Linear Regression using Single variable.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

#Reading Data
data = pd.read_csv('/content/headbrain.csv')
data.head()
```

	Gender	Age Range	Head Size(cm^3)	Brain Weight(grams)
0	1	1	4512	1530
1	1	1	3738	1297
2	1	1	4261	1335
3	1	1	3777	1282
4	1	1	4177	1590

```
[ ] # collecting X and Y
    X= data['Head Size(cm^3)'].values
    Y = data['Brain Weight(grams)'].values
    # data.iloc[0:5,0:1]
    m = len(X)
    X = X.reshape((m,1))
    # Model Initialization
    reg = LinearRegression()
    # Data Fitting
```

Linear Regression using Single variable.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
[ ] # collecting X and Y
    X= data['Head Size(cm^3)'].values
    Y = data['Brain Weight(grams)'].values
    # data.iloc[0:5,0:1]
    m = len(X)
    X = X.reshape((m,1))
    # Model Initialization
    reg = LinearRegression()
    # Data Fitting
    reg = reg.fit(X,Y)
    # y prediction
    Y_pred = reg.predict(X)
    print(Y_pred)
```

```
[1514.16660083 1310.27229206 1448.04583661 1320.5460363  1425.9177721
 1269.96760312 1322.65347102 1263.11844029 1277.34362462 1374.5490509
 1232.56063691 1377.44677363 1284.45621679 1434.08408162 1335.03464997
 1346.62554091 1246.78582124 1238.61951172 1140.88722677 1490.98481895
 1347.15239959 1391.40852862 1327.65862847 1302.89627056 1576.07249561
 1490.72138961 1388.77423523 1385.08622448 1235.45835965 1425.39091342
 1323.1803297  1325.55119375 1406.42400097 1421.70290266 1420.91261465
 1330.29292186 1254.68870143 1312.90658546 1203.84683891 1245.46867455
 1369.28046411 1248.89325596 1462.27102095 1327.65862847 1343.2009595
 1388.24737655 1460.42701557 1159.32728053 1284.71964613 1285.50993415
 1350.576981   1324.76090573 1450.41670067 1395.88682739 1382.45193108
 1236.512077   1350.31355166 1423.02004936 1362.16787194 1292.09566764
 1343.99124752 1219.38916994 1495.72654706 1445.9384019  1307.37456933
 1202.52969221 1225.18461541 1213.85715381 1492.56539499 1479.65735735
 1381.39821372 1341.8838128  1411.16572909 1394.30625136 1332.92721526
 1219.65259928 1262.85501095 1211.22286041 1360.85072524 1335.82493799
 1334.50779129 1341.35695412 1181.45534505 1267.8601684  1361.64101326
```

Type here to search

Linear Regression using Single Variable.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
[ ]  1216.22801786 1080.03504935 1310.79915074 1242.04409313 1286.30022217
     1304.47684659 1190.67537193 1433.03036426 1309.48200404 1173.28903553
     1296.31053707 1301.31569452 1172.23531817 1219.65259928 1177.24047562
     1208.58856702 1218.86231126]
```

```
[ ]  print(Y)
```

```
[1530 1297 1335 1282 1590 1300 1400 1255 1355 1375 1340 1380 1355 1522
 1208 1405 1358 1292 1340 1400 1357 1287 1275 1270 1635 1505 1490 1485
 1310 1420 1318 1432 1364 1405 1432 1207 1375 1350 1236 1250 1350 1320
 1525 1570 1340 1422 1506 1215 1311 1300 1224 1350 1335 1390 1400 1225
 1310 1560 1330 1222 1415 1175 1330 1485 1470 1135 1310 1154 1510 1415
 1468 1390 1380 1432 1240 1195 1225 1188 1252 1315 1245 1430 1279 1245
 1309 1412 1120 1220 1280 1440 1370 1192 1230 1346 1290 1165 1240 1132
 1242 1270 1218 1430 1588 1320 1290 1260 1425 1226 1360 1620 1310 1250
 1295 1290 1290 1275 1250 1270 1362 1300 1173 1256 1440 1180 1306 1350
 1125 1165 1312 1300 1270 1335 1450 1310 1027 1235 1260 1165 1080 1127
 1270 1252 1200 1290 1334 1380 1140 1243 1340 1168 1322 1249 1321 1192
 1373 1170 1265 1235 1302 1241 1078 1520 1460 1075 1280 1180 1250 1190
 1374 1306 1202 1240 1316 1280 1350 1180 1210 1127 1324 1210 1290 1100
 1280 1175 1160 1205 1163 1022 1243 1350 1237 1204 1090 1355 1250 1076
 1120 1220 1240 1220 1095 1235 1105 1405 1150 1305 1220 1296 1175  955
 1070 1320 1060 1130 1250 1225 1180 1178 1142 1130 1185 1012 1280 1103
 1408 1300 1246 1380 1350 1060 1350 1220 1110 1215 1104 1170 1120]
```

```
[ ]  print('Intercept: \n', reg.intercept_)
     print('Coefficient:\n', reg.coef_)
```

```
Intercept:
```

Linear Regression using Single variable.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
Intercept:
 325.5734210494426
Coefficient:
 [0.26342934]
```

```
# Model Evaluation
rmse = np.sqrt(mean_squared_error(Y,Y_pred))
r2 = reg.score(X,Y)
print("RMSE")
print(rmse)
print("R2")
print(r2)
```

```
RMSE
72.1206213783709
R2
0.639311719957
```

```
```

Linear_regression_using_multiple_variable.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
[ ] import numpy as np
    import pandas as pd
    from sklearn.linear_model import LinearRegression
    from sklearn.metrics import mean_squared_error

    #Reading Data
    data = pd.read_csv('/content/50_Startups.csv')
    data.head()
```

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

```
[ ] X = data[['R&D Spend','Administration','Marketing Spend']].values

    y = data['Profit'].values
    print(X)
    print(y)
```

Linear_regression_using_multiple_variable.ipynb ☆

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
# Model Initialization
reg = LinearRegression()
# Data Fitting
reg = reg.fit(X,y)
# y prediction
Y_pred = reg.predict(X)
print(Y_pred)
```

```
[192521.25289008 189156.76823227 182147.2790962  173696.70002553
 172139.51418327 163580.7805712  158114.09666865 160021.36304781
 151741.69969865 154884.68410995 135509.01636714 135573.71296074
 129138.05418243 127487.99166275 149548.64633453 146235.1599852
 116915.40540144 130192.44720781 129014.2268059  115635.21636716
 116639.6692309  117319.45164029 114706.98171695 109996.61522126
 113362.96611314 102237.72506481 110600.5753503  114408.07145684
 101660.02600497 101794.98345176  99452.37293606  97687.85627575
  99001.32898549  97915.00780465  89039.27374116  90511.59956753
  75286.17458546  89619.5377079   69697.43064804  83729.01197692
  74815.95399105  74802.55623866  70620.41182056  60167.03996335
  64611.3549157   47650.64968691  56166.20685261  46490.58898335
  49171.38815763  48215.1341113 ]
```

+ Code

```
[ ] print(y)
```

```
[192261.83 191792.06 191050.39 182901.99 166187.94 156991.12 156122.51
 155752.6  152211.77 149759.96 146121.95 144259.4  141585.52 134307.35
 132602.65 129917.04 126992.93 125370.37 124266.9  122776.86 118474.03
 111313.02 110352.25 108733.99 108552.04 107404.34 105733.54 105008.31
 103282.38 101004.64  99937.59  97483.56  97427.84  96778.92  96712.8
```

Linear_regression_using_multiple_variable.ipynb ☆

File Edit View Insert Runtime Tools Help Last edited on December 2

+ Code + Text

```
[ ] print('Intercept: \n', reg.intercept_)
    print('Coefficient:\n', reg.coef_)
```

```
Intercept:
 50122.19298986523
Coefficient:
 [ 0.80571505 -0.02681597  0.02722806]
```

```
[ ] # Model Evaluation
    rmse = np.sqrt(mean_squared_error(y,Y_pred))
    r2 = reg.score(X,y)
    print("RMSE")
    print(rmse)
    print("R2")
    print(r2)
```

```
RMSE
8855.34448901514
R2
0.9507459940683246
```

Logistic_Regression.ipynb ☆

File Edit View Insert Runtime Tools Help Last edited on December 3

+ Code + Text

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import mean_squared_error

#Reading Data
data = pd.read_csv('/content/diabetes.csv')
data.head()
```

	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

```
[ ] # collecting X and Y
    X= data[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction','Age']].values
    Y = data['Outcome'].values

    # Data Fitting
    reg = reg.fit(X,Y)
    # y prediction
    Y_pred = reg.predict(X)
    print(Y_pred)
```

Logistic_Regression.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 3

+ Code + Text

```
# Model Initialization
reg = LogisticRegression()
```

```
print(Y)
```

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

```
X = df.iloc[:,0:8]
y = df.iloc[:,8]
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train, Y_test
```

Logistic_Regression.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 3

+ Code + Text

```
[ ] df = pd.read_csv('/content/diabetes.csv', delimiter=',', nrows = None)
    df.dataframeName = 'diabetes.csv'
    nRow, nCol = df.shape
    print(f'rows: {nRow}\ncolumns: {nCol}')
```

```
rows: 768
columns: 9
```

```
[ ] X = data.iloc[:,0:8]
    y = data.iloc[:,8]
    from sklearn.model_selection import train_test_split
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.3,random_state=101)

    from sklearn.linear_model import LogisticRegression

    logmodel = LogisticRegression()

    logmodel.fit(X_train,Y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:818: ConvergenceWarning: lbfgs failed to converge
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
LogisticRegression()
```

```
[ ] predictions = logmodel.predict(X_test)
    from sklearn.metrics import classification_report
```

Logistic_Regression.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 3

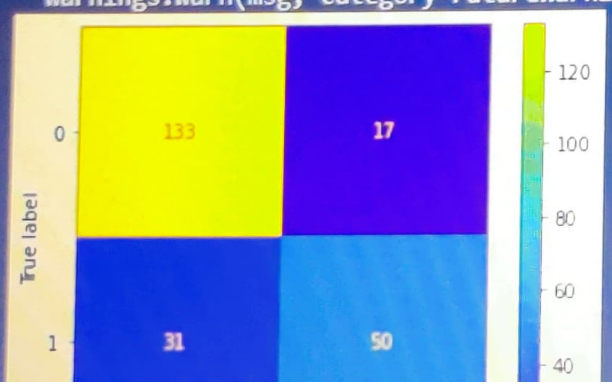
+ Code + Text

```
[ ] predictions = logmodel.predict(X_test)
    from sklearn.metrics import classification_report
    print(predictions)
```

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

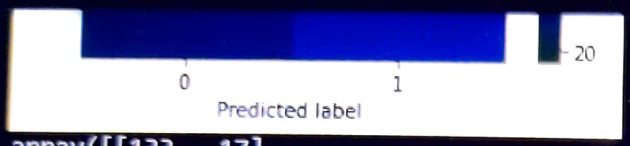
```
[ ] import matplotlib.pyplot as plt
    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import plot_confusion_matrix
    plot_confusion_matrix(logmodel, X_test, Y_test)
    plt.show()
    confusion_matrix(Y_test, predictions)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning:
  warnings.warn(msg, category=FutureWarning)
```



+ Code + Text

[ ]




```
array([[133,  17],
       [ 31,  50]])
```

[ ] `print(classification_report(Y_test, predictions))`

	precision	recall	f1-score	support
0	0.81	0.89	0.85	150
1	0.75	0.62	0.68	81
accuracy			0.79	231
macro avg	0.78	0.75	0.76	231
weighted avg	0.79	0.79	0.79	231

Logistic_Regression_Multiple_variable.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import mean_squared_error

#Reading Data
data = pd.read_csv('/content/50_Startups.csv')
data.head()
```

	R&D Spend	Administration	Marketing Spend	State	Profit
0	165349.20	136897.80	471784.10	New York	192261.83
1	162597.70	151377.59	443898.53	California	191792.06
2	153441.51	101145.55	407934.54	Florida	191050.39
3	144372.41	118671.85	383199.62	New York	182901.99
4	142107.34	91391.77	366168.42	Florida	166187.94

```
[ ] x= data.iloc[:,[0,1,2]].values
```

```
[ ] print(x)
```

```
[[165349.2  136897.8  471784.1 ]
 [162597.7  151377.59 443898.53]
 [153441.51 101145.55 407934.54]
 [144372.41 118671.85 383199.62]
 [142107.34  91391.77 366168.42]
```

Logistic_Regression_Multiple_variable.ipynb ☆

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
[ ] y = data.Profit.values.reshape(-1,1)
```

```
print(y)
```

```
[[192261.83]
 [191792.06]
 [191050.39]
 [182901.99]
 [166187.94]
 [156991.12]
 [156122.51]
 [155752.6 ]
 [152211.77]
 [149759.96]
 [146121.95]
 [144259.4 ]
 [141585.52]
 [134307.35]
 [132602.65]
 [129917.04]
 [126992.93]
 [125370.37]
 [124266.9 ]
 [122776.86]
 [118474.03]
 [111313.02]
 [110352.25]
 [108733.99]
 [108552.04]
```

DecisionTreeClassifier.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

#Reading Data
data = pd.read_csv('/content/diabetes.csv')
data.head()
```

	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

```
[ ] X= data[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction','Age']].values
    Y = data['Outcome'].values
```

```
[ ] x = data.iloc[:,0:8]
    y = data.iloc[:,8]
    print(x)
    print(y)
```

DecisionTreeClassifier.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
[ ] from sklearn.model_selection import train_test_split
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.25,random_state=0)
```

```
[ ] from sklearn.tree import DecisionTreeClassifier
    model = DecisionTreeClassifier()

    model.fit(X_train,Y_train)
```

```
DecisionTreeClassifier()
```

```
[ ] predictions = model.predict(X_test)
    from sklearn.metrics import classification_report
    print(predictions)
```

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

```
[ ] import matplotlib.pyplot as plt
    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import plot_confusion_matrix
    plot_confusion_matrix(model, X_test, Y_test)
    plt.show()
```

DecisionTreeClassifier.ipynb

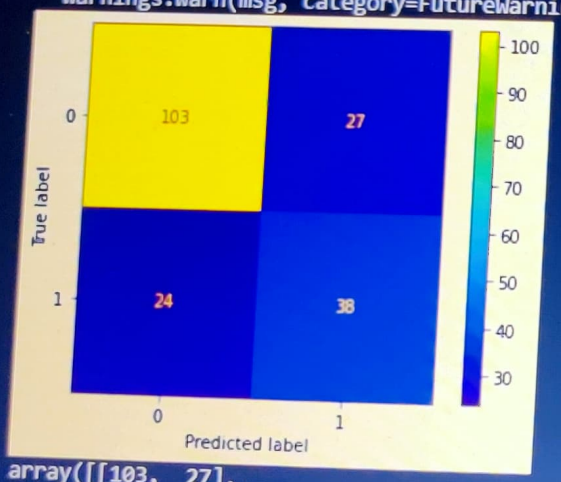
File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

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

```
[ ]  import matplotlib.pyplot as plt
     from sklearn.metrics import confusion_matrix
     from sklearn.metrics import plot_confusion_matrix
     plot_confusion_matrix(model, X_test, Y_test)
     plt.show()
     confusion_matrix(Y_test, predictions)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_co
  warnings.warn(msg, category=FutureWarning)
```




```
array([[103,  27],
       [ 24,  38]])
```

```
[ ]  print(classification_report(Y_test, predictions))
```

Naive_bayesipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
from sklearn.naive_bayes import GaussianNB
import numpy as np
import pandas as pd
from sklearn.metrics import mean_squared_error

#Reading Data
data = pd.read_csv('/content/diabetes.csv')
data.head()
```

	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

```
X= data[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction','Age']].values
Y = data['Outcome'].values
print(X)
print(Y)
```

```
[[  6.   148.    72.   ...  33.6    0.627  50.  ]
 [  1.    85.    66.   ...  26.6    0.351  31.  ]
 [  8.   183.    64.   ...  23.3    0.672  32.  ]
 ...
```

Naive_bayesipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
[ ] x = data.iloc[:,0:8]
    y = data.iloc[:,8]
    print(x)
    print(y)
```

```
     Pregnancies  Glucose  BloodPressure  ...   BMI  DiabetesPedigreeFunction  Age
0              6      148             72  ...  33.6                     0.627   50
1              1       85             66  ...  26.6                     0.351   31
2              8      183             64  ...  23.3                     0.672   32
3              1       89             66  ...  28.1                     0.167   21
4              0      137             40  ...  43.1                     2.288   33
..           ...      ...            ...  ...   ...                       ...  ...
763           10      101             76  ...  32.9                     0.171   63
764            2      122             70  ...  36.8                     0.340   27
765            5      121             72  ...  26.2                     0.245   30
766            1      126             60  ...  30.1                     0.349   47
767            1       93             70  ...  30.4                     0.315   23

[768 rows x 8 columns]
0      1
1      0
2      1
3      0
4      1
      ..
763    0
764    0
765    0
766    1
767    0
Name: Outcome, Length: 768, dtype: int64
```

Type here to search

Naive_Bayes.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
[ ] from sklearn.model_selection import train_test_split
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.25,random_state=0)
```

```
[ ] from sklearn.naive_bayes import GaussianNB
    model = GaussianNB()

    model.fit(X_train,Y_train)
```

```
GaussianNB()
```

```
[ ] predictions = model.predict(X_test)
    from sklearn.metrics import classification_report
    print(predictions)
```

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

```
[ ] import matplotlib.pyplot as plt
    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import plot_confusion_matrix
    plot_confusion_matrix(model, X_test, Y_test)
    plt.show()
    confusion_matrix(Y_test, predictions)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_
  warnings.warn(msg, category=FutureWarning)
```

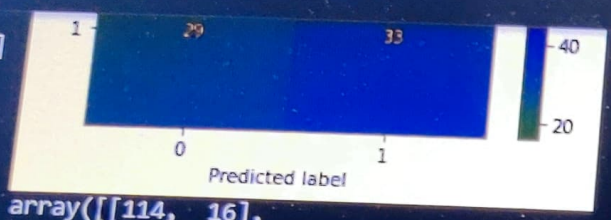
Type here to search

Naive_bayesipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

[ ]




```
array([[114,  16],
       [ 29,  33]])
```

[ ] `print(classification_report(Y_test, predictions))`

	precision	recall	f1-score	support
0	0.80	0.88	0.84	130
1	0.67	0.53	0.59	62
accuracy			0.77	192
macro avg	0.74	0.70	0.71	192
weighted avg	0.76	0.77	0.76	192

Support_Vector_machine.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
from sklearn.naive_bayes import GaussianNB
import numpy as np
import pandas as pd
from sklearn.metrics import mean_squared_error
from sklearn.svm import SVC

#Reading Data
data = pd.read_csv('/content/diabetes.csv')
data.head()
```

	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

```
[ ] X= data[['Pregnancies','Glucose','BloodPressure','SkinThickness','Insulin','BMI','DiabetesPedigreeFunction','Age']].values
    Y = data['Outcome'].values
```

```
[ ] x = data.iloc[:,0:8]
    y = data.iloc[:,8]
    print(x)
    print(y)
```

Type here to search

Support_Vector_machine.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
[ ] from sklearn.model_selection import train_test_split
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.25,random_state=0)
```

```
from sklearn.svm import SVC
model = SVC(kernel = 'linear')

model.fit(X_train,Y_train)
```

```
SVC(kernel='linear')
```

```
[ ] predictions = model.predict(X_test)
    from sklearn.metrics import classification_report
    print(predictions)
```

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

```
[ ] import matplotlib.pyplot as plt
    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import plot_confusion_matrix
    plot_confusion_matrix(model, X_test, Y_test)
    plt.show()
    confusion_matrix(Y_test, predictions)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_co
  warnings.warn(msg, category=FutureWarning)
```

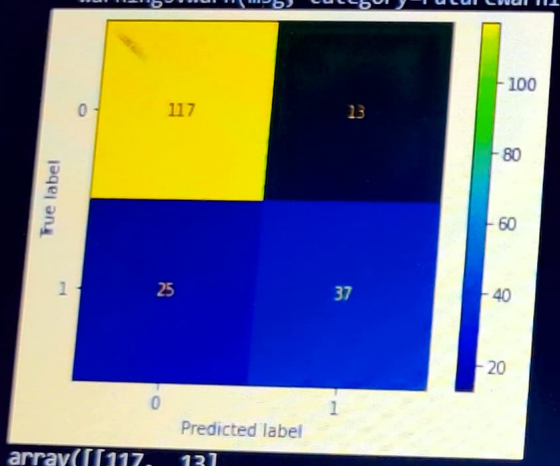
Support_Vector_machine.ipynb

File Edit View Insert Runtime Tools Help Last edited on December 9

+ Code + Text

```
plt.show()
confusion_matrix(Y_test, predictions)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Function plot_con
  warnings.warn(msg, category=FutureWarning)
```




```
array([[117,  13],
       [ 25,  37]])
```

```
[ ] print(classification_report(Y_test, predictions))
```

	precision	recall	f1-score	support
0	0.82	0.90	0.86	130
1	0.74	0.60	0.66	62
accuracy			0.80	192
macro avg	0.78	0.75	0.76	192