dia

June 28, 2024

[2]:
```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

[1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[3]:
```
dia = pd.read_csv(r"/content/drive/MyDrive/diabetcsv.csv")
```

[4]:
```
x = dia[['preg', 'plas', 'pres', 'skin', 'insu', 'mass', 'pedi', 'age']]
dia['class'] = dia['class'].replace({'tested_negative': 0, 'tested_positive':␣
↪1})
y = dia['class']
```

[5]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
↪random_state=42)
```

[15]:
```
lr = LinearRegression()
lr.fit(x_train, y_train)
y_pred_lr = lr.predict(x_test)
y_pred_lr = [1 if i >= 0.5 else 0 for i in y_pred_lr]
accuracy_lr = accuracy_score(y_test, y_pred_lr)
print(f"Linear Regression Accuracy: {accuracy_lr}")

log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(x_train, y_train)
y_pred_log_reg = log_reg.predict(x_test)
accuracy_log_reg = accuracy_score(y_test, y_pred_log_reg)
print(f"Logistic Regression Accuracy: {accuracy_log_reg}")
```

```
knn = KNeighborsClassifier(n_neighbors=5)
knn.fit(x_train, y_train)
y_pred_knn = knn.predict(x_test)
accuracy_knn = accuracy_score(y_test, y_pred_knn)
print(f"K-Nearest Neighbors Accuracy: {accuracy_knn}")

dt = DecisionTreeClassifier()
dt.fit(x_train, y_train)
y_pred_dt = dt.predict(x_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print(f"Decision Tree Accuracy: {accuracy_dt}")

rf = RandomForestClassifier(n_estimators=100)
rf.fit(x_train, y_train)
y_pred_rf = rf.predict(x_test)
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f"Random Forest Accuracy: {accuracy_rf}")
```

```
Linear Regression Accuracy: 0.7597402597402597
Logistic Regression Accuracy: 0.7467532467532467
K-Nearest Neighbors Accuracy: 0.6623376623376623
Decision Tree Accuracy: 0.7337662337662337
Random Forest Accuracy: 0.7532467532467533
```

[17]:
```
a = int(input("Enter the preg: "))
b = float(input("Enter the plas: "))
c = float(input("Enter the pres: "))
d = float(input("Enter the skin: "))
e = float(input("Enter the insu: "))
f = float(input("Enter the mass: "))
g = float(input("Enter the pedi: "))
h = float(input("Enter the age: "))

input_data = pd.DataFrame([[a, b, c, d, e, f, g, h]],
                          columns=['preg', 'plas', 'pres', 'skin', 'insu',␣
 ↪'mass', 'pedi', 'age'])
```

```
Enter the preg: 23
Enter the plas: 56
Enter the pres: 45
Enter the skin: 32
Enter the insu: 56
Enter the mass: 12
Enter the pedi: 32
Enter the age: 56
```

[18]:
```
pred_lr = lr.predict(input_data)
pred_lr = 1 if pred_lr[0] >= 0.5 else 0
print(f"Linear Regression Prediction: {'tested_positive' if pred_lr == 1 else␣
 ↪'tested_negative'}")

pred_log_reg = log_reg.predict(input_data)
print(f"Logistic Regression Prediction: {'tested_positive' if pred_log_reg[0]␣
 ↪== 1 else 'tested_negative'}")

pred_knn = knn.predict(input_data)
print(f"K-Nearest Neighbors Prediction: {'tested_positive' if pred_knn[0] == 1␣
 ↪else 'tested_negative'}")

pred_dt = dt.predict(input_data)
print(f"Decision Tree Prediction: {'tested_positive' if pred_dt[0] == 1 else␣
 ↪'tested_negative'}")

pred_rf = rf.predict(input_data)
print(f"Random Forest Prediction: {'tested_positive' if pred_rf[0] == 1 else␣
 ↪'tested_negative'}")
```

```
Linear Regression Prediction: tested_positive
Logistic Regression Prediction: tested_positive
K-Nearest Neighbors Prediction: tested_negative
Decision Tree Prediction: tested_negative
Random Forest Prediction: tested_negative
```