Name: Kanwal Ahmed

*MAKE SURE EACH SCREENSHOT SHOWS YOUR NAME SOMEWHERE*

1) Include a screenshot of the code showing how you separated your dataset into target (y) and predictor (X) variables.

```
[ ]: Kanwal Ahmed

[279]: y = data["Outcome"]
       X = data.drop(["Outcome"], axis=1)
```

2) Include a screenshot showing that you normalized/scaled your predictor variables.

```
[ ]: Kanwal Ahmed

[280]: #normalize our data
       from sklearn.preprocessing import StandardScaler
       scaler = StandardScaler()
       x_scaled = scaler.fit_transform(data.drop('Outcome', axis=1))
       x_scaled

[280]: array([[ 0.63994726,  0.84832379,  0.14964075, ...,  0.20401277,
                0.46849198,  1.4259954 ],
              [-0.84488505, -1.12339636, -0.16054575, ..., -0.68442195,
               -0.36506078, -0.19067191],
              [ 1.23388019,  1.94372388, -0.26394125, ..., -1.10325546,
                0.60439732, -0.10558415],
              ...,
              [ 0.3429808 ,  0.00330087,  0.14964075, ..., -0.73518964,
               -0.68519336, -0.27575966],
              [-0.84488505,  0.1597866 , -0.47073225, ..., -0.24020459,
               -0.37110101,  1.17073215],
              [-0.84488505, -0.8730192 ,  0.04624525, ..., -0.20212881,
               -0.47378505, -0.87137393]])
```

3) Include a screenshot of the code that splits your dataset into training and testing sets.

```
[ ]: Kanwal Ahmed

[281]: # define y (the target column)
       y = data['Outcome']

[282]: #split our data into training and testing
       from sklearn.model_selection import train_test_split
       x_train, x_test, y_train, y_test = train_test_split(x_scaled, y , test_size=0.30, random_state=22)
```

4) Include a screenshot of the code that trains your logistic regression model.

```
[ ]: Kanwal Ahmed

[283]: #train the model
       from sklearn.linear_model import LogisticRegression
       lr = LogisticRegression()
       lr.fit(x_train, y_train)
       y_pred = lr.predict(x_test)
```

5) Include a screenshot of your predicted values (y_pred) and actual values (y_test).

```
[ ]: Kanwal Ahmed
```

```
[284]: y_pred
```

```
[284]: array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 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, 0, 0, 1, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 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, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1,
       0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1])
```

```
[285]: y_test
```

```
[285]: 645    0
       767    0
       31     1
       148    0
       59     0
             ..
       575    0
       360    1
       46     0
       454    0
       215    1
       Name: Outcome, Length: 231, dtype: int64
```

6) Include a screenshot of the accuracy score of your model.

```
[ ]: Kanwal Ahmed
```

```
[286]: from sklearn.metrics import accuracy_score
```

```
[287]: accuracy = accuracy_score(y_test, y_pred)
       print(accuracy)
```

```
0.7705627705627706
```

The Screenshot of everything below

jupyter TD1Logistic Regression Kanwal Ahmed Last Checkpoint: yesterday

File Edit View Run Kernel Settings Help Trusted

Code JupyterLab Python [conda env:base] *

```
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

[279]:
```
y = data["Outcome"]
X = data.drop(["Outcome"], axis=1)
```

[280]:
```
#normalize our data
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
x_scaled = scaler.fit_transform(data.drop('Outcome', axis=1))
x_scaled
```

[280]:
```
array([[ 0.63994726,  0.84832379,  0.14964075, ...,  0.20401277,
         0.46849198,  1.4259954 ],
       [-0.84488505, -1.12339636, -0.16054575, ..., -0.68442195,
        -0.36506078, -0.19067191],
       [ 1.23388019,  1.94372388, -0.26394125, ..., -1.10325546,
         0.60439732, -0.10558415],
       ...,
       [ 0.3429808 ,  0.00330087,  0.14964075, ..., -0.73518964,
        -0.68519336, -0.27575966],
       [-0.84488505,  0.1597866 , -0.47073225, ..., -0.24020459,
        -0.37110101,  1.17073215],
       [-0.84488505, -0.8730192 ,  0.04624525, ..., -0.20212881,
        -0.47378505, -0.87137393]])
```

[281]:
```
# define y (the target column)
y = data['Outcome']
```

[282]:
```
#split our data into training and testing
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x_scaled, y , test_size=0.30, random_state=22)
```

[283]:
```
#train the model
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(x_train, y_train)
y_pred = lr.predict(x_test)
```

[284]:
```
y_pred
```

[284]:
```
array([0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       1, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1,
       0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1,
       0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1])
```

[285]:
```
y_test
```

[285]:
```
645    0
767    0
31     1
148    0
59     0
      ..
575    0
360    1
46     0
454    0
215    1
Name: Outcome, Length: 231, dtype: int64
```

[286]:
```
from sklearn.metrics import accuracy_score
```

[287]:
```
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
```

```
0.7705627705627706
```