Fuel_Economy_Dataset

July 28, 2020

[111]:
```
# load libraries
import pandas as pd
import matplotlib.pyplot as plt
```

[4]:
```
#load datasets
df_08 = pd.read_csv(r"C:\Users\noama\all_alpha_08.csv")
df_18 = pd.read_csv(r"C:\Users\noama\all_alpha_18.csv")
```

[10]:
```
#number of rows (samples) and columns (variables) in 2008 dataset
df_08.shape
```

[10]:
```
(2404, 18)
```

[11]:
```
#number of duplicate rows in 2008 dataset
sum(df_08.duplicated())
```

[11]:
```
25
```

[12]:
```
#number of rows with missing data in 2008 dataset
df_08.isnull().sum()
```

[12]:
```
Model                    0
Displ                    0
Cyl                    199
Trans                  199
Drive                   93
Fuel                     0
Sales Area               0
Stnd                     0
Underhood ID             0
Veh Class                0
Air Pollution Score      0
FE Calc Appr           199
City MPG               199
Hwy MPG                199
Cmb MPG                199
Unadj Cmb MPG          199
```

```
Greenhouse Gas Score    199
SmartWay                  0
dtype: int64
```

[13]:
```
#same for 2018 data
print(df_18.shape)
print(sum(df_18.duplicated()))
print(df_18.isnull().sum())
```

```
(1611, 18)
0
Model                   0
Displ                   2
Cyl                     2
Trans                   0
Drive                   0
Fuel                    0
Cert Region             0
Stnd                    0
Stnd Description        0
Underhood ID            0
Veh Class               0
Air Pollution Score     0
City MPG                0
Hwy MPG                 0
Cmb MPG                 0
Greenhouse Gas Score    0
SmartWay                0
Comb CO2                0
dtype: int64
```

[15]:
```
#data types and number of missing values in 2008 dataset
df_08.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2404 entries, 0 to 2403
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Model                 2404 non-null   object
 1   Displ                 2404 non-null   float64
 2   Cyl                   2205 non-null   object
 3   Trans                 2205 non-null   object
 4   Drive                 2311 non-null   object
 5   Fuel                  2404 non-null   object
 6   Sales Area            2404 non-null   object
 7   Stnd                  2404 non-null   object
 8   Underhood ID          2404 non-null   object
```

```
 9   Veh Class              2404 non-null   object
 10  Air Pollution Score    2404 non-null   object
 11  FE Calc Appr           2205 non-null   object
 12  City MPG               2205 non-null   object
 13  Hwy MPG                2205 non-null   object
 14  Cmb MPG                2205 non-null   object
 15  Unadj Cmb MPG          2205 non-null   float64
 16  Greenhouse Gas Score   2205 non-null   object
 17  SmartWay               2404 non-null   object
dtypes: float64(2), object(16)
memory usage: 338.2+ KB
```

[16]:
```
#data types and number of missing values in 2018 dataset
df_18.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1611 entries, 0 to 1610
Data columns (total 18 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Model                  1611 non-null   object
 1   Displ                  1609 non-null   float64
 2   Cyl                    1609 non-null   float64
 3   Trans                  1611 non-null   object
 4   Drive                  1611 non-null   object
 5   Fuel                   1611 non-null   object
 6   Cert Region            1611 non-null   object
 7   Stnd                   1611 non-null   object
 8   Stnd Description       1611 non-null   object
 9   Underhood ID           1611 non-null   object
 10  Veh Class              1611 non-null   object
 11  Air Pollution Score    1611 non-null   int64
 12  City MPG               1611 non-null   object
 13  Hwy MPG                1611 non-null   object
 14  Cmb MPG                1611 non-null   object
 15  Greenhouse Gas Score   1611 non-null   int64
 16  SmartWay               1611 non-null   object
 17  Comb CO2               1611 non-null   object
dtypes: float64(2), int64(2), object(14)
memory usage: 226.7+ KB
```

[18]:
```
#number of unique values in 2008 dataset
df_08.nunique()
```

[18]:
```
Model                  436
Displ                   47
Cyl                      8
```

```
Trans                    14
Drive                     2
Fuel                      5
Sales Area                3
Stnd                     12
Underhood ID            343
Veh Class                 9
Air Pollution Score      13
FE Calc Appr              2
City MPG                 39
Hwy MPG                  43
Cmb MPG                  38
Unadj Cmb MPG           721
Greenhouse Gas Score     20
SmartWay                  2
dtype: int64
```

[19]:
```
#number of unique values in 2008 dataset
df_18.nunique()
```

[19]:
```
Model                   367
Displ                    36
Cyl                       7
Trans                    26
Drive                     2
Fuel                      5
Cert Region               2
Stnd                     19
Stnd Description         19
Underhood ID            230
Veh Class                 9
Air Pollution Score       6
City MPG                 58
Hwy MPG                  62
Cmb MPG                  57
Greenhouse Gas Score     10
SmartWay                  3
Comb CO2                299
dtype: int64
```

[20]:
```
#number of vehicles assigned to each fuel type
df_08.Fuel.value_counts()
```

[20]:
```
Gasoline       2318
ethanol/gas      72
diesel           11
CNG               2
```

```
ethanol           1
Name: Fuel, dtype: int64
```

[21]:
```
#number of vehicles assigned to each fuel type
df_18.Fuel.value_counts()
```

[21]:
```
Gasoline               1492
Ethanol/Gas              55
Diesel                   38
Gasoline/Electricity     24
Electricity               2
Name: Fuel, dtype: int64
```

Drop Superflous Columns

[22]:
```
# drop columns from 2008 dataset
df_08.drop(['Stnd', 'Underhood ID', 'FE Calc Appr', 'Unadj Cmb MPG'], axis=1,␣
 ↪inplace=True)

# confirm changes
df_08.head(1)
```

[22]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Sales Area	Veh Class	Air Pollution Score	City MPG	Hwy MPG	Cmb MPG	Greenhouse Gas Score	SmartWay
0	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	15	20	17	4	no

[23]:
```
# drop columns from 2018 dataset
df_18.drop(['Stnd', 'Stnd Description', 'Underhood ID', 'Comb CO2'], axis=1,␣
 ↪inplace=True)

# confirm changes
df_18.head(1)
```

[23]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Cert Region	Veh Class	Air Pollution Score	City MPG	Hwy MPG	Cmb MPG	Greenhouse Gas Score	SmartWay
0	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	FA	small SUV	3	20	28	23	5	No

Rename Columns

[24]:
```
# rename Sales Area to Cert Region
df_08.rename(columns={'Sales Area':'Cert Region'}, inplace=True)

# confirm changes
df_08.head(1)
```

[24]:
```
        Model  Displ       Cyl    Trans Drive      Fuel Cert Region Veh Class  \
0  ACURA MDX    3.7  (6 cyl)  Auto-S5   4WD  Gasoline          CA       SUV

  Air Pollution Score City MPG Hwy MPG Cmb MPG Greenhouse Gas Score SmartWay
0                   7       15      20      17                    4       no
```

[25]:
```
# replace spaces with underscores and lowercase labels for 2008 dataset
df_08.rename(columns=lambda x: x.strip().lower().replace(" ", "_"),␣
 ↪inplace=True)

# confirm changes
df_08.head(1)
```

[25]:
```
        model  displ       cyl    trans drive      fuel cert_region veh_class  \
0  ACURA MDX    3.7  (6 cyl)  Auto-S5   4WD  Gasoline          CA       SUV

  air_pollution_score city_mpg hwy_mpg cmb_mpg greenhouse_gas_score smartway
0                   7       15      20      17                    4       no
```

[26]:
```
# replace spaces with underscores and lowercase labels for 2018 dataset
df_18.rename(columns=lambda x: x.strip().lower().replace(" ", "_"),␣
 ↪inplace=True)

# confirm changes
df_18.head(1)
```

[26]:
```
        model  displ  cyl       trans drive      fuel cert_region  veh_class  \
0  ACURA RDX    3.5  6.0  SemiAuto-6   2WD  Gasoline          FA  small SUV

   air_pollution_score city_mpg hwy_mpg cmb_mpg  greenhouse_gas_score smartway
0                    3       20      28      23                     5       No
```

[27]:
```
# confirm column labels for 2008 and 2018 datasets are identical
df_08.columns == df_18.columns
```

[27]:
```
array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True])
```

[28]:
```
# make sure they're all identical like this
(df_08.columns == df_18.columns).all()
```

[28]:
```
True
```

Filter by Certification Region

[29]:
```
# filter datasets for rows following California standards
df_08 = df_08.query('cert_region == "CA"')
```

```
df_18 = df_18.query('cert_region == "CA"')
```

[30]:
```
# confirm only certification region is California
df_08['cert_region'].unique()
```

[30]:
```
array(['CA'], dtype=object)
```

[31]:
```
# confirm only certification region is California
df_18['cert_region'].unique()
```

[31]:
```
array(['CA'], dtype=object)
```

[32]:
```
# drop certification region columns form both datasets
df_08.drop('cert_region', axis=1, inplace=True)
df_18.drop('cert_region', axis=1, inplace=True)
```

[33]:
```
df_08.shape
```

[33]:
```
(1084, 13)
```

[34]:
```
df_18.shape
```

[34]:
```
(798, 13)
```

Drop Rows with Missing Values

[35]:
```
# view missing value count for each feature in 2008
df_08.isnull().sum()
```

[35]:
```
model                   0
displ                   0
cyl                    75
trans                  75
drive                  37
fuel                    0
veh_class               0
air_pollution_score     0
city_mpg               75
hwy_mpg                75
cmb_mpg                75
greenhouse_gas_score   75
smartway                0
dtype: int64
```

[36]:
```
# view missing value count for each feature in 2018
df_18.isnull().sum()
```

[36]:
```
model                   0
displ                   1
cyl                     1
trans                   0
drive                   0
fuel                    0
veh_class               0
air_pollution_score     0
city_mpg                0
hwy_mpg                 0
cmb_mpg                 0
greenhouse_gas_score    0
smartway                0
dtype: int64
```

[37]:
```
# drop rows with any null values in both datasets
df_08.dropna(inplace=True)
df_18.dropna(inplace=True)
```

[38]:
```
# checks if any of columns in 2008 have null values
df_08.isnull().sum().any()
```

[38]:
```
False
```

[40]:
```
# checks if any of columns in 2018 have null values
df_18.isnull().sum().any()
```

[40]:
```
False
```

Dedupe Data

[41]:
```
# print number of duplicates in 2008 and 2018 datasets
print(sum(df_08.duplicated()))
print(sum(df_18.duplicated()))
```

```
23
3
```

[42]:
```
# drop duplicates in both datasets
df_08.drop_duplicates(inplace=True)
df_18.drop_duplicates(inplace=True)
```

[43]:
```
# print number of duplicates again to confirm dedupe
print(sum(df_08.duplicated()))
print(sum(df_18.duplicated()))
```

```
0
0
```

[44]:
```
# check value counts for the 2008 cyl column
df_08['cyl'].value_counts()
```

[44]:
```
(6 cyl)     409
(4 cyl)     283
(8 cyl)     199
(5 cyl)      48
(12 cyl)     30
(10 cyl)     14
(2 cyl)       2
(16 cyl)      1
Name: cyl, dtype: int64
```

[45]:
```
# Extract integer from strings in the 2008 cyl column
df_08['cyl'] = df_08['cyl'].str.extract('(\d+)').astype(int)
```

[46]:
```
# Check value counts for 2008 cyl column again to confirm the change
df_08['cyl'].value_counts()
```

[46]:
```
6     409
4     283
8     199
5      48
12     30
10     14
2       2
16      1
Name: cyl, dtype: int64
```

[47]:
```
# convert 2018 cyl column to int
df_18['cyl'] = df_18['cyl'].astype(int)
```

[51]:
```
# search for dual fuel types (hybrids) in 2018 dataset
hb_08 = df_08[df_08['fuel'].str.contains('/')]
hb_08
```

[51]:
```
                    model  displ  cyl    trans drive         fuel  veh_class  \
1550  MERCEDES-BENZ C300    3.0    6  Auto-L7   2WD  ethanol/gas  small car

     air_pollution_score city_mpg hwy_mpg cmb_mpg greenhouse_gas_score  \
1550                 6/4    13/18   19/25   15/21                 7/6

     smartway
1550       no
```

[52]:
```
# search for dual fuel types (hybrids) in 2018 dataset
hb_18 = df_18[df_18['fuel'].str.contains('/')]
```

```
hb_18
```

[52]:

	model	displ	cyl	trans	drive	\
108	BMW 330e	2.0	4	SemiAuto-8	2WD	
160	BMW 530e	2.0	4	SemiAuto-8	2WD	
162	BMW 530e	2.0	4	SemiAuto-8	4WD	
188	BMW 740e	2.0	4	SemiAuto-8	4WD	
382	CHEVROLET Impala	3.6	6	SemiAuto-6	2WD	
394	CHEVROLET Silverado 15	4.3	6	Auto-6	2WD	
396	CHEVROLET Silverado 15	4.3	6	Auto-6	4WD	
398	CHEVROLET Silverado 15	5.3	8	Auto-6	2WD	
428	CHEVROLET Suburban 1500	5.3	8	Auto-6	2WD	
432	CHEVROLET Suburban 1500	5.3	8	Auto-6	4WD	
436	CHEVROLET Tahoe 1500	5.3	8	Auto-6	2WD	
440	CHEVROLET Tahoe 1500	5.3	8	Auto-6	4WD	
454	CHEVROLET Volt	1.5	4	CVT	2WD	
456	CHEVROLET Volt	1.5	4	CVT	2WD	
458	CHRYSLER 300	3.6	6	Auto-8	2WD	
462	CHRYSLER 300	3.6	6	Auto-8	4WD	
492	DODGE Charger	3.6	6	Auto-8	2WD	
496	DODGE Charger	3.6	6	Auto-8	4WD	
605	FORD Fusion Energi Plug-in Hybrid	2.0	4	CVT	2WD	
659	GMC Sierra 15	4.3	6	Auto-6	2WD	
661	GMC Sierra 15	4.3	6	Auto-6	4WD	
663	GMC Sierra 15	5.3	8	Auto-6	2WD	
697	GMC Yukon 1500	5.3	8	Auto-6	2WD	
701	GMC Yukon 1500	5.3	8	Auto-6	4WD	
709	GMC Yukon 1500 XL	5.3	8	Auto-6	2WD	
715	GMC Yukon XL 1500	5.3	8	Auto-6	4WD	
892	JEEP Cherokee	2.4	4	Auto-9	2WD	
896	JEEP Cherokee	2.4	4	Auto-9	4WD	
933	KARMA Revero	2.0	4	Auto-1	2WD	
1162	MERCEDES-BENZ CLA250 4Matic	2.0	4	AutoMan-7	4WD	
1179	MERCEDES-BENZ GLA250 4Matic	2.0	4	AutoMan-7	4WD	
1192	MERCEDES-BENZ GLE350 4Matic	3.5	6	Auto-7	4WD	
1256	MINI Cooper SE Countryman All4	1.5	3	SemiAuto-6	4WD	
1507	TOYOTA Sequoia FFV	5.7	8	SemiAuto-6	4WD	
1517	TOYOTA Tundra FFV	5.7	8	SemiAuto-6	4WD	
1577	VOLVO S90	2.0	4	SemiAuto-8	4WD	
1601	VOLVO XC 60	2.0	4	SemiAuto-8	4WD	
1609	VOLVO XC 90	2.0	4	SemiAuto-8	4WD	

	fuel	veh_class	air_pollution_score	city_mpg	\
108	Gasoline/Electricity	small car	3	28/66	
160	Gasoline/Electricity	small car	7	27/70	
162	Gasoline/Electricity	small car	7	27/66	
188	Gasoline/Electricity	large car	3	25/62	

382	Ethanol/Gas	large car		5	14/18
394	Ethanol/Gas	pickup		5	12/18
396	Ethanol/Gas	pickup		5	12/17
398	Ethanol/Gas	pickup		3	12/16
428	Ethanol/Gas	standard SUV		3	12/16
432	Ethanol/Gas	standard SUV		3	11/16
436	Ethanol/Gas	standard SUV		3	12/16
440	Ethanol/Gas	standard SUV		3	11/16
454	Gasoline/Electricity	small car		3	43/113
456	Gasoline/Electricity	small car		7	43/113
458	Ethanol/Gas	large car		3	14/19
462	Ethanol/Gas	large car		3	13/18
492	Ethanol/Gas	large car		3	14/19
496	Ethanol/Gas	large car		3	13/18
605	Gasoline/Electricity	midsize car		7	43/102
659	Ethanol/Gas	pickup		5	12/18
661	Ethanol/Gas	pickup		5	12/17
663	Ethanol/Gas	pickup		3	12/16
697	Ethanol/Gas	standard SUV		3	12/16
701	Ethanol/Gas	standard SUV		3	11/16
709	Ethanol/Gas	standard SUV		3	12/16
715	Ethanol/Gas	standard SUV		3	11/16
892	Ethanol/Gas	small SUV		3	15/21
896	Ethanol/Gas	small SUV		3	14/21
933	Gasoline/Electricity	small car		1	20/59
1162	Ethanol/Gas	small car		5	17/24
1179	Ethanol/Gas	small SUV		5	17/23
1192	Ethanol/Gas	standard SUV		3	13/18
1256	Gasoline/Electricity	midsize car		3	28/63
1507	Ethanol/Gas	standard SUV		5	9/13
1517	Ethanol/Gas	pickup		5	9/13
1577	Gasoline/Electricity	midsize car		7	26/70
1601	Gasoline/Electricity	small SUV		7	26/60
1609	Gasoline/Electricity	standard SUV		7	26/63

	hwy_mpg	cmb_mpg	greenhouse_gas_score	smartway
108	34/78	30/71	10	Yes
160	31/75	29/72	10	Elite
162	31/68	28/67	10	Elite
188	29/68	27/64	9	Yes
382	20/28	16/22	4	No
394	16/24	14/20	4	No
396	15/22	13/19	3	No
398	17/23	14/19	3	No
428	17/23	14/19	3	No
432	15/22	12/18	3	No
436	17/23	14/19	3	No

```
440    16/22   13/18                    3       No
454    42/99  42/106                   10      Yes
456    42/99  42/106                   10    Elite
458    22/30   17/23                    5       No
462    20/27   16/21                    4       No
492    22/30   17/23                    5       No
496    20/27   16/21                    4       No
605    41/91   42/97                   10    Elite
659    16/24   14/20                    4       No
661    15/22   13/19                    3       No
663    17/23   14/19                    3       No
697    17/23   14/19                    3       No
701    16/22   13/18                    3       No
709    17/23   14/19                    3       No
715    15/22   12/18                    3       No
892    22/30   18/25                    5       No
896    21/28   17/23                    5       No
933    21/61   20/60                   10       No
1162   24/32   20/27                    6       No
1179   23/31   19/26                    5       No
1192   17/22   14/19                    3       No
1256   27/66   27/65                    9      Yes
1507   13/17   10/14                    1       No
1517   12/17   10/15                    2       No
1577   33/72   29/71                   10    Elite
1601   28/58   26/59                   10    Elite
1609   30/61   27/62                   10    Elite
```

[53]:
```
# create two copies of the 2008 hybrids dataframe
df1 = hb_08.copy()  # data on first fuel type of each hybrid vehicle
df2 = hb_08.copy()  # data on second fuel type of each hybrid vehicle
```

[54]:
```
# columns to split by "/"
split_columns = ['fuel', 'air_pollution_score', 'city_mpg', 'hwy_mpg',␣
 ↪'cmb_mpg', 'greenhouse_gas_score']

# apply split function to each column of each dataframe copy
for c in split_columns:
    df1[c] = df1[c].apply(lambda x: x.split("/")[0])
    df2[c] = df2[c].apply(lambda x: x.split("/")[1])
```

[55]:
```
# this dataframe holds info for the FIRST fuel type of the hybrid
df1
```

[55]:
```
                   model  displ  cyl    trans drive     fuel  veh_class  \
1550  MERCEDES-BENZ C300    3.0    6  Auto-L7   2WD  ethanol  small car
```

```
     air_pollution_score city_mpg hwy_mpg cmb_mpg greenhouse_gas_score  \
1550                   6       13      19      15                     7

     smartway
1550       no
```

[57]:
```
# this dataframe holds info for the SECOND fuel type of the hybrid
df2
```

[57]:
```
                   model  displ  cyl    trans drive fuel  veh_class  \
1550  MERCEDES-BENZ C300    3.0    6  Auto-L7   2WD  gas  small car

     air_pollution_score city_mpg hwy_mpg cmb_mpg greenhouse_gas_score  \
1550                   4       18      25      21                     6

     smartway
1550       no
```

[58]:
```
# combine dataframes to add to the original dataframe
new_rows = df1.append(df2)
new_rows
```

[58]:
```
                   model  displ  cyl    trans drive     fuel  veh_class  \
1550  MERCEDES-BENZ C300    3.0    6  Auto-L7   2WD  ethanol  small car
1550  MERCEDES-BENZ C300    3.0    6  Auto-L7   2WD      gas  small car

     air_pollution_score city_mpg hwy_mpg cmb_mpg greenhouse_gas_score  \
1550                   6       13      19      15                     7
1550                   4       18      25      21                     6

     smartway
1550       no
1550       no
```

[59]:
```
# drop the original hybrid rows
df_08.drop(hb_08.index, inplace=True)

# add in our newly separated rows
df_08 = df_08.append(new_rows, ignore_index=True)
```

[60]:
```
# check that all the original hybrid rows with "/"s are gone
df_08[df_08['fuel'].str.contains('/')]
```

[60]:
```
Empty DataFrame
Columns: [model, displ, cyl, trans, drive, fuel, veh_class, air_pollution_score,
city_mpg, hwy_mpg, cmb_mpg, greenhouse_gas_score, smartway]
Index: []
```

Repeat this process for the 2018 dataset¶

[62]:
```
# create two copies of the 2018 hybrids dataframe, hb_18
df1 = hb_18.copy()
df2 = hb_18.copy()
df2.head(1)
```

[62]:
```
        model  displ  cyl       trans drive                  fuel  veh_class  \
108  BMW 330e    2.0    4  SemiAuto-8   2WD  Gasoline/Electricity  small car

     air_pollution_score city_mpg hwy_mpg cmb_mpg  greenhouse_gas_score  \
108                    3    28/66   34/78   30/71                    10

    smartway
108      Yes
```

[63]:
```
# list of columns to split
split_columns = ['fuel', 'city_mpg', 'hwy_mpg', 'cmb_mpg']

# apply split function to each column of each dataframe copy
for c in split_columns:
    df1[c] = df1[c].apply(lambda x: x.split("/")[0])
    df2[c] = df2[c].apply(lambda x: x.split("/")[1])
```

[64]:
```
# append the two dataframes
new_rows = df1.append(df2)

# drop each hybrid row from the original 2018 dataframe
# do this by using pandas' drop function with hb_18's index
df_18.drop(hb_18.index, inplace=True)

# append new_rows to df_18
df_18 = new_rows.append(df_18)
```

[65]:
```
# check that they're gone
df_18[df_18['fuel'].str.contains('/')]
```

[65]:
```
Empty DataFrame
Columns: [model, displ, cyl, trans, drive, fuel, veh_class, air_pollution_score,
city_mpg, hwy_mpg, cmb_mpg, greenhouse_gas_score, smartway]
Index: []
```

[66]:
```
# convert string to float for 2008 air pollution column
df_08.air_pollution_score = df_08.air_pollution_score.astype(float)
df_08.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 987 entries, 0 to 986
```

```
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   model                 987 non-null    object
 1   displ                 987 non-null    float64
 2   cyl                   987 non-null    int32
 3   trans                 987 non-null    object
 4   drive                 987 non-null    object
 5   fuel                  987 non-null    object
 6   veh_class             987 non-null    object
 7   air_pollution_score   987 non-null    float64
 8   city_mpg              987 non-null    object
 9   hwy_mpg               987 non-null    object
 10  cmb_mpg               987 non-null    object
 11  greenhouse_gas_score  987 non-null    object
 12  smartway              987 non-null    object
dtypes: float64(2), int32(1), object(10)
memory usage: 96.5+ KB
```

[67]:
```
# convert int to float for 2018 air pollution column
df_18.air_pollution_score = df_18.air_pollution_score.astype(float)
df_18.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 832 entries, 108 to 1607
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   model                 832 non-null    object
 1   displ                 832 non-null    float64
 2   cyl                   832 non-null    int32
 3   trans                 832 non-null    object
 4   drive                 832 non-null    object
 5   fuel                  832 non-null    object
 6   veh_class             832 non-null    object
 7   air_pollution_score   832 non-null    float64
 8   city_mpg              832 non-null    object
 9   hwy_mpg               832 non-null    object
 10  cmb_mpg               832 non-null    object
 11  greenhouse_gas_score  832 non-null    int64
 12  smartway              832 non-null    object
dtypes: float64(2), int32(1), int64(1), object(9)
memory usage: 87.8+ KB
```

[68]:
```
# convert mpg columns to floats
mpg_columns = ['city_mpg', 'hwy_mpg', 'cmb_mpg']
for c in mpg_columns:
```

```
    df_18[c] = df_18[c].astype(float)
    df_08[c] = df_08[c].astype(float)
df_08.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 987 entries, 0 to 986
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   model                 987 non-null    object
 1   displ                 987 non-null    float64
 2   cyl                   987 non-null    int32
 3   trans                 987 non-null    object
 4   drive                 987 non-null    object
 5   fuel                  987 non-null    object
 6   veh_class             987 non-null    object
 7   air_pollution_score   987 non-null    float64
 8   city_mpg              987 non-null    float64
 9   hwy_mpg               987 non-null    float64
 10  cmb_mpg               987 non-null    float64
 11  greenhouse_gas_score  987 non-null    object
 12  smartway              987 non-null    object
dtypes: float64(5), int32(1), object(7)
memory usage: 96.5+ KB
```

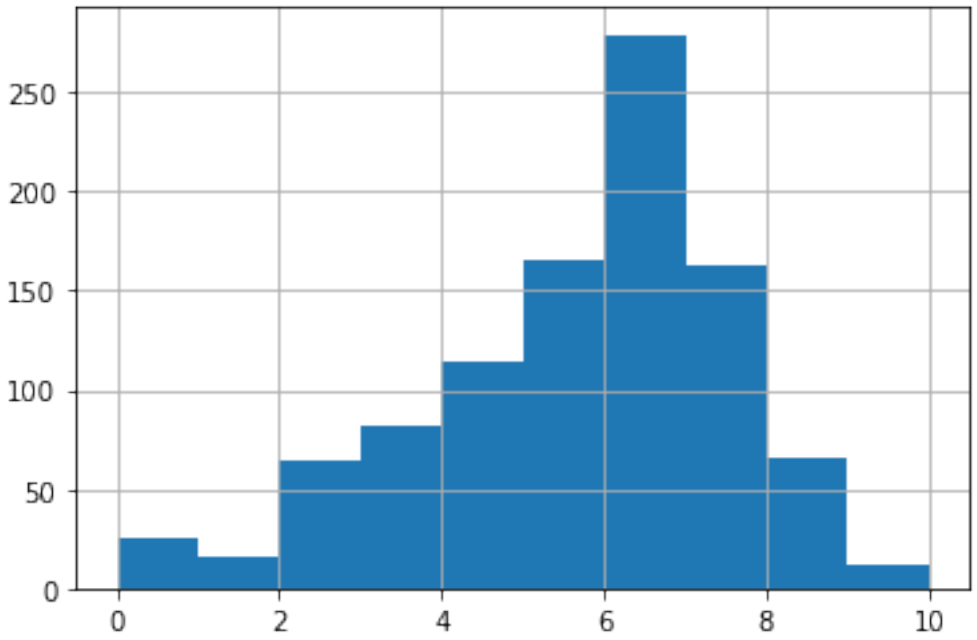
[71]:
```
# convert from float to int
df_08['greenhouse_gas_score'] = df_08['greenhouse_gas_score'].astype(int)
df_18['greenhouse_gas_score'] = df_18['greenhouse_gas_score'].astype(int)
```

[73]:
```
#check data types for each dataset match
df_08.dtypes == df_18.dtypes
```

[73]:
```
model                 True
displ                 True
cyl                   True
trans                 True
drive                 True
fuel                  True
veh_class             True
air_pollution_score   True
city_mpg              True
hwy_mpg               True
cmb_mpg               True
greenhouse_gas_score  True
smartway              True
dtype: bool
```

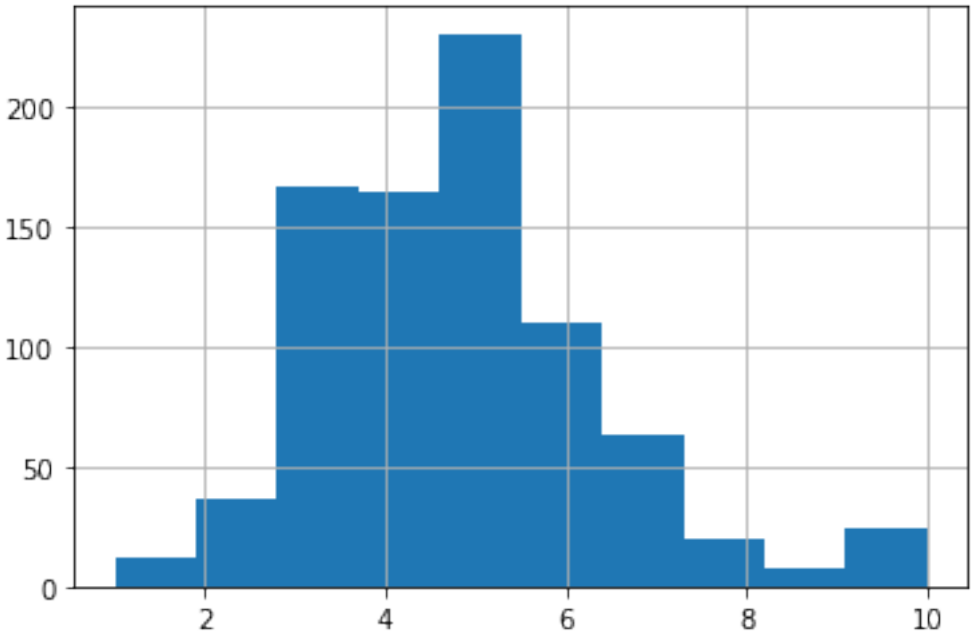
[74]:
```
#distribution of greenhouse gas emissions in 2008 dataset
df_08.greenhouse_gas_score.hist()
```

[74]: <matplotlib.axes._subplots.AxesSubplot at 0x20970b14308>



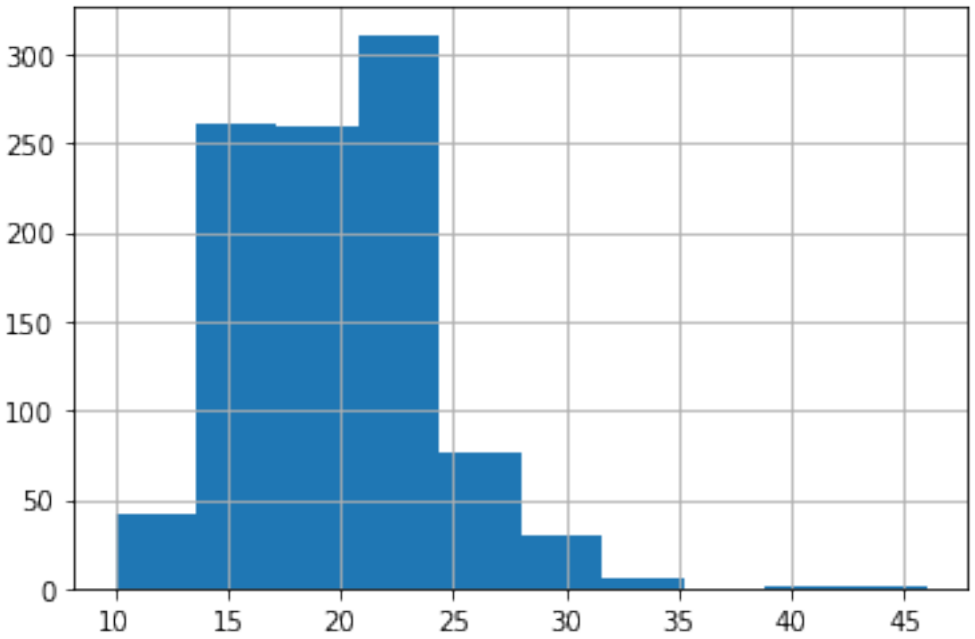
[75]:
```
#distribution of greenhouse gas emissions in 2018 dataset
df_18.greenhouse_gas_score.hist()
```

[75]: <matplotlib.axes._subplots.AxesSubplot at 0x20971279c48>



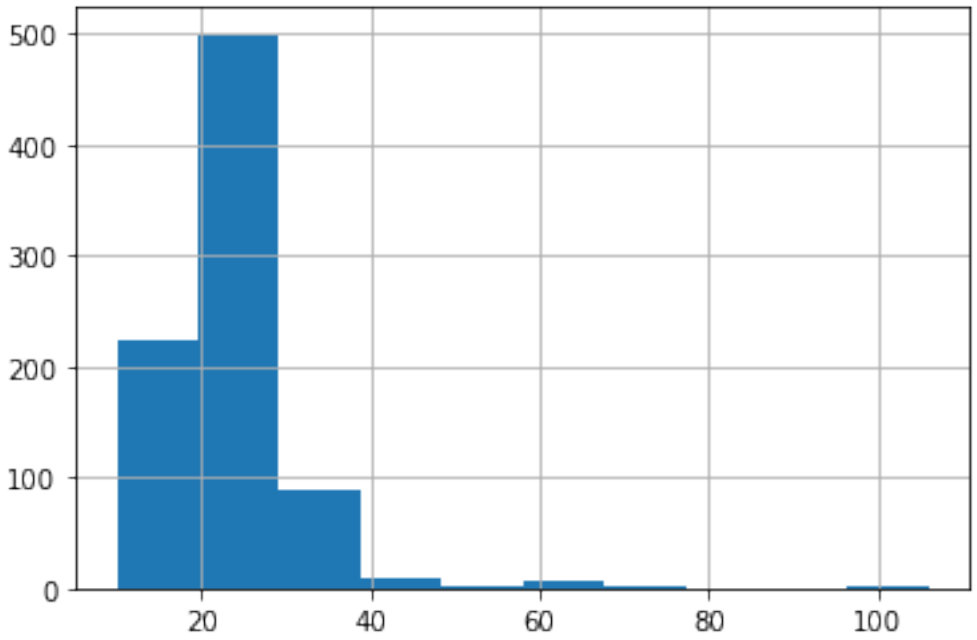
[82]:
```
#distribution of city mileage in 2008 dataset
df_08.cmb_mpg.hist()
```

[82]: <matplotlib.axes._subplots.AxesSubplot at 0x263e1950588>



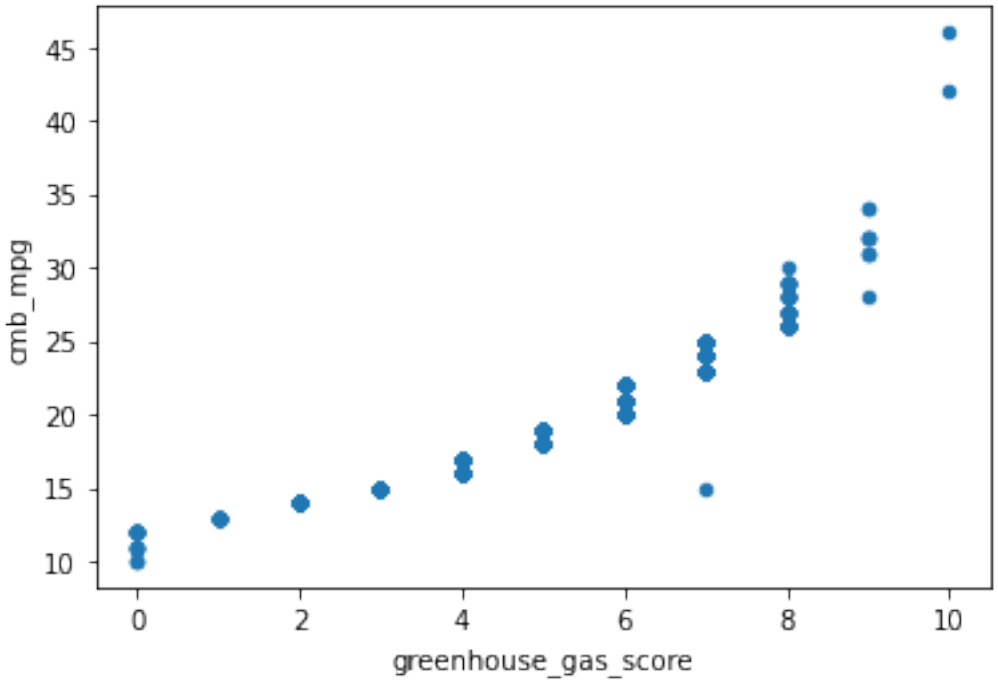
[76]:
```
#distribution of city mileage in 2018 dataset
df_18.cmb_mpg.hist()
```

[76]: <matplotlib.axes._subplots.AxesSubplot at 0x2097131d888>



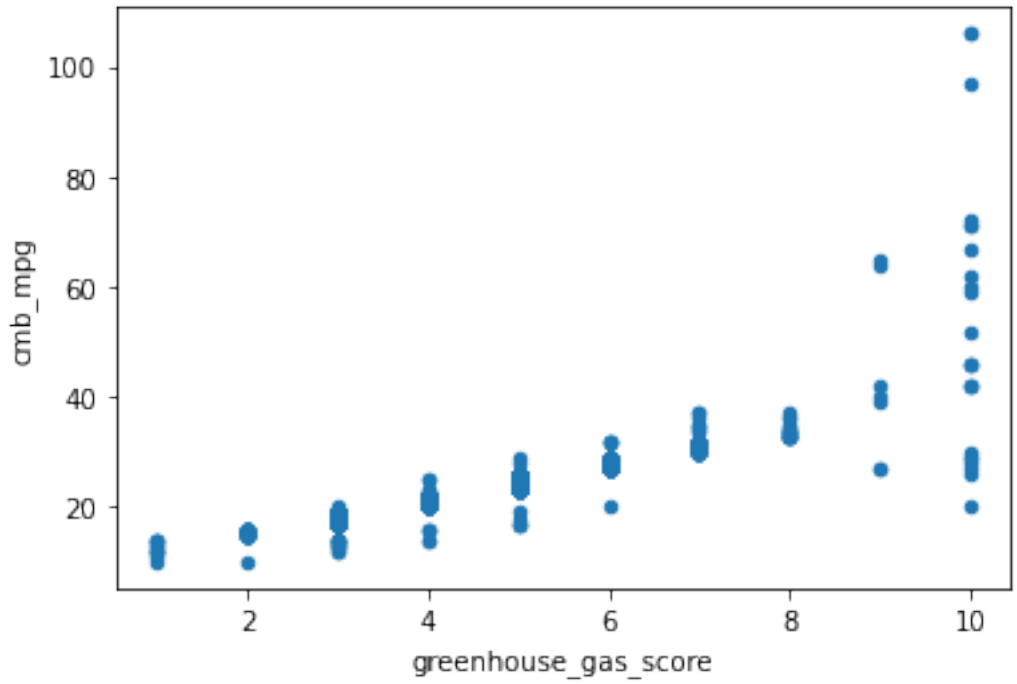
[77]:
```
#relationship between emissions and city mileage in 2008 dataset
df_08.plot(x='greenhouse_gas_score', y='cmb_mpg', kind='scatter')
```

[77]: <matplotlib.axes._subplots.AxesSubplot at 0x2097134e188>



[78]:
```
#relationship between emissions and city mileage in 2018 dataset
df_18.plot(x='greenhouse_gas_score', y='cmb_mpg', kind='scatter')
```

[78]: <matplotlib.axes._subplots.AxesSubplot at 0x20971416508>



Question: Are more models in 2018 using alternative sources of fuel? By how much?

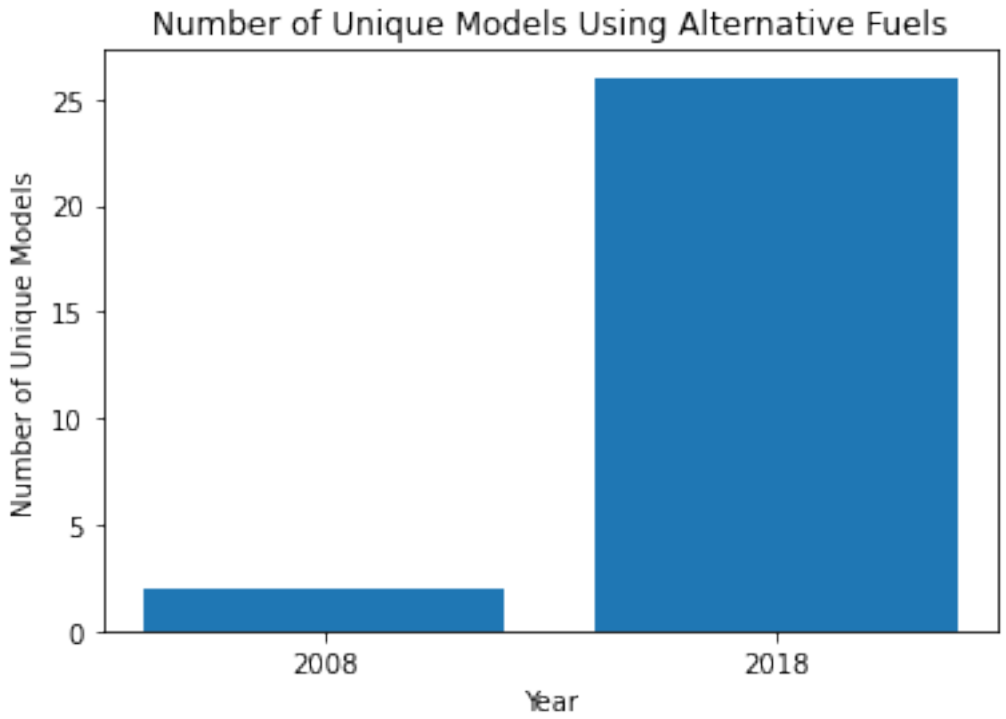
[79]:
```
# how many unique models used alternative sources of fuel in 2008
alt_08 = df_08.query('fuel in ["CNG", "ethanol"]').model.nunique()
alt_08
```

[79]: 2

[80]:
```
# how many unique models used alternative sources of fuel in 2018
alt_18 = df_18.query('fuel in ["Ethanol", "Electricity"]').model.nunique()
alt_18
```

[80]: 26

[83]:
```
#bar chart with labelled title and axis
plt.bar(["2008", "2018"], [alt_08, alt_18])
plt.title("Number of Unique Models Using Alternative Fuels")
plt.xlabel("Year")
plt.ylabel("Number of Unique Models");
```



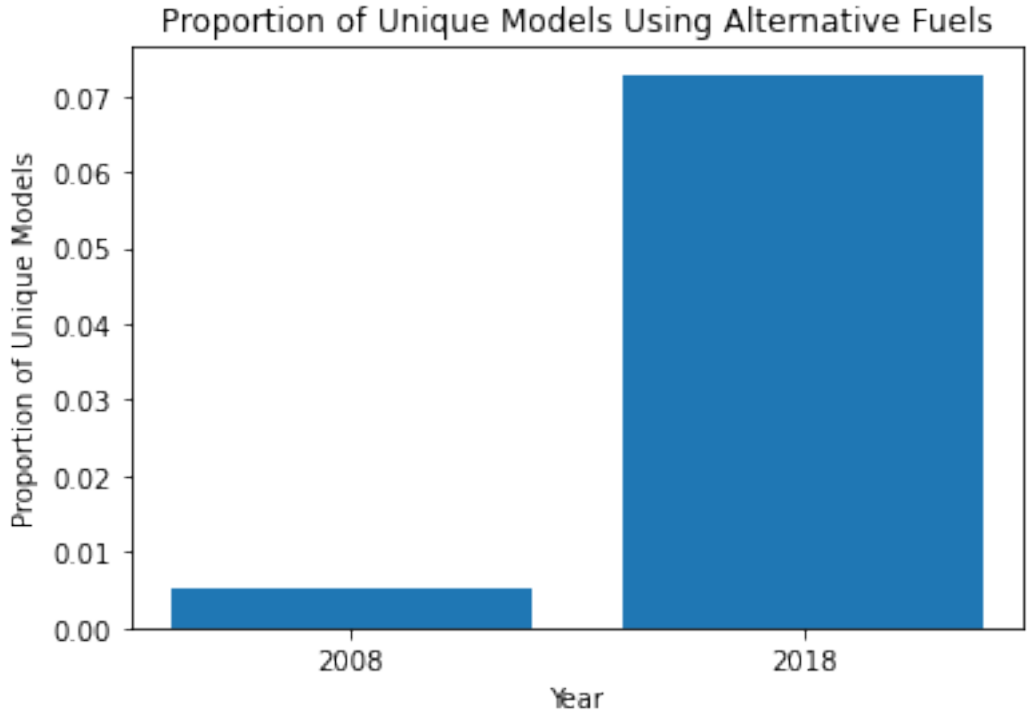
[84]:
```
# total unique models each year
total_08 = df_08.model.nunique()
total_18 = df_18.model.nunique()
total_08, total_18
```

[84]: (377, 357)

[85]:
```
#calculate proportions
prop_08 = alt_08/total_08
prop_18 = alt_18/total_18
prop_08, prop_18
```

[85]: (0.005305039787798408, 0.07282913165266107)

[86]:
```
#bar chart with labelled title and axis
plt.bar(["2008", "2018"], [prop_08, prop_18])
plt.title("Proportion of Unique Models Using Alternative Fuels")
plt.xlabel("Year")
plt.ylabel("Proportion of Unique Models");
```



Question: How much have vehicle classes improved in fuel economy?

[87]:
```
#calculate mean fuel economy by vehicle class in 2008 dataset
veh_08 = df_08.groupby('veh_class').cmb_mpg.mean()
```

```
veh_08
```

[87]:
```
veh_class
SUV               18.471429
large car         18.509091
midsize car       21.601449
minivan           19.117647
pickup            16.277108
small car         21.105105
station wagon     22.366667
van               14.952381
Name: cmb_mpg, dtype: float64
```

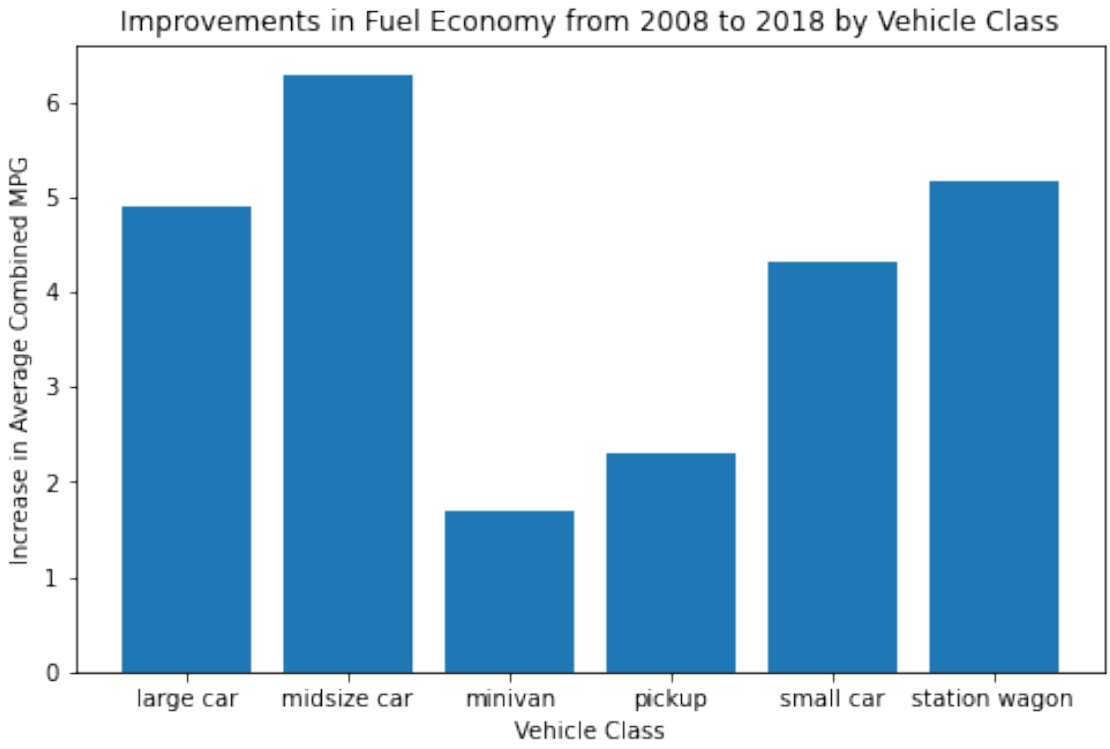
[88]:
```
#calculate mean fuel economy by vehicle class in 2008 dataset
veh_18 = df_18.groupby('veh_class').cmb_mpg.mean()
veh_18
```

[88]:
```
veh_class
large car          23.409091
midsize car        27.884058
minivan            20.800000
pickup             18.589744
small SUV          24.074074
small car          25.421053
special purpose    18.500000
standard SUV       18.197674
station wagon      27.529412
Name: cmb_mpg, dtype: float64
```

[89]:
```
#calculate difference between 2018 and 2008
inc = veh_18 - veh_08
inc
```

[89]:
```
veh_class
SUV                     NaN
large car          4.900000
midsize car        6.282609
minivan            1.682353
pickup             2.312635
small SUV               NaN
small car          4.315948
special purpose         NaN
standard SUV            NaN
station wagon      5.162745
van                     NaN
Name: cmb_mpg, dtype: float64
```

[91]:
```
# only plot vehicle classes that exist in both years
inc.dropna(inplace=True)
plt.subplots(figsize=(8, 5))
plt.bar(inc.index, inc)
plt.title('Improvements in Fuel Economy from 2008 to 2018 by Vehicle Class')
plt.xlabel('Vehicle Class')
plt.ylabel('Increase in Average Combined MPG');
```



Question: What are the characteristics of SmartWay vehicles? Have they changed over time?

[94]:
```
#unique values in smartway column for 2008 dataset
df_08.smartway.unique()
```

[94]:
```
array(['no', 'yes'], dtype=object)
```

[92]:
```
#isolate smartway vehicles in 2008 dataset
smart_08 = df_08.query('smartway == "yes"')
```

[93]:
```
#numerical summary of smartway vehicles in 2008 dataset
smart_08.describe()
```

[93]:
```
              displ         cyl  air_pollution_score    city_mpg     hwy_mpg  \
count  380.000000  380.000000           380.000000  380.000000  380.000000
```

```
mean      2.602895    4.826316             7.365789   20.984211   28.413158
std       0.623436    1.002025             1.148195    3.442672    3.075194
min       1.300000    4.000000             6.000000   17.000000   22.000000
25%       2.275000    4.000000             7.000000   19.000000   26.000000
50%       2.400000    4.000000             7.000000   20.000000   28.000000
75%       3.000000    6.000000             7.000000   22.000000   30.000000
max       5.000000    8.000000             9.500000   48.000000   45.000000

          cmb_mpg  greenhouse_gas_score
count  380.000000            380.000000
mean    23.736842              6.868421
std      3.060379              0.827338
min     20.000000              6.000000
25%     22.000000              6.000000
50%     23.000000              7.000000
75%     25.000000              7.000000
max     46.000000             10.000000
```

[95]:
```
#unique values in smartway column for 2018 dataset
df_18.smartway.unique()
```

[95]:
```
array(['Yes', 'Elite', 'No'], dtype=object)
```

[96]:
```
#isolate smartway vehicles in 2018 dataset
smart_18 = df_18.query('smartway in ["Yes", "Elite"]')
```

[97]:
```
#numerical summary of smartway vehicles in 2018 dataset
smart_18.describe()
```

[97]:
```
            displ         cyl  air_pollution_score    city_mpg     hwy_mpg  \
count  108.000000  108.000000           108.000000  108.000000  108.000000
mean     1.787963    3.935185             5.212963   34.907407   41.472222
std      0.408031    0.416329             1.798498   16.431982   13.095236
min      1.200000    3.000000             3.000000   25.000000   27.000000
25%      1.500000    4.000000             3.000000   28.000000   36.000000
50%      1.700000    4.000000             5.500000   28.500000   37.000000
75%      2.000000    4.000000             7.000000   31.250000   40.250000
max      3.500000    6.000000             7.000000  113.000000   99.000000

          cmb_mpg  greenhouse_gas_score
count  108.000000            108.000000
mean    37.361111              7.925926
std     14.848429              1.197378
min     26.000000              7.000000
25%     31.000000              7.000000
50%     32.000000              7.000000
75%     35.000000              9.000000
```

```
max    106.000000             10.000000
```

Question: What features are associated with better fuel economy?

[98]:
```
#isolate vehicles with mpg above the dataset mean for 2008 dataset
top_08 = df_08.query('cmb_mpg > cmb_mpg.mean()')
top_08.describe()
```

[98]:
```
            displ         cyl  air_pollution_score    city_mpg     hwy_mpg  \
count  519.000000  519.000000           519.000000  519.000000  519.000000
mean     2.667823    4.890173             6.998073   20.317919   27.603083
std      0.665551    1.034856             1.159565    3.198257    3.051120
min      1.300000    4.000000             4.000000   17.000000   20.000000
25%      2.300000    4.000000             6.000000   18.000000   25.000000
50%      2.500000    4.000000             7.000000   20.000000   27.000000
75%      3.000000    6.000000             7.000000   21.000000   29.000000
max      6.000000    8.000000             9.500000   48.000000   45.000000

          cmb_mpg  greenhouse_gas_score
count  519.000000            519.000000
mean    22.992293              6.639692
std      2.926371              0.804935
min     20.000000              6.000000
25%     21.000000              6.000000
50%     22.000000              6.000000
75%     24.000000              7.000000
max     46.000000             10.000000
```

[99]:
```
#isolate vehicles with mpg above the dataset mean for 2018 dataset
top_18 = df_18.query('cmb_mpg > cmb_mpg.mean()')
top_18.describe()
```

[99]:
```
            displ         cyl  air_pollution_score    city_mpg     hwy_mpg  \
count  328.000000  328.000000           328.000000  328.000000  328.000000
mean     1.964329    4.021341             4.856707   27.472561   35.304878
std      0.398593    0.465477             1.860802   11.033692    9.024857
min      1.200000    3.000000             1.000000   21.000000   27.000000
25%      1.600000    4.000000             3.000000   23.000000   31.000000
50%      2.000000    4.000000             5.000000   25.000000   33.000000
75%      2.000000    4.000000             7.000000   28.000000   36.000000
max      3.500000    6.000000             7.000000  113.000000   99.000000

          cmb_mpg  greenhouse_gas_score
count  328.000000            328.000000
mean    30.411585              6.329268
std     10.081539              1.410358
min     25.000000              4.000000
```

```
25%     26.000000              5.000000
50%     28.000000              6.000000
75%     31.000000              7.000000
max    106.000000             10.000000
```

[100]:
```
# rename 2008 columns
df_08 = df_08.rename(lambda x: x[:10] + "_2008", axis='columns')
```

[101]:
```
# view to check names
df_08.head()
```

[101]:
```
  model_2008  displ_2008  cyl_2008 trans_2008 drive_2008 fuel_2008  \
0  ACURA MDX         3.7         6    Auto-S5        4WD  Gasoline
1  ACURA RDX         2.3         4    Auto-S5        4WD  Gasoline
2   ACURA RL         3.5         6    Auto-S5        4WD  Gasoline
3   ACURA TL         3.2         6    Auto-S5        2WD  Gasoline
4   ACURA TL         3.5         6    Auto-S5        2WD  Gasoline

  veh_class_2008  air_pollut_2008  city_mpg_2008  hwy_mpg_2008  cmb_mpg_2008  \
0            SUV              7.0           15.0          20.0          17.0
1            SUV              7.0           17.0          22.0          19.0
2    midsize car              7.0           16.0          24.0          19.0
3    midsize car              7.0           18.0          26.0          21.0
4    midsize car              7.0           17.0          26.0          20.0

   greenhouse_2008 smartway_2008
0                4           no
1                5           no
2                5           no
3                6          yes
4                6          yes
```

[102]:
```
# merge datasets
df_combined = df_08.merge(df_18, left_on='model_2008', right_on='model',␣
↪how='inner')
```

[103]:
```
#view first five rows of combined dataset
df_combined.head()
```

[103]:
```
  model_2008  displ_2008  cyl_2008 trans_2008 drive_2008 fuel_2008  \
0  ACURA RDX         2.3         4    Auto-S5        4WD  Gasoline
1  ACURA RDX         2.3         4    Auto-S5        4WD  Gasoline
2    AUDI A3         2.0         4      Man-6        2WD  Gasoline
3    AUDI A3         2.0         4      Man-6        2WD  Gasoline
4    AUDI A3         2.0         4    Auto-S6        2WD  Gasoline

  veh_class_2008  air_pollut_2008  city_mpg_2008  hwy_mpg_2008  …  \
```

```
0            SUV              7.0           17.0          22.0  …
1            SUV              7.0           17.0          22.0  …
2  station wagon              7.0           21.0          29.0  …
3  station wagon              7.0           21.0          29.0  …
4  station wagon              7.0           22.0          29.0  …

        trans  drive      fuel  veh_class  air_pollution_score  city_mpg  \
0  SemiAuto-6    2WD  Gasoline  small SUV                  3.0      20.0
1  SemiAuto-6    4WD  Gasoline  small SUV                  3.0      19.0
2       AMS-6    4WD  Gasoline  small car                  7.0      24.0
3       AMS-7    2WD  Gasoline  small car                  7.0      26.0
4       AMS-6    4WD  Gasoline  small car                  7.0      24.0

  hwy_mpg cmb_mpg greenhouse_gas_score smartway
0    28.0    23.0                    5       No
1    27.0    22.0                    4       No
2    31.0    27.0                    6       No
3    35.0    29.0                    6       No
4    31.0    27.0                    6       No

[5 rows x 26 columns]
```

[104]:
```
#view number of rows and columns in combined dataset
df_combined.shape
```

[104]:
```
(922, 26)
```

Question: For all of the models that were produced in 2008 that are still being produced now, how much has the mpg improved and which vehicle improved the most?

[107]:
```
#compare mpg by model between 2008 and 2018
model_mpg = df_combined.groupby('model')[['cmb_mpg_2008', 'cmb_mpg']].mean()
model_mpg.head()
```

[107]:
```
           cmb_mpg_2008    cmb_mpg
model
ACURA RDX     19.000000  22.500000
AUDI A3       23.333333  28.000000
AUDI A4       21.000000  27.000000
AUDI A6       19.666667  25.666667
AUDI A8 L     16.500000  22.000000
```

[108]:
```
#calculate difference in mpg
model_mpg['mpg_change'] = model_mpg.cmb_mpg - model_mpg.cmb_mpg_2008
model_mpg.head()
```

[108]:
model	cmb_mpg_2008	cmb_mpg	mpg_change
ACURA RDX	19.000000	22.500000	3.500000
AUDI A3	23.333333	28.000000	4.666667
AUDI A4	21.000000	27.000000	6.000000
AUDI A6	19.666667	25.666667	6.000000
AUDI A8 L	16.500000	22.000000	5.500000

[109]:
```
#find model with largest increase in fuel effiency between 2008 and 2018
model_mpg.query('mpg_change == mpg_change.max()')
```

[109]:
model	cmb_mpg_2008	cmb_mpg	mpg_change
VOLVO XC 90	15.666667	32.2	16.533333