ML with sklearn

April 14, 2023

[3]:

```
# Name: Emmanuel Asante
# Class: CS 4375
# Assg: Sklearn Python

import pandas as pd

print("Machine Learning with SKLearn - Emmanuel␣
 ↪Asante\n--------------------------------------------------\n")
# 1a) Reading csv - file_name auto.csv
try:
    print("QUESTION 1:\n-------------")
    print("1a) Attempting to Open the file: <Auto.csv>")
    data_frame = pd.read_csv("C:\\Users\\Manny\\Box\\University\\Sem 5 (Spring␣
 ↪2023)\\CS 4375\\Assignments\\SK Learn\\Auto.csv")
    print("File Opened\n")

except:
    print("1b) Outputting the first few rows pf the data")

# 1b) Outputting the first few rows
print("1b) Attempting to Open the file: <Auto.csv>")
print(data_frame.head(), "\n")
data_frame.shape

# 1c) Dimensions of the data
print("1c) Dimensions of the data:\n    Rows: {} \n    Cols: {} ".
 ↪format(len(data_frame), len(data_frame.axes[1])))
```

```
Machine Learning with SKLearn - Emmanuel Asante
--------------------------------------------------

QUESTION 1:
-------------
1a) Attempting to Open the file: <Auto.csv>
File Opened

1b) Attempting to Open the file: <Auto.csv>
    mpg  cylinders  displacement  horsepower  weight  acceleration  year
```

```
0  18.0          8         307.0         130    3504          12.0  70.0  \
1  15.0          8         350.0         165    3693          11.5  70.0
2  18.0          8         318.0         150    3436          11.0  70.0
3  16.0          8         304.0         150    3433          12.0  70.0
4  17.0          8         302.0         140    3449           NaN  70.0

   origin                       name
0       1  chevrolet chevelle malibu
1       1          buick skylark 320
2       1         plymouth satellite
3       1              amc rebel sst
4       1                ford torino

1c) Dimensions of the data:
    Rows: 392
    Cols: 9
```

[4]:

```
# Question 2
print("\nQUESTION 2:\n-------------\n")
print("2a) Describing mpg, weight, and year columns \n")
print (data_frame[ ['mpg', 'weight', 'year']].describe())


print("\n2b) Summary of Description")
print("MPG:  Range = [{}, {}]  Average = {:.2f}".format(data_frame['mpg'].
 ↪min(), data_frame['mpg'].max(), data_frame['mpg'].mean()))
print("Cyl:  Range = [{}, {}]       Average = {:.2f}".
 ↪format(data_frame['cylinders'].min(), data_frame['cylinders'].max(),␣
 ↪data_frame['cylinders'].mean()))
print("Year: Range = [{}, {}] Average = {:.2f}".format(data_frame['year'].
 ↪min(), data_frame['year'].max(), data_frame['year'].mean()))
```

```
QUESTION 2:
-------------

2a) Describing mpg, weight, and year columns

              mpg       weight        year
count  392.000000   392.000000  390.000000
mean    23.445918  2977.584184   76.010256
std      7.805007   849.402560    3.668093
min      9.000000  1613.000000   70.000000
25%     17.000000  2225.250000   73.000000
50%     22.750000  2803.500000   76.000000
75%     29.000000  3614.750000   79.000000
max     46.600000  5140.000000   82.000000
```

```
2b) Summary of Description
MPG:  Range = [9.0, 46.6]  Average = 23.45
Cyl:  Range = [3, 8]       Average = 5.47
Year: Range = [70.0, 82.0] Average = 76.01
```

[5]:

```python
# Question 3
print("\nQUESTION 3:\n-------------\n")
print("3a) Data Types of each column")
print (data_frame.dtypes, "\n")

print("3b) Changing cylinder to categorical")
data_frame['origin'] = data_frame['origin'].astype('category')
print ("changed")


print("3c) Changing cylinder to categorical witouht cat.codes")
data_frame['cylinders'] = data_frame['cylinders'].astype('category')
print ("changed")
print (data_frame.dtypes)
```

```

QUESTION 3:
-------------

3a) Data Types of each column
mpg             float64
cylinders         int64
displacement    float64
horsepower        int64
weight            int64
acceleration    float64
year            float64
origin            int64
name             object
dtype: object

3b) Changing cylinder to categorical
changed
3c) Changing cylinder to categorical witouht cat.codes
changed
mpg              float64
cylinders       category
displacement     float64
horsepower         int64
weight             int64
acceleration     float64
year             float64
origin          category
```

```
name                 object
dtype: object
```

[6]:

```
# Question 4
print("\nQUESTION 4:\n-------------\n")
print("4a) Dealing wiht NA's")
# data_frame.dropna(inplace= True)
# print("4b) New dimensions of the data:\n    Rows: {} \n    Cols: {} ".
 ↪format(len(new_data_frame), len(data_frame.axes[1])))
```

```
QUESTION 4:
-------------

4a) Dealing wiht NA's
```

[7]:

```
# Question 5
print("\nQUESTION 5:\n-------------\n")
print("5a) Creating a new column mpg_high which is a categorical version of␣
 ↪mpg")
pd.set_option('display.max_rows', 10)

mpg_high = data_frame['mpg'].copy()
mpg_avg = mpg_high.mean()
print (mpg_avg)

for i in range(0, len(mpg_high)):
    val = mpg_high.iloc[i]
    if val > mpg_avg:
        mpg_high.at[i] = 1
    else:
        mpg_high.at[i] = 0

mpg_high = mpg_high.astype('category')

data_frame['mpg_high'] = mpg_high
data_frame

print("5b) Removing mpg and name")
data_frame = data_frame.drop(['mpg', 'name'], axis = 1)
data_frame
```

```
QUESTION 5:
-------------

5a) Creating a new column mpg_high which is a categorical version of mpg
23.445918367346938
5b) Removing mpg and name
```

[7]:

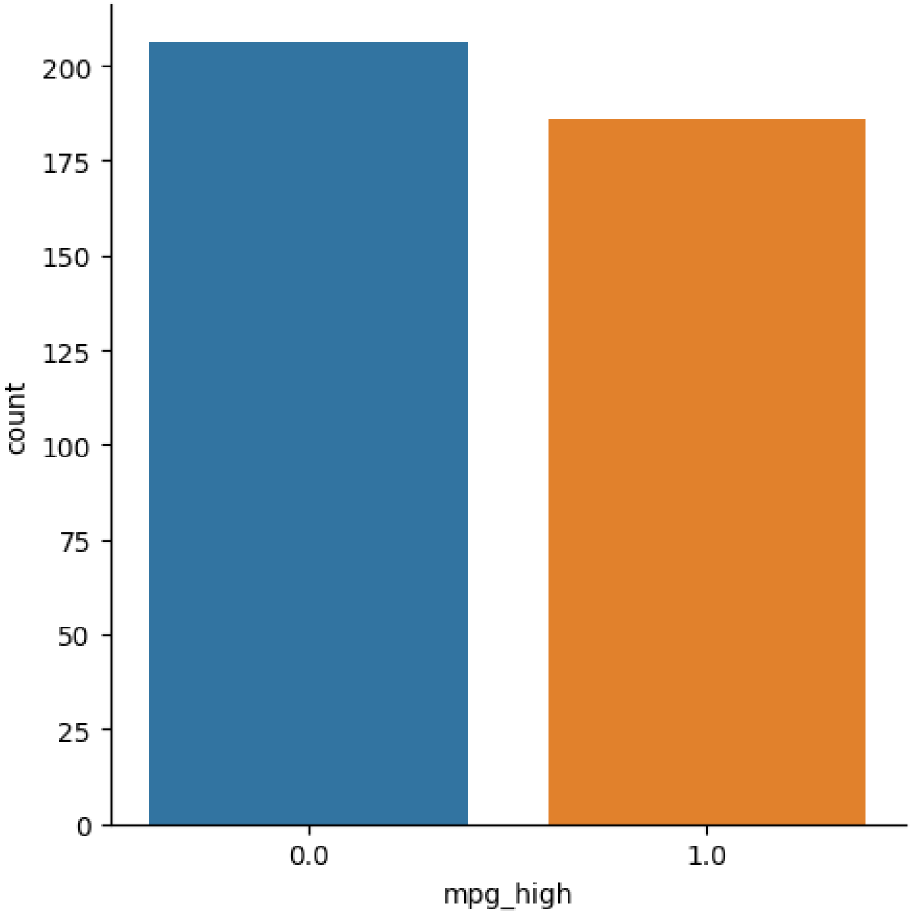
```
    cylinders  displacement  horsepower  weight  acceleration  year origin
0           8         307.0         130    3504          12.0  70.0      1  \
1           8         350.0         165    3693          11.5  70.0      1
2           8         318.0         150    3436          11.0  70.0      1
3           8         304.0         150    3433          12.0  70.0      1
4           8         302.0         140    3449           NaN  70.0      1
..        …           …         …     …           …   …    …
387         4         140.0          86    2790          15.6  82.0      1
388         4          97.0          52    2130          24.6  82.0      2
389         4         135.0          84    2295          11.6  82.0      1
390         4         120.0          79    2625          18.6  82.0      1
391         4         119.0          82    2720          19.4  82.0      1

    mpg_high
0        0.0
1        0.0
2        0.0
3        0.0
4        0.0
..       …
387      1.0
388      1.0
389      1.0
390      1.0
391      1.0

[392 rows x 8 columns]
```

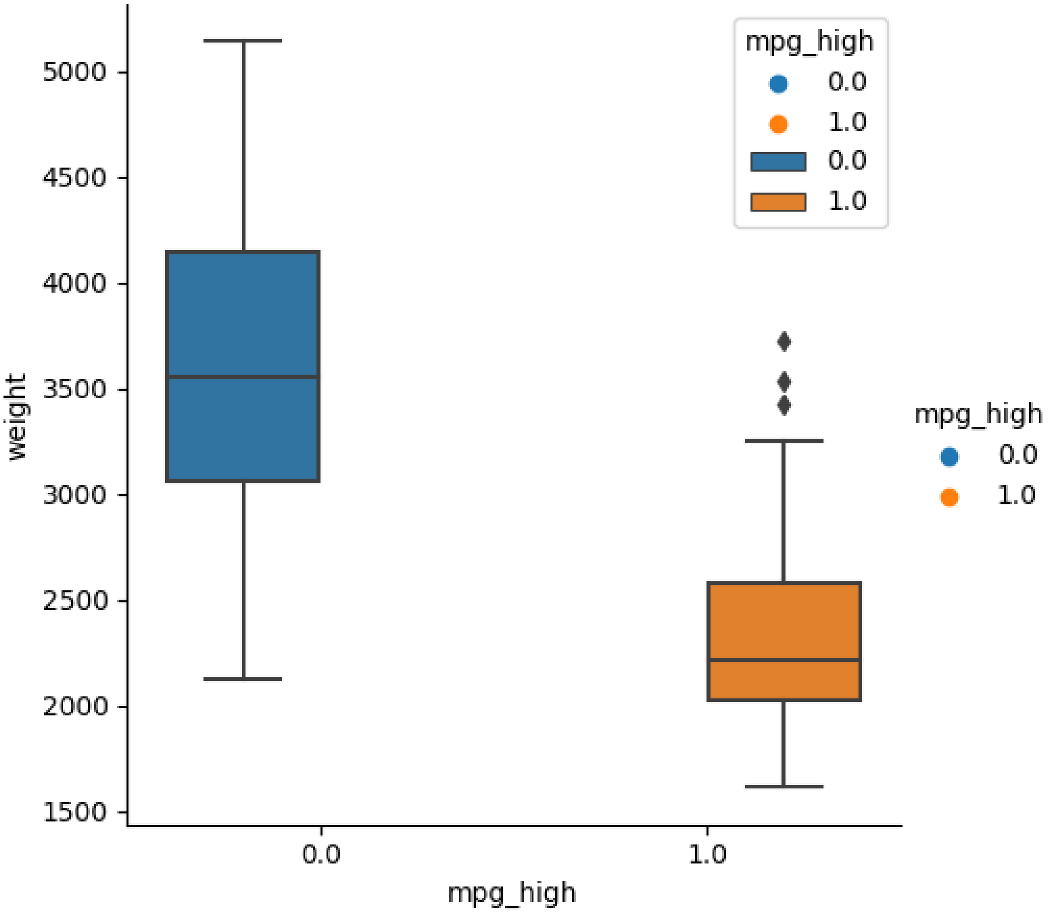
[8]:

```python
# Question 6
print("\nQUESTION 6:\n-------------\n")
import seaborn as sns

print("6a)Seaborn catplot on mpg_high")
sns.catplot(x = 'mpg_high', data=data_frame, kind='count')

print("6b)Seaborn catplot on mpg_high")
sns.relplot(x='horsepower', y = 'weight', data=data_frame, hue='mpg_high')

print("6c)Seaborn boxplot on mpg_high")
sns.boxplot(x='mpg_high', y = 'weight', data=data_frame, hue='mpg_high')
```

```
QUESTION 6:
-------------

6a)Seaborn catplot on mpg_high
6b)Seaborn catplot on mpg_high
6c)Seaborn boxplot on mpg_high
```

```
[8]: <Axes: xlabel='mpg_high', ylabel='weight'>
```





```
[9]: # Question 7
     print("\nQUESTION 7:\n-------------\n")
     print("7a) Dividing Data set in test and train data")
     import random

     train_size = int (len(data_frame)*0.8)
     print("\n     - Train Size: ", train_size)
     print("     - Test Size:   ", len(data_frame)-train_size)

     print("\n7b) Setting Seed")
     random_seed = 1234
     print ("Ransom seed:",random_seed)
     data_frame.sample(frac=1)


     train_data = data_frame.iloc[0:train_size, data_frame.columns != "mpg_high"]
     test_data = data_frame.iloc[train_size:len(data_frame), data_frame.columns !=␣
       ↪"mpg_high"]
```

```
print("\n7c) Train and Test Data with mpg_high removed\n")
print("Train Data")
print("----------\n")
print(train_data)

print("Test Data")
print("---------\n")
print(test_data)

print("\n7d) Dimensions of the Test and Train Data\n")
print("    - Dimensions of Autos Data Frame:",data_frame.shape)
print("    - Dimensions of Train Data Frame:",train_data.shape)
print("    - Dimensions of Test  Data Frame:",test_data.shape)
```

```
QUESTION 7:
-------------

7a) Dividing Data set in test and train data

    - Train Size:  313
    - Test Size:   79

7b) Setting Seed
Ransom seed: 1234

7c) Train and Test Data with mpg_high removed

Train Data
----------

    cylinders  displacement  horsepower  weight  acceleration  year origin
0           8         307.0         130    3504          12.0  70.0      1
1           8         350.0         165    3693          11.5  70.0      1
2           8         318.0         150    3436          11.0  70.0      1
3           8         304.0         150    3433          12.0  70.0      1
4           8         302.0         140    3449           NaN  70.0      1
..        …           …         …     …           …   …    …
308         4          89.0          60    1968          18.8  80.0      3
309         4          98.0          70    2120          15.5  80.0      1
310         4          86.0          65    2019          16.4  80.0      3
311         4         151.0          90    2678          16.5  80.0      1
312         4         140.0          88    2870          18.1  80.0      1

[313 rows x 7 columns]
Test Data
---------
```

```
    cylinders  displacement  horsepower  weight  acceleration  year origin
313         4         151.0          90    3003          20.1  80.0      1
314         6         225.0          90    3381          18.7  80.0      1
315         4          97.0          78    2188          15.8  80.0      2
316         4         134.0          90    2711          15.5  80.0      3
317         4         120.0          75    2542          17.5  80.0      3
..        …           …         …     …            …   …    …
387         4         140.0          86    2790          15.6  82.0      1
388         4          97.0          52    2130          24.6  82.0      2
389         4         135.0          84    2295          11.6  82.0      1
390         4         120.0          79    2625          18.6  82.0      1
391         4         119.0          82    2720          19.4  82.0      1

[79 rows x 7 columns]

7d) Dimensions of the Test and Train Data

    - Dimensions of Autos Data Frame: (392, 8)
    - Dimensions of Train Data Frame: (313, 7)
    - Dimensions of Test  Data Frame: (79, 7)
```

[10]:

```
# Question 8
print("\nQUESTION 8:\n-------------\n")
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

print("8a) Dividing Data set in test and train data")

# fit the model with data
Y = 'horsepower'

data_frame.drop (Y, axis=1)

X_train, X_test, Y_train, Y_test = train_test_split(data_frame.drop(Y, axis=1),␣
 ↪data_frame[Y], test_size = 0.2)
print(X_train)
print(Y_train)

LogReg = LogisticRegression()
# LogReg.fit(X_train, Y_train)
```

```
QUESTION 8:
-------------

8a) Dividing Data set in test and train data
    cylinders  displacement  weight  acceleration  year origin mpg_high
```

```
116         4          68.0    1867          19.5  73.0      2      1.0
248         8         318.0    3735          13.2  78.0      1      0.0
35          6         250.0    3302          15.5  71.0      1      0.0
245         4          85.0    2070          18.6  78.0      3      1.0
146         4         116.0    2246          14.0  74.0      2      1.0
..        …           …     …           …   …    …      …
345         4          91.0    1985          16.0  81.0      3      1.0
318         4         119.0    2434          15.0  80.0      3      1.0
33          6         225.0    3439          15.5  71.0      1      0.0
88          8         318.0    3777          12.5  73.0      1      0.0
8           8         455.0    4425          10.0  70.0      1      0.0

[313 rows x 7 columns]
116     49
248    140
35      88
245     70
146     75
      …
345     68
318     92
33     105
88     150
8      225
Name: horsepower, Length: 313, dtype: int64
```

[ ]: