ARTIFICIAL INTELLIGENCE

PROJECT REPORT

Cancer

Detection

Group Member:	Faculty:
Kainat Fatima (9413)	
Ali Sher Fazil (9411)	**Sir Siraj Munir**

Date: 22nd April 2021

Problem description

In this modern world, we have different types of deceases around us. We hear lots of news about those deceases. In these all deceases, Cancel is very dangerous and many times it couldn’t be diagnosed early so patient have chances to suffer a lot and in the end, God forbid patient could have been died.

Although in this modern world, we have different type of mechanism to detect the cancer types. So I worked on this model and try to figure out that what type of cancer is this.

This assignment is bring us to classify the type of cancer by recognizing the various features and classify it in to 2 classes. I performed classification by using three models Random Forest, Decision Tree and Logistic Regression. For all these working, we use python as language and colab.research.google.com as our tool to classify and get dataset and working on it. I successfully classify the cancer type and successfully train our model to classify it.

Use Case

Admin

Admin will gather all dataset regarding cancer types and label them. He is responsible to prepare a proper excel or csv file so we can use it on future when it's needed.

Also he can delete or do feature engineering to maximize the accuracy of dataset and retain all necessary attributes and delete all unnecessary data from dataset. He can choose any model or machine learning algorithm to train machine by giving prepared dataset and examine that which one is best for this given dataset. He can check the model accuracy as well as check the precision and recall.

Tester

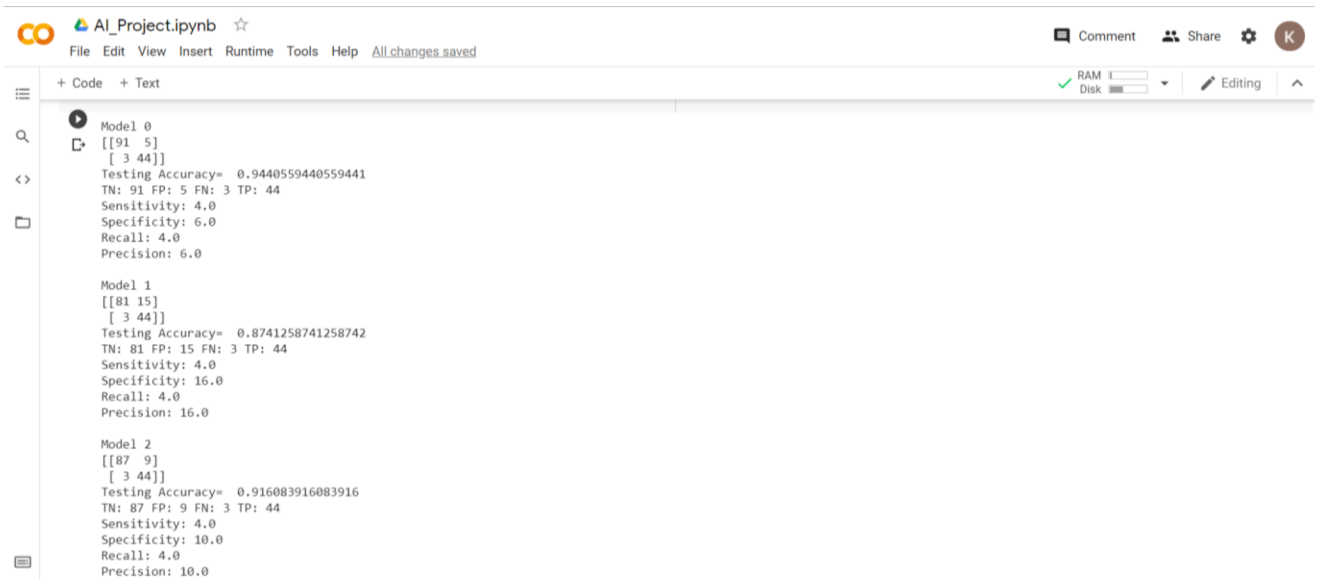
Tester can gather any of the data about cancer and choose any of the model which are chosen by Admin and test it that model can predict on new data or that. And if model can predict then what's the accuracy of it.

Doctors, Patient

After developed the system, Doctor or patient use this system to recognize the type of cancer. They can give the new data and machine will tell them that what type of cancer is this.

Accuracy

Overall performance of all models which we have used in this system.
Model 2 is Random forest.



Methods and Procedures

1- Import all necessary libraries

2- Import Dataset

3- All necessary steps of features engineering if needed.

4- Split data in to x and y for input and output.

5- Split data for training and testing. (Usually up to 75% of data for training)

6- Train the model on training data using one of the following:

1- Random Forest

We have a big dataset so we choose random forest as our training model. Because Random Forest choose the data from data set randomly like some trees in forest and train them and at last, voting will be done that how many result is in maximum number.

2- Decision Tree

Decision trees provide an effective method of Decision Making because they: Clearly lay out the problem so that all options can be challenged. Allow us to analyze fully the possible consequences of a decision. Provide a framework to quantify the values of outcomes and the probabilities of achieving them.

3- Logistic Regression

Logistic regression is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables.

7- Check the accuracy on test data with actual result and predicted result.

8- Construct the confusion matrix for detailed result and performance.

Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files
upl=files.upload()
data=pd.read_csv("data.csv")
data.head(5)
data.shape
data.isna().sum()
data=data.dropna(axis=1)
data.shape
data['diagnosis'].value_counts()
sns.countplot(data['diagnosis'],label='count')
from sklearn.preprocessing import LabelEncoder
labelEncoder_Y=LabelEncoder()
data.iloc[:,1]=labelEncoder_Y.fit_transform(data.iloc[:,1].values)
data.iloc[:,1]
sns.pairplot(data.iloc[:,1:5], hue="diagnosis")
plt.figure(figsize=(10,10))
#sns.heatmap(data.iloc[:,1:12].corr(), annot=True)
sns.heatmap(data.iloc[:,1:12].corr(), annot=True ,fmt='.0%')
X=data.iloc[:,2:31].values
Y=data.iloc[:,1].values
type(data)
```

Splitting data into testing and training

```
from sklearn.model_selection import train_test_split
trainx, testx ,trainy, testy = train_test_split(X,Y,test_size=0.25)
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
trainx=sc.fit_transform(trainx)
testx=sc.fit_transform(testx)

trainx
```

Random Forest

```
from sklearn.ensemble import RandomForestClassifier
```

```
  RF=RandomForestClassifier(n_estimators=10,criterion="entropy",random_sta
te=0)
  RF.fit(trainx,trainy)
```

Decision Tree

```
from sklearn.tree import DecisionTreeClassifier
  DT=DecisionTreeClassifier(criterion="entropy", random_state=0)
  DT.fit(trainx,trainy)
```

Logistic Regression

```
def models(trainx,trainy):
  from sklearn.linear_model import LogisticRegression
  log=LogisticRegression(random_state=0)
  log.fit(trainx,trainy)
```

Which one is better ?

```
print('[0]Logistic Regression Training Accuracy:' ,log.score(trainx,trainy
))
  print('[1]Decision Tree Classifier Training Accuracy:' ,DT.score(trainx,
trainy))
  print('[2]Random Forest Classifier Training Accuracy:' ,RF.score(trainx,
trainy))

  return log, DT , RF
```

Confusion Matrix

```
from sklearn.metrics import confusion_matrix

for i in range(len(model)) :
  print('Model', i)

  cm=confusion_matrix(testy,model[i].predict(testx))

  TP=cm[0][0]
  TN=cm[1][1]
  FN=cm[1][0]
  FP=cm[0][1]
  print(cm)
  test_accuracy=(TP+TN)/(TP+TN+FN+FP)
  print('Testing Accuracy= ',test_accuracy)

  tn,fp,fn,tp=confusion_matrix(testy,model[i].predict(testx)).ravel()
  print("TN:",tn,"FP:",fp,"FN:",fn,"TP:",tp)
```

```
  sensitivity=tp/tp+fn
  print("Sensitivity:",sensitivity)
  spec=tn/tn+fp
  print("Specificity:",spec)
  recall=tp/tp+fn
  print("Recall:",recall)
  Prec=tp/tp+fp
  print("Precision:",Prec)
  print()
```

Model 2(Random forest) better working:

```
pred=model[2].predict(testx)
print(pred)
print()
print(testy)

for i in range(len(model)) :
  print('Model', i)
  y_pred = model[i].predict(testx)
  cm = confusion_matrix(testy, y_pred)
  df_cm = pd.DataFrame(cm, range(2),
                range(2))
  plt.figure(figsize=(10,7))
  sns.set(font_scale=1.4)#for label size
  cm_plot = sns.heatmap(df_cm, annot=True, fmt='n', annot_kws={"size": 10}
)
```

Output Screenshots

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files
upl=files.upload()
data=pd.read_csv("data.csv")
data.head(5)
```

Choose Files data.csv

- **data.csv**(application/vnd.ms-excel) - 125204 bytes, last modified: 4/21/2020 - 100% done

```
Saving data.csv to data.csv
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	fractal_dimer
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	

✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
[ ] data.shape
```

```
(569, 33)
```

```
data.isna().sum()
```

```
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
```

✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

```
[3] data=data.dropna(axis=1)
```

```
[4] data.shape
```

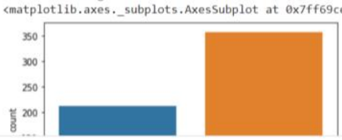
```
(569, 32)
```

```
[5] data['diagnosis'].value_counts()
```

```
B    357
M    212
Name: diagnosis, dtype: int64
```

```
[6] sns.countplot(data['diagnosis'],label='count')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid posit
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ff69ce601d0>
```



AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

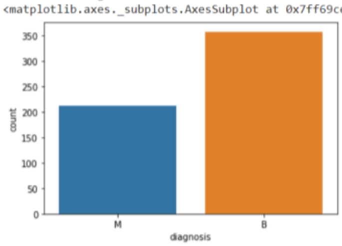
Comment Share

+ Code + Text RAM Disk Editing

```
M    212
Name: diagnosis, dtype: int64
```

```
sns.countplot(data['diagnosis'],label='count')
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid posit
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7ff69ce601d0>
```




```
[7] data.dtypes
```

```
id                         int64
```

✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text RAM Disk Editing

```
fractal_dimension_worst    float64
dtype: object
```

```
[ ] from sklearn.preprocessing import LabelEncoder
    labelEncoder_Y=LabelEncoder()
    data.iloc[:,1]=labelEncoder_Y.fit_transform(data.iloc[:,1].values)
    data.iloc[:,1]
```

```
0      1
1      1
2      1
3      1
4      1
      ..
564    1
565    1
566    1
567    1
568    0
Name: diagnosis, Length: 569, dtype: int64
```

```
[ ] #sns.pairplot(data.iloc[:,1:5])
    sns.pairplot(data.iloc[:,1:5], hue="diagnosis")
```

```
<seaborn.axisgrid.PairGrid at 0x7f011b7f9be0>
```

✓ 0s completed at 8:39 AM

AI_Project.ipynb

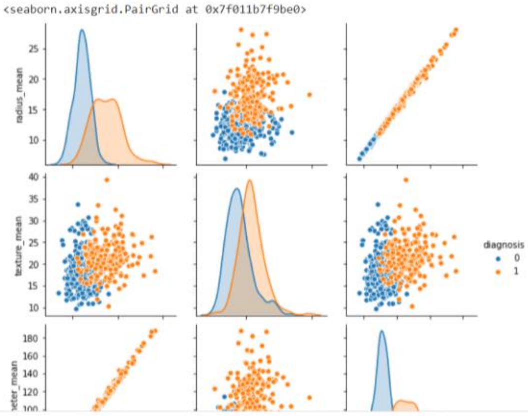
File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text RAM Disk Editing

```
#sns.pairplot(data.iloc[:,1:5])
sns.pairplot(data.iloc[:,1:5], hue="diagnosis")
```

```
<seaborn.axisgrid.PairGrid at 0x7f011b7f9be0>
```



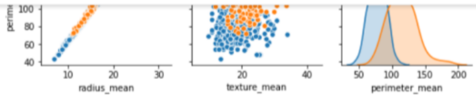
✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share K

+ Code + Text RAM Disk Editing




```
[ ] data.head(5)
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	fractal_dimer
0	842302	1	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2419	
1	842517	1	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1812	
2	84300903	1	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2069	
3	84348301	1	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2597	
4	84358402	1	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1809	

```
[ ] data.iloc[:,1:12].corr()
```

diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	fr

✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share K

+ Code + Text RAM Disk Editing

✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share K

+ Code + Text RAM Disk Editing


```
[ ] X=data.iloc[:,2:31].values
    Y=data.iloc[:,1].values
    type(data)

pandas.core.frame.DataFrame
```

```
[ ] from sklearn.model_selection import train_test_split
    trainx, testx ,trainy, testy = train_test_split(X,Y,test_size=0.25)
```

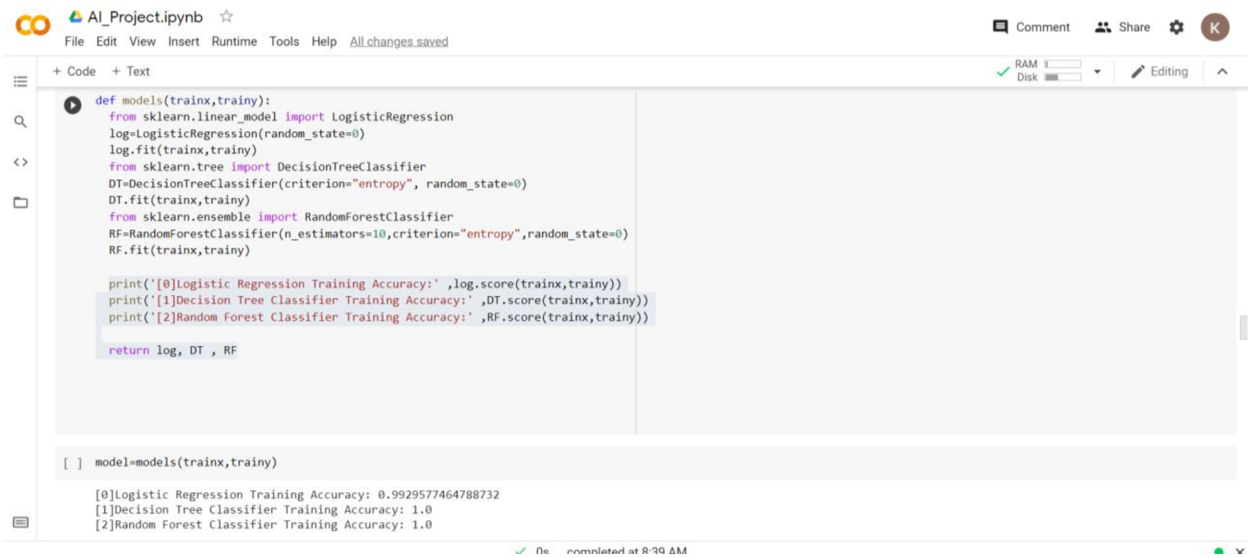
```
[ ] from sklearn.preprocessing import StandardScaler
    sc=StandardScaler()
    trainx=sc.fit_transform(trainx)
    testx=sc.fit_transform(testx)

    trainx

array([[ 1.05657846,  0.29671443,  1.04367797, ..., -0.10851178,
         0.50373099,  0.54701847],
       [ 1.49576352, -0.26664406,  1.54034385, ...,  1.89525489,
         1.6456224 ,  1.17033107],
       [-0.55561987, -1.21570444, -0.55615147, ...,  0.14137192,
        -0.36247951, -0.36568927],
       ...,
       [ 1.17888316,  0.61228869,  1.11636078, ...,  0.55894074,
         0.33802115,  0.32122666],
       [-0.51948439, -1.64114529, -0.54969078, ..., -0.46220055,
```


✓ 0s completed at 8:39 AM

Model Trainings



Confusion Matrix

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing

```python
from sklearn.metrics import confusion_matrix

for i in range(len(model)) :
  print('Model', i)

  cm=confusion_matrix(testy,model[i].predict(testx))

  TP=cm[0][0]
  TN=cm[1][1]
  FN=cm[1][0]
  FP=cm[0][1]
  print(cm)
  test_accuracy=(TP+TN)/(TP+TN+FN+FP)
  print('Testing Accuracy= ',test_accuracy)

  tn,fp,fn,tp=confusion_matrix(testy,model[i].predict(testx)).ravel()
  print("TN:",tn,"FP:",fp,"FN:",fn,"TP:",tp)
  sensitivity=tp/tp+fn
  print("Sensitivity:",sensitivity)
  spec=tn/tn+fp
  print("Specificity:",spec)
  recall=tp/tp+fn
  print("Recall:",recall)
  Prec=tp/tp+fp
  print("Precision:",Prec)
  print()
```

✓ 0s completed at 8:39 AM

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing


```
Model 0
[[91  5]
 [ 3 44]]
Testing Accuracy=  0.9440559440559441
TN: 91 FP: 5 FN: 3 TP: 44
Sensitivity: 4.0
Specificity: 6.0
Recall: 4.0
Precision: 6.0

Model 1
[[81 15]
 [ 3 44]]
Testing Accuracy=  0.8741258741258742
TN: 81 FP: 15 FN: 3 TP: 44
Sensitivity: 4.0
Specificity: 16.0
Recall: 4.0
Precision: 16.0

Model 2
[[87  9]
 [ 3 44]]
Testing Accuracy=  0.916083916083916
TN: 87 FP: 9 FN: 3 TP: 44
Sensitivity: 4.0
Specificity: 10.0
Recall: 4.0
Precision: 10.0
```


AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

RAM Disk Editing


```python
pred=model[2].predict(testx)
print(pred)
print()
print(testy)
```

```
[1 1 0 1 1 1 0 0 1 1 1 1 0 0 0 1 0 1 0 0 1 1 0 0 1 1 0 0 0 0 1 1 0 1 0 1 0
 1 1 0 0 0 1 0 0 0 0 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0 1 1 0 0 1 0 1
 0 0 0 1 0 0 1 0 0 1 0 0 0 0 0 1 1 0 1 0 1 0 1 1 1 0 0 0 1 0 0 0 1 1 0 1 1
 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 0 0 0 0 0]

[1 1 0 0 1 1 0 0 1 1 1 1 0 0 0 1 1 1 0 0 1 1 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0
 1 1 0 0 0 1 0 0 0 0 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 1
 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 1 1 1 0 0 0 1 0 0 0 1 1 0 1 1
 0 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0]
```

```python
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn.metrics import roc_curve, auc
y_score = model[0].fit(trainx, trainy).decision_function(testx)
fpr, tpr, thresholds = roc_curve(testy, y_score)
def ROC_curve(fpr,tpr):

    sns.set_style('darkgrid', {'axes.facecolor': '0.9'})
    print('AUC: {}'.format(auc(fpr, tpr)))
    plt.figure(figsize=(10, 8))
    lw = 2
```

AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved

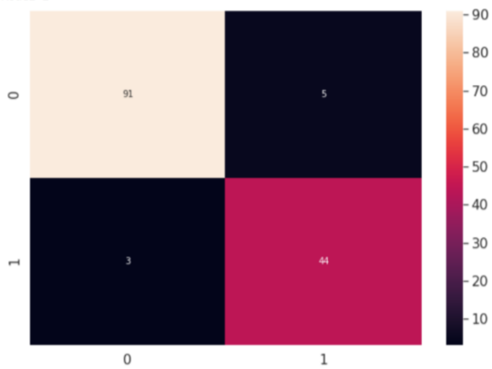
```
for i in range(len(model)) :
  print('Model', i)
  y_pred = model[i].predict(testx)
  cm = confusion_matrix(testy, y_pred)
  df_cm = pd.DataFrame(cm, range(2),
                  range(2))
  plt.figure(figsize=(10,7))
  sns.set(font_scale=1.4)#for label size
  cm_plot = sns.heatmap(df_cm, annot=True, fmt='n', annot_kws={"size": 10})
```

```
Model 0
Model 1
Model 2
```

AI_Project.ipynb

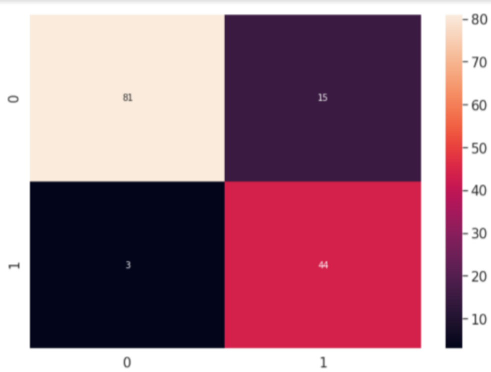
File Edit View Insert Runtime Tools Help All changes saved

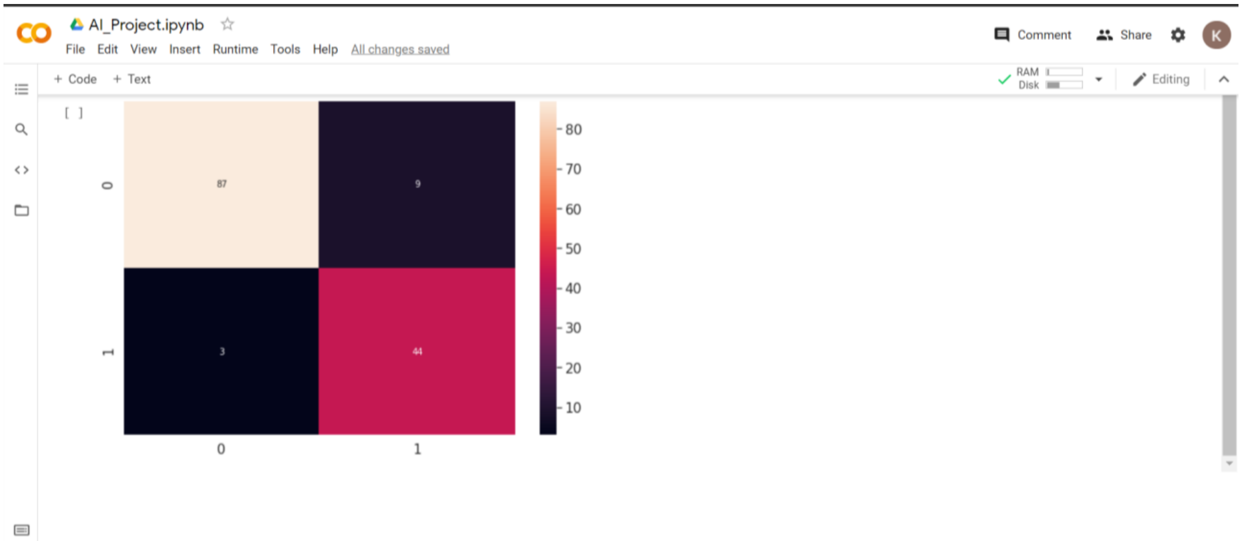
```
Model 0
Model 1
Model 2
```



AI_Project.ipynb

File Edit View Insert Runtime Tools Help All changes saved





Supporting Tools and Techniques

Basic Library
numpy
pandas

Graph Library
matplotlib.pyplot
seaborn

Models and Split data Library
sklearn.model_selection import train_test_split
sklearn.ensemble import RandomForestClassifier
sklearn.linear_model import LogisticRegression
sklearn.tree import DecisionTreeClassifier

Models Performances Library
from sklearn.metrics import confusion_matrix,
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_recall_fscore_support

Python Version
`3.8.2`

Tool
Colaboratory by google

References

Dr. Affan’s Machine Learning Lecture where he gave students this dataset to classify that.