Module 1 Homework

ISE-529 Predictive Analytics

1. For this problem, we will be using the file "Cars Data.csv"

a. Load the file Cars Data.csv into a dataframe named cars and display the first 10 rows of the dataframe.

In [117…
```
import pandas;

cars = pandas.read_csv(filepath_or_buffer = "Cars Data.csv", header = 3);
print(cars.head(10));
```

```
    Make                    Model DriveTrain  Origin    Type  Cylinders  \
0  Acura             3.5 RL 4dr      Front    Asia   Sedan        6.0
1  Acura  3.5 RL w/Navigation 4dr      Front    Asia   Sedan        6.0
2  Acura                     MDX        All    Asia     SUV        6.0
3  Acura   NSX coupe 2dr manual S       Rear    Asia  Sports        6.0
4  Acura           RSX Type S 2dr      Front    Asia   Sedan        4.0
5  Acura                  TL 4dr      Front    Asia   Sedan        6.0
6  Acura                 TSX 4dr      Front    Asia   Sedan        4.0
7   Audi              A4 1.8T 4dr      Front  Europe   Sedan        4.0
8   Audi               A4 3.0 4dr      Front  Europe   Sedan        6.0
9   Audi   A4 3.0 convertible 2dr      Front  Europe   Sedan        6.0

   Engine Size (L)  Horsepower   Invoice  Length (IN)  MPG (City)  \
0              3.5         225  $39,014           197          18
1              3.5         225  $41,100           197          18
2              3.5         265  $33,337           189          17
3              3.2         290  $79,978           174          17
4              2.0         200  $21,761           172          24
5              3.2         270  $30,299           186          20
6              2.4         200  $24,647           183          22
7              1.8         170  $23,508           179          22
8              3.0         220  $28,846           179          20
9              3.0         220  $38,325           180          20

   MPG (Highway)      MSRP  Weight (LBS)  Wheelbase (IN)
0             24  $43,755          3880             115
1             24  $46,100          3893             115
2             23  $36,945          4451             106
3             24  $89,765          3153             100
4             31  $23,820          2778             101
5             28  $33,195          3575             108
6             29  $26,990          3230             105
7             31  $25,940          3252             104
8             28  $31,840          3462             104
9             27  $42,490          3814             105
```

b. Use the describe() function to produce a numerical summary of the variables in the dataset. (10 points)

In [118…
```
print(cars.describe());
```

```
        Cylinders  Engine Size (L)  Horsepower  Length (IN)  MPG (City)  \
count  426.000000       428.000000  428.000000   428.000000  428.000000
mean     5.807512         3.196729  215.885514   186.362150   20.060748
std      1.558443         1.108595   71.836032    14.357991    5.238218
min      3.000000         1.300000   73.000000   143.000000   10.000000
25%      4.000000         2.375000  165.000000   178.000000   17.000000
50%      6.000000         3.000000  210.000000   187.000000   19.000000
75%      6.000000         3.900000  255.000000   194.000000   21.250000
max     12.000000         8.300000  500.000000   238.000000   60.000000

       MPG (Highway)  Weight (LBS)  Wheelbase (IN)
count     428.000000    428.000000      428.000000
mean       26.843458   3577.953271      108.154206
std         5.741201    758.983215        8.311813
min        12.000000   1850.000000       89.000000
25%        24.000000   3104.000000      103.000000
50%        26.000000   3474.500000      107.000000
75%        29.000000   3977.750000      112.000000
max        66.000000   7190.000000      144.000000
```

c. Use the groupby() and size() functions to create a table of the number of observations in the dataset of each car Make (e.g., Acura, Audi, etc.) (15 points)

In [119…
```
print(cars.groupby("Make").size());
```

```
Make
Acura             7
Audi             19
BMW              20
Buick             9
Cadillac          8
Chevrolet        27
Chrysler         15
Dodge            13
Ford             23
GMC               8
Honda            17
Hummer            1
Hyundai          12
Infiniti          8
Isuzu             2
Jaguar           12
Jeep              3
Kia              11
Land Rover        3
Lexus            11
Lincoln           9
MINI              2
Mazda            11
Mercedes-Benz    26
Mercury           9
Mitsubishi       13
Nissan           17
Oldsmobile        3
Pontiac          11
Porsche           7
Saab              7
Saturn            8
Scion             2
Subaru           11
Suzuki            8
Toyota           28
Volkswagen       15
Volvo            12
dtype: int64
```

d. Use the corr() function to create a correlation matrix of the numeric attributes in the cars dataset. Use the "pearson" correlation coefficient algorithm (15 points)

In [120…
```
print(cars.corr(method = "pearson"));
```

```
                Cylinders  Engine Size (L)  Horsepower  Length (IN)  \
Cylinders        1.000000         0.908002    0.810341     0.547783
Engine Size (L)  0.908002         1.000000    0.787435     0.637448
Horsepower       0.810341         0.787435    1.000000     0.381554
Length (IN)      0.547783         0.637448    0.381554     1.000000
MPG (City)      -0.684402        -0.709471   -0.676699    -0.501526
MPG (Highway)   -0.676100        -0.717302   -0.647195    -0.466092
Weight (LBS)     0.742209         0.807867    0.630796     0.690021
Wheelbase (IN)   0.546730         0.636517    0.387398     0.889195

                MPG (City)  MPG (Highway)  Weight (LBS)  Wheelbase (IN)
Cylinders        -0.684402      -0.676100      0.742209        0.546730
Engine Size (L)  -0.709471      -0.717302      0.807867        0.636517
Horsepower       -0.676699      -0.647195      0.630796        0.387398
Length (IN)      -0.501526      -0.466092      0.690021        0.889195
MPG (City)        1.000000       0.941021     -0.737966       -0.507284
MPG (Highway)     0.941021       1.000000     -0.790989       -0.524661
Weight (LBS)     -0.737966      -0.790989      1.000000        0.760703
Wheelbase (IN)   -0.507284      -0.524661      0.760703        1.000000
```

e. Add a new attribute to the dataframe called HP_Groups which has one of the following values:

- "Low": Horsepower is <= 200
- "Medium": Horsepower is > 200 and <= 300
- "High": Horsepower is > 300

Hint: create a function that takes a number and return one of the three bins and then use the apply method

In [121…
```
hpList = [];
for index, row in cars.iterrows():
    temp = "";
    if row["Horsepower"] <= 200:
        temp = "Low";
    elif row["Horsepower"] > 200 and row["Horsepower"] <= 300:
        temp = "Medium";
    else:
        temp = "High";
    hpList.append(temp);

cars["HP_Groups"] = hpList;
print(cars);
```

	Make	Model	DriveTrain	Origin	Type	Cylinders
0	Acura	3.5 RL 4dr	Front	Asia	Sedan	6.0
1	Acura	3.5 RL w/Navigation 4dr	Front	Asia	Sedan	6.0
2	Acura	MDX	All	Asia	SUV	6.0
3	Acura	NSX coupe 2dr manual S	Rear	Asia	Sports	6.0
4	Acura	RSX Type S 2dr	Front	Asia	Sedan	4.0
..	...	...	...	...	...	...
423	Volvo	S80 2.9 4dr	Front	Europe	Sedan	6.0
424	Volvo	S80 T6 4dr	Front	Europe	Sedan	6.0
425	Volvo	V40	Front	Europe	Wagon	4.0
426	Volvo	XC70	All	Europe	Wagon	5.0
427	Volvo	XC90 T6	All	Europe	SUV	6.0

	Engine Size (L)	Horsepower	Invoice	Length (IN)	MPG (City)
0	3.5	225	$39,014	197	18
1	3.5	225	$41,100	197	18
2	3.5	265	$33,337	189	17
3	3.2	290	$79,978	174	17
4	2.0	200	$21,761	172	24
..	...	...	...	...	...
423	2.9	208	$35,542	190	20
424	2.9	268	$42,573	190	19
425	1.9	170	$24,641	180	22
426	2.5	208	$33,112	186	20
427	2.9	268	$38,851	189	15

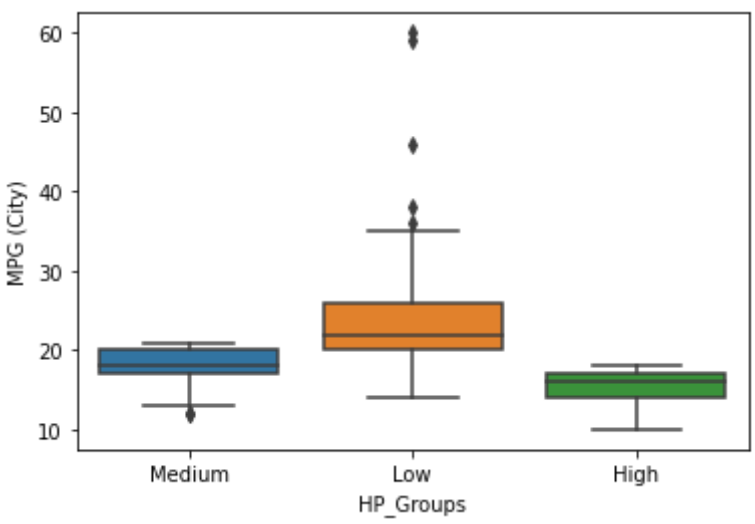
	MPG (Highway)	MSRP	Weight (LBS)	Wheelbase (IN)	HP_Groups
0	24	$43,755	3880	115	Medium
1	24	$46,100	3893	115	Medium
2	23	$36,945	4451	106	Medium
3	24	$89,765	3153	100	Medium
4	31	$23,820	2778	101	Low
..	...	...	...	...	...
423	28	$37,730	3576	110	Medium
424	26	$45,210	3653	110	Medium
425	29	$26,135	2822	101	Low
426	27	$35,145	3823	109	Medium
427	20	$41,250	4638	113	Medium

[428 rows x 16 columns]

f. Using a Seaborn, create side-by-side boxplots showing the MPG (City) for each of the three HP_Groups

In [122…
```
import seaborn;

seaborn.boxplot(x = cars["HP_Groups"], y = cars["MPG (City)"]);
```



2. Function Definition

A common metric of health is the Body Mass Index (BMI), which is calculated simply as the weight in kilograms divided by the height in meters squared (BMI = kg/m2)

a. Create a function called calculate_bmi() that takes height in inches and weight in pounds as parameters and returns the BMI. Use the conversions inches * 0.025 = m and pounds * 0.453592 = kg. Test the function with a height of 72" and a weight of 190 lbs. (15 points)

In [123…
```
def calculate_bmi(inches, pounds):
    kg = pounds * 0.453592;
    m = inches * 0.025;
    return kg / pow(m, 2);

print(calculate_bmi(72, 190));
```

```
26.59953086419753
```

b. General standard categories for BMI are given by:

- BMI < 18.5: Underweight
- BMI 18.5 - 24.9: Normal weight
- BMI 25 - 29.9: Overweight
- BMI 30 or greater: Obese

Create a function called determine_weight_category that takes height in inches and weight in pounds as parameters and returns the category the person falls into. Test your function with the following values: (15 points)

- Height 60" / Weight 160 lbs
- Height 68" / Weight 160 lbs
- Height 72" / Weight 160 lbs

In [124…
```
def determine_weight_category(inches, pounds):
    bmi = calculate_bmi(inches, pounds);
    print(bmi);
    if bmi < 18.5:
        return "Underweight";
    elif bmi >= 18.5 and bmi < 25:
        return "Normal weight";
    elif bmi >= 25 and bmi < 30:
        return "Overweight";
    else:
        return "Obese";

print(determine_weight_category(60, 160));
print(determine_weight_category(68, 160));
print(determine_weight_category(72, 160));
```

```
32.25543111111111
Obese
25.112359861591692
Overweight
22.399604938271604
Normal weight
```

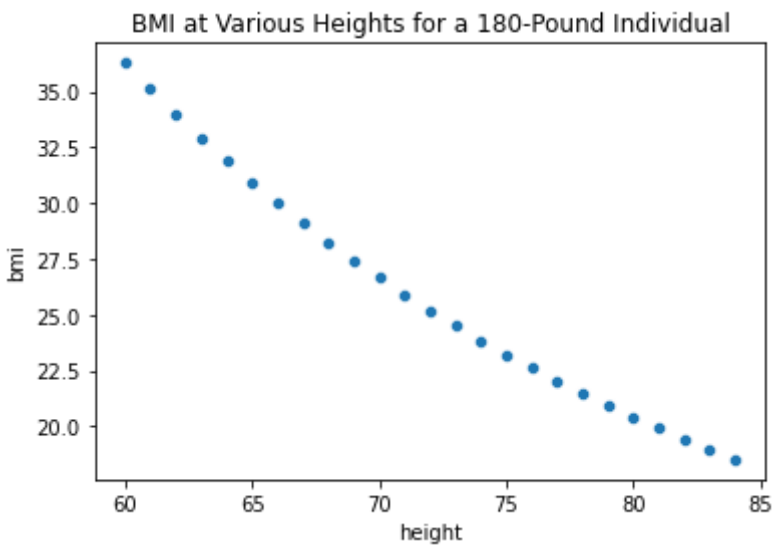
c. You want to create a plot of the BMI for a 180-pound individual at various heights from 60" to 84" (in one-inch increments). Create a vector 'heights' to hold the various heights from 60 to 84 and write a for-loop to call your calculate_bmi() function for each value of heights. Then, use these two vectors to create a scatter plot showing the relationship. Give the chart a title of 'BMI at Various Heights for a 180-Pound Individual'. For full credit, label the axes and give the chart a title.

In [125…
```
heights = [];
bmiList = [];

for a in range(60, 85):
    heights.append(a);
for h in heights:
    bmiList.append(calculate_bmi(h, 180));

df = pandas.DataFrame(
    {
        "height" : heights,
        "bmi" : bmiList
    }
)

seaborn.scatterplot(data = df, x = "height", y = "bmi").set(title = "BMI at Various Heights for a 180-Pound Individual");
```



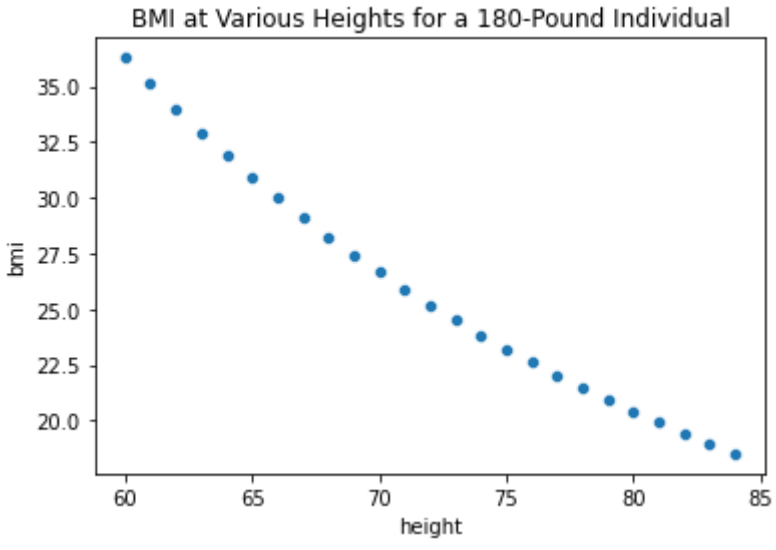
d. Repeat part c from above but without using a loop

In [126…
```
def calculate_bmi_2(inches):
    kg = 180 * 0.453592;
    m = inches * 0.025;
    return kg / pow(m, 2);

series_heights = pandas.Series(heights);
series_bmi = series_heights.apply(calculate_bmi_2);

df2 = pandas.DataFrame(
    {
        "height" : series_heights,
        "bmi" : series_bmi
    }
)

seaborn.scatterplot(data = df2, x = "height", y = "bmi").set(title = "BMI at Various Heights for a 180-Pound Individual");
```



3) Demographics Dataset Analysis

a. For this problem, load the file "demographics.csv" into a DataFrame called "demos". Uset the "type" function to verify that it is a dataframe

In [127…
```
demos = pandas.read_csv(filepath_or_buffer = "demographics.csv");
print(type(demos));
print(demos);
```

```
<class 'pandas.core.frame.DataFrame'>
     CONT              NAME Region  Population   Urban Pop Percentage   \
0      91           BAHAMAS    AMR      323063                   0.90
1      91            BELIZE    AMR      269736                   0.49
2      91            CANADA    AMR    32268243                   0.81
3      91        COSTA RICA    AMR     4327228                   0.62
4      91              CUBA    AMR    11269400                   0.76
..    ...               ...    ...         ...                    ...
192    96              NIUE    WPR        1445                   0.37
193    96  PAPUA NEW GUINEA    WPR     5887138                   0.13
194    96             TONGA    WPR      102311                   0.34
195    96            TUVALU    WPR       10441                   0.57
196    96             SAMOA    WPR      184984                   0.23

      AdultLiteracyPct   MaleSchoolPct  FemaleSchoolPct      GNI  \
0                  NaN           0.85             0.88  16140.0
1                 0.77           0.98             1.00   6510.0
2                  NaN           1.00             1.00  30660.0
3                 0.96           0.90             0.91   9530.0
4                 1.00           0.96             0.95      NaN
..                 ...            ...              ...      ...
192                NaN           0.99             0.98      NaN
193               0.57           0.79             0.69   2300.0
194               0.99           1.00             1.00   7220.0
195                NaN            NaN              NaN      NaN
196               0.99           0.99             0.96   5670.0

     PopPovertyPct
0              NaN
1              NaN
2              NaN
3             0.02
4              NaN
..             ...
192            NaN
193            NaN
194            NaN
195            NaN
196            NaN

[197 rows x 10 columns]
```

b. For each continent, summarize the average (mean) city size

In [128…
```
print(demos.groupby("CONT")[["Population"]].mean());
```

```
        Population
CONT
91    3.182618e+07
92    1.984816e+07
93    1.695122e+07
94    1.707178e+07
95    8.953362e+07
96    4.030014e+06
```

c. Find the size of the largest country in each continent

In [129…
```
print(demos.groupby("CONT")[["NAME", "Population"]].max());
```

```
                NAME  Population
CONT
91     UNITED STATES   298212895
92         VENEZUELA   186404913
93    UNITED KINGDOM    82689210
94          ZIMBABWE   131529669
95             YEMEN  1323344591
96           VANUATU    20155129
```

d. What is the average percentage of males and females that attend school in each continent?

In [130…
```
print(demos.groupby("CONT")[["MaleSchoolPct", "FemaleSchoolPct"]].mean());
```

```
      MaleSchoolPct  FemaleSchoolPct
CONT
91         0.932143         0.931429
92         0.930588         0.919412
93         0.942381         0.939286
94         0.734444         0.676222
95         0.885429         0.856000
96         0.933077         0.923077
```

e. What is the average percentage of females that attend school in each continent. Sort the list from highest to lowest.

In [131…
```
print(demos.groupby("CONT")[["FemaleSchoolPct"]].mean().sort_values(["FemaleSchoolPct"], ascending = False));
```

```
      FemaleSchoolPct
CONT
93           0.939286
91           0.931429
96           0.923077
92           0.919412
95           0.856000
94           0.676222
```