Train Test & Split

We have a dataset containing prices of used BMW cars. We are going to analyze this dataset and build a prediction function that can predict a price by taking mileage and age of the car as input. We will use sklearn train_test_split method to split training and testing dataset

Usually when you have a dataset like below sometimes we train the model using the entire dataset but that's not a good strategy. The good strategy is to split dataset into 2 parts where you use 80% of the samples for actual training and the other 20% for testing your model. The reason is you can see the accuracy of your model in testing samples because the model has not seen those 20% testing samples.

In [50]:
```
import pandas as pd
df = pd.read_csv("D:/Data_Science/My Github/Machine-Learning-with-Python/6. train_
df.head()
```

Out[50]:

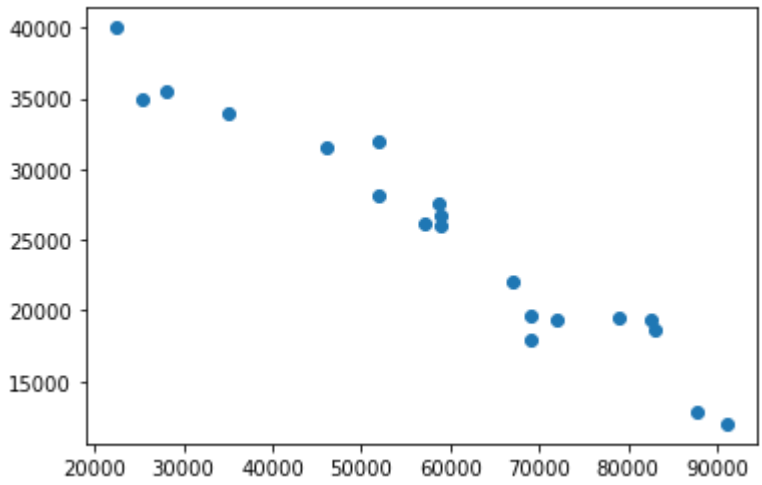
	Mileage	Age(yrs)	Sell Price($)
0	69000	6	18000
1	35000	3	34000
2	57000	5	26100
3	22500	2	40000
4	46000	4	31500

In [51]:
```
import matplotlib.pyplot as plt
%matplotlib inline
```

Car Mileage Vs Sell Price ($)

In [52]:
```
plt.scatter(df['Mileage'],df['Sell Price($)'])
```

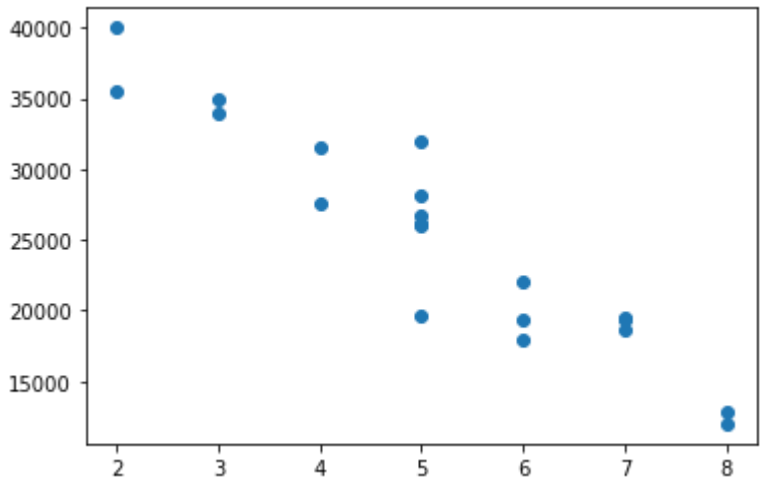
Out[52]:
```
<matplotlib.collections.PathCollection at 0xbff0820>
```



Car Age Vs Sell Price ($)

In [53]:
```
plt.scatter(df['Age(yrs)'],df['Sell Price($)'])
```

Out[53]:
```
<matplotlib.collections.PathCollection at 0xc04c940>
```



Looking at above two scatter plots, using linear regression model makes sense as we can clearly see a linear relationship between our dependant (i.e. Sell Price) and independant variables (i.e. car age and car mileage)

The approach we are going to use here is to split available data in two sets

1. **Training: We will train our model on this dataset**
2. **Testing: We will use this subset to make actual predictions using trained model**

The reason we don't use same training set for testing is because our model has seen those samples before, using same samples for making predictions might give us wrong impression about accuracy of our model. It is like you ask same questions in exam paper as you tought the students in the class.

In [54]:
```
X = df[['Mileage','Age(yrs)']]
y = df['Sell Price($)']
```

In [55]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.2)
```

In [7]:
```
len(X)
```

Out[7]: 20

In [8]:
```
len(X_train)
```

Out[8]: 16

In [9]:
```
len(X_test)
```

Out[9]: 4

In [56]:
```
X_train
```

Out[56]:

	Mileage	Age(yrs)
12	59000	5
18	87600	8
11	79000	7
7	72000	6
15	25400	3
16	28000	2
0	69000	6
4	46000	4
6	52000	5
17	69000	5
5	59000	5

by utilizing random_state=10, the samples are fixed

In [57]:
```
X_test
```

Out[57]:

	Mileage	Age(yrs)
3	22500	2
2	57000	5
8	91000	8
19	52000	5

Lets run linear regression model now

In [58]:
```
from sklearn.linear_model import LinearRegression
clf = LinearRegression()
clf.fit(X_train, y_train)
```

Out[58]:
```
LinearRegression()
```

In [59]:
```
X_test
```

Out[59]:

	Mileage	Age(yrs)
3	22500	2
2	57000	5
8	91000	8
19	52000	5

In [60]:
```
clf.predict(X_test)
```

Out[60]:
```
array([38286.31105651, 26141.08911954, 14160.30954509, 27785.51274481])
```

In [61]:
```
y_test
```

Out[61]:
```
3     40000
2     26100
8     12000
19    28200
Name: Sell Price($), dtype: int64
```

Check the accuracy of the model

In [62]:
```
clf.score(X_test, y_test)
```

Out[62]:
```
0.9803372581524511
```

Date	Author
2021-08-25	Ehsan Zia