In [56]:
```
# import libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

Step 1:Basic Data Quality Checks

In [57]:
```
# Import data set

cars = pd.read_csv('UsedCarsSA_Clean_ENG.csv')
cars.head()
```

Out[57]:

	Make	Type	Year	Origin	Options	Gear_Type	Mileage	Region	Price
0	Chrysler	C300	2018	Saudi	Full	Automatic	103000	Riyadh	114000
1	Nissan	Patrol	2016	Saudi	Full	Automatic	5448	Riyadh	0
2	Nissan	Sunny	2019	Saudi	Standard	Automatic	72418	Riyadh	27500
3	Hyundai	Elantra	2019	Saudi	Standard	Automatic	114154	Riyadh	43000
4	Hyundai	Elantra	2019	Saudi	Semi Full	Automatic	41912	Riyadh	59500

In [58]:
```
print ("Rows     : " ,cars.shape[0])  #get number of rows/observations
print ("Columns  : " , cars.shape[1]) #get number of columns
```

```
Rows     :  8035
Columns  :  9
```

In [59]:
```
cars.head()
```

Out[59]:

	Make	Type	Year	Origin	Options	Gear_Type	Mileage	Region	Price
0	Chrysler	C300	2018	Saudi	Full	Automatic	103000	Riyadh	114000
1	Nissan	Patrol	2016	Saudi	Full	Automatic	5448	Riyadh	0
2	Nissan	Sunny	2019	Saudi	Standard	Automatic	72418	Riyadh	27500
3	Hyundai	Elantra	2019	Saudi	Standard	Automatic	114154	Riyadh	43000
4	Hyundai	Elantra	2019	Saudi	Semi Full	Automatic	41912	Riyadh	59500

In [285]:
```
cars.tail()
```

Out[285]:

	Make	Type	Year	Origin	Options	Gear_Type	Mileage	Region	Price	age
8030	1	17	44	2	0	0	877	20	242	1977
8031	8	77	39	2	0	0	1395	20	191	1982
8032	54	215	42	0	0	0	140	17	0	1979
8033	44	28	40	2	0	0	456	7	120	1981
8034	5	253	42	2	0	0	1742	4	163	1979

In [253]:
```
cars.describe()
```

Out[253]:

	Make	Type	Year	Origin	Options	Gear_Type	Mileag
count	8035.000000	8035.000000	8035.000000	8035.000000	8035.000000	8035.000000	8035.0000
mean	32.827629	197.102427	43.101680	1.638208	0.927442	0.132421	1071.9917
std	16.640902	106.874745	5.726115	0.709064	0.846782	0.338969	585.3312
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	19.000000	106.000000	41.000000	2.000000	0.000000	0.000000	604.0000
50%	32.000000	198.000000	45.000000	2.000000	1.000000	0.000000	1084.0000
75%	54.000000	291.000000	47.000000	2.000000	2.000000	0.000000	1551.0000
max	58.000000	380.000000	51.000000	3.000000	2.000000	1.000000	2174.0000

In [254]:
```
cars.columns
```

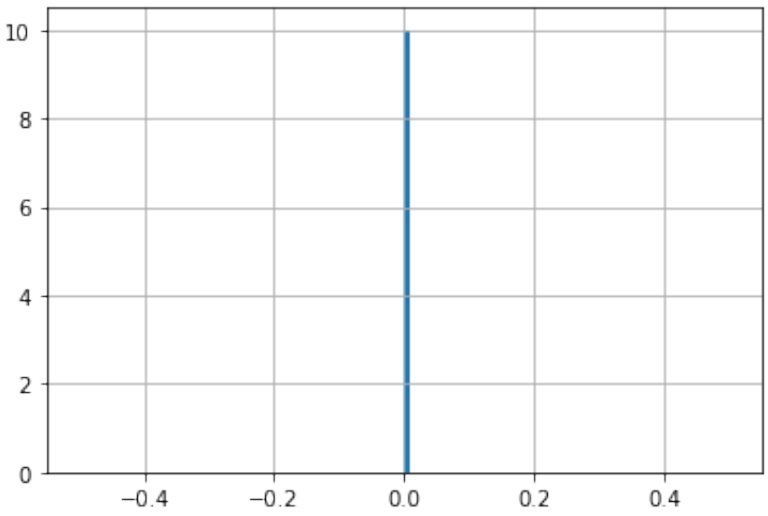
Out[254]:
```
Index(['Make', 'Type', 'Year', 'Origin', 'Options', 'Gear_Type', 'Mile
age',
       'Region', 'Price', 'age'],
      dtype='object')
```

In [255]:
```
cars.dtypes
```

Out[255]:
```
Make         int64
Type         int64
Year         int64
Origin       int64
Options      int64
Gear_Type    int64
Mileage      int64
Region       int64
Price        int64
age          int64
dtype: object
```

In [257]:
```
missing= pd.isnull(cars)
missing_ratio= missing.sum()/len(missing)
missing_ratio.hist(bins=100)
```

Out[257]:
```
<AxesSubplot:>
```



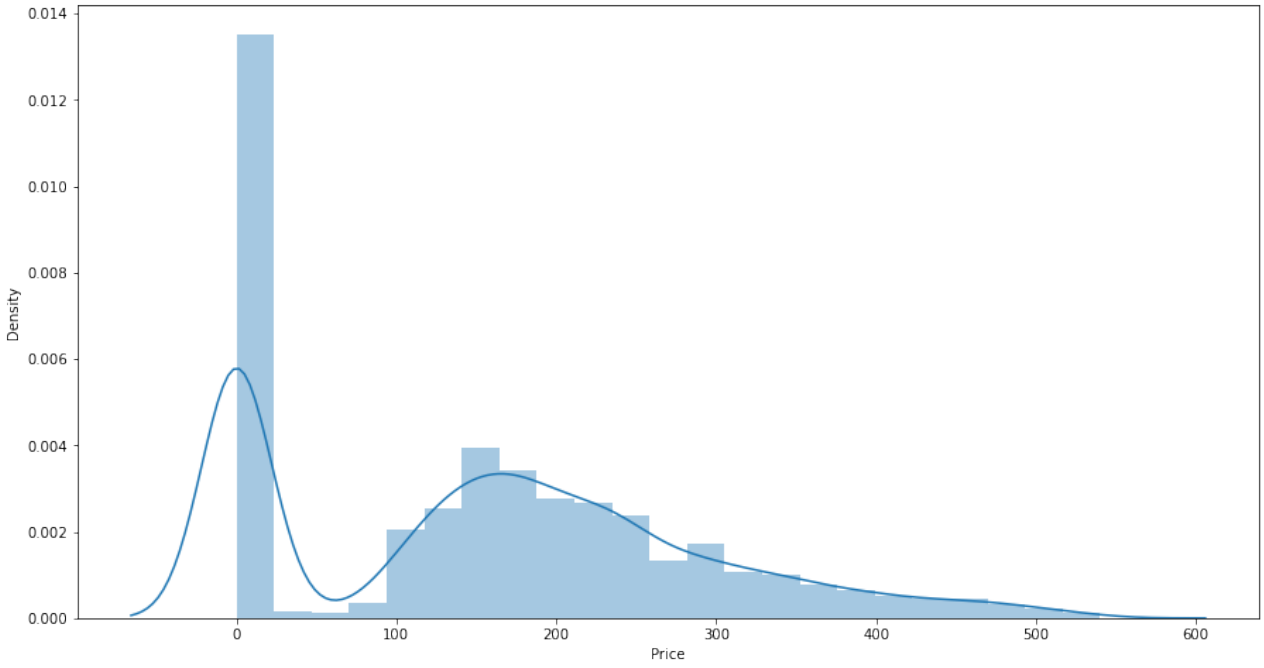
In [258]:
```
cars.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8035 entries, 0 to 8034
Data columns (total 10 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Make       8035 non-null   int64
 1   Type       8035 non-null   int64
 2   Year       8035 non-null   int64
 3   Origin     8035 non-null   int64
 4   Options    8035 non-null   int64
 5   Gear_Type  8035 non-null   int64
 6   Mileage    8035 non-null   int64
 7   Region     8035 non-null   int64
 8   Price      8035 non-null   int64
 9   age        8035 non-null   int64
dtypes: int64(10)
memory usage: 627.9 KB
```

In [294]:
```
fig, ax = plt.subplots(figsize=(15,8))
sns.distplot(cars['Price'])
plt.xlim
```

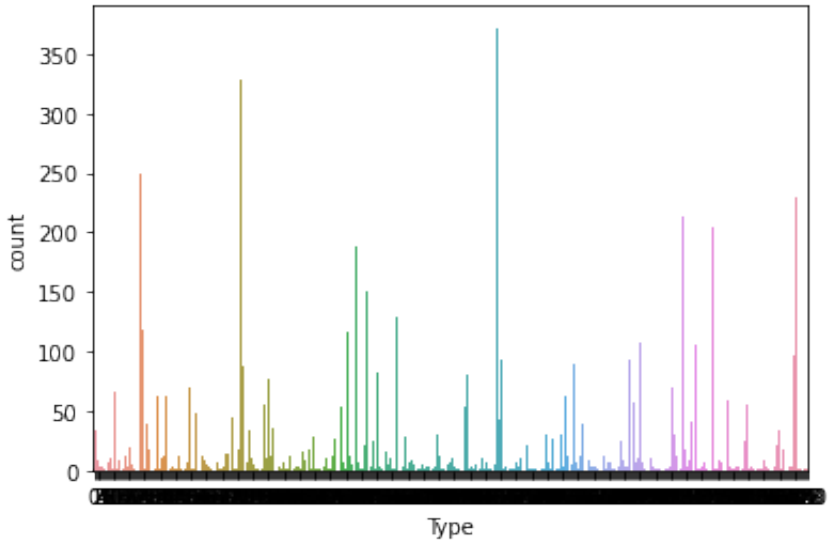
```
/Users/hadelalenezi/opt/anaconda3/lib/python3.8/site-packages/seaborn/
distributions.py:2557: FutureWarning: `distplot` is a deprecated funct
ion and will be removed in a future version. Please adapt your code to
use either `displot` (a figure-level function with similar flexibility
) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[294]:
```
<function matplotlib.pyplot.xlim(*args, **kwargs)>
```

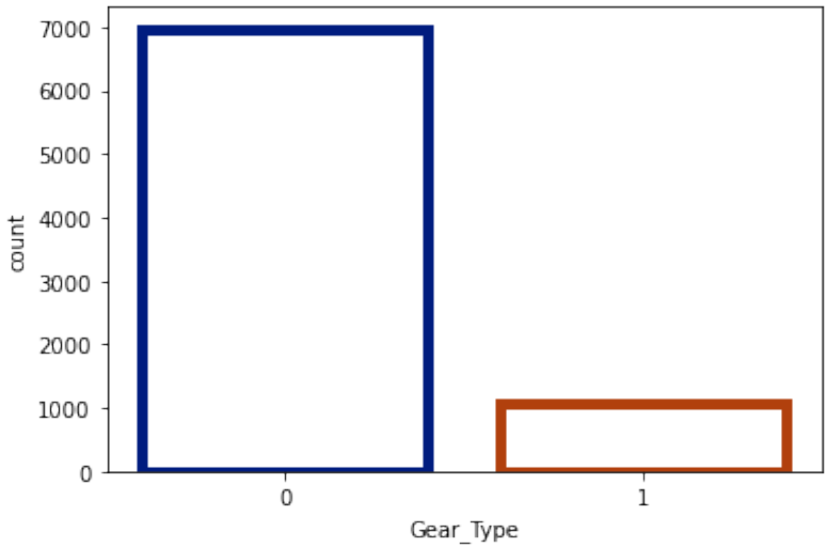


In [260]:
```
sns.countplot(x='Type',data=cars,)
```

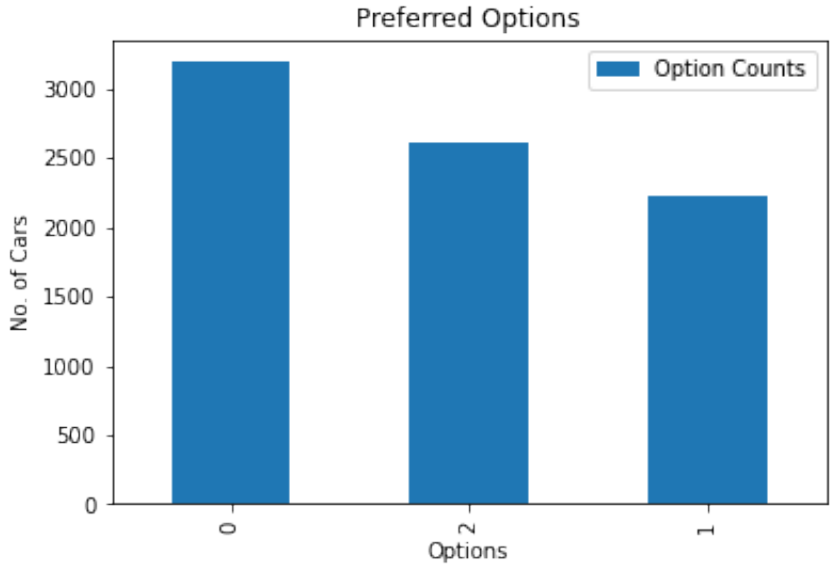
Out[260]: <AxesSubplot:xlabel='Type', ylabel='count'>



In [261]:
```
ax = sns.countplot(x="Gear_Type", data=cars,
                   facecolor=(0, 0, 0, 0),
                   linewidth=5,
                   edgecolor=sns.color_palette("dark", 3))
```



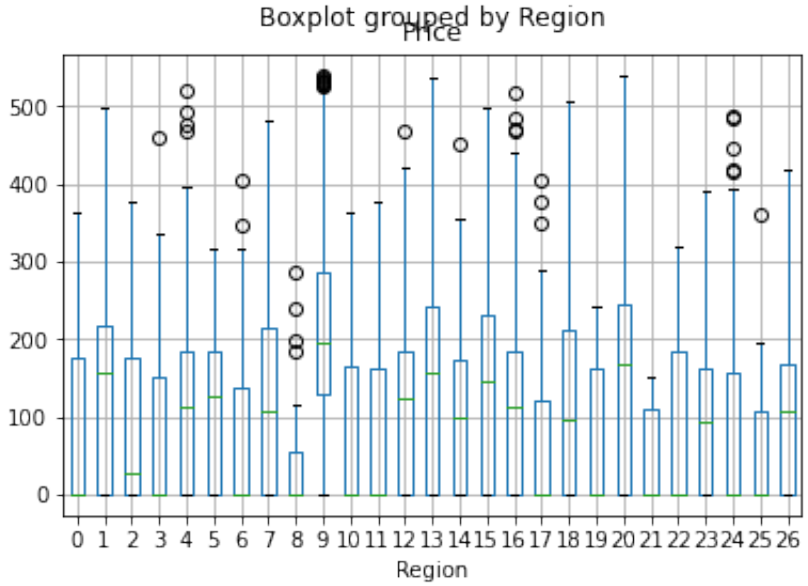
In [287]:
```
option= cars.groupby('Options')['Options'].count()
option = pd.DataFrame(option)
option.columns = ['Option Counts']
option.sort_values(by=['Option Counts'], inplace=True, ascending=False)
option = option.head(5)

option.plot.bar();

plt.title('Preferred Options')

plt.xlabel('Options')
plt.ylabel('No. of Cars');
```



From the above bar graph it can be easily visulaized that the people in Saudi prefer to drive Full cars. Semi Full cars have the lowest numbers of drivers in Saudi as this fuel is not available in abundant in Saudi

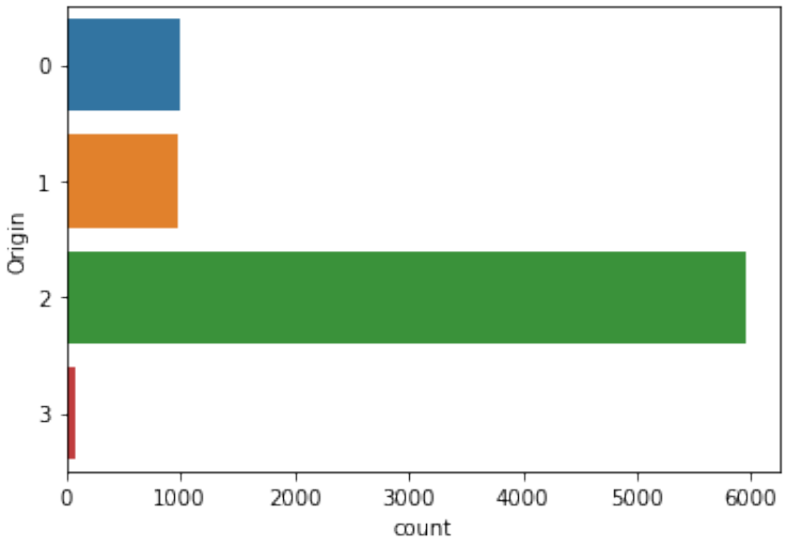
In [286]:
```
cars.boxplot(column='Price',by = 'Region')
```

Out[286]: <AxesSubplot:title={'center':'Price'}, xlabel='Region'>

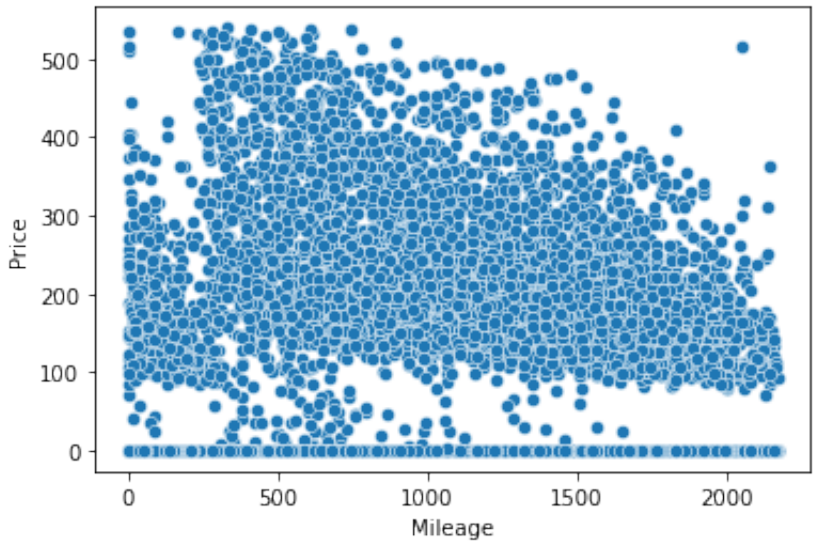


In [288]:
```
sns.countplot(data = cars
              ,y = 'Origin'
              )
```

Out[288]: <AxesSubplot:xlabel='count', ylabel='Origin'>

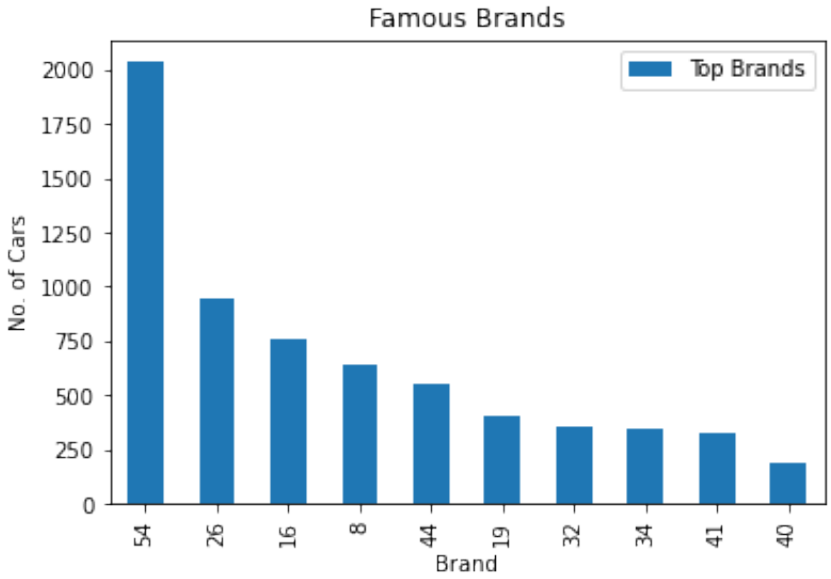


In [293]:
```
sns.scatterplot(x=cars['Mileage'], y=cars['Price'], s=40);
```



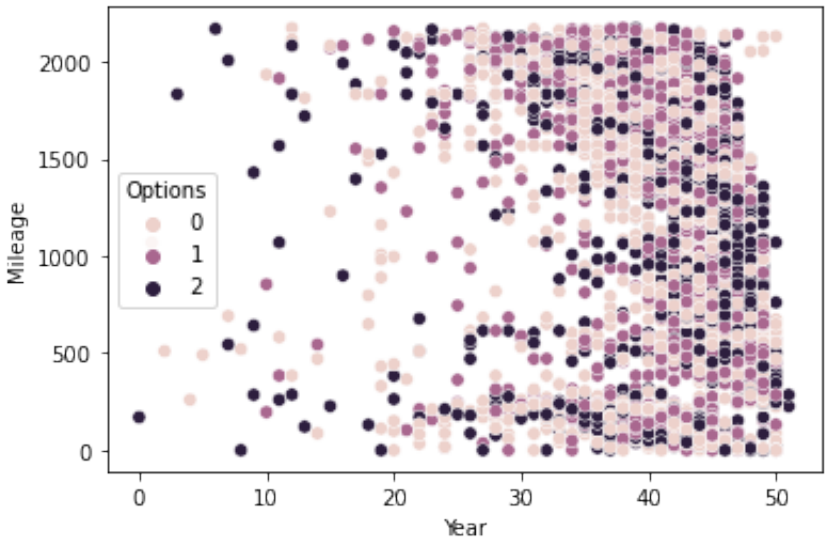
The price of car decreases sharply with increasing mileage

In [290]:
```
topbrands= cars.groupby('Make')['Make'].count()
topbrands = pd.DataFrame(topbrands)
topbrands.columns = ['Top Brands']
topbrands.sort_values(by=['Top Brands'], inplace=True, ascending=False)
topbrands = topbrands.head(10)

topbrands.plot.bar();

plt.title('Famous Brands')

plt.xlabel('Brand')
plt.ylabel('No. of Cars');
```



The bar plot shows that the people in Saudi prefer cars from Japenese brand as the most selling cars from the top 10 brands in Saudi are from Japenese brand. This is obvious as the most of the Japenese car companies have manufacturing plants in saudi

In [292]:
```
sns.scatterplot(x=cars.Year, y=cars['Mileage'], hue=cars.Options, s=40);
```



The above scatterplot shows that Saudi people like to drive cars of the used models i.e. released between 1970 and 2020. For the cars between 1980 and 2000 models, the preferred Options was standers as it is the cheapest available in Saudi. For the 2000 - 2020 model cars, the people tend to shift to the Full cars due to the shortage of standers in saudi lately

Step 2: EDA

2.1 Visualizing Numerical Data

In [265]:
```
# Calculate the age of the car

cars['age'] = 2021 - cars['Year']
cars.head()
```

Out[265]:

	Make	Type	Year	Origin	Options	Gear_Type	Mileage	Region	Price	age
0	9	54	47	2	0	0	1094	20	308	1974
1	44	262	45	2	0	0	336	20	0	1976
2	44	321	48	2	2	0	841	20	135	1973
3	26	140	48	2	2	0	1200	20	171	1973
4	26	140	48	2	1	0	627	20	209	1973

In [266]:
```
cars.dtypes
```

Out[266]:
```
Make          int64
Type          int64
Year          int64
Origin        int64
Options       int64
Gear_Type     int64
Mileage       int64
Region        int64
Price         int64
age           int64
dtype: object
```

In [267]:
```
X = cars.iloc[:,[0,1,2,3,4,5,6,7,9]].values
y=cars.iloc[:,[8]].values
```

In [268]:
```
X
```

Out[268]:
```
array([[   9,   54,   47, ..., 1094,   20, 1974],
       [  44,  262,   45, ...,  336,   20, 1976],
       [  44,  321,   48, ...,  841,   20, 1973],
       ...,
       [  54,  215,   42, ...,  140,   17, 1979],
       [  44,   28,   40, ...,  456,    7, 1981],
       [   5,  253,   42, ..., 1742,    4, 1979]])
```

In [269]:
```
y
```

Out[269]:
```
array([[308],
       [  0],
       [135],
       ...,
       [  0],
       [120],
       [163]])
```

We need to do encodeing beause Machine Learning will understand numerical

data preproessing

Label Encoding

In [215]:
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
cars = cars.apply(le.fit_transform)

X = cars.values[:, 0:7]
Y = cars.values[:,8]
```

Feature scaling because speed traning process

In [219]:
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X = scaler.fit_transform(cars)
```

In [220]:
```
X
```

Out[220]:
```
array([[-1.43196034, -1.3390566 ,  0.68083904, ...,  0.86123736,
         1.16172286, -0.68083904],
       [ 0.6714219 ,  0.60726799,  0.33154033, ...,  0.86123736,
        -1.16234267, -0.33154033],
       [ 0.6714219 ,  1.15935045,  0.85548839, ...,  0.86123736,
        -0.14367758, -0.85548839],
       ...,
       [ 1.27238826,  0.16747349, -0.19240772, ...,  0.37223452,
        -1.16234267,  0.19240772],
       [ 0.6714219 , -1.58234717, -0.54170643, ..., -1.25777495,
        -0.25686259,  0.54170643],
       [-1.67234689,  0.52305202, -0.19240772, ..., -1.7467778 ,
         0.0676011 ,  0.19240772]])
```

Spiltting Data into Traning and Test set because we need some data for traning and we dont have some data some data will use for training and other for testinh

In [224]:
```
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,
```

traning model

In [228]:
```
from sklearn.ensemble import RandomForestRegressor
regr = RandomForestRegressor(random_state=0)
regr.fit(X_train, y_train)
```

```
<ipython-input-228-84ae2596759d>:3: DataConversionWarning: A column-ve
ctor y was passed when a 1d array was expected. Please change the shap
e of y to (n_samples,), for example using ravel().
  regr.fit(X_train, y_train)
```

Out[228]:
```
RandomForestRegressor(random_state=0)
```

Predict

In [229]:
```
y_pred= regr.predict(X_test)
```

Testing RESULT

In [232]:
```
print(y_pred)
```

```
[49000. 37000.     0. ... 17390. 32000. 85960.]
```

In [233]:
```
print(y_test)
```

```
[[49000]
 [37000]
 [    0]
 ...
 [17500]
 [32000]
 [86000]]
```

In [270]:
```
mydata =np.concatenate((y_pred.reshape(len(y_pred),1),y_test.reshape(len
print(mydata)
```

```
[[49000. 49000.]
 [37000. 37000.]
 [    0.     0.]
 ...
 [17390. 17500.]
 [32000. 32000.]
 [85960. 86000.]]
```

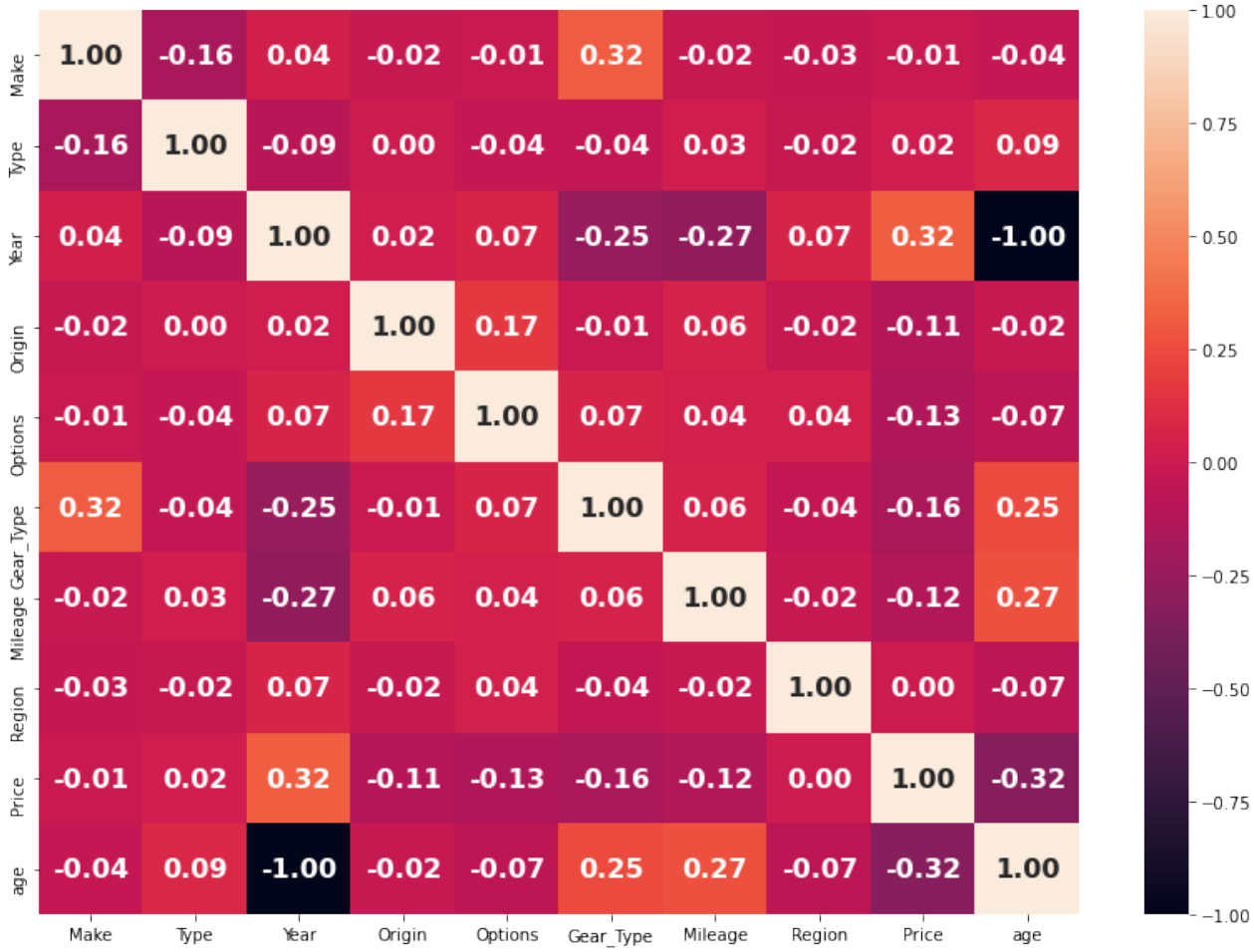
Calculating Accuracy

In [ ]:

In [240]:
```
from sklearn.metrics import r2_score
r2_score(y_test, y_pred)*100
```

Out[240]: 98.6406467114643

In [283]:
```
corr = cars.corr()
f, ax = plt.subplots(figsize=(14, 10))
sns.heatmap(corr, annot=True, fmt='.2f', ax=ax,annot_kws={'fontsize':16,
```



making dataFram

In [291]:
```
dataframe= pd.DataFrame(mydata,columns=['Predicted value','Real Value'])
print(dataframe)
```

```
      Predicted value  Real Value
0            49000.00     49000.0
1            37000.00     37000.0
2                0.00         0.0
3            92989.92     93000.0
4                0.00         0.0
...                ...         ...
1602             0.00         0.0
1603        155080.00    155000.0
1604         17390.00     17500.0
1605         32000.00     32000.0
1606         85960.00     86000.0

[1607 rows x 2 columns]
```

In [ ]: