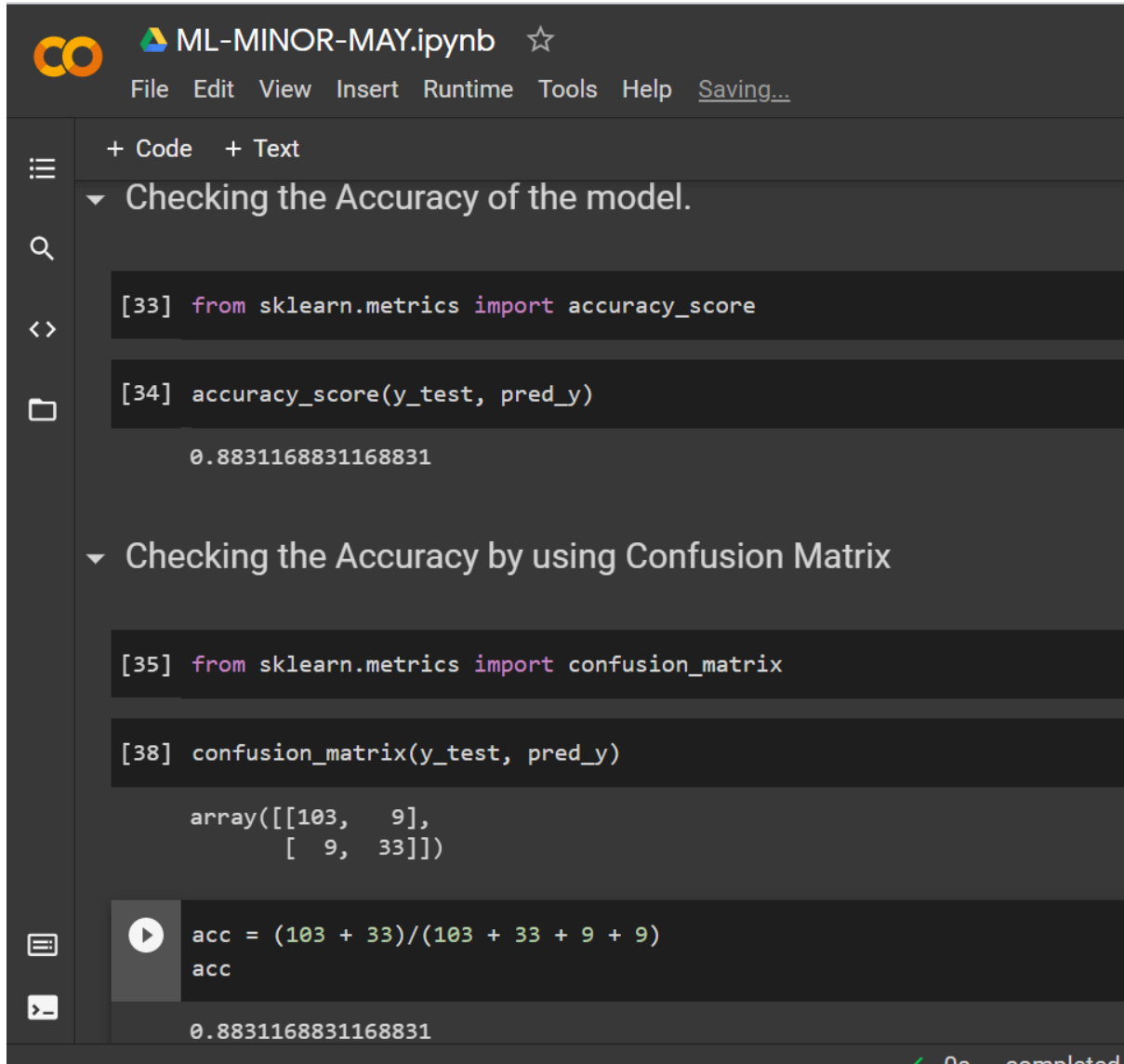
[30] pred_y = classifier.predict(x_test)

[31] pred_y
```

ML – MINOR PROJECT- MAY

Conclusion:

After predicting the value, I've checked the accuracy of the model. I've calculated the accuracy_score and also by using the confusion matrix using sklearn library. Here's the screenshot of the code,



```
ML-MINOR-MAY.ipynb ☆
File Edit View Insert Runtime Tools Help Saving...

+ Code + Text

▼ Checking the Accuracy of the model.

[33] from sklearn.metrics import accuracy_score

[34] accuracy_score(y_test, pred_y)

0.8831168831168831

▼ Checking the Accuracy by using Confusion Matrix

[35] from sklearn.metrics import confusion_matrix

[38] confusion_matrix(y_test, pred_y)

array([[103,  9],
       [ 9,  33]])

acc = (103 + 33)/(103 + 33 + 9 + 9)
acc

0.8831168831168831
```

Hereby, I conclude that the above Logistic Regression model is ready to predict whether a patient has diabetes or not, based on certain diagnostic measurements included in the dataset with an accuracy of 88.3%.