Predicting the sales price of bulldozers using Machine learning algorithm

1. Definition

> How well can we predict the future sale price of a bulldozer, given its characteristics and previous examples of how much similar bulldozers have been sold for?

2. Data

The data we have taken from kaggle : https://www.kaggle.com/c/bluebook-for-bulldozers/data (https://www.kaggle.com/c/bluebook-for-bulldozers/data)

There are 3 main datasets:

- Train.csv is the training set, which contains data through the end of 2011.
- Valid.csv is the validation set, which contains data from January 1, 2012 - April 30, 2012 You make predictions on this set throughout the majority of the competition. Your score on this set is used to create the public leaderboard.
- Test.csv is the test set, which won't be released until the last week of the competition. It contains data from May 1, 2012 - November 2012. Your score on the test set determines your final rank for the competition.

3. Evaluation

The evaluation metric for this competition is the RMSLE (root mean squared log error) between the actual and predicted auction prices.

For more on the evaluation of this project check : https://www.kaggle.com/c/bluebook-for-bulldozers/overview/evaluation (https://www.kaggle.com/c/bluebook-for-bulldozers/overview/evaluation)

Note. The goal for most regression evaluation metrics is to minimises the the error. For example, our goal for this project is to build a machine learning model which minimises RMSLE.

4. Features

Kaggle provides a data dictionary detailing all the features of the dataset.
https://www.kaggle.com/c/bluebook-for-bulldozers/data?select=Data+Dictionary.xlsx (https://www.kaggle.com/c/bluebook-for-bulldozers/data?select=Data+Dictionary.xlsx)

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
```

In [120]:
```
# Importing training and validation sets
df=pd.read_csv("data/TrainAndValid.csv", low_memory=False)
```

In [121]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 412698 entries, 0 to 412697
Data columns (total 53 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   SalesID                   412698 non-null  int64  
 1   SalePrice                 412698 non-null  float64
 2   MachineID                 412698 non-null  int64  
 3   ModelID                   412698 non-null  int64  
 4   datasource                412698 non-null  int64  
 5   auctioneerID              392562 non-null  float64
 6   YearMade                  412698 non-null  int64  
 7   MachineHoursCurrentMeter  147504 non-null  float64
 8   UsageBand                 73670 non-null   object 
 9   saledate                  412698 non-null  object 
 10  fiModelDesc               412698 non-null  object 
 11  fiBaseModel               412698 non-null  object 
 12  fiSecondaryDesc           271971 non-null  object 
 13  fiModelSeries             58667 non-null   object 
 14  fiModelDescriptor         74816 non-null   object 
 15  ProductSize               196093 non-null  object 
 16  fiProductClassDesc        412698 non-null  object 
 17  state                     412698 non-null  object 
 18  ProductGroup              412698 non-null  object 
 19  ProductGroupDesc          412698 non-null  object 
 20  Drive_System              107087 non-null  object 
 21  Enclosure                 412364 non-null  object 
 22  Forks                     197715 non-null  object 
 23  Pad_Type                  81096 non-null   object 
 24  Ride_Control              152728 non-null  object 
 25  Stick                     81096 non-null   object 
 26  Transmission              188007 non-null  object 
 27  Turbocharged              81096 non-null   object 
 28  Blade_Extension           25983 non-null   object 
 29  Blade_Width               25983 non-null   object 
 30  Enclosure_Type            25983 non-null   object 
 31  Engine_Horsepower         25983 non-null   object 
 32  Hydraulics                330133 non-null  object 
 33  Pushblock                 25983 non-null   object 
 34  Ripper                    106945 non-null  object 
 35  Scarifier                 25994 non-null   object 
 36  Tip_Control               25983 non-null   object 
 37  Tire_Size                 97638 non-null   object 
 38  Coupler                   220679 non-null  object 
 39  Coupler_System            44974 non-null   object 
 40  Grouser_Tracks            44875 non-null   object 
 41  Hydraulics_Flow           44875 non-null   object 
 42  Track_Type                102193 non-null  object 
 43  Undercarriage_Pad_Width   102916 non-null  object 
 44  Stick_Length              102261 non-null  object 
 45  Thumb                     102332 non-null  object 
 46  Pattern_Changer           102261 non-null  object 
 47  Grouser_Type              102193 non-null  object 
 48  Backhoe_Mounting          80712 non-null   object 
 49  Blade_Type                81875 non-null   object 
 50  Travel_Controls           81877 non-null   object
 51  Differential_Type         71564 non-null   object
 52  Steering_Controls         71522 non-null   object
dtypes: float64(3), int64(5), object(45)
memory usage: 166.9+ MB
```

```
In [122]: df.isna().sum()
```

```
Out[122]: SalesID                          0
          SalePrice                        0
          MachineID                        0
          ModelID                          0
          datasource                       0
          auctioneerID                 20136
          YearMade                         0
          MachineHoursCurrentMeter    265194
          UsageBand                   339028
          saledate                         0
          fiModelDesc                      0
          fiBaseModel                      0
          fiSecondaryDesc             140727
          fiModelSeries               354031
          fiModelDescriptor           337882
          ProductSize                 216605
          fiProductClassDesc               0
          state                            0
          ProductGroup                     0
          ProductGroupDesc                 0
          Drive_System                305611
          Enclosure                      334
          Forks                       214983
          Pad_Type                    331602
          Ride_Control                259970
          Stick                       331602
          Transmission                224691
          Turbocharged                331602
          Blade_Extension             386715
          Blade_Width                 386715
          Enclosure_Type              386715
          Engine_Horsepower           386715
          Hydraulics                   82565
          Pushblock                   386715
          Ripper                      305753
          Scarifier                   386704
          Tip_Control                 386715
          Tire_Size                   315060
          Coupler                     192019
          Coupler_System              367724
          Grouser_Tracks              367823
          Hydraulics_Flow             367823
          Track_Type                  310505
          Undercarriage_Pad_Width     309782
          Stick_Length                310437
          Thumb                       310366
          Pattern_Changer             310437
          Grouser_Type                310505
          Backhoe_Mounting            331986
          Blade_Type                  330823
          Travel_Controls             330821
          Differential_Type           341134
          Steering_Controls           341176
          dtype: int64
```

In [123]:
```
df.saledate
```

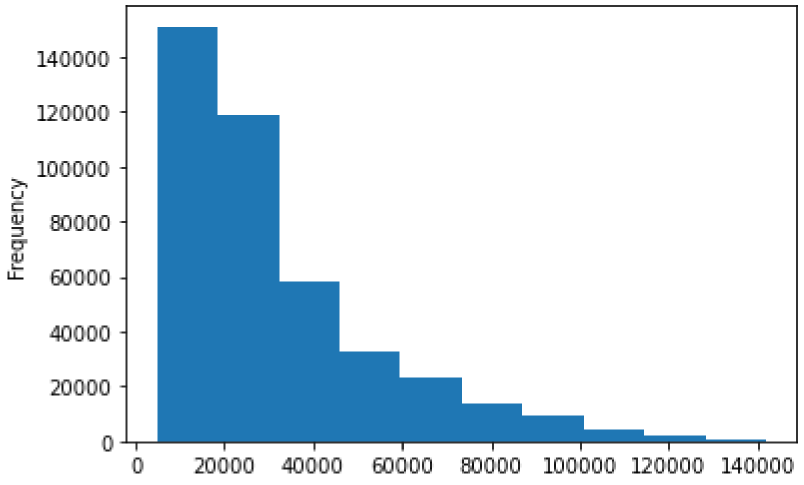
Out[123]:
```
0         11/16/2006 0:00
1          3/26/2004 0:00
2          2/26/2004 0:00
3          5/19/2011 0:00
4          7/23/2009 0:00
               ...
412693      3/7/2012 0:00
412694     1/28/2012 0:00
412695     1/28/2012 0:00
412696      3/7/2012 0:00
412697     1/28/2012 0:00
Name: saledate, Length: 412698, dtype: object
```

In [124]:
```
df.saledate.dtype
```

Out[124]:
```
dtype('O')
```

In [125]:
```
df.SalePrice.plot.hist()
```

Out[125]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1816ebcf1c8>
```



Parsing dates

When we work with time series data, we want to enrich the time & date component as much as possible.

We can do that by telling which of our columns has dates in it using the `parse_dates` parameter.

In [126]:
```
# Import data agin but this time parse dates
df=pd.read_csv("data/TrainAndValid.csv",
               low_memory=False,
               parse_dates=["saledate"])
```

```
In [127]: df.saledate.dtype
```

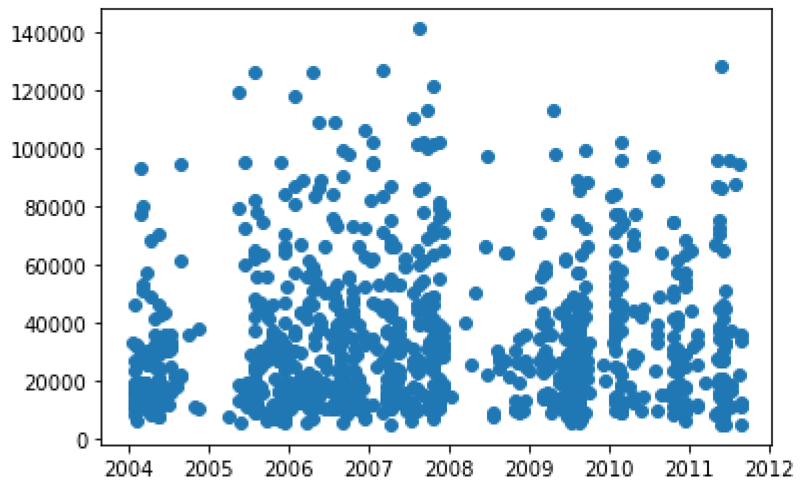
```
Out[127]: dtype('<M8[ns]')
```

```
In [128]: df.saledate[:1000]
```

```
Out[128]: 0      2006-11-16
          1      2004-03-26
          2      2004-02-26
          3      2011-05-19
          4      2009-07-23
                    ...
          995    2009-07-16
          996    2007-06-14
          997    2005-09-22
          998    2005-07-28
          999    2011-06-16
          Name: saledate, Length: 1000, dtype: datetime64[ns]
```

```
In [129]: fig, ax =plt.subplots()
          ax.scatter(df["saledate"][:1000],df["SalePrice"][:1000])
```

```
Out[129]: <matplotlib.collections.PathCollection at 0x18137af4308>
```



```
In [130]: df.head()
```

Out[130]:

	SalesID	SalePrice	MachineID	ModelID	datasource	auctioneerID	YearMade	MachineHoursCurı
0	1139246	66000.0	999089	3157	121	3.0	2004	
1	1139248	57000.0	117657	77	121	3.0	1996	
2	1139249	10000.0	434808	7009	121	3.0	2001	
3	1139251	38500.0	1026470	332	121	3.0	2001	
4	1139253	11000.0	1057373	17311	121	3.0	2007	

5 rows × 53 columns

In [131]:
```
df.head().T
```

Out[131]:

	0	1	2	3	4
SalesID	1139246	1139248	1139249	1139251	1139253
SalePrice	66000	57000	10000	38500	11000
MachineID	999089	117657	434808	1026470	1057373
ModelID	3157	77	7009	332	17311
datasource	121	121	121	121	121
auctioneerID	3	3	3	3	3
YearMade	2004	1996	2001	2001	2007
MachineHoursCurrentMeter	68	4640	2838	3486	722
UsageBand	Low	Low	High	High	Medium
saledate	2006-11-16 00:00:00	2004-03-26 00:00:00	2004-02-26 00:00:00	2011-05-19 00:00:00	2009-07-23 00:00:00
fiModelDesc	521D	950FII	226	PC120-6E	S175
fiBaseModel	521	950	226	PC120	S175
fiSecondaryDesc	D	F	NaN	NaN	NaN
fiModelSeries	NaN	II	NaN	-6E	NaN
fiModelDescriptor	NaN	NaN	NaN	NaN	NaN
ProductSize	NaN	Medium	NaN	Small	NaN
fiProductClassDesc	Wheel Loader - 110.0 to 120.0 Horsepower	Wheel Loader - 150.0 to 175.0 Horsepower	Skid Steer Loader - 1351.0 to 1601.0 Lb Operat...	Hydraulic Excavator, Track - 12.0 to 14.0 Metr...	Skid Steer Loader - 1601.0 to 1751.0 Lb Operat...
state	Alabama	North Carolina	New York	Texas	New York
ProductGroup	WL	WL	SSL	TEX	SSL
ProductGroupDesc	Wheel Loader	Wheel Loader	Skid Steer Loaders	Track Excavators	Skid Steer Loaders
Drive_System	NaN	NaN	NaN	NaN	NaN
Enclosure	EROPS w AC	EROPS w AC	OROPS	EROPS w AC	EROPS
Forks	None or Unspecified	None or Unspecified	None or Unspecified	NaN	None or Unspecified
Pad_Type	NaN	NaN	NaN	NaN	NaN
Ride_Control	None or Unspecified	None or Unspecified	NaN	NaN	NaN
Stick	NaN	NaN	NaN	NaN	NaN
Transmission	NaN	NaN	NaN	NaN	NaN
Turbocharged	NaN	NaN	NaN	NaN	NaN
Blade_Extension	NaN	NaN	NaN	NaN	NaN

	0	1	2	3	4
Blade_Width	NaN	NaN	NaN	NaN	NaN
Enclosure_Type	NaN	NaN	NaN	NaN	NaN
Engine_Horsepower	NaN	NaN	NaN	NaN	NaN
Hydraulics	2 Valve	2 Valve	Auxiliary	2 Valve	Auxiliary
Pushblock	NaN	NaN	NaN	NaN	NaN
Ripper	NaN	NaN	NaN	NaN	NaN
Scarifier	NaN	NaN	NaN	NaN	NaN
Tip_Control	NaN	NaN	NaN	NaN	NaN
Tire_Size	None or Unspecified	23.5	NaN	NaN	NaN
Coupler	None or Unspecified	None or Unspecified	None or Unspecified	None or Unspecified	None or Unspecified
Coupler_System	NaN	NaN	None or Unspecified	NaN	None or Unspecified
Grouser_Tracks	NaN	NaN	None or Unspecified	NaN	None or Unspecified
Hydraulics_Flow	NaN	NaN	Standard	NaN	Standard
Track_Type	NaN	NaN	NaN	NaN	NaN
Undercarriage_Pad_Width	NaN	NaN	NaN	NaN	NaN
Stick_Length	NaN	NaN	NaN	NaN	NaN
Thumb	NaN	NaN	NaN	NaN	NaN
Pattern_Changer	NaN	NaN	NaN	NaN	NaN
Grouser_Type	NaN	NaN	NaN	NaN	NaN
Backhoe_Mounting	NaN	NaN	NaN	NaN	NaN
Blade_Type	NaN	NaN	NaN	NaN	NaN
Travel_Controls	NaN	NaN	NaN	NaN	NaN
Differential_Type	Standard	Standard	NaN	NaN	NaN
Steering_Controls	Conventional	Conventional	NaN	NaN	NaN

```
In [132]: df.saledate[:20]
```

```
Out[132]: 0     2006-11-16
          1     2004-03-26
          2     2004-02-26
          3     2011-05-19
          4     2009-07-23
          5     2008-12-18
          6     2004-08-26
          7     2005-11-17
          8     2009-08-27
          9     2007-08-09
          10    2008-08-21
          11    2006-08-24
          12    2005-10-20
          13    2006-01-26
          14    2006-01-03
          15    2006-11-16
          16    2007-06-14
          17    2010-01-28
          18    2006-03-09
          19    2005-11-17
          Name: saledate, dtype: datetime64[ns]
```

Sort DataFrame by saledate

When working with time series data, its a good practice to sort the data in ascending order of date.

```
In [133]: df.sort_values(by=["saledate"],inplace=True,ascending=True)
          df.saledate.head(20)
```

```
Out[133]: 205615   1989-01-17
          274835   1989-01-31
          141296   1989-01-31
          212552   1989-01-31
          62755    1989-01-31
          54653    1989-01-31
          81383    1989-01-31
          204924   1989-01-31
          135376   1989-01-31
          113390   1989-01-31
          113394   1989-01-31
          116419   1989-01-31
          32138    1989-01-31
          127610   1989-01-31
          76171    1989-01-31
          127000   1989-01-31
          128130   1989-01-31
          127626   1989-01-31
          55455    1989-01-31
          55454    1989-01-31
          Name: saledate, dtype: datetime64[ns]
```

In [134]:
```
df.head().T
```

Out[134]:

	205615	274835	141296	212552	62755
SalesID	1646770	1821514	1505138	1671174	1329056
SalePrice	9500	14000	50000	16000	22000
MachineID	1126363	1194089	1473654	1327630	1336053
ModelID	8434	10150	4139	8591	4089
datasource	132	132	132	132	132
auctioneerID	18	99	99	99	99
YearMade	1974	1980	1978	1980	1984
MachineHoursCurrentMeter	NaN	NaN	NaN	NaN	NaN
UsageBand	NaN	NaN	NaN	NaN	NaN
saledate	1989-01-17 00:00:00	1989-01-31 00:00:00	1989-01-31 00:00:00	1989-01-31 00:00:00	1989-01-31 00:00:00
fiModelDesc	TD20	A66	D7G	A62	D3B
fiBaseModel	TD20	A66	D7	A62	D3
fiSecondaryDesc	NaN	NaN	G	NaN	B
fiModelSeries	NaN	NaN	NaN	NaN	NaN
fiModelDescriptor	NaN	NaN	NaN	NaN	NaN
ProductSize	Medium	NaN	Large	NaN	NaN
fiProductClassDesc	Track Type Tractor, Dozer - 105.0 to 130.0 Hor...	Wheel Loader - 120.0 to 135.0 Horsepower	Track Type Tractor, Dozer - 190.0 to 260.0 Hor...	Wheel Loader - Unidentified	Track Type Tractor, Dozer - 20.0 to 75.0 Horse...
state	Texas	Florida	Florida	Florida	Florida
ProductGroup	TTT	WL	TTT	WL	TTT
ProductGroupDesc	Track Type Tractors	Wheel Loader	Track Type Tractors	Wheel Loader	Track Type Tractors
Drive_System	NaN	NaN	NaN	NaN	NaN
Enclosure	OROPS	OROPS	OROPS	EROPS	OROPS
Forks	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Pad_Type	NaN	NaN	NaN	NaN	NaN
Ride_Control	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Stick	NaN	NaN	NaN	NaN	NaN
Transmission	Direct Drive	NaN	Standard	NaN	Standard
Turbocharged	NaN	NaN	NaN	NaN	NaN
Blade_Extension	NaN	NaN	NaN	NaN	NaN
Blade_Width	NaN	NaN	NaN	NaN	NaN

	205615	274835	141296	212552	62755
Enclosure_Type	NaN	NaN	NaN	NaN	NaN
Engine_Horsepower	NaN	NaN	NaN	NaN	NaN
Hydraulics	2 Valve	2 Valve	2 Valve	2 Valve	2 Valve
Pushblock	NaN	NaN	NaN	NaN	NaN
Ripper	None or Unspecified	NaN	None or Unspecified	NaN	None or Unspecified
Scarifier	NaN	NaN	NaN	NaN	NaN
Tip_Control	NaN	NaN	NaN	NaN	NaN
Tire_Size	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Coupler	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Coupler_System	NaN	NaN	NaN	NaN	NaN
Grouser_Tracks	NaN	NaN	NaN	NaN	NaN
Hydraulics_Flow	NaN	NaN	NaN	NaN	NaN
Track_Type	NaN	NaN	NaN	NaN	NaN
Undercarriage_Pad_Width	NaN	NaN	NaN	NaN	NaN
Stick_Length	NaN	NaN	NaN	NaN	NaN
Thumb	NaN	NaN	NaN	NaN	NaN
Pattern_Changer	NaN	NaN	NaN	NaN	NaN
Grouser_Type	NaN	NaN	NaN	NaN	NaN
Backhoe_Mounting	None or Unspecified	NaN	None or Unspecified	NaN	None or Unspecified
Blade_Type	Straight	NaN	Straight	NaN	PAT
Travel_Controls	None or Unspecified	NaN	None or Unspecified	NaN	Lever
Differential_Type	NaN	Standard	NaN	Standard	NaN
Steering_Controls	NaN	Conventional	NaN	Conventional	NaN

In [135]:
```
## Make a copy of original dataframe, so that when we manipulate the copy we stil
df_tmp=df.copy()
```

In [136]:
```
df_tmp.head()
```

Out[136]:

	SalesID	SalePrice	MachineID	ModelID	datasource	auctioneerID	YearMade	MachineHou
205615	1646770	9500.0	1126363	8434	132	18.0	1974	
274835	1821514	14000.0	1194089	10150	132	99.0	1980	
141296	1505138	50000.0	1473654	4139	132	99.0	1978	
212552	1671174	16000.0	1327630	8591	132	99.0	1980	
62755	1329056	22000.0	1336053	4089	132	99.0	1984	

5 rows × 53 columns

In [137]:
```
## We have done some feature engineering which required in timeSeries kind of pro
df_tmp["saleYear"]=df_tmp.saledate.dt.year
df_tmp["saleMonth"]=df_tmp.saledate.dt.month
df_tmp["saleDay"]=df_tmp.saledate.dt.day
df_tmp["saleDayOfWeek"]=df_tmp.saledate.dt.dayofweek
df_tmp["saleDayOfYear"]=df_tmp.saledate.dt.dayofyear
```

In [138]:
```
df_tmp.head().T
```

Out[138]:

	205615	274835	141296	212552	62755
SalesID	1646770	1821514	1505138	1671174	1329056
SalePrice	9500	14000	50000	16000	22000
MachineID	1126363	1194089	1473654	1327630	1336053
ModelID	8434	10150	4139	8591	4089
datasource	132	132	132	132	132
auctioneerID	18	99	99	99	99
YearMade	1974	1980	1978	1980	1984
MachineHoursCurrentMeter	NaN	NaN	NaN	NaN	NaN
UsageBand	NaN	NaN	NaN	NaN	NaN
saledate	1989-01-17 00:00:00	1989-01-31 00:00:00	1989-01-31 00:00:00	1989-01-31 00:00:00	1989-01-31 00:00:00
fiModelDesc	TD20	A66	D7G	A62	D3B
fiBaseModel	TD20	A66	D7	A62	D3
fiSecondaryDesc	NaN	NaN	G	NaN	B
fiModelSeries	NaN	NaN	NaN	NaN	NaN
fiModelDescriptor	NaN	NaN	NaN	NaN	NaN
ProductSize	Medium	NaN	Large	NaN	NaN
fiProductClassDesc	Track Type Tractor, Dozer - 105.0 to 130.0 Hor...	Wheel Loader - 120.0 to 135.0 Horsepower	Track Type Tractor, Dozer - 190.0 to 260.0 Hor...	Wheel Loader - Unidentified	Track Type Tractor, Dozer - 20.0 to 75.0 Horse...
state	Texas	Florida	Florida	Florida	Florida
ProductGroup	TTT	WL	TTT	WL	TTT
ProductGroupDesc	Track Type Tractors	Wheel Loader	Track Type Tractors	Wheel Loader	Track Type Tractors
Drive_System	NaN	NaN	NaN	NaN	NaN
Enclosure	OROPS	OROPS	OROPS	EROPS	OROPS
Forks	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Pad_Type	NaN	NaN	NaN	NaN	NaN
Ride_Control	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Stick	NaN	NaN	NaN	NaN	NaN
Transmission	Direct Drive	NaN	Standard	NaN	Standard
Turbocharged	NaN	NaN	NaN	NaN	NaN
Blade_Extension	NaN	NaN	NaN	NaN	NaN
Blade_Width	NaN	NaN	NaN	NaN	NaN

	205615	274835	141296	212552	62755
Enclosure_Type	NaN	NaN	NaN	NaN	NaN
Engine_Horsepower	NaN	NaN	NaN	NaN	NaN
Hydraulics	2 Valve	2 Valve	2 Valve	2 Valve	2 Valve
Pushblock	NaN	NaN	NaN	NaN	NaN
Ripper	None or Unspecified	NaN	None or Unspecified	NaN	None or Unspecified
Scarifier	NaN	NaN	NaN	NaN	NaN
Tip_Control	NaN	NaN	NaN	NaN	NaN
Tire_Size	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Coupler	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Coupler_System	NaN	NaN	NaN	NaN	NaN
Grouser_Tracks	NaN	NaN	NaN	NaN	NaN
Hydraulics_Flow	NaN	NaN	NaN	NaN	NaN
Track_Type	NaN	NaN	NaN	NaN	NaN
Undercarriage_Pad_Width	NaN	NaN	NaN	NaN	NaN
Stick_Length	NaN	NaN	NaN	NaN	NaN
Thumb	NaN	NaN	NaN	NaN	NaN
Pattern_Changer	NaN	NaN	NaN	NaN	NaN
Grouser_Type	NaN	NaN	NaN	NaN	NaN
Backhoe_Mounting	None or Unspecified	NaN	None or Unspecified	NaN	None or Unspecified
Blade_Type	Straight	NaN	Straight	NaN	PAT
Travel_Controls	None or Unspecified	NaN	None or Unspecified	NaN	Lever
Differential_Type	NaN	Standard	NaN	Standard	NaN
Steering_Controls	NaN	Conventional	NaN	Conventional	NaN
saleYear	1989	1989	1989	1989	1989
saleMonth	1	1	1	1	1
saleDay	17	31	31	31	31
saleDayOfWeek	1	1	1	1	1
saleDayOfYear	17	31	31	31	31

In [139]:
```
## Now we have saleDay, saleMonth, saleYear, So we don't require saledate anymore
df_tmp.drop(labels="saledate",axis=1,inplace=True)
```

In [140]:
```
df_tmp.head(20).T
```

Out[140]:

	205615	274835	141296	212552	62755	5
SalesID	1646770	1821514	1505138	1671174	1329056	130
SalePrice	9500	14000	50000	16000	22000	2
MachineID	1126363	1194089	1473654	1327630	1336053	118
ModelID	8434	10150	4139	8591	4089	
datasource	132	132	132	132	132	
auctioneerID	18	99	99	99	99	
YearMade	1974	1980	1978	1980	1984	
MachineHoursCurrentMeter	NaN	NaN	NaN	NaN	NaN	
UsageBand	NaN	NaN	NaN	NaN	NaN	
fiModelDesc	TD20	A66	D7G	A62	D3B	
fiBaseModel	TD20	A66	D7	A62	D3	

In [141]:
```
df_tmp.saledate
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-141-0b8254724953> in <module>
----> 1 df_tmp.saledate

~\AppData\Roaming\Python\Python37\site-packages\pandas\core\generic.py in __get
attr__(self, name)
   5139             if self._info_axis._can_hold_identifiers_and_holds_name(nam
e):
   5140                 return self[name]
-> 5141             return object.__getattribute__(self, name)
   5142 
   5143     def __setattr__(self, name: str, value) -> None:

AttributeError: 'DataFrame' object has no attribute 'saledate'
```

```
In [144]: df_tmp.state.value_counts()
```

```
Out[144]: Florida           67320
          Texas             53110
          California        29761
          Washington        16222
          Georgia           14633
          Maryland          13322
          Mississippi       13240
          Ohio              12369
          Illinois          11540
          Colorado          11529
          New Jersey        11156
          North Carolina    10636
          Tennessee         10298
          Alabama           10292
          Pennsylvania      10234
          South Carolina     9951
          Arizona            9364
          New York           8639
          Connecticut        8276
          Minnesota          7885
          Missouri           7178
          Nevada             6932
          Louisiana          6627
          Kentucky           5351
          Maine              5096
          Indiana            4124
          Arkansas           3933
          New Mexico         3631
          Utah               3046
          Unspecified        2801
          Wisconsin          2745
          New Hampshire      2738
          Virginia           2353
          Idaho              2025
          Oregon             1911
          Michigan           1831
          Wyoming            1672
          Montana            1336
          Iowa               1336
          Oklahoma           1326
          Nebraska            866
          West Virginia       840
          Kansas              667
          Delaware            510
          North Dakota        480
          Alaska              430
          Massachusetts       347
          Vermont             300
          South Dakota        244
          Hawaii              118
          Rhode Island         83
          Puerto Rico          42
          Washington DC         2
          Name: state, dtype: int64
```

```
In [145]: len(df_tmp)
```

```
Out[145]: 412698
```

5. Modelling

We have done enough EDA (we could always do more) but let's start to do some model-driven EDA

```
In [146]: ## Let's build a machine learning model
          from sklearn.ensemble import RandomForestRegressor
          model=RandomForestRegressor(n_jobs=1,
                                      random_state=42)  # same as np.seed(42)
          model.fit(df_tmp.drop("SalePrice", axis=1), df_tmp["SalePrice"])
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-146-b3ff0195a76f> in <module>
      3 model=RandomForestRegressor(n_jobs=1,
      4                             random_state=42)  # same as np.seed(42)
----> 5 model.fit(df_tmp.drop("SalePrice", axis=1), df_tmp["SalePrice"])

~\AppData\Roaming\Python\Python37\site-packages\sklearn\ensemble\_forest.py in
fit(self, X, y, sample_weight)
    303         )
    304         X, y = self._validate_data(X, y, multi_output=True,
--> 305                                    accept_sparse="csc", dtype=DTYPE)
    306         if sample_weight is not None:
    307             sample_weight = _check_sample_weight(sample_weight, X)

~\AppData\Roaming\Python\Python37\site-packages\sklearn\base.py in _validate_da
ta(self, X, y, reset, validate_separately, **check_params)
    431                 y = check_array(y, **check_y_params)
    432             else:
--> 433                 X, y = check_X_y(X, y, **check_params)
    434             out = X, y
    435 

~\AppData\Roaming\Python\Python37\site-packages\sklearn\utils\validation.py in
inner_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

~\AppData\Roaming\Python\Python37\site-packages\sklearn\utils\validation.py in
check_X_y(X, y, accept_sparse, accept_large_sparse, dtype, order, copy, force_a
ll_finite, ensure_2d, allow_nd, multi_output, ensure_min_samples, ensure_min_fe
atures, y_numeric, estimator)
    876                     ensure_min_samples=ensure_min_samples,
    877                     ensure_min_features=ensure_min_features,
--> 878                     estimator=estimator)
    879     if multi_output:
    880         y = check_array(y, accept_sparse='csr', force_all_finite=True,

~\AppData\Roaming\Python\Python37\site-packages\sklearn\utils\validation.py in
inner_f(*args, **kwargs)
     61             extra_args = len(args) - len(all_args)
     62             if extra_args <= 0:
---> 63                 return f(*args, **kwargs)
     64 
     65             # extra_args > 0

~\AppData\Roaming\Python\Python37\site-packages\sklearn\utils\validation.py in
check_array(array, accept_sparse, accept_large_sparse, dtype, order, copy, forc
e_all_finite, ensure_2d, allow_nd, ensure_min_samples, ensure_min_features, est
imator)
    671                     array = array.astype(dtype, casting="unsafe", copy=
False)
    672                 else:
--> 673                     array = np.asarray(array, order=order, dtype=dtype)
    674             except ComplexWarning as complex_warning:
    675                 raise ValueError("Complex data not supported\n"

~\AppData\Roaming\Python\Python37\site-packages\numpy\core\_asarray.py in asarr
ay(a, dtype, order)
     81
     82     """
---> 83     return array(a, dtype, copy=False, order=order)
     84
     85

~\AppData\Roaming\Python\Python37\site-packages\pandas\core\generic.py in __arr
ay__(self, dtype)
   1779
   1780     def __array__(self, dtype=None) -> np.ndarray:
-> 1781         return np.asarray(self._values, dtype=dtype)
   1782
   1783     def __array_wrap__(self, result, context=None):

~\AppData\Roaming\Python\Python37\site-packages\numpy\core\_asarray.py in asarr
ay(a, dtype, order)
     81
     82     """
---> 83     return array(a, dtype, copy=False, order=order)
     84
     85

ValueError: could not convert string to float: 'Low'
```

In [147]:
```
df_tmp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 412698 entries, 205615 to 409203
Data columns (total 57 columns):
 #   Column                    Non-Null Count   Dtype  
---  ------                    --------------   -----  
 0   SalesID                   412698 non-null  int64  
 1   SalePrice                 412698 non-null  float64
 2   MachineID                 412698 non-null  int64  
 3   ModelID                   412698 non-null  int64  
 4   datasource                412698 non-null  int64  
 5   auctioneerID              392562 non-null  float64
 6   YearMade                  412698 non-null  int64  
 7   MachineHoursCurrentMeter  147504 non-null  float64
 8   UsageBand                 73670 non-null   object 
 9   fiModelDesc               412698 non-null  object 
 10  fiBaseModel               412698 non-null  object 
 11  fiSecondaryDesc           271971 non-null  object 
 12  fiModelSeries             58667 non-null   object 
 13  fiModelDescriptor         74816 non-null   object 
 14  ProductSize               196093 non-null  object 
 15  fiProductClassDesc        412698 non-null  object 
 16  state                     412698 non-null  object 
 17  ProductGroup              412698 non-null  object 
 18  ProductGroupDesc          412698 non-null  object 
 19  Drive_System              107087 non-null  object 
 20  Enclosure                 412364 non-null  object 
 21  Forks                     197715 non-null  object 
 22  Pad_Type                  81096 non-null   object 
 23  Ride_Control              152728 non-null  object 
 24  Stick                     81096 non-null   object 
 25  Transmission              188007 non-null  object 
 26  Turbocharged              81096 non-null   object 
 27  Blade_Extension           25983 non-null   object 
 28  Blade_Width               25983 non-null   object 
 29  Enclosure_Type            25983 non-null   object 
 30  Engine_Horsepower         25983 non-null   object 
 31  Hydraulics                330133 non-null  object 
 32  Pushblock                 25983 non-null   object 
 33  Ripper                    106945 non-null  object 
 34  Scarifier                 25994 non-null   object 
 35  Tip_Control               25983 non-null   object 
 36  Tire_Size                 97638 non-null   object 
 37  Coupler                   220679 non-null  object 
 38  Coupler_System            44974 non-null   object 
 39  Grouser_Tracks            44875 non-null   object 
 40  Hydraulics_Flow           44875 non-null   object 
 41  Track_Type                102193 non-null  object 
 42  Undercarriage_Pad_Width   102916 non-null  object 
 43  Stick_Length              102261 non-null  object 
 44  Thumb                     102332 non-null  object 
 45  Pattern_Changer           102261 non-null  object 
 46  Grouser_Type              102193 non-null  object 
 47  Backhoe_Mounting          80712 non-null   object 
 48  Blade_Type                81875 non-null   object 
 49  Travel_Controls           81877 non-null   object 
 50  Differential_Type         71564 non-null   object
 51  Steering_Controls         71522 non-null   object
 52  saleYear                  412698 non-null  int64
 53  saleMonth                 412698 non-null  int64
 54  saleDay                   412698 non-null  int64
 55  saleDayOfWeek             412698 non-null  int64
 56  saleDayOfYear             412698 non-null  int64
dtypes: float64(3), int64(10), object(44)
memory usage: 182.6+ MB
```

```
In [148]: df_tmp["UsageBand"].dtype

Out[148]: dtype('O')
```

In [149]:
```
df_tmp.isna().sum()
```

Out[149]:
```
SalesID                          0
SalePrice                        0
MachineID                        0
ModelID                          0
datasource                       0
auctioneerID                 20136
YearMade                         0
MachineHoursCurrentMeter    265194
UsageBand                   339028
fiModelDesc                      0
fiBaseModel                      0
fiSecondaryDesc             140727
fiModelSeries               354031
fiModelDescriptor           337882
ProductSize                 216605
fiProductClassDesc               0
state                            0
ProductGroup                     0
ProductGroupDesc                 0
Drive_System                305611
Enclosure                      334
Forks                       214983
Pad_Type                    331602
Ride_Control                259970
Stick                       331602
Transmission                224691
Turbocharged                331602
Blade_Extension             386715
Blade_Width                 386715
Enclosure_Type              386715
Engine_Horsepower           386715
Hydraulics                   82565
Pushblock                   386715
Ripper                      305753
Scarifier                   386704
Tip_Control                 386715
Tire_Size                   315060
Coupler                     192019
Coupler_System              367724
Grouser_Tracks              367823
Hydraulics_Flow             367823
Track_Type                  310505
Undercarriage_Pad_Width     309782
Stick_Length                310437
Thumb                       310366
Pattern_Changer             310437
Grouser_Type                310505
Backhoe_Mounting            331986
Blade_Type                  330823
Travel_Controls             330821
Differential_Type           341134
Steering_Controls           341176
saleYear                         0
saleMonth                        0
saleDay                          0
```

```
saleDayOfWeek                    0
saleDayOfYear                    0
dtype: int64
```

Convert String to Categorical data

One way we can turn all our data into numbers is by converting them into pandas categories.

```
In [150]: pd.api.types.is_string_dtype(df_tmp["UsageBand"])
```

```
Out[150]: True
```

In [151]:
```python
## Getting all the column names which has dtype as string

for label,content in df_tmp.items():
        if(pd.api.types.is_string_dtype(content)):
            print(label)
```

```
UsageBand
fiModelDesc
fiBaseModel
fiSecondaryDesc
fiModelSeries
fiModelDescriptor
ProductSize
fiProductClassDesc
state
ProductGroup
ProductGroupDesc
Drive_System
Enclosure
Forks
Pad_Type
Ride_Control
Stick
Transmission
Turbocharged
Blade_Extension
Blade_Width
Enclosure_Type
Engine_Horsepower
Hydraulics
Pushblock
Ripper
Scarifier
Tip_Control
Tire_Size
Coupler
Coupler_System
Grouser_Tracks
Hydraulics_Flow
Track_Type
Undercarriage_Pad_Width
Stick_Length
Thumb
Pattern_Changer
Grouser_Type
Backhoe_Mounting
Blade_Type
Travel_Controls
Differential_Type
Steering_Controls
```

In [152]:
```
# This will turn all the string values into categorical values

for label, content in df_tmp.items():
    if pd.api.types.is_string_dtype(content):
        df_tmp[label]=content.astype("category").cat.as_ordered()
```

```
In [153]: df_tmp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 412698 entries, 205615 to 409203
Data columns (total 57 columns):
 #   Column                    Non-Null Count   Dtype   
---  ------                    --------------   -----   
 0   SalesID                   412698 non-null  int64   
 1   SalePrice                 412698 non-null  float64 
 2   MachineID                 412698 non-null  int64   
 3   ModelID                   412698 non-null  int64   
 4   datasource                412698 non-null  int64   
 5   auctioneerID              392562 non-null  float64 
 6   YearMade                  412698 non-null  int64   
 7   MachineHoursCurrentMeter  147504 non-null  float64 
 8   UsageBand                 73670 non-null   category
 9   fiModelDesc               412698 non-null  category
 10  fiBaseModel               412698 non-null  category
 11  fiSecondaryDesc           271971 non-null  category
 12  fiModelSeries             58667 non-null   category
 13  fiModelDescriptor         74816 non-null   category
 14  ProductSize               196093 non-null  category
 15  fiProductClassDesc        412698 non-null  category
 16  state                     412698 non-null  category
 17  ProductGroup              412698 non-null  category
 18  ProductGroupDesc          412698 non-null  category
 19  Drive_System              107087 non-null  category
 20  Enclosure                 412364 non-null  category
 21  Forks                     197715 non-null  category
 22  Pad_Type                  81096 non-null   category
 23  Ride_Control              152728 non-null  category
 24  Stick                     81096 non-null   category
 25  Transmission              188007 non-null  category
 26  Turbocharged              81096 non-null   category
 27  Blade_Extension           25983 non-null   category
 28  Blade_Width               25983 non-null   category
 29  Enclosure_Type            25983 non-null   category
 30  Engine_Horsepower         25983 non-null   category
 31  Hydraulics                330133 non-null  category
 32  Pushblock                 25983 non-null   category
 33  Ripper                    106945 non-null  category
 34  Scarifier                 25994 non-null   category
 35  Tip_Control               25983 non-null   category
 36  Tire_Size                 97638 non-null   category
 37  Coupler                   220679 non-null  category
 38  Coupler_System            44974 non-null   category
 39  Grouser_Tracks            44875 non-null   category
 40  Hydraulics_Flow           44875 non-null   category
 41  Track_Type                102193 non-null  category
 42  Undercarriage_Pad_Width   102916 non-null  category
 43  Stick_Length              102261 non-null  category
 44  Thumb                     102332 non-null  category
 45  Pattern_Changer           102261 non-null  category
 46  Grouser_Type              102193 non-null  category
 47  Backhoe_Mounting          80712 non-null   category
 48  Blade_Type                81875 non-null   category
 49  Travel_Controls           81877 non-null   category
 50  Differential_Type         71564 non-null   category
 51  Steering_Controls         71522 non-null   category
 52  saleYear                  412698 non-null  int64
 53  saleMonth                 412698 non-null  int64
 54  saleDay                   412698 non-null  int64
 55  saleDayOfWeek             412698 non-null  int64
 56  saleDayOfYear             412698 non-null  int64
dtypes: category(44), float64(3), int64(10)
memory usage: 63.3 MB
```

In [154]:
```
df_tmp.state.cat.categories
```

Out[154]:
```
Index(['Alabama', 'Alaska', 'Arizona', 'Arkansas', 'California', 'Colorado',
       'Connecticut', 'Delaware', 'Florida', 'Georgia', 'Hawaii', 'Idaho',
       'Illinois', 'Indiana', 'Iowa', 'Kansas', 'Kentucky', 'Louisiana',
       'Maine', 'Maryland', 'Massachusetts', 'Michigan', 'Minnesota',
       'Mississippi', 'Missouri', 'Montana', 'Nebraska', 'Nevada',
       'New Hampshire', 'New Jersey', 'New Mexico', 'New York',
       'North Carolina', 'North Dakota', 'Ohio', 'Oklahoma', 'Oregon',
       'Pennsylvania', 'Puerto Rico', 'Rhode Island', 'South Carolina',
       'South Dakota', 'Tennessee', 'Texas', 'Unspecified', 'Utah', 'Vermont',
       'Virginia', 'Washington', 'Washington DC', 'West Virginia', 'Wisconsin',
       'Wyoming'],
      dtype='object')
```

In [155]:
```
df_tmp.state.cat.codes
```

Out[155]:
```
205615    43
274835     8
141296     8
212552     8
62755      8
          ..
410879     4
412476     4
411927     4
407124     4
409203     4
Length: 412698, dtype: int8
```

Save preprocessed data

In [156]:
```
# Export current tmp dataFrame
df_tmp.to_csv("data/train_tmp.csv",index=False)
```

In [157]:
```
# Import preprocessed data
df_tmp=pd.read_csv("data/train_tmp.csv",
                   low_memory=False)
```

In [158]:
```
df_tmp.head().T
```

Out[158]:

	0	1	2	3	4
SalesID	1646770	1821514	1505138	1671174	1329056
SalePrice	9500	14000	50000	16000	22000
MachineID	1126363	1194089	1473654	1327630	1336053
ModelID	8434	10150	4139	8591	4089
datasource	132	132	132	132	132
auctioneerID	18	99	99	99	99
YearMade	1974	1980	1978	1980	1984
MachineHoursCurrentMeter	NaN	NaN	NaN	NaN	NaN
UsageBand	NaN	NaN	NaN	NaN	NaN
fiModelDesc	TD20	A66	D7G	A62	D3B
fiBaseModel	TD20	A66	D7	A62	D3
fiSecondaryDesc	NaN	NaN	G	NaN	B
fiModelSeries	NaN	NaN	NaN	NaN	NaN
fiModelDescriptor	NaN	NaN	NaN	NaN	NaN
ProductSize	Medium	NaN	Large	NaN	NaN
fiProductClassDesc	Track Type Tractor, Dozer - 105.0 to 130.0 Hor...	Wheel Loader - 120.0 to 135.0 Horsepower	Track Type Tractor, Dozer - 190.0 to 260.0 Hor...	Wheel Loader - Unidentified	Track Type Tractor, Dozer - 20.0 to 75.0 Horse...
state	Texas	Florida	Florida	Florida	Florida
ProductGroup	TTT	WL	TTT	WL	TTT
ProductGroupDesc	Track Type Tractors	Wheel Loader	Track Type Tractors	Wheel Loader	Track Type Tractors
Drive_System	NaN	NaN	NaN	NaN	NaN
Enclosure	OROPS	OROPS	OROPS	EROPS	OROPS
Forks	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Pad_Type	NaN	NaN	NaN	NaN	NaN
Ride_Control	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Stick	NaN	NaN	NaN	NaN	NaN
Transmission	Direct Drive	NaN	Standard	NaN	Standard
Turbocharged	NaN	NaN	NaN	NaN	NaN
Blade_Extension	NaN	NaN	NaN	NaN	NaN
Blade_Width	NaN	NaN	NaN	NaN	NaN
Enclosure_Type	NaN	NaN	NaN	NaN	NaN

	0	1	2	3	4
Engine_Horsepower	NaN	NaN	NaN	NaN	NaN
Hydraulics	2 Valve	2 Valve	2 Valve	2 Valve	2 Valve
Pushblock	NaN	NaN	NaN	NaN	NaN
Ripper	None or Unspecified	NaN	None or Unspecified	NaN	None or Unspecified
Scarifier	NaN	NaN	NaN	NaN	NaN
Tip_Control	NaN	NaN	NaN	NaN	NaN
Tire_Size	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Coupler	NaN	None or Unspecified	NaN	None or Unspecified	NaN
Coupler_System	NaN	NaN	NaN	NaN	NaN
Grouser_Tracks	NaN	NaN	NaN	NaN	NaN
Hydraulics_Flow	NaN	NaN	NaN	NaN	NaN
Track_Type	NaN	NaN	NaN	NaN	NaN
Undercarriage_Pad_Width	NaN	NaN	NaN	NaN	NaN
Stick_Length	NaN	NaN	NaN	NaN	NaN
Thumb	NaN	NaN	NaN	NaN	NaN
Pattern_Changer	NaN	NaN	NaN	NaN	NaN
Grouser_Type	NaN	NaN	NaN	NaN	NaN
Backhoe_Mounting	None or Unspecified	NaN	None or Unspecified	NaN	None or Unspecified
Blade_Type	Straight	NaN	Straight	NaN	PAT
Travel_Controls	None or Unspecified	NaN	None or Unspecified	NaN	Lever
Differential_Type	NaN	Standard	NaN	Standard	NaN
Steering_Controls	NaN	Conventional	NaN	Conventional	NaN
saleYear	1989	1989	1989	1989	1989
saleMonth	1	1	1	1	1
saleDay	17	31	31	31	31
saleDayOfWeek	1	1	1	1	1
saleDayOfYear	17	31	31	31	31

In [159]:
```
df_tmp.isnull().sum()
```

Out[159]:
```
SalesID                          0
SalePrice                        0
MachineID                        0
ModelID                          0
datasource                       0
auctioneerID                 20136
YearMade                         0
MachineHoursCurrentMeter    265194
UsageBand                   339028
fiModelDesc                      0
fiBaseModel                      0
fiSecondaryDesc             140727
fiModelSeries               354031
fiModelDescriptor           337882
ProductSize                 216605
fiProductClassDesc               0
state                            0
ProductGroup                     0
ProductGroupDesc                 0
Drive_System                305611
Enclosure                      334
Forks                       214983
Pad_Type                    331602
Ride_Control                259970
Stick                       331602
Transmission                224691
Turbocharged                331602
Blade_Extension             386715
Blade_Width                 386715
Enclosure_Type              386715
Engine_Horsepower           386715
Hydraulics                   82565
Pushblock                   386715
Ripper                      305753
Scarifier                   386704
Tip_Control                 386715
Tire_Size                   315060
Coupler                     192019
Coupler_System              367724
Grouser_Tracks              367823
Hydraulics_Flow             367823
Track_Type                  310505
Undercarriage_Pad_Width     309782
Stick_Length                310437
Thumb                       310366
Pattern_Changer             310437
Grouser_Type                310505
Backhoe_Mounting            331986
Blade_Type                  330823
Travel_Controls             330821
Differential_Type           341134
Steering_Controls           341176
saleYear                         0
saleMonth                        0
saleDay                          0
```

```
saleDayOfWeek                   0
saleDayOfYear                   0
dtype: int64
```

Fill the missing values

First fill the numeric missing values

In [160]:
```
# First check the numerical columns
for label,content in df_tmp.items():
    if pd.api.types.is_numeric_dtype(content):
        print(label)
```

```
SalesID
SalePrice
MachineID
ModelID
datasource
auctioneerID
YearMade
MachineHoursCurrentMeter
saleYear
saleMonth
saleDay
saleDayOfWeek
saleDayOfYear
```

In [161]:
```
# Check for which numeric columns have null values
for label, content in df_tmp.items():
    if pd.api.types.is_numeric_dtype(content):
        if pd.isnull(content).sum():
            print(label)
```

```
auctioneerID
MachineHoursCurrentMeter
```

In [162]:
```
# Fill numeric rows with median
for label, content in df_tmp.items():
    if pd.api.types.is_numeric_dtype(content):
        if pd.isnull(content).sum():
            # Add a binary column which tells us if the data was missing or not
            df_tmp[label+"_is_missing"]=pd.isnull(content)
            # Fill missing numeric values with median
            df_tmp[label]=content.fillna(content.median())
```

In [164]:
```
# Check if there is null numeric value
for label, content in df_tmp.items():
    if pd.api.types.is_numeric_dtype(content):
        if pd.isnull(content).sum():
            print(label)
```

In [165]:
```
# Checkk to see how many examples were missing
df_tmp.auctioneerID_is_missing.value_counts()
```

Out[165]:
```
False    392562
True      20136
Name: auctioneerID_is_missing, dtype: int64
```

```
In [166]: df_tmp.isna().sum()
```

```
Out[166]: SalesID                          0
          SalePrice                        0
          MachineID                        0
          ModelID                          0
          datasource                       0
          auctioneerID                     0
          YearMade                         0
          MachineHoursCurrentMeter         0
          UsageBand                   339028
          fiModelDesc                      0
          fiBaseModel                      0
          fiSecondaryDesc             140727
          fiModelSeries               354031
          fiModelDescriptor           337882
          ProductSize                 216605
          fiProductClassDesc               0
          state                            0
          ProductGroup                     0
          ProductGroupDesc                 0
          Drive_System                305611
          Enclosure                      334
          Forks                       214983
          Pad_Type                    331602
          Ride_Control                259970
          Stick                       331602
          Transmission                224691
          Turbocharged                331602
          Blade_Extension             386715
          Blade_Width                 386715
          Enclosure_Type              386715
          Engine_Horsepower           386715
          Hydraulics                   82565
          Pushblock                   386715
          Ripper                      305753
          Scarifier                   386704
          Tip_Control                 386715
          Tire_Size                   315060
          Coupler                     192019
          Coupler_System              367724
          Grouser_Tracks              367823
          Hydraulics_Flow             367823
          Track_Type                  310505
          Undercarriage_Pad_Width     309782
          Stick_Length                310437
          Thumb                       310366
          Pattern_Changer             310437
          Grouser_Type                310505
          Backhoe_Mounting            331986
          Blade_Type                  330823
          Travel_Controls             330821
          Differential_Type           341134
          Steering_Controls           341176
          saleYear                         0
          saleMonth                        0
          saleDay                          0
```

```
saleDayOfWeek                               0
saleDayOfYear                               0
auctioneerID_is_missing                     0
MachineHoursCurrentMeter_is_missing         0
dtype: int64
```

Filling and turning categorical values with numerical values

In [167]:
```
# Find the categorical column with null values
for label, content in df_tmp.items():
    if not pd.api.types.is_numeric_dtype(content):
        if pd.isnull(content).sum():
            print(label)
```

```
UsageBand
fiSecondaryDesc
fiModelSeries
fiModelDescriptor
ProductSize
Drive_System
Enclosure
Forks
Pad_Type
Ride_Control
Stick
Transmission
Turbocharged
Blade_Extension
Blade_Width
Enclosure_Type
Engine_Horsepower
Hydraulics
Pushblock
Ripper
Scarifier
Tip_Control
Tire_Size
Coupler
Coupler_System
Grouser_Tracks
Hydraulics_Flow
Track_Type
Undercarriage_Pad_Width
Stick_Length
Thumb
Pattern_Changer
Grouser_Type
Backhoe_Mounting
Blade_Type
Travel_Controls
Differential_Type
Steering_Controls
```

In [168]:
```
pd.Categorical(df_tmp["UsageBand"]).codes+1
```

Out[168]:
```
array([0, 0, 0, ..., 0, 0, 0], dtype=int8)
```

In [169]:
```
# Turn categorical values into number and fill missing values
for label, content in df_tmp.items():
    if not pd.api.types.is_numeric_dtype(content):
        # Add binary columns to indicate whether sample had missing values or not
        df_tmp[label+"_is_missing"]=pd.isnull(content)
        # Turn categories into numbers and add +1
        df_tmp[label]=pd.Categorical(content).codes+1
```

In [170]:
```
df_tmp.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 412698 entries, 0 to 412697
Columns: 103 entries, SalesID to Steering_Controls_is_missing
dtypes: bool(46), float64(3), int16(4), int64(10), int8(40)
memory usage: 77.9 MB
```

In [171]:
```
df_tmp.head().T
```

Out[171]:

	0	1	2	3	4
SalesID	1646770	1821514	1505138	1671174	1329056
SalePrice	9500	14000	50000	16000	22000
MachineID	1126363	1194089	1473654	1327630	1336053
ModelID	8434	10150	4139	8591	4089
datasource	132	132	132	132	132
...	...	...	...	...	...
Backhoe_Mounting_is_missing	False	True	False	True	False
Blade_Type_is_missing	False	True	False	True	False
Travel_Controls_is_missing	False	True	False	True	False
Differential_Type_is_missing	True	False	True	False	True
Steering_Controls_is_missing	True	False	True	False	True

103 rows × 5 columns

In [172]:
```
df_tmp.isna().sum()
```

Out[172]:
```
SalesID                        0
SalePrice                      0
MachineID                      0
ModelID                        0
datasource                     0
                              ..
Backhoe_Mounting_is_missing    0
Blade_Type_is_missing          0
Travel_Controls_is_missing     0
Differential_Type_is_missing   0
Steering_Controls_is_missing   0
Length: 103, dtype: int64
```

Now let's build the Machine Learning model

In [173]:
```
len(df_tmp)
```

Out[173]:
```
412698
```

In [174]:
```
df_tmp.head()
```

Out[174]:

	SalesID	SalePrice	MachineID	ModelID	datasource	auctioneerID	YearMade	MachineHoursCur
0	1646770	9500.0	1126363	8434	132	18.0	1974	
1	1821514	14000.0	1194089	10150	132	99.0	1980	
2	1505138	50000.0	1473654	4139	132	99.0	1978	
3	1671174	16000.0	1327630	8591	132	99.0	1980	
4	1329056	22000.0	1336053	4089	132	99.0	1984	

5 rows × 103 columns

In [175]:
```
%%time
# Instantiate model
model = RandomForestRegressor(n_jobs=-1,
                              random_state=42)
# Fit the model
model.fit(df_tmp.drop("SalePrice", axis=1), df_tmp["SalePrice"])
```

```
Wall time: 7min 44s
```

Out[175]:
```
RandomForestRegressor(n_jobs=-1, random_state=42)
```

In [176]:
```
# Score the model
model.score(df_tmp.drop("SalePrice", axis=1), df_tmp["SalePrice"])
```

Out[176]:
```
0.9875468079970562
```

Split the data in train and valid sets (Plzz see the data)

In [177]:
```
df_tmp.saleYear
```

Out[177]:
```
0         1989
1         1989
2         1989
3         1989
4         1989
          ...
412693    2012
412694    2012
412695    2012
412696    2012
412697    2012
Name: saleYear, Length: 412698, dtype: int64
```

In [178]:
```
df_tmp.saleYear.value_counts()
```

Out[178]:
```
2009    43849
2008    39767
2011    35197
2010    33390
2007    32208
2006    21685
2005    20463
2004    19879
2001    17594
2000    17415
2002    17246
2003    15254
1998    13046
1999    12793
2012    11573
1997     9785
1996     8829
1995     8530
1994     7929
1993     6303
1992     5519
1991     5109
1989     4806
1990     4529
Name: saleYear, dtype: int64
```

In [179]:
```
df_valid = df_tmp[df_tmp.saleYear==2012]
df_train = df_tmp[df_tmp.saleYear!=2012]
len(df_valid),len(df_train)
```

Out[179]:
```
(11573, 401125)
```

In [180]:
```
# Split the train and valid data in X and y
X_train, y_train = df_train.drop("SalePrice", axis=1), df_train["SalePrice"]
X_valid, y_valid = df_valid.drop("SalePrice", axis=1), df_valid["SalePrice"]
X_train.shape, y_train.shape, X_valid.shape, y_valid.shape
```

Out[180]:
```
((401125, 102), (401125,), (11573, 102), (11573,))
```

In [181]:
```
y_train
```

Out[181]:
```
0          9500.0
1         14000.0
2         50000.0
3         16000.0
4         22000.0
           ...
401120    29000.0
401121    11000.0
401122    11000.0
401123    18000.0
401124    13500.0
Name: SalePrice, Length: 401125, dtype: float64
```

Build an Evaluation Metrics

In [182]:
```
#  Create a evaluatio metrics i.e  RMSLE
from sklearn.metrics import mean_squared_log_error, mean_absolute_error, r2_score

def rmsle(y_test, y_preds):
    """
    Calculate root mean sqaured log error between prediction and true labels.
    """
    return np.sqrt(mean_squared_log_error(y_test, y_preds))

# Create function to evaluate model on a few different levels.
def show_score(model):
    train_preds=model.predict(X_train)
    valid_preds=model.predict(X_valid)
    scores={"Training MAE":mean_absolute_error(y_train,train_preds),
            "Valid MAE":mean_absolute_error(y_valid,valid_preds),
            "Training RMSLE":rmsle(y_train,train_preds),
            "Valid RMSLE":rmsle(y_valid,valid_preds),
            "Training R^2":r2_score(y_train,train_preds),
            "Valid R^2":r2_score(y_valid,valid_preds)}
    return scores
```

Testing our model on Subset (to tune the hyperparameters)

In [183]:
```
# This takes far too long .... for experimenting

# %%time
# model=RandomForestRegressor(n_jobs=-1,
#                            random_state=42)
#  model.fit(X_train,y_train)
```

In [184]:
```
len(X_train)
```

Out[184]:
```
401125
```

In [185]:
```
# Change max samples value
from sklearn.ensemble import RandomForestRegressor
model=RandomForestRegressor(n_jobs=-1,
                            random_state=42,
                            max_samples=10000)
```

In [186]:
```
%%time
# Cutting down on the max number of samples each estimator can see improve traini
model.fit(X_train, y_train)
```

```
Wall time: 25.4 s
```

Out[186]:
```
RandomForestRegressor(max_samples=10000, n_jobs=-1, random_state=42)
```

In [187]:
```
show_score(model)
```

Out[187]:
```
{'Training MAE': 5561.2988092240585,
 'Valid MAE': 7177.26365505919,
 'Training RMSLE': 0.257745378256977,
 'Valid RMSLE': 0.29362638671089003,
 'Training R^2': 0.8606658995199189,
 'Valid R^2': 0.8320374995090507}
```

Hyperparameter tuning with RandomizedSearchCV

In [188]:
```python
%%time
from sklearn.model_selection import RandomizedSearchCV

# Different RandomForestRegressor hyperparameters
rf_grid={"n_estimators":np.arange(10,100,10),
         "max_depth":[None,3,5,10],
         "min_samples_split":np.arange(2,20,2),
         "min_samples_leaf":np.arange(1,20,2),
         "max_features":[0.5,1,"sqrt","auto"],
         "max_samples":[10000]}

# Instantiate RandomizedSearchCV model
rs_model = RandomizedSearchCV(RandomForestRegressor(n_jobs=-1,
                                                     random_state=42),
                              param_distributions=rf_grid,
                              n_iter=2,
                              cv=5,
                              verbose=True)

# Fit the model
rs_model.fit(X_train,y_train)
```

```
Fitting 5 folds for each of 2 candidates, totalling 10 fits
Wall time: 3min 25s
```

Out[188]:
```
RandomizedSearchCV(cv=5,
                   estimator=RandomForestRegressor(n_jobs=-1, random_state=42),
                   n_iter=2,
                   param_distributions={'max_depth': [None, 3, 5, 10],
                                        'max_features': [0.5, 1, 'sqrt',
                                                         'auto'],
                                        'max_samples': [10000],
                                        'min_samples_leaf': array([ 1,  3,  5,
7,  9, 11, 13, 15, 17, 19]),
                                        'min_samples_split': array([ 2,  4,  6,
8, 10, 12, 14, 16, 18]),
                                        'n_estimators': array([10, 20, 30, 40,
50, 60, 70, 80, 90])},
                   verbose=True)
```

In [189]:
```python
# Find the best model hyperparameters
rs_model.best_params_
```

Out[189]:
```
{'n_estimators': 40,
 'min_samples_split': 12,
 'min_samples_leaf': 9,
 'max_samples': 10000,
 'max_features': 'auto',
 'max_depth': None}
```

In [190]:
```
# Evaluate the RandomizedSearchCV model
show_score(rs_model)
```

Out[190]:
```
{'Training MAE': 6086.99131166347,
 'Valid MAE': 7674.949511960607,
 'Training RMSLE': 0.27648998624118665,
 'Valid RMSLE': 0.3072434602008128,
 'Training R^2': 0.8312379190695804,
 'Valid R^2': 0.8002384558444482}
```

Train a model with the best hyperparameters

Note: These were found after 100 iterations of `RandomizedSearchCV`

In [191]:
```
%%time

# Most ideal hyperparameter
ideal_model = RandomForestRegressor(n_estimators=40,
                                    min_samples_leaf=1,
                                    min_samples_split=14,
                                    max_features=0.5,
                                    n_jobs=-1,
                                    max_samples=None,
                                    random_state=42)

# Fit the ideal model
ideal_model.fit(X_train,y_train)
```

```
Wall time: 1min 34s
```

Out[191]:
```
RandomForestRegressor(max_features=0.5, min_samples_split=14, n_estimators=40,
                      n_jobs=-1, random_state=42)
```

In [192]:
```
# Scores for ideal model (trained on all the data)
show_score(ideal_model)
```

Out[192]:
```
{'Training MAE': 2953.8161137163484,
 'Valid MAE': 5951.247761444453,
 'Training RMSLE': 0.14469006962371858,
 'Valid RMSLE': 0.24524163989538328,
 'Training R^2': 0.9588145522577225,
 'Valid R^2': 0.8818019502450094}
```

In [193]:
```
# Scores for rs_model (only trained on 10,000 examples)
show_score(rs_model)
```

Out[193]:
```
{'Training MAE': 6086.99131166347,
 'Valid MAE': 7674.949511960607,
 'Training RMSLE': 0.27648998624118665,
 'Valid RMSLE': 0.3072434602008128,
 'Training R^2': 0.8312379190695804,
 'Valid R^2': 0.8002384558444482}
```

Test our model using test data

In [203]:
```python
# Import the test data
df_test=pd.read_csv("data/Test.csv",
                    low_memory=False,
                    parse_dates=["saledate"])
df_test.head().T
```

Out[203]:

	0	1	2	3	4
SalesID	1227829	1227844	1227847	1227848	1227863
MachineID	1006309	1022817	1031560	56204	1053887
ModelID	3168	7271	22805	1269	22312
datasource	121	121	121	121	121
auctioneerID	3	3	3	3	3
YearMade	1999	1000	2004	2006	2005
MachineHoursCurrentMeter	3688	28555	6038	8940	2286
UsageBand	Low	High	Medium	High	Low
saledate	2012-05-03 00:00:00	2012-05-10 00:00:00	2012-05-10 00:00:00	2012-05-10 00:00:00	2012-05-10 00:00:00
fiModelDesc	580G	936	EC210BLC	330CL	650K
fiBaseModel	580	936	EC210	330	650
fiSecondaryDesc	G	NaN	B	C	K
fiModelSeries	NaN	NaN	NaN	NaN	NaN
fiModelDescriptor	NaN	NaN	LC	L	NaN
ProductSize	NaN	Medium	Large / Medium	Large / Medium	NaN
fiProductClassDesc	Backhoe Loader - 14.0 to 15.0 Ft Standard Digg...	Wheel Loader - 135.0 to 150.0 Horsepower	Hydraulic Excavator, Track - 21.0 to 24.0 Metr...	Hydraulic Excavator, Track - 33.0 to 40.0 Metr...	Track Type Tractor, Dozer - 20.0 to 75.0 Horse...
state	Wyoming	Virginia	New Jersey	New Jersey	Florida
ProductGroup	BL	WL	TEX	TEX	TTT
ProductGroupDesc	Backhoe Loaders	Wheel Loader	Track Excavators	Track Excavators	Track Type Tractors
Drive_System	Two Wheel Drive	NaN	NaN	NaN	NaN
Enclosure	OROPS	EROPS	EROPS w AC	EROPS w AC	OROPS
Forks	Yes	Yes	NaN	NaN	NaN
Pad_Type	None or Unspecified	NaN	NaN	NaN	NaN
Ride_Control	No	None or Unspecified	NaN	NaN	NaN
Stick	Standard	NaN	NaN	NaN	NaN
Transmission	Standard	NaN	NaN	NaN	Hydrostatic

	0	1	2	3	4
Turbocharged	None or Unspecified	NaN	NaN	NaN	NaN
Blade_Extension	NaN	NaN	NaN	NaN	NaN
Blade_Width	NaN	NaN	NaN	NaN	NaN
Enclosure_Type	NaN	NaN	NaN	NaN	NaN
Engine_Horsepower	NaN	NaN	NaN	NaN	NaN
Hydraulics	NaN	2 Valve	Auxiliary	Standard	2 Valve
Pushblock	NaN	NaN	NaN	NaN	NaN
Ripper	NaN	NaN	NaN	NaN	None or Unspecified
Scarifier	NaN	NaN	NaN	NaN	NaN
Tip_Control	NaN	NaN	NaN	NaN	NaN
Tire_Size	NaN	20.5	NaN	NaN	NaN
Coupler	NaN	None or Unspecified	None or Unspecified	None or Unspecified	NaN
Coupler_System	NaN	NaN	NaN	NaN	NaN
Grouser_Tracks	NaN	NaN	NaN	NaN	NaN
Hydraulics_Flow	NaN	NaN	NaN	NaN	NaN
Track_Type	NaN	NaN	Steel	Steel	NaN
Undercarriage_Pad_Width	NaN	NaN	None or Unspecified	None or Unspecified	NaN
Stick_Length	NaN	NaN	9' 6"	None or Unspecified	NaN
Thumb	NaN	NaN	Manual	Manual	NaN
Pattern_Changer	NaN	NaN	None or Unspecified	Yes	NaN
Grouser_Type	NaN	NaN	Double	Triple	NaN
Backhoe_Mounting	NaN	NaN	NaN	NaN	None or Unspecified
Blade_Type	NaN	NaN	NaN	NaN	PAT
Travel_Controls	NaN	NaN	NaN	NaN	None or Unspecified
Differential_Type	NaN	Standard	NaN	NaN	NaN
Steering_Controls	NaN	Conventional	NaN	NaN	NaN

```
In [204]: df_test.saledate[:1000]
```

```
Out[204]: 0     2012-05-03
          1     2012-05-10
          2     2012-05-10
          3     2012-05-10
          4     2012-05-10
                   ...
          995   2012-06-12
          996   2012-06-12
          997   2012-05-01
          998   2012-05-01
          999   2012-06-05
          Name: saledate, Length: 1000, dtype: datetime64[ns]
```

In [205]:
```
df_test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12457 entries, 0 to 12456
Data columns (total 52 columns):
 #   Column                    Non-Null Count  Dtype         
---  ------                    --------------  -----         
 0   SalesID                   12457 non-null  int64         
 1   MachineID                 12457 non-null  int64         
 2   ModelID                   12457 non-null  int64         
 3   datasource                12457 non-null  int64         
 4   auctioneerID              12457 non-null  int64         
 5   YearMade                  12457 non-null  int64         
 6   MachineHoursCurrentMeter  2129 non-null   float64       
 7   UsageBand                 1834 non-null   object        
 8   saledate                  12457 non-null  datetime64[ns]
 9   fiModelDesc               12457 non-null  object        
 10  fiBaseModel               12457 non-null  object        
 11  fiSecondaryDesc           8482 non-null   object        
 12  fiModelSeries             2006 non-null   object        
 13  fiModelDescriptor         3024 non-null   object        
 14  ProductSize               6048 non-null   object        
 15  fiProductClassDesc        12457 non-null  object        
 16  state                     12457 non-null  object        
 17  ProductGroup              12457 non-null  object        
 18  ProductGroupDesc          12457 non-null  object        
 19  Drive_System              2759 non-null   object        
 20  Enclosure                 12455 non-null  object        
 21  Forks                     6308 non-null   object        
 22  Pad_Type                  2108 non-null   object        
 23  Ride_Control              4241 non-null   object        
 24  Stick                     2108 non-null   object        
 25  Transmission              4818 non-null   object        
 26  Turbocharged              2108 non-null   object        
 27  Blade_Extension           651 non-null    object        
 28  Blade_Width               651 non-null    object        
 29  Enclosure_Type            651 non-null    object        
 30  Engine_Horsepower         651 non-null    object        
 31  Hydraulics                10315 non-null  object        
 32  Pushblock                 651 non-null    object        
 33  Ripper                    2704 non-null   object        
 34  Scarifier                 651 non-null    object        
 35  Tip_Control               651 non-null    object        
 36  Tire_Size                 2778 non-null   object        
 37  Coupler                   7601 non-null   object        
 38  Coupler_System            2066 non-null   object        
 39  Grouser_Tracks            2066 non-null   object        
 40  Hydraulics_Flow           2066 non-null   object        
 41  Track_Type                3394 non-null   object        
 42  Undercarriage_Pad_Width   3398 non-null   object        
 43  Stick_Length              3394 non-null   object        
 44  Thumb                     3395 non-null   object        
 45  Pattern_Changer           3394 non-null   object        
 46  Grouser_Type              3394 non-null   object        
 47  Backhoe_Mounting          2051 non-null   object        
 48  Blade_Type                2058 non-null   object        
 49  Travel_Controls           2058 non-null   object        
 50  Differential_Type         2129 non-null   object
 51  Steering_Controls         2129 non-null   object
dtypes: datetime64[ns](1), float64(1), int64(6), object(44)
memory usage: 4.9+ MB
```

```
In [206]: df_test.isna().sum()
```

```
Out[206]: SalesID                         0
          MachineID                       0
          ModelID                         0
          datasource                      0
          auctioneerID                    0
          YearMade                        0
          MachineHoursCurrentMeter    10328
          UsageBand                   10623
          saledate                        0
          fiModelDesc                     0
          fiBaseModel                     0
          fiSecondaryDesc              3975
          fiModelSeries               10451
          fiModelDescriptor            9433
          ProductSize                  6409
          fiProductClassDesc              0
          state                           0
          ProductGroup                    0
          ProductGroupDesc                0
          Drive_System                 9698
          Enclosure                       2
          Forks                        6149
          Pad_Type                    10349
          Ride_Control                 8216
          Stick                       10349
          Transmission                 7639
          Turbocharged                10349
          Blade_Extension             11806
          Blade_Width                 11806
          Enclosure_Type              11806
          Engine_Horsepower           11806
          Hydraulics                   2142
          Pushblock                   11806
          Ripper                       9753
          Scarifier                   11806
          Tip_Control                 11806
          Tire_Size                    9679
          Coupler                      4856
          Coupler_System              10391
          Grouser_Tracks              10391
          Hydraulics_Flow             10391
          Track_Type                   9063
          Undercarriage_Pad_Width      9059
          Stick_Length                 9063
          Thumb                        9062
          Pattern_Changer              9063
          Grouser_Type                 9063
          Backhoe_Mounting            10406
          Blade_Type                  10399
          Travel_Controls             10399
          Differential_Type           10328
          Steering_Controls           10328
          dtype: int64
```

Preprocessing the data (getting the test dataset in the same format as our training dataset)

In [207]:
```
def preprocess_data(df):
    """
    Performs some transformationon df and returns transformed df.
    """
    df["saleYear"]=df.saledate.dt.year
    df["saleMonth"]=df.saledate.dt.month
    df["saleDay"]=df.saledate.dt.day
    df["saleDayOfWeek"]=df.saledate.dt.dayofweek
    df["saleDayOfYear"]=df.saledate.dt.dayofyear

    df.drop(labels="saledate",axis=1,inplace=True)

    # Fill the numeric rows with median
    for label,content in df.items():
        if pd.api.types.is_numeric_dtype(content):
            if pd.isnull(content).sum():
                # Add a binary column which tells us if the data was missing or r
                df[label+"_is_missing"]=pd.isnull(content)
                # Fill missing numeric values with median
                df[label]=content.fillna(content.median())


    # Fill the categorical missing data and turn categories into numbers
        if not pd.api.types.is_numeric_dtype(content):
            df[label+"_is_missing"]=pd.isnull(content)
            df[label]=pd.Categorical(content).codes+1

    return df
```

In [208]:
```
# Preprocess the test data
df_test=preprocess_data(df_test)
df_test.head()
```

Out[208]:

	SalesID	MachineID	ModelID	datasource	auctioneerID	YearMade	MachineHoursCurrentMeter
0	1227829	1006309	3168	121	3	1999	3688.0
1	1227844	1022817	7271	121	3	1000	28555.0
2	1227847	1031560	22805	121	3	2004	6038.0
3	1227848	56204	1269	121	3	2006	8940.0
4	1227863	1053887	22312	121	3	2005	2286.0

5 rows × 101 columns

```
In [209]: # Make predicytions on update test data
          test_preds=ideal_model.predict(df_test)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-209-2cc34114db25> in <module>
      1 # Make predicytions on update test data
----> 2 test_preds=ideal_model.predict(df_test)

~\AppData\Roaming\Python\Python37\site-packages\sklearn\ensemble\_forest.py in
predict(self, X)
    782         check_is_fitted(self)
    783         # Check data
--> 784         X = self._validate_X_predict(X)
    785 
    786         # Assign chunk of trees to jobs

~\AppData\Roaming\Python\Python37\site-packages\sklearn\ensemble\_forest.py in
_validate_X_predict(self, X)
    420         check_is_fitted(self)
    421 
--> 422         return self.estimators_[0]._validate_X_predict(X, check_input=T
rue)
    423 
    424     @property

~\AppData\Roaming\Python\Python37\site-packages\sklearn\tree\_classes.py in _va
lidate_X_predict(self, X, check_input)
    406         if check_input:
    407             X = self._validate_data(X, dtype=DTYPE, accept_sparse="cs
r",
--> 408                                     reset=False)
    409             if issparse(X) and (X.indices.dtype != np.intc or
    410                                 X.indptr.dtype != np.intc):

~\AppData\Roaming\Python\Python37\site-packages\sklearn\base.py in _validate_da
ta(self, X, y, reset, validate_separately, **check_params)
    435 
    436         if check_params.get('ensure_2d', True):
--> 437             self._check_n_features(X, reset=reset)
    438 
    439         return out

~\AppData\Roaming\Python\Python37\site-packages\sklearn\base.py in _check_n_fea
tures(self, X, reset)
    364         if n_features != self.n_features_in_:
    365             raise ValueError(
--> 366                 f"X has {n_features} features, but {self.__class__.__na
me__} "
    367                 f"is expecting {self.n_features_in_} features as inpu
t.")
    368 

ValueError: X has 101 features, but DecisionTreeRegressor is expecting 102 feat
ures as input.
```

In [210]:
```
X_train.head()
```

Out[210]:

	SalesID	MachineID	ModelID	datasource	auctioneerID	YearMade	MachineHoursCurrentMeter
0	1646770	1126363	8434	132	18.0	1974	0.0
1	1821514	1194089	10150	132	99.0	1980	0.0
2	1505138	1473654	4139	132	99.0	1978	0.0
3	1671174	1327630	8591	132	99.0	1980	0.0
4	1329056	1336053	4089	132	99.0	1984	0.0

5 rows × 102 columns

In [211]:
```
# we can find how the columns differ using sets
set(X_train.columns)-set(df_test.columns)
```

Out[211]:
```
{'auctioneerID_is_missing'}
```

In [212]:
```
# Manually adjust df_test to have auctioneerID_is_missing column
df_test["auctioneerID_is_missing"]=False
df_test.head()
```

Out[212]:

	SalesID	MachineID	ModelID	datasource	auctioneerID	YearMade	MachineHoursCurrentMeter
0	1227829	1006309	3168	121	3	1999	3688.0
1	1227844	1022817	7271	121	3	1000	28555.0
2	1227847	1031560	22805	121	3	2004	6038.0
3	1227848	56204	1269	121	3	2006	8940.0
4	1227863	1053887	22312	121	3	2005	2286.0

5 rows × 102 columns

In [213]:
```
# Make predicitions on the test data
test_preds=ideal_model.predict(df_test)
```

In [214]:
```
test_preds
```

Out[214]:
```
array([20614.36780887, 19897.80170658, 44852.21959446, ...,
       14296.98620472, 22164.85757662, 31683.80063427])
```

```
In [215]: # Format predicitons into the same format kaggle is asking
          df_preds=pd.DataFrame()
          df_preds["SalesID"]=df_test["SalesID"]
          df_preds["SalesPrice"]=test_preds
          df_preds
```

```
Out[215]:
              SalesID     SalesPrice
          0   1227829   20614.367809
          1   1227844   19897.801707
          2   1227847   44852.219594
          3   1227848   68346.325323
          4   1227863   39487.349708
        ...       ...            ...
      12452   6643171   46466.092910
      12453   6643173   17500.493352
      12454   6643184   14296.986205
      12455   6643186   22164.857577
      12456   6643196   31683.800634

12457 rows × 2 columns
```

```
In [216]: # Export predictive data
          df_preds.to_csv("data/test_predicitions.csv", index=False)
```

```
In [ ]:
```