#####################################

# Week 3: Exploratory Data Analysis #

#####################################

# importing libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from scipy import stats

#######################

# Reading the Dataset #

#######################

# read url

path = "https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass/DA0101EN/automobileEDA.csv"

df = pd.read\_csv(path)

df.head()

# listing data type

df.dtypes

##############

# Question 1 #

##############

# The data type of column peak-rpm is float type

# the following prints a correlation matrix

df.corr()

##############

# Question 2 #

##############

# now printing corr matrix for select columns of interest

df[['bore', 'stroke', 'compression-ratio', 'horsepower']].corr()

##################################

# Continuous numerical variables #

##################################

# Positive linear relationship

# using engine size as a predictor of price

#sns.regplot(x="engine-size", y="price", data=df)

#plt.ylim(0,)

# plt.show()

# obtaining metrics on fit, such as strenght of correlation

df[["engine-size", "price"]].corr()

# Negative linear relationship

# likewise regressing price on highway mpg

#sns.regplot(x="highway-mpg", y="price", data=df)

# plt.show()

df[['highway-mpg', 'price']].corr()

# here we see a negative correlation

# Weak Linear Relationship

#sns.regplot(x="peak-rpm", y="price", data=df)

# plt.show()

df[['peak-rpm', 'price']].corr()

# weak inconclusive relation

#################

# Question 3 a) #

#################

df[['stroke', 'price']].corr()

# the correlation between stroke and price is 0.08

# on the weaker side

#################

# Question 3 b) #

#################

# given the above results, we can expect a weak linear positive result

#sns.regplot(x="stroke", y="price", data=df)

#plt.show()

# as expected, we find a weak linear relation

#########################

# Categorical variables #

#########################

# exploring the boxplot

# price vs body style boxplot

# sns.boxplot(x="body-style", y="price", data=df)

# plt.show()

# price vs drive wheels boxplot

# sns.boxplot(x="drive-wheels", y="price", data=df)

# plt.show()

####################################

# Descriptive Statistical Analysis #

####################################

# running some descriptive stats

df.describe

# printing some descriptives

df.describe(include=['object'])

################

# Value Counts #

################

# basic frequency tabs

df['drive-wheels'].value\_counts()

# writing frequency tabs to dataframe

df['drive-wheels'].value\_counts().to\_frame()

# saving the written dataframe explicitly

drive\_wheels\_counts = df['drive-wheels'].value\_counts().to\_frame()

# renaming columns for convenience

drive\_wheels\_counts.rename(columns={'drive-wheels': 'value\_counts'}, inplace=True)

drive\_wheels\_counts

# renaming index

drive\_wheels\_counts.index.name = 'drive\_wheels'

drive\_wheels\_counts

# similarly for engine-location we have

engine\_loc\_counts = df['engine-location'].value\_counts().to\_frame()

# renaming columns for clarity

engine\_loc\_counts.rename(columns={'engine-location': 'value\_counts'}, inplace=True)

# renaming index

engine\_loc\_counts.index.name = 'engine-location'

engine\_loc\_counts.head(10)

######################

# Basics of Grouping #

######################

# obtaining distinct values

df['drive-wheels'].unique()

# creating a subset of dataframe

df\_group\_one = df[['drive-wheels', 'body-style', 'price']]

df\_group\_one

# grouping results

df\_group\_one = df\_group\_one.groupby(['drive-wheels'], as\_index=False).mean()

df\_group\_one

# grouping by multiple variables

df\_gptest = df[['drive-wheels', 'body-style', 'price']]

grouped\_test1 = df\_gptest.groupby(['drive-wheels','body-style'], as\_index=False).mean()

grouped\_test1

# creating a pivot table of means

grouped\_pivot = grouped\_test1.pivot(index='drive-wheels', columns='body-style')

grouped\_pivot

# filling missing cells in pivot with 0's

grouped\_pivot = grouped\_pivot.fillna(0)

grouped\_pivot

##############

# Question 4 #

##############

# creating subset

df\_group\_2 = df[['drive-wheels', 'body-style', 'price']]

# grouping by body style to determine price

df\_group\_2 = df\_group\_2.groupby(['body-style'], as\_index=False).mean()

df\_group\_2

# Variables: Drive Wheels and Body Style vs Price

# using the grouped results

#plt.pcolor(grouped\_pivot, cmap='RdBu')

#plt.colorbar()

#plt.show()

# now setting up intricacies of our plot

fig, ax = plt.subplots()

im = ax.pcolor(grouped\_pivot, cmap='RdBu')

# label names

row\_labels = grouped\_pivot.columns.levels[1]

col\_labels = grouped\_pivot.index

# centering labels

ax.set\_xticks(np.arange(grouped\_pivot.shape[1]) + 0.5, minor=False)

ax.set\_yticks(np.arange(grouped\_pivot.shape[0]) + 0.5, minor=False)

# insert labels

ax.set\_xticklabels(row\_labels, minor=False)

ax.set\_yticklabels(row\_labels, minor=False)

# rotate label if too long

plt.xticks(rotation=90)

fig.colorbar(im)

# plt.show()

#############################

# Correlation and Causation #

#############################

# basic overall correlation

df.corr()

# Wheel-base vs Price

# calculating pearson correlation coefficient

pearson\_coef, p\_value = stats.pearsonr(df['wheel-base'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Horsepower vs Price

pearson\_coef, p\_value = stats.pearsonr(df['horsepower'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Length vs Price

pearson\_coef, p\_value = stats.pearsonr(df['length'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Width vs Price

pearson\_coef, p\_value = stats.pearsonr(df['width'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Curb-weight vs Price

pearson\_coef, p\_value = stats.pearsonr(df['curb-weight'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Engine-size vs Price

pearson\_coef, p\_value = stats.pearsonr(df['engine-size'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Bore vs Price

pearson\_coef, p\_value = stats.pearsonr(df['bore'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# City-mpg vs Price

pearson\_coef, p\_value = stats.pearsonr(df['city-mpg'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

# Highway-mpg vs Price

pearson\_coef, p\_value = stats.pearsonr(df['highway-mpg'], df['price'])

# print("The Pearson Correlation Coefficient is", pearson\_coef, " with a P-value of P =", p\_value)

#########

# ANOVA #

#########

# ANOVA: Analysis of Variance

# Drive wheels

# anova compares same variable for different groups

grouped\_test2=df\_gptest[['drive-wheels', 'price']].groupby(['drive-wheels'])

grouped\_test2.head(2)

df\_gptest

# getting group averaged values for 4wd

grouped\_test2.get\_group('4wd')['price']

# ANOVA testing across drive wheel types

f\_val, p\_val = stats.f\_oneway(grouped\_test2.get\_group('fwd')['price'], grouped\_test2.get\_group('rwd')['price'], grouped\_test2.get\_group('4wd')['price'])

# print("ANOVA results: F=", f\_val, ", P=", p\_val)

# conducting ANOVA for two groups at a time: fwd and rwd

f\_val, p\_val = stats.f\_oneway(grouped\_test2.get\_group('fwd')['price'], grouped\_test2.get\_group('rwd')['price'])

# print("ANOVA results: F=", f\_val, ", P=", p\_val)

# conducting ANOVA for: 4wd and rwd

f\_val, p\_val = stats.f\_oneway(grouped\_test2.get\_group('rwd')['price'], grouped\_test2.get\_group('4wd')['price'])

# print("ANOVA results: F=", f\_val, ", P=", p\_val)

# conducting ANOVA for: 4wd and fwd

f\_val, p\_val = stats.f\_oneway(grouped\_test2.get\_group('fwd')['price'], grouped\_test2.get\_group('4wd')['price'])

# print("ANOVA results: F=", f\_val, ", P=", p\_val)

# anova is still most significant for diff bw fwd and rwd, 3 way results are driven by that difference.

# in order to display plot within window

# plt.show()