new car data

```
import numpy as np
import pandas as pd

# Data Visulaization Libraries
import matplotlib.pyplot as plt
import seaborn as sns


%matplotlib inline
```

```
df = pd.read_csv('data.csv')
```

```
df.head()
```

	Make	Model	Year	Engine Fuel Type	Engine HP	Engine Cylinders	Transmission Type	Driven_Wheels	Number of Doors	Market Category	Vehicle Size	Vehicle Style	highwa MP
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Coupe	2
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convertible	2
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Coupe	2
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Coupe	2
4	BMW	1 Series	2011	premium unleaded	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convertible	2

```
df.shape
```

```
(11914, 16)
```

```
df.describe()
```

	Year	Engine HP	Engine Cylinders	Number of Doors	highway MPG	city mpg	Popularity	MSRP
count	11914.000000	11845.00000	11884.000000	11908.000000	11914.000000	11914.000000	11914.000000	1.191400e+04
mean	2010.384338	249.38607	5.628829	3.436093	26.637485	19.733255	1554.911197	4.059474e+04
std	7.579740	109.19187	1.780559	0.881315	8.863001	8.987798	1441.855347	6.010910e+04
min	1990.000000	55.00000	0.000000	2.000000	12.000000	7.000000	2.000000	2.000000e+03
25%	2007.000000	170.00000	4.000000	2.000000	22.000000	16.000000	549.000000	2.100000e+04
50%	2015.000000	227.00000	6.000000	4.000000	26.000000	18.000000	1385.000000	2.999500e+04
75%	2016.000000	300.00000	6.000000	4.000000	30.000000	22.000000	2009.000000	4.223125e+04
max	2017.000000	1001.00000	16.000000	4.000000	354.000000	137.000000	5657.000000	2.065902e+06

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11914 entries, 0 to 11913
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Make               11914 non-null  object
 1   Model              11914 non-null  object
 2   Year               11914 non-null  int64
 3   Engine Fuel Type   11911 non-null  object
 4   Engine HP          11845 non-null  float64
 5   Engine Cylinders   11884 non-null  float64
 6   Transmission Type  11914 non-null  object
 7   Driven_Wheels      11914 non-null  object
 8   Number of Doors    11908 non-null  float64
 9   Market Category    8172 non-null   object
```

```
 10  Vehicle Size       11914 non-null  object
 11  Vehicle Style      11914 non-null  object
 12  highway MPG        11914 non-null  int64
 13  city mpg           11914 non-null  int64
 14  Popularity         11914 non-null  int64
 15  MSRP               11914 non-null  int64
dtypes: float64(3), int64(5), object(8)
memory usage: 1.5+ MB
```

```
df.dtypes
```

	0
Make	object
Model	object
Year	int64
Engine Fuel Type	object
Engine HP	float64
Engine Cylinders	float64
Transmission Type	object
Driven_Wheels	object
Number of Doors	float64
Market Category	object
Vehicle Size	object
Vehicle Style	object
highway MPG	int64
city mpg	int64
Popularity	int64
MSRP	int64

dtype: object

```
df.columns
```

```
Index(['Make', 'Model', 'Year', 'Engine Fuel Type', 'Engine HP',
       'Engine Cylinders', 'Transmission Type', 'Driven_Wheels',
       'Number of Doors', 'Market Category', 'Vehicle Size', 'Vehicle Style',
       'highway MPG', 'city mpg', 'Popularity', 'MSRP'],
      dtype='object')
```

```
df.rename(columns={
    'Make': 'Brand',
    'Engine Fuel Type': 'Fuel_Type',
    'Engine HP': 'Horsepower',
    'Engine Cylinders': 'Cylinders',
    'Transmission Type': 'Transmission',
    'Driven_Wheels': 'Drive_Type',
    'Number of Doors': 'Doors',
    'Market Category': 'Market_Category',
    'Vehicle Size': 'Vehicle_Size',
    'Vehicle Style': 'Vehicle_Style',
    'highway MPG': 'Highway_MPG',
    'city mpg': 'city_mpg',
    'MSRP': 'Price'
}, inplace=True)
```

```
df.head()
```

	Brand	Model	Year	Fuel_Type	Horsepower	Cylinders	Transmission	Drive_Type	Doors	Market_Category	Vehicle_Size	Vehicle_Sty
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	2.0	Factory Tuner,Luxury,High-Performance	Compact	Co
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Convert
2	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,High-Performance	Compact	Co
3	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury,Performance	Compact	Co
4	BMW	1 Series	2011	premium unleaded (required)	230.0	6.0	MANUAL	rear wheel drive	2.0	Luxury	Compact	Convert

```
df.columns
```

```
Index(['Brand', 'Model', 'Year', 'Fuel_Type', 'Horsepower', 'Cylinders',
       'Transmission', 'Drive_Type', 'Doors', 'Market_Category',
       'Vehicle_Size', 'Vehicle_Style', 'Highway_MPG', 'city_mpg',
       'Popularity', 'Price'],
      dtype='object')
```

```
numerical_cols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()
categorical_cols = df.select_dtypes(include=['object']).columns.tolist()
other_cols = df.select_dtypes(exclude=['int64', 'float64', 'object']).columns.tolist()

# Output the classified columns
print("Numerical Columns:", numerical_cols)
print("Categorical Columns:", categorical_cols)
print("Other Columns (e.g., DateTime):", other_cols)
```

```
Numerical Columns: ['Year', 'Horsepower', 'Cylinders', 'Doors', 'Highway_MPG', 'city_mpg', 'Popularity', 'Price']
Categorical Columns: ['Brand', 'Model', 'Fuel_Type', 'Transmission', 'Drive_Type', 'Market_Category', 'Vehicle_Size', 'Vehicle_Style
Other Columns (e.g., DateTime): []
```

```
float_columns = df.select_dtypes(include=['float']).columns
print(float_columns)
```

```
Index(['Horsepower', 'Cylinders', 'Doors'], dtype='object')
```

```
df.isnull().sum()
```

	0
Brand	0
Model	0
Year	0
Fuel_Type	3
Horsepower	69
Cylinders	30
Transmission	0
Drive_Type	0
Doors	6
Market_Category	3742
Vehicle_Size	0
Vehicle_Style	0
Highway_MPG	0
city_mpg	0
Popularity	0
Price	0

dtype: int64

```
df = df.drop(columns =['Market_Category', 'Doors'])
```

```
df.columns
```

```
Index(['Brand', 'Model', 'Year', 'Fuel_Type', 'Horsepower', 'Cylinders',
       'Transmission', 'Drive_Type', 'Vehicle_Size', 'Vehicle_Style',
       'Highway_MPG', 'city_mpg', 'Popularity', 'Price'],
      dtype='object')
```

```
# Remove rows where "Cylinders" column values are empty
df.dropna(subset = ['Cylinders'], inplace=True)
```

```
df[df['Fuel_Type'].isnull()]
```

	Brand	Model	Year	Fuel_Type	Horsepower	Cylinders	Transmission	Drive_Type	Vehicle_Size	Vehicle_Style	Highway_MPG	ci
11321	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	Midsize	Sedan	25	
11322	Suzuki	Verona	2004	NaN	155.0	6.0	AUTOMATIC	front wheel drive	Midsize	Sedan	25	

```
# Fill the 3 empty values of Fuel Type column with most frequent values
df['Fuel_Type'] = df['Fuel_Type'].fillna(df['Fuel_Type'].mode()[0])
```

```
df['Horsepower'].mode()[0]
```

```
200.0
```

```
df['Horsepower'].describe()
```

	Horsepower
count	11816.000000
mean	249.514472
std	109.261297
min	55.000000
25%	170.000000
50%	227.000000
75%	300.000000
max	1001.000000

dtype: float64

```
# Fill the null values in "Horsepower" columns with median / 50% of data
df['Horsepower'] = df['Horsepower'].fillna(df['Horsepower'].median())
```

```
# Lets Check null values now
df.isnull().sum()
```

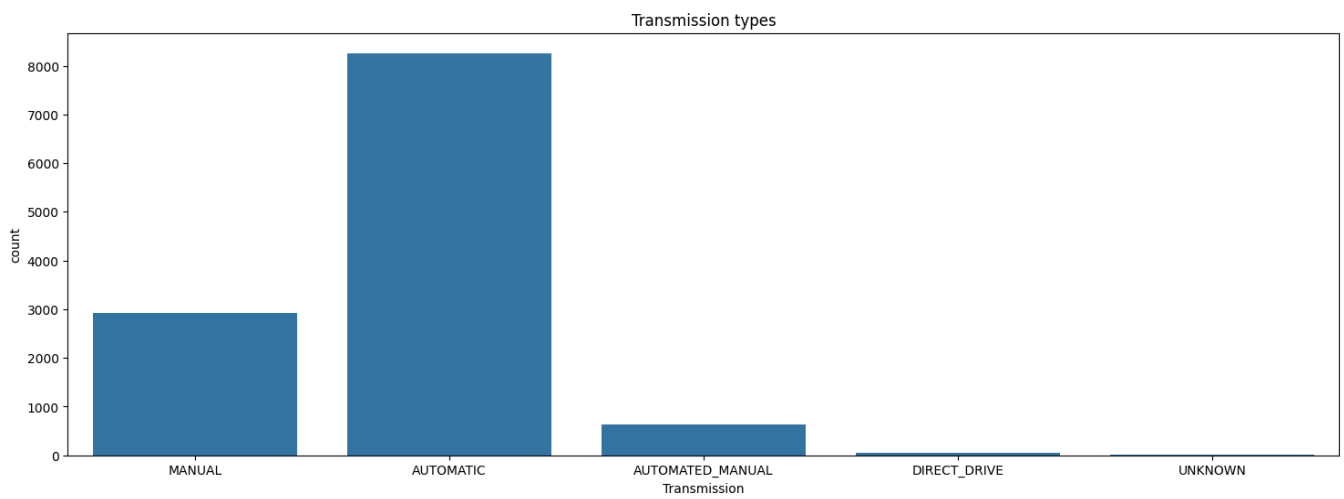
	0
Brand	0
Model	0
Year	0
Fuel_Type	0
Horsepower	0
Cylinders	0
Transmission	0
Drive_Type	0
Vehicle_Size	0
Vehicle_Style	0
Highway_MPG	0
city_mpg	0
Popularity	0
Price	0

dtype: int64

```
df.shape
```

```
(11884, 14)
```

```
# Count the occurrences of each transmission type in dataset
plt.figure(figsize=(18, 6))
sns.countplot(x='Transmission', data=df)
plt.title('Transmission types')
plt.show()
```




```
df[df['Transmission'] == 'UNKNOWN'].shape
```

```
(19, 14)
```

```
# Remove rows where the Transmission type is unknown
df = df[df['Transmission'] != 'UNKNOWN']
```

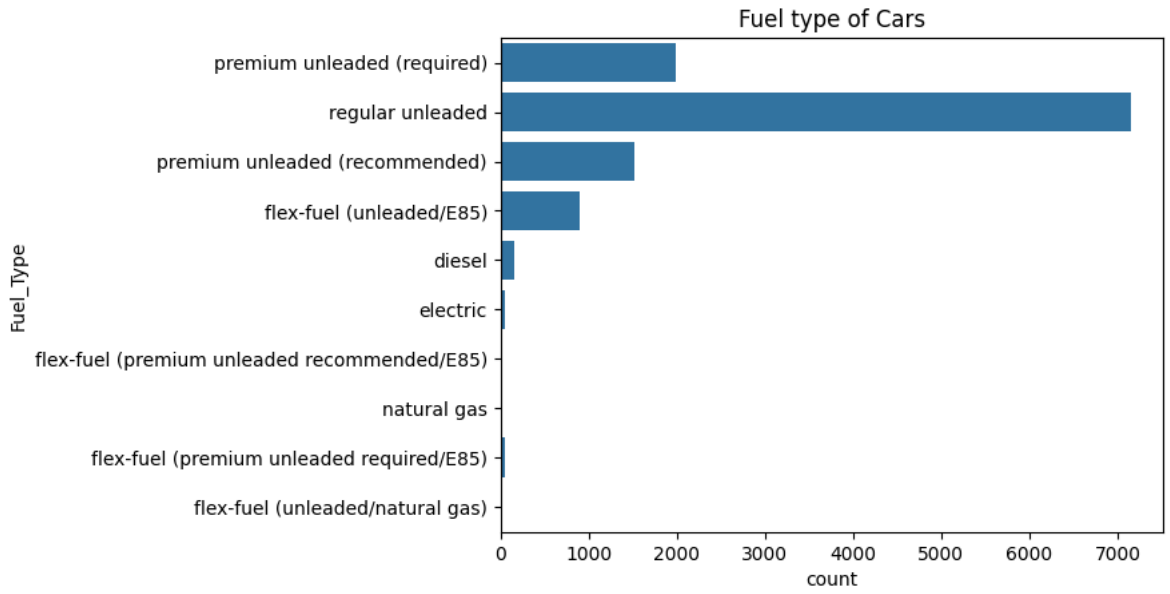
```
df.shape
```

```
(11865, 14)
```

```
df['Fuel_Type'].unique()
```

```
array(['premium unleaded (required)', 'regular unleaded',
       'premium unleaded (recommended)', 'flex-fuel (unleaded/E85)',
       'diesel', 'electric',
       'flex-fuel (premium unleaded recommended/E85)', 'natural gas',
       'flex-fuel (premium unleaded required/E85)',
       'flex-fuel (unleaded/natural gas)'], dtype=object)
```

```
# Count the occurrences of each fuel type in the dataset
sns.countplot(y='Fuel_Type', data=df)
plt.title('Fuel type of Cars')
plt.show()
```



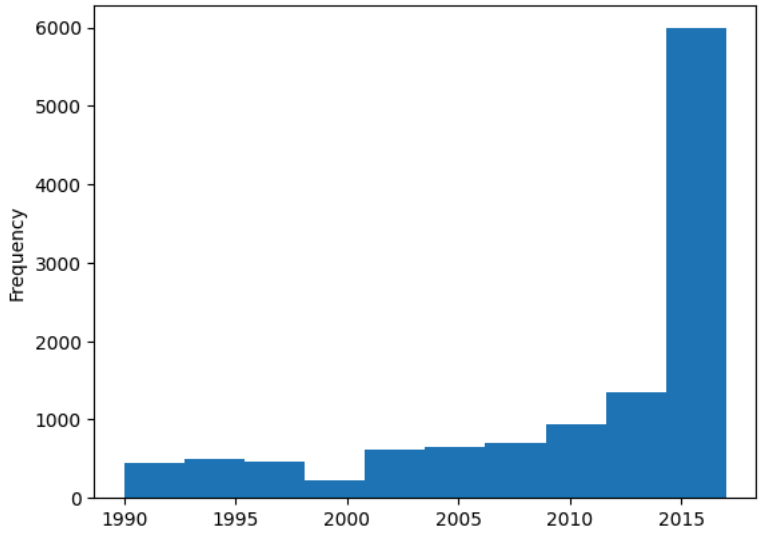

```
df['Fuel_Type'].value_counts()
```

Fuel_Type	count
regular unleaded	7153
premium unleaded (required)	1992
premium unleaded (recommended)	1523
flex-fuel (unleaded/E85)	899
diesel	154
electric	56
flex-fuel (premium unleaded required/E85)	54
flex-fuel (premium unleaded recommended/E85)	26
flex-fuel (unleaded/natural gas)	6
natural gas	2

dtype: int64

```
df['Year'].plot(kind='hist')
```

```
<Axes: ylabel='Frequency'>
```




```
# Count the occurrences of each drive type in descending order
drive_type = df['Drive_Type'].value_counts(ascending=False).reset_index()
drive_type
```

	Drive_Type	count
0	front wheel drive	4776
1	rear wheel drive	3335
2	all wheel drive	2353
3	four wheel drive	1401

```
df.columns
df.info()
df.describe()
df.head()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 11865 entries, 0 to 11913
Data columns (total 14 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Brand          11865 non-null  object
 1   Model          11865 non-null  object
 2   Year           11865 non-null  int64
 3   Fuel_Type      11865 non-null  object
 4   Horsepower     11865 non-null  float64
 5   Cylinders      11865 non-null  float64
 6   Transmission   11865 non-null  object
 7   Drive_Type     11865 non-null  object
 8   Vehicle_Size   11865 non-null  object
 9   Vehicle_Style  11865 non-null  object
 10  Highway_MPG    11865 non-null  int64
 11  city_mpg       11865 non-null  int64
 12  Popularity     11865 non-null  int64
 13  Price          11865 non-null  int64
dtypes: float64(2), int64(5), object(7)
memory usage: 1.4+ MB
```

	Brand	Model	Year	Fuel_Type	Horsepower	Cylinders	Transmission	Drive_Type	Vehicle_Size	Vehicle_Style	Highway_MPG	city_mpg
0	BMW	1 Series M	2011	premium unleaded (required)	335.0	6.0	MANUAL	rear wheel drive	Compact	Coupe	26	19
1	BMW	1 Series	2011	premium unleaded (required)	300.0	6.0	MANUAL	rear wheel drive	Compact	Convertible	28	19

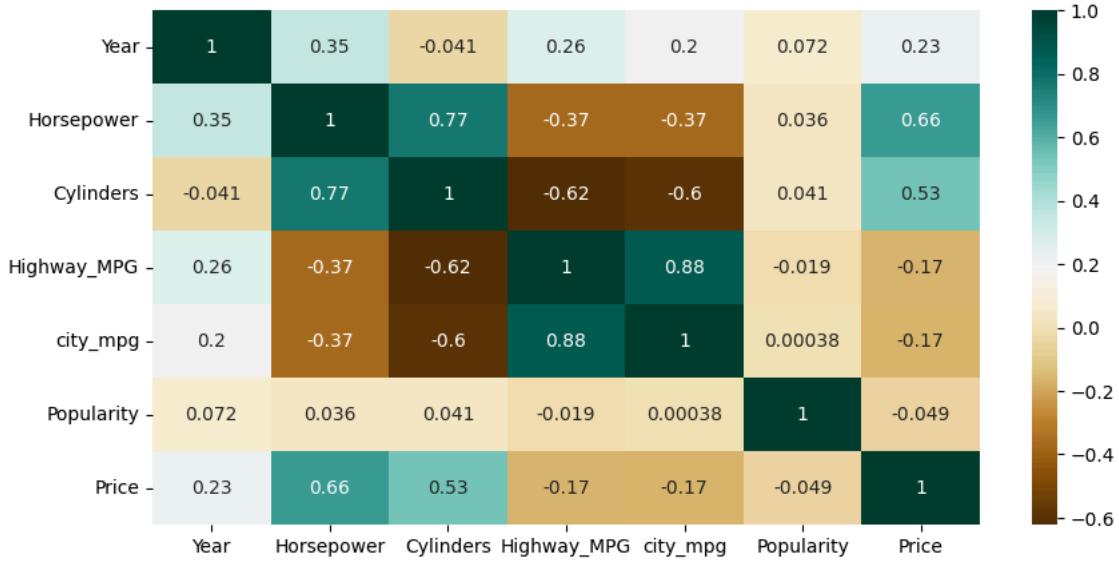
```
df.isnull().sum()
```

	0
Brand	0
Model	0
Year	0
Fuel_Type	0
Horsepower	0
Cylinders	0
Transmission	0
Drive_Type	0
Vehicle_Size	0
Vehicle_Style	0
Highway_MPG	0
city_mpg	0
Popularity	0
Price	0

dtype: int64

```
plt.figure(figsize=(10,5))
# Select only numerical features for correlation calculation
numerical_df = df.select_dtypes(include=['number'])
c= numerical_df.corr()
sns.heatmap(c,cmap="BrBG",annot=True)
c
```

	Year	Horsepower	Cylinders	Highway_MPG	city_mpg	Popularity	Price
Year	1.000000	0.350301	-0.041344	0.259944	0.201516	0.072474	0.226382
Horsepower	0.350301	1.000000	0.768329	-0.365298	-0.366172	0.036168	0.661576
Cylinders	-0.041344	0.768329	1.000000	-0.621835	-0.600898	0.041287	0.531804
Highway_MPG	0.259944	-0.365298	-0.621835	1.000000	0.880434	-0.019182	-0.165256
city_mpg	0.201516	-0.366172	-0.600898	0.880434	1.000000	0.000380	-0.166608
Popularity	0.072474	0.036168	0.041287	-0.019182	0.000380	1.000000	-0.049226
Price	0.226382	0.661576	0.531804	-0.165256	-0.166608	-0.049226	1.000000




```
from sklearn.model_selection import train_test_split

train_df, test_df = train_test_split(df, test_size=0.2, random_state=42)

# Print the shapes of the resulting DataFrames
print("Training data shape:", train_df.shape)
print("Testing data shape:", test_df.shape)
```

```
Training data shape: (9492, 14)
Testing data shape: (2373, 14)
```