19IT016
BHARAT SRINIVAS R

Applied Datascience with Python

Programming Assignment -III

Code:

```
import pandas as pd

df = pd.read_csv("diabetes dataset.csv")
df.head()


y = df["Outcome"]

y = y.map(dict(yes=1, no=0))

x = df.drop(['Outcome'], axis=1)

from sklearn.tree import DecisionTreeClassifier
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=1)
model = DecisionTreeClassifier()
model = model.fit(x_train,y_train)
pred = model.predict(x_test)
from sklearn import metrics
print("Accuracy:",metrics.accuracy_score(y_test, pred)*100)

from sklearn.metrics import classification_report
print(classification_report(y_test,pred))

X_train, X_test, y_train, y_test = train_test_split(x, y,test_size=0.2,random_state=0)
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier()
rfc.fit(X_train, y_train)
r_pred = rfc.predict(X_test)
rfc.score(X_test, y_test)
from sklearn.metrics import confusion_matrix
mat = confusion_matrix(y_test, r_pred)
Mat
print("Accuracy:",metrics.accuracy_score(y_test, r_pred)*100)
print(classification_report(y_test,r_pred))
```

OUTPUT:

jupyter Untitled Last Checkpoint: 24 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

```
In [1]: import pandas as pd

In [2]: df = pd.read_csv("diabetes dataset.csv")

In [3]: df.head()
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	6	148	72	35	0	33.6	0.627	50	yes
1	1	85	66	29	0	26.6	0.351	31	no
2	8	183	64	0	0	23.3	0.672	32	yes
3	1	89	66	23	94	28.1	0.167	21	no
4	0	137	40	35	168	43.1	2.288	33	yes

```
In [4]: y = df["Outcome"]

In [5]: y
```

Out[5]:
```
0      yes
1       no
2      yes
3       no
4      yes
      ...
763     no
764     no
765     no
766    yes
767     no
Name: Outcome, Length: 768, dtype: object
```

```
In [11]: y = y.map(dict(yes=1, no=0))

In [12]: y
```

Out[12]:
```
0      1
1      0
2      1
3      0
4      1
      ..
763    0
```

jupyter Untitled Last Checkpoint: 25 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

```
Name: Outcome, Length: 768, dtype: int64

In [14]: x = df.drop(['Outcome'], axis=1)
```

DECISION TREE

```
In [16]: from sklearn.tree import DecisionTreeClassifier
         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=1)

In [17]: model = DecisionTreeClassifier()
         model = model.fit(x_train,y_train)
         pred = model.predict(x_test)

In [18]: from sklearn import metrics
         print("Accuracy:",metrics.accuracy_score(y_test, pred)*100)

         Accuracy: 68.83116883116884

In [20]: from sklearn.metrics import classification_report
         print(classification_report(y_test,pred))

                       precision    recall  f1-score   support

                    0       0.73      0.79      0.76       146
                    1       0.59      0.51      0.54        85

             accuracy                           0.69       231
            macro avg       0.66      0.65      0.65       231
         weighted avg       0.68      0.69      0.68       231
```

RANDOM FOREST

```
In [22]: X_train, X_test, y_train, y_test = train_test_split(x, y,test_size=0.2,random_state=0)

In [24]: from sklearn.ensemble import RandomForestClassifier
         rfc = RandomForestClassifier()
         rfc.fit(X_train, y_train)
         r_pred = rfc.predict(X_test)
         rfc.score(X_test, y_test)
```

Out[24]:
```
0.8181818181818182
```

jupyter Untitled Last Checkpoint: 25 minutes ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

RANDOM FOREST

```
In [22]: X_train, X_test, y_train, y_test = train_test_split(x, y,test_size=0.2,random_state=0)

In [24]: from sklearn.ensemble import RandomForestClassifier
         rfc = RandomForestClassifier()
         rfc.fit(X_train, y_train)
         r_pred = rfc.predict(X_test)
         rfc.score(X_test, y_test)
```

Out[24]:
```
0.8181818181818182
```

```
In [26]: from sklearn.metrics import confusion_matrix
         mat = confusion_matrix(y_test, r_pred)
         mat
```

Out[26]:
```
array([[95, 12],
       [16, 31]], dtype=int64)
```

```
In [27]: print("Accuracy:",metrics.accuracy_score(y_test, r_pred)*100)

         Accuracy: 81.81818181818183

In [28]: print(classification_report(y_test,r_pred))

                       precision    recall  f1-score   support

                    0       0.86      0.89      0.87       107
                    1       0.72      0.66      0.69        47

             accuracy                           0.82       154
            macro avg       0.79      0.77      0.78       154
         weighted avg       0.81      0.82      0.82       154

In [ ]:
```