Consider the `CarPrice_Assignment.csv` data file (under the In-Class 10 assignment link). This data is public available on the Kaggle website, and has information on cars (characteristics related to car dimensions, engine and more). The goal is to use car information to predict the price of the car. **In Python**, answer the following:

1. (5 points) Load the data file to you S3 bucket. Using the pandas library, read the csv data file and create a data-frame called `car_price`.

```
import boto3
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
import tensorflow as tf

## Defining the bucket
s3 = boto3.resource('s3')
bucket_name = 'data-445'
bucket = s3.Bucket(bucket_name)

## Defining the csv file
file_key = 'Demos/CarPrice_Assignment.csv'

bucket_object = bucket.Object(file_key)
file_object = bucket_object.get()
file_content_stream = file_object.get('Body')

## Reading the csv file
car_price = pd.read_csv(file_content_stream)
car_price.head()
```

2. (5 points) Using the `wheelbase`, `enginesize`, `compressionratio`, `horsepower`, `peakrpm`, `citympg`, as the predictor variables, and `price` is the target variable, split the data into train (80%) and test (20%).

```
## Defining the input and target variables
X = car_price[['wheelbase', 'enginesize', 'compressionratio', 'horsepower',
               'peakrpm', 'citympg']]

Y = car_price['price']

## Split the data into train & testing
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.2)
```

3. (5 points) Using the `MinMaxScaler`, transform the input variables in the train and test dataset to 0-1 scale.

```
## Transforming input data to 0-1
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.fit_transform(X_test)
```

4. (7 points) Using the train dataset, build a MLP with a single hidden layer with 10 neurons and the `ReLU` as the activation function. Also use `optimizer = 'adam'`, `epochs = 100` and `batch_size = 100`. After that, use this model to predict on the test dataset. Report the MSE of this model.

```
md1 = tf.keras.models.Sequential([
      tf.keras.layers.Dense(10, input_dim = 6, activation = 'relu'),
      tf.keras.layers.Dense(1)
])

md1.compile(optimizer = 'adam', loss = 'mse')
md1.fit(X_train, Y_train, epochs = 100, batch_size = 100, verbose = 0)

md1.evaluate(X_test, Y_test)
```

5. (7 points) Using the train dataset, build a MLP with two hidden layers: the first one with 10 neurons and the `ReLU` as the activation function, the second one with 8 neurons and `ReLU` as the activation function. Also use `optimizer = 'adam'`, `epochs = 100` and `batch_size = 100`. After that, use this model to predict on the test dataset. Report the MSE of this model.

```
md2 = tf.keras.models.Sequential([
      tf.keras.layers.Dense(10, input_dim = 6, activation = 'relu'),
      tf.keras.layers.Dense(8, activation = 'relu'),
      tf.keras.layers.Dense(1)
])

md2.compile(optimizer = 'adam', loss = 'mse')
md2.fit(X_train, Y_train, epochs = 100, batch_size = 100, verbose = 0)

md2.evaluate(X_test, Y_test)
```

6. (3 points) Using the results from parts (4) and (5), what model would you use to predict car prices? Explain.

Based on the mean squared error, I would use the model from part (5) since its mse is smaller.

The End.