Best Fuel Economy by Car Model Analysis

In [1]:
```
import pandas as pd
import numpy as np

%matplotlib inline
import matplotlib.pyplot as plt
```

In [2]:
```
# importing CSV files
df_08 = pd.read_csv('./all_alpha_08.csv')
df_18 = pd.read_csv('./all_alpha_18.csv')
```

In [3]:
```
#First Five data from 2008 dataset
df_08.head()
```

Out[3]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Sales Area	Stnd	Underhood ID	Veh Class	Air Pollution Score
0	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	CA	U2	8HNXT03.7PKR	SUV	7
1	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	FA	B5	8HNXT03.7PKR	SUV	6
2	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	CA	U2	8HNXT02.3DKR	SUV	7
3	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	FA	B5	8HNXT02.3DKR	SUV	6
4	ACURA RL	3.5	(6 cyl)	Auto-S5	4WD	Gasoline	CA	U2	8HNXV03.5HKR	midsize car	7

In [4]:
```
#First Five data from 2018 dataset
df_18.head()
```

Out[4]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Cert Region	Stnd	Stnd Description	Underhooc
0	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	FA	T3B125	Federal Tier 3 Bin 125	JHNXT03.5C
1	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	CA	U2	California LEV-II ULEV	JHNXT03.5C
2	ACURA RDX	3.5	6.0	SemiAuto-6	4WD	Gasoline	FA	T3B125	Federal Tier 3 Bin 125	JHNXT03.5C
3	ACURA RDX	3.5	6.0	SemiAuto-6	4WD	Gasoline	CA	U2	California LEV-II ULEV	JHNXT03.5C
4	ACURA TLX	2.4	4.0	AMS-8	2WD	Gasoline	CA	L3ULEV125	California LEV-III ULEV125	JHNXV02.4W

Cleaning Data

In [5]:
```
# duplicates for 2008 dataset
df_08.duplicated().sum()
```

Out[5]: 25

In [6]:
```
# duplicates for 2008 dataset
df_18.duplicated().sum()
```

Out[6]: 0

In [7]:
```
#Find records with the missing values in 2008 dataset
df_08.isnull().sum()
```

Out[7]:
```
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
```

In [8]:
```
#Find records with the missing values in 2018 dataset
df_18.isnull().sum()
```

Out[8]:
```
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
```

In [9]:
```
#missing value 2008 has been fixed
df_08.query('Drive == "NaN"')
```

Out[9]:

Model	Displ	Cyl	Trans	Drive	Fuel	Sales Area	Stnd	Underhood ID	Veh Class	Air Pollution Score	FE Calc Appr	City MPG

In [10]:
```
#Total number of unique values for 2008 dataset
df_08.nunique().sum()
```

Out[10]: 1757

In [11]:
```
#Total number of unique values for 2018 dataset
df_18.nunique().sum()
```

Out[11]: 1217

In [12]:
```
#droped columns which is not needed for dataset 2008
df_08.drop(['Stnd','Underhood ID', 'FE Calc Appr', 'Unadj Cmb MPG'], axis=1, i
nplace=True)
df_08.head()
```

Out[12]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Sales Area	Veh Class	Air Pollution Score	City MPG	Hwy MPG	Cmb MPG	Gre Ga
0	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	15	20	17	
1	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	FA	SUV	6	15	20	17	
2	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	17	22	19	
3	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	FA	SUV	6	17	22	19	
4	ACURA RL	3.5	(6 cyl)	Auto-S5	4WD	Gasoline	CA	midsize car	7	16	24	19	

In [13]:
```
#droped columns which is not needed for dataset 2018
df_18.drop(['Stnd', 'Stnd Description', 'Underhood ID', 'Comb CO2'], axis=1, i
nplace=True)
df_18.head()
```

Out[13]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Cert Region	Veh Class	Air Pollution Score	City MPG	Hwy MPG	Cmb MPG
0	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	FA	small SUV	3	20	28	23
1	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	CA	small SUV	3	20	28	23
2	ACURA RDX	3.5	6.0	SemiAuto-6	4WD	Gasoline	FA	small SUV	3	19	27	22
3	ACURA RDX	3.5	6.0	SemiAuto-6	4WD	Gasoline	CA	small SUV	3	19	27	22
4	ACURA TLX	2.4	4.0	AMS-8	2WD	Gasoline	CA	small car	3	23	33	27

Rename ' Sales Area' to 'Cert Region' in 2008 dataframe

In [14]:
```
df_08.rename(columns={"Sales Area": "Cert Region"}, inplace=True)
df_08.head()
```

Out[14]:

	Model	Displ	Cyl	Trans	Drive	Fuel	Cert Region	Veh Class	Air Pollution Score	City MPG	Hwy MPG	Cmb MPG	Gr
0	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	15	20	17	
1	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	FA	SUV	6	15	20	17	
2	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	17	22	19	
3	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	FA	SUV	6	17	22	19	
4	ACURA RL	3.5	(6 cyl)	Auto-S5	4WD	Gasoline	CA	midsize car	7	16	24	19	

In [15]:
```
# rename and change column name in 2008 dataset
df_08.rename(columns=lambda x: x.strip().lower().replace(" ", "_"), inplace=True)
df_08.head()
```

Out[15]:

	model	displ	cyl	trans	drive	fuel	cert_region	veh_class	air_pollution_score	city_m
0	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	
1	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	FA	SUV	6	
2	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	
3	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	FA	SUV	6	
4	ACURA RL	3.5	(6 cyl)	Auto-S5	4WD	Gasoline	CA	midsize car	7	

In [16]:
```
# rename and change column name in 2018
df_18.rename(columns=lambda x: x.strip().lower().replace(" ", "_"), inplace=True)
df_18.head()
```

Out[16]:

	model	displ	cyl	trans	drive	fuel	cert_region	veh_class	air_pollution_score	cit
0	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	FA	small SUV	3	
1	ACURA RDX	3.5	6.0	SemiAuto-6	2WD	Gasoline	CA	small SUV	3	
2	ACURA RDX	3.5	6.0	SemiAuto-6	4WD	Gasoline	FA	small SUV	3	
3	ACURA RDX	3.5	6.0	SemiAuto-6	4WD	Gasoline	CA	small SUV	3	
4	ACURA TLX	2.4	4.0	AMS-8	2WD	Gasoline	CA	small car	3	

Save Dataframe to csv

In [17]:
```
df_08.to_csv("df_08_v1.csv", index=False)
df_18.to_csv("df_18_v1.csv", index=False)
```

Cars driven in the U.S are certified only through California, Knowing this, I decidec to remove all records that don't have 'CA' certification region

In [18]:
```
#kept only "CA" certification region for dataset 2008
df_08 = df_08.query('cert_region == "CA"')
df_08.head()
```

Out[18]:

	model	displ	cyl	trans	drive	fuel	cert_region	veh_class	air_pollution_score	city_m
0	ACURA MDX	3.7	(6 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	
2	ACURA RDX	2.3	(4 cyl)	Auto-S5	4WD	Gasoline	CA	SUV	7	
4	ACURA RL	3.5	(6 cyl)	Auto-S5	4WD	Gasoline	CA	midsize car	7	
6	ACURA TL	3.2	(6 cyl)	Auto-S5	2WD	Gasoline	CA	midsize car	7	
7	ACURA TL	3.5	(6 cyl)	Auto-S5	2WD	Gasoline	CA	midsize car	7	

In [20]:
```
# only kept "CA" certification region 2018
df_18 = df_18.query('cert_region == "CA"')
```

In [21]:
```
#drop "cert_region" column because do not need any more
df_08.drop(columns=['cert_region'], axis=1, inplace=True)
```

In [22]:
```
#drop "cert_region" column because do not need any more
df_18.drop(columns=['cert_region'], axis=1, inplace=True)
```

Drop records with missing values

In [23]:
```
df_08.dropna(inplace=True)
df_18.dropna(inplace=True)
```

In [24]:
```
# drop all duplicates from datasets
df_08.drop_duplicates(inplace=True)
df_18.drop_duplicates(inplace=True)
```

In [25]:
```
#saved new CSV files
df_08.to_csv('df_08_v2.csv', index=False)
df_18.to_csv('df_18_v2.csv', index=False)
```

In [26]:
```
### changing str to int for 2008 dataset
df_08.cyl = df_08.cyl.str.extract('(\d+)').astype(int)
df_08.cyl.value_counts()
```

Out[26]:
```
6     409
4     283
8     199
5      48
12     30
10     14
2       2
16      1
Name: cyl, dtype: int64
```

In [27]:
```
### changing flot to int dor 2018 dataset
df_18.cyl = df_18.cyl.astype(int)
df_08.cyl.value_counts()
```

Out[27]:
```
6     409
4     283
8     199
5      48
12     30
10     14
2       2
16      1
Name: cyl, dtype: int64
```

Save latest version to CSV

In [28]:
```
#Latest version of CSV files
df_08.to_csv('df_08_v3.csv', index=False)
df_18.to_csv('df_18_v3.csv', index=False)
```

In [29]:
```
#Look for data which have uncount value
df_08.query('air_pollution_score == "6/4"')
```

Out[29]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg
1550	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol/gas	small car	6/4	13/18

Find all records with ' slashed' value and creat one record for each value

In [30]:
```
hybrid_08 = df_08[df_08['fuel'].str.contains('/')]
hybrid_08.head()
```

Out[30]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg
1550	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol/gas	small car	6/4	13/18

In [31]:
```
hybrid_18 = df_18[df_18['fuel'].str.contains('/')]
hybrid_18.head()
```

Out[31]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score
108	BMW 330e	2.0	4	SemiAuto-8	2WD	Gasoline/Electricity	small car	3
160	BMW 530e	2.0	4	SemiAuto-8	2WD	Gasoline/Electricity	small car	7
162	BMW 530e	2.0	4	SemiAuto-8	4WD	Gasoline/Electricity	small car	7
188	BMW 740e	2.0	4	SemiAuto-8	4WD	Gasoline/Electricity	large car	3
382	CHEVROLET Impala	3.6	6	SemiAuto-6	2WD	Ethanol/Gas	large car	5

In [32]:
```
#created copy of hybrid object for 2008 dataset
df1 = hybrid_08.copy()
df2 = hybrid_08.copy()

df2.head()
```

Out[32]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg
1550	MERCEDES-BENZ C300	3.0	6	Auto-L7	2WD	ethanol/gas	small car	6/4	13/18

In [33]:
```
# split the "/" in two rows using lambda method
columns_to_spilt = ['fuel', 'air_pollution_score', 'city_mpg', 'hwy_mpg', 'cmb
_mpg', 'greenhouse_gas_score']

for i in columns_to_spilt:
    df1[i] = df1[i].apply(lambda x: x.split('/')[0])
    df2[i] = df2[i].apply(lambda x: x.split('/')[1])
```

In [34]:
```
# drop index and append splited "/"
df_08.drop(hybrid_08.index, inplace=True)
df_08 = df_08.append(df1.append(df2), ignore_index=True)
```

In [35]:
```
# created copy of hybrid object for 2018 dataset
df3 = hybrid_18.copy()
df4 = hybrid_18.copy()

df4.head()
```

Out[35]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score
108	BMW 330e	2.0	4	SemiAuto-8	2WD	Gasoline/Electricity	small car	3
160	BMW 530e	2.0	4	SemiAuto-8	2WD	Gasoline/Electricity	small car	7
162	BMW 530e	2.0	4	SemiAuto-8	4WD	Gasoline/Electricity	small car	7
188	BMW 740e	2.0	4	SemiAuto-8	4WD	Gasoline/Electricity	large car	3
382	CHEVROLET Impala	3.6	6	SemiAuto-6	2WD	Ethanol/Gas	large car	5

In [36]:
```
## split the "/" in two rows using lambda method
columns_to_spilt = ['fuel', 'city_mpg', 'hwy_mpg', 'cmb_mpg']

for i in columns_to_spilt:
    df3[i] = df3[i].apply(lambda x: x.split('/')[0])
    df4[i] = df4[i].apply(lambda x: x.split('/')[1])
```

In [37]:
```
## drop index and append splited "/"
df_18.drop(hybrid_18.index, inplace=True)
df_18 = df_18.append(df3.append(df4), ignore_index=True)
```

In [38]:
```
# changed objects in to "int"
df_08.city_mpg = df_08.city_mpg.astype(int)
df_08.hwy_mpg = df_08.hwy_mpg.astype(int)
df_08.cmb_mpg = df_08.cmb_mpg.astype(int)

df_18.city_mpg = df_18.city_mpg.astype(int)
df_18.hwy_mpg = df_18.hwy_mpg.astype(int)
df_18.cmb_mpg = df_18.cmb_mpg.astype(int)
```

In [39]:
```
#changed objects in to "float" for 2008 dataset
df_08.air_pollution_score =df_08.air_pollution_score.astype(float)
df_08.greenhouse_gas_score = df_08.greenhouse_gas_score.astype(float)
df_08.dtypes
```

Out[39]:
```
model                  object
displ                 float64
cyl                     int32
trans                  object
drive                  object
fuel                   object
veh_class              object
air_pollution_score   float64
city_mpg                int32
hwy_mpg                 int32
cmb_mpg                 int32
greenhouse_gas_score  float64
smartway               object
dtype: object
```

In [40]:
```
#changed objects in to "float" for dataset 2018
df_18.air_pollution_score =df_18.air_pollution_score.astype(float)
df_18.greenhouse_gas_score = df_18.greenhouse_gas_score.astype(float)
df_18.dtypes
```

Out[40]:
```
model                  object
displ                 float64
cyl                     int32
trans                  object
drive                  object
fuel                   object
veh_class              object
air_pollution_score   float64
city_mpg                int32
hwy_mpg                 int32
cmb_mpg                 int32
greenhouse_gas_score  float64
smartway               object
dtype: object
```

In [41]:
```
df_08.to_csv('df_08_v4.csv', index=False)
df_18.to_csv('df_18_v4.csv', index=False)
```

Q1. How many more car models use alternative fuels in 2018 as opposed to 2008?

In [42]:
```
# altternative fuels for 2008
df_08.fuel.value_counts()
```

Out[42]:
```
Gasoline    984
ethanol       1
gas           1
CNG           1
Name: fuel, dtype: int64
```

In [43]:
```
# alternative fuel for 2018
df_18.fuel.value_counts()
```

Out[43]:
```
Gasoline       749
Ethanol         26
Gas             26
Diesel          19
Electricity     12
Name: fuel, dtype: int64
```

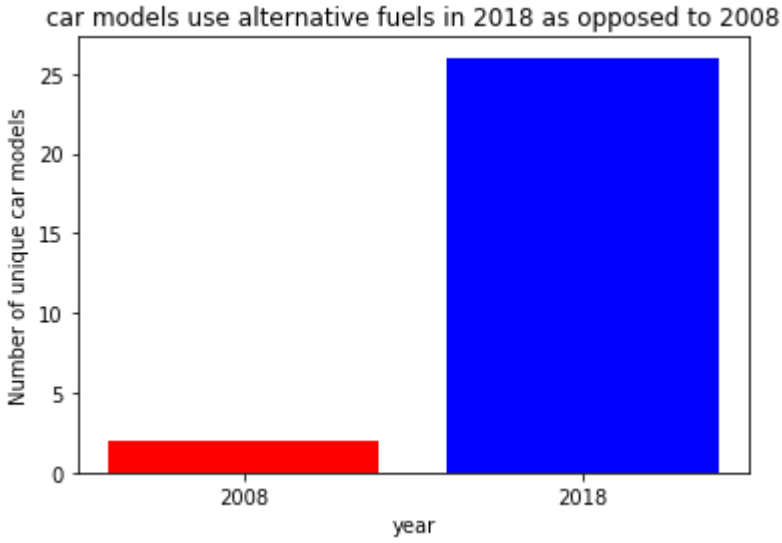
In [44]:
```
# unique numbers for car models which use alternative fuel in 2008
alt_08 = df_08.query('fuel in ["CNG", "ethanol"]').model.nunique()
alt_08
```

Out[44]: 2

In [45]:
```
# unique numbers for car models which use alternative fuel in 2018
alt_18 = df_18.query('fuel in ["Ethanol", "Electricity"]').model.nunique()
alt_18
```

Out[45]: 26

In [46]:
```
#bar chart for visualization
plt.bar(['2008', '2018'], [alt_08, alt_18], color=('red','blue'))
plt.title("car models use alternative fuels in 2018 as opposed to 2008")
plt.xlabel('year')
plt.ylabel('Number of unique car models')
plt.show()
```



It's seems like 24 car models use alternative fuels in 2018 opposed to 2008.</p>

Q2. How much has the average fuel economy improved since 2008?

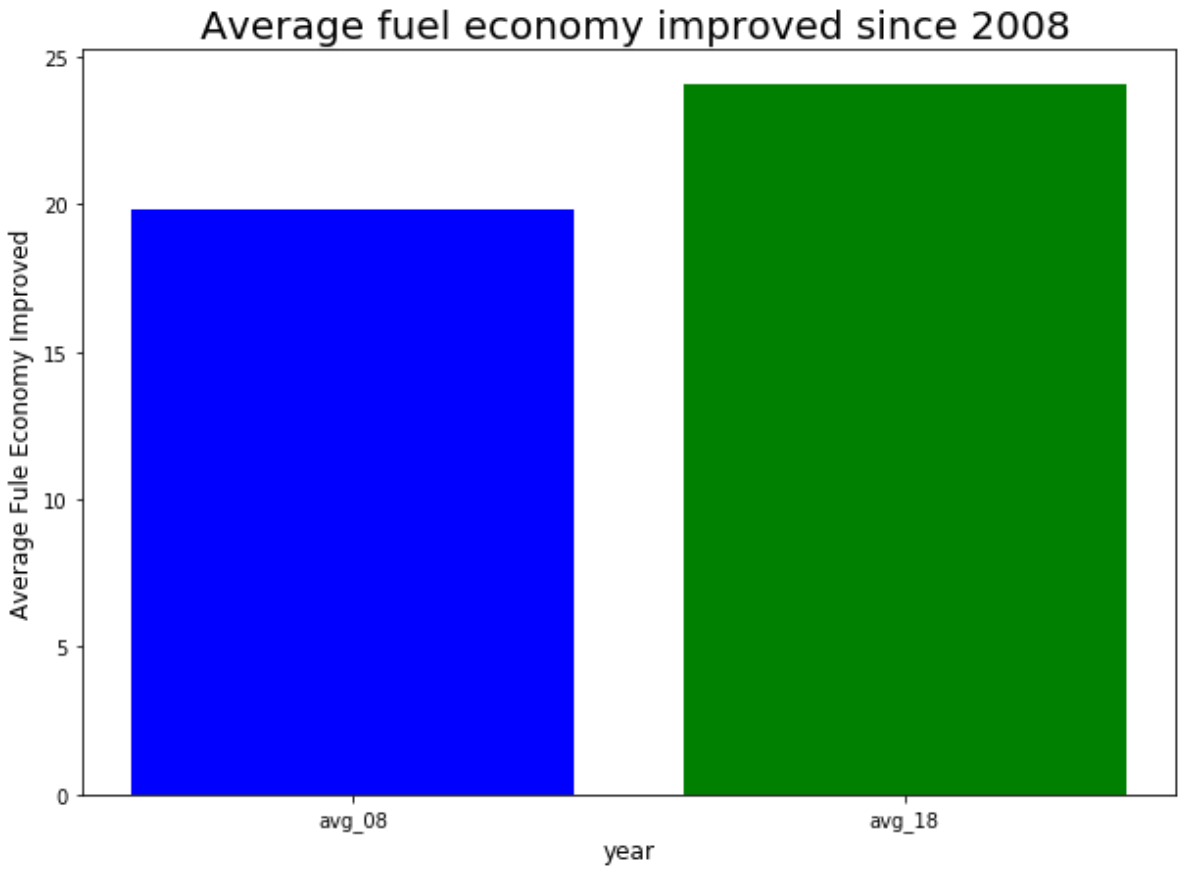
In [47]:
```
# average fuel economy in 2008
df_08.cmb_mpg.mean()
```

Out[47]: 19.78824721377913

In [48]:
```
# average fuel economy in 2018
df_18.cmb_mpg.mean()
```

Out[48]: 24.030048076923077

In [49]:
```
#Plot for improved fule economy since 2008
plt.subplots(figsize=(10,7))
plt.bar(["avg_08", "avg_18"], [df_08.cmb_mpg.mean(), df_18.cmb_mpg.mean()], co
lor=("blue", "green"))
plt.title(" Average fuel economy improved since 2008", fontsize=20)
plt.xlabel("year", fontsize=12)
plt.ylabel("Average Fule Economy Improved", fontsize=12)
plt.show()
```



By Looking this graph I can say that the average fule economy has improved by 4 mpg since 2008.

Q3. Based on vehicle type, how much has average fuel economy improved since 2008?

In [50]:
```
#The average fuel economy for each vehicle class for both years.

veh_08 = df_08.groupby("veh_class").cmb_mpg.mean()
veh_08

veh_18 = df_18.groupby("veh_class").cmb_mpg.mean()
veh_18
```

Out[50]:
```
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
```

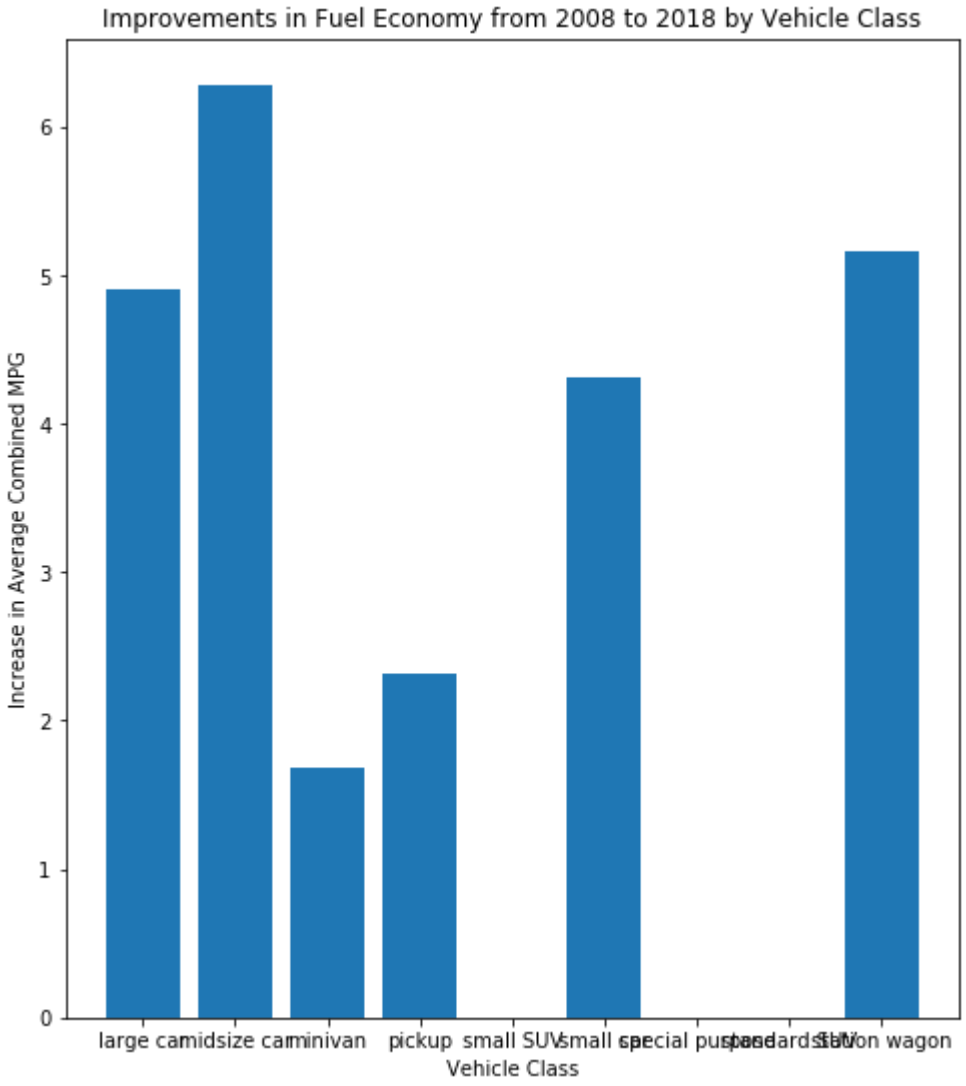
In [51]:
```
# how much they have increased
inc = veh_18 - veh_08
inc
```

Out[51]:
```
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
```

In [52]:
```
# vasual for vehicle type and how much has average fuel economy improved since
2008 for that fuel economy.

plt.subplots(figsize=(8, 9))
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")
```

Out[52]: Text(0, 0.5, 'Increase in Average Combined MPG')



Midsize car has improved highest fuel economy comperd to 2008, it's 6 mpg. large car and sation wagon has improved 5 mpg. Minivan, pickup and small car also has improved their fuel economy by 1 mpg, 2 mpg, and 4 mpg respectively. In all six vehical types has improved their fuel economy since 2008.

Q4. Which model has the highest average air_pollution_score and fuel economy in 2008 and 2018? Comper them

In [53]:
```
# Average air pollution score from 2008 dataset
df_08.groupby('model').air_pollution_score.max().max()
```

Out[53]: 9.5

In [54]:
```
# Best Fuel economy in 2008 dataset
df_08.groupby('model').cmb_mpg.max().max()
```

Out[54]: 46

In [55]:
```
# Best Fuel economy Car for 2008
fuel_08= df_08.query('air_pollution_score == 9.5 and cmb_mpg == 46')
fuel_08
```

Out[55]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	hwy_r
878	TOYOTA Prius	1.5	4	Auto-AV	2WD	Gasoline	midsize car	9.5	48	

In [56]:
```
# Average air pollution score from 2018 dataset
df_18.groupby('model').air_pollution_score.max().max()
```

Out[56]: 7.0

In [57]:
```
# Best Fuel economy in 2018 dataset
df_18.groupby("model").cmb_mpg.max().max()
```

Out[57]: 106

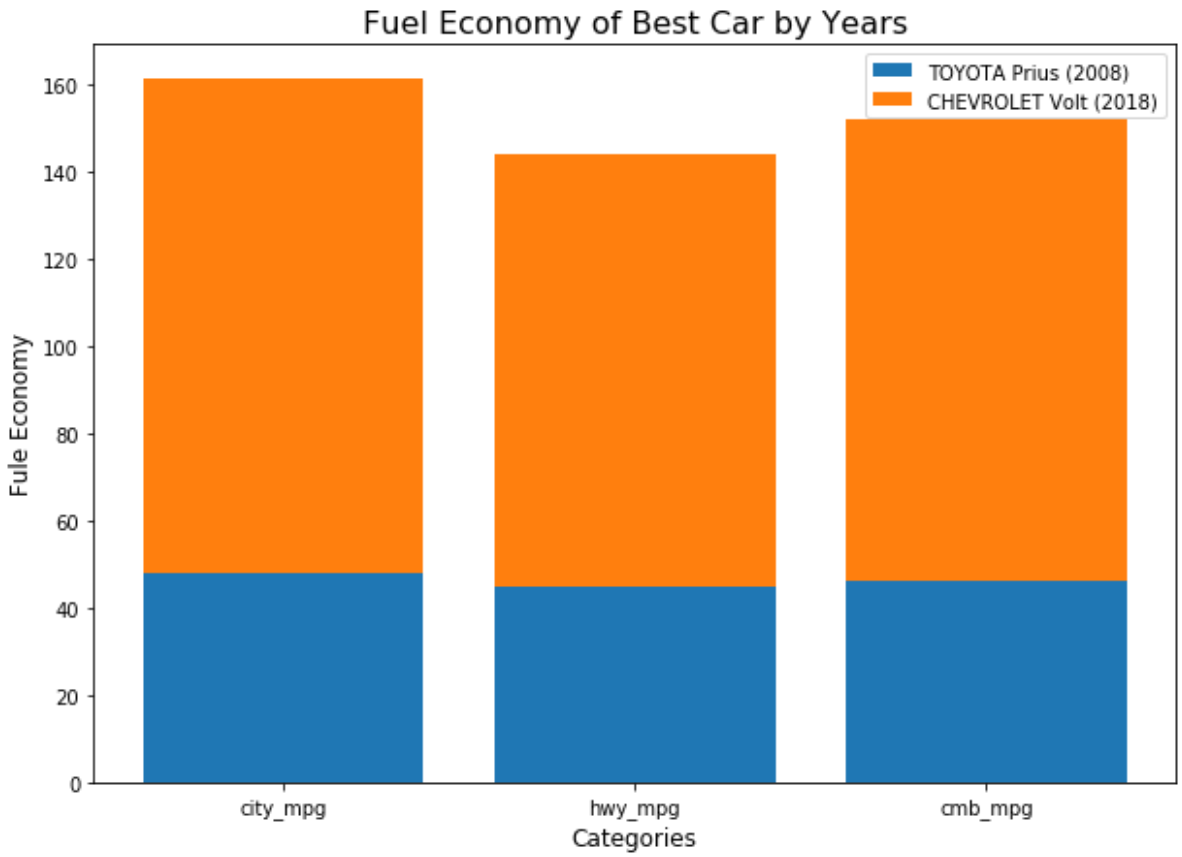
In [58]:
```
# Best Fuel economy Car for 2018
fuel_18 = df_18.query("air_pollution_score == 7.0 and cmb_mpg == 106")
fuel_18
```

Out[58]:

	model	displ	cyl	trans	drive	fuel	veh_class	air_pollution_score	city_mpg	h
807	CHEVROLET Volt	1.5	4	CVT	2WD	Electricity	small car	7.0	113	

In [59]:
```
# vasualisation best cars by Year
dp_08 = fuel_08.iloc[:, 8:11].iloc[0]
dp_18 = fuel_18.iloc[:, 8:11].iloc[0]

plt.subplots(figsize=(10,7))
plt.bar(dp_08.index, dp_08, label="TOYOTA Prius (2008)")
plt.bar(dp_18.index, dp_18, bottom=dp_08, label="CHEVROLET Volt (2018)")
plt.title("Fuel Economy of Best Car by Years", fontsize=16)
plt.xlabel("Categories", fontsize=12)
plt.ylabel("Fule Economy", fontsize=12)
plt.legend(loc= 'best')
plt.show()
```



As we can see here The best fuel economy car in 2008 was TOYOTA Prius with 46 mpg and in 2018 the mpg reach to 106, and the best car for 2018 is CHEVROLET Volt.

In [ ]: