Lab 3 - Logistic regression

July 21, 2022

[25]:

```
pip install sklearn
```

```
Requirement already satisfied: sklearn in
c:\users\admin\.conda\envs\fdp\lib\site-packages (0.0)
Requirement already satisfied: scikit-learn in
c:\users\admin\.conda\envs\fdp\lib\site-packages (from sklearn) (1.0.2)
Requirement already satisfied: scipy>=1.1.0 in
c:\users\admin\.conda\envs\fdp\lib\site-packages (from scikit-learn->sklearn)
(1.5.2)
Requirement already satisfied: numpy>=1.14.6 in
c:\users\admin\.conda\envs\fdp\lib\site-packages (from scikit-learn->sklearn)
(1.19.2)
Requirement already satisfied: joblib>=0.11 in
c:\users\admin\.conda\envs\fdp\lib\site-packages (from scikit-learn->sklearn)
(1.1.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\admin\.conda\envs\fdp\lib\site-packages (from scikit-learn->sklearn)
(3.1.0)
Note: you may need to restart the kernel to use updated packages.
```

[26]:

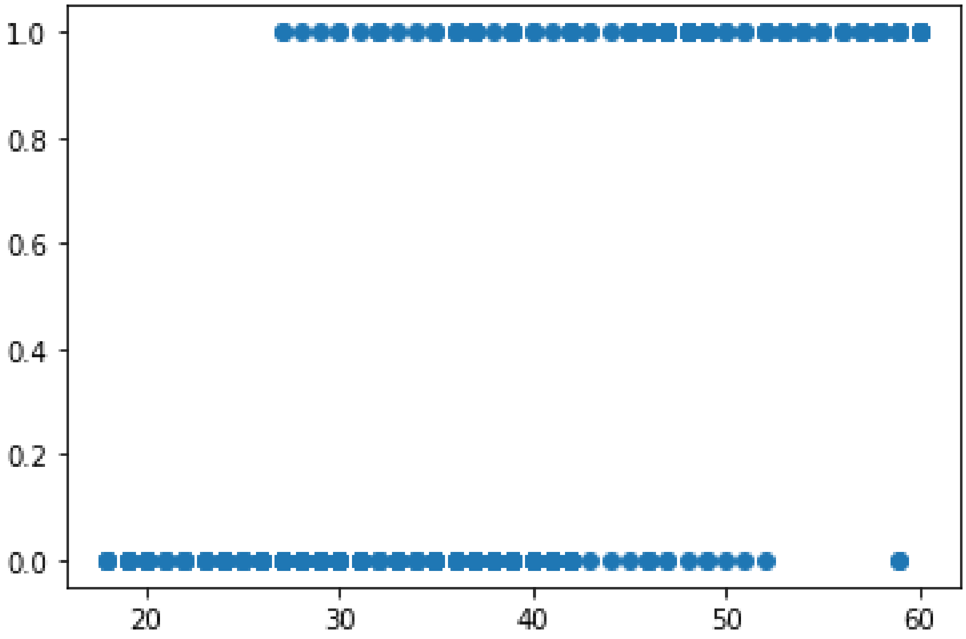
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from math import exp
import pandas as pd
import matplotlib.pyplot as plt
```

[27]:

```
data = pd.read_csv("Social_Network_Ads.csv")
```

[28]:

```
data.head()
# Visualizing the dataset
plt.scatter(data['Age'], data['Purchased'])
plt.show()
```



[29]:
```
# Divide the data to training set and test set
X_train, X_test, y_train, y_test = train_test_split(data['Age'],␣
↪data['Purchased'], test_size=0.20)
```

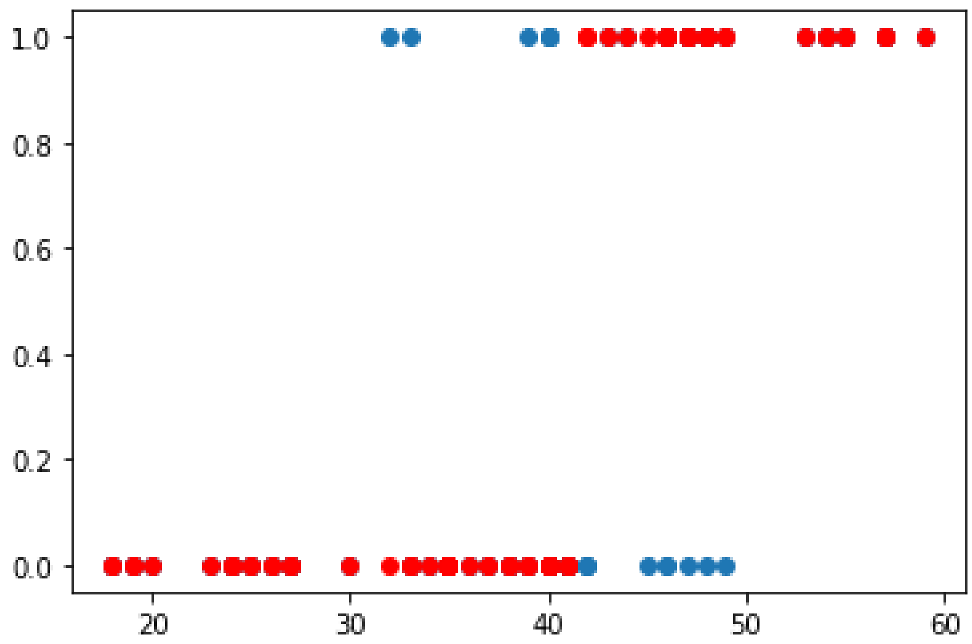
[30]:
```
# Create an instance and fit the model
lr_model = LogisticRegression()
lr_model.fit(X_train.values.reshape(-1, 1), y_train.values.reshape(-1, 1))
```

```
C:\Users\ADMIN\.conda\envs\FDP\lib\site-
packages\sklearn\utils\validation.py:993: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

[30]:
```
LogisticRegression()
```

[31]:
```
# Making predictions
y_pred_sk = lr_model.predict(X_test.values.reshape(-1, 1))
```

[32]:
```
plt.clf()
plt.scatter(X_test, y_test)
plt.scatter(X_test, y_pred_sk, c="red")
plt.show()
```



[33]:

```
# Accuracy
print(f"Accuracy = {lr_model.score(X_test.values.reshape(-1, 1), y_test.values.
 ↪reshape(-1, 1))}")
```

```
Accuracy = 0.8125
```

[ ]: