SC1101_u5613415_Tanapat_t

June 7, 2018

In [2]:
```
# Follow www.analyticsvidhya.com to do the pandas
import pandas as pd
import numpy as np
import matplotlib as plt
```

In [5]:
```
df = pd.read_csv("C:/Users/EGAT/Downloads/train.csv") #Reading the dataset in a datafram
```

In [7]:
```
#Quick Data Exploration
df.head(10)
```

Out[7]:

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0.0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1.0	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0.0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0.0	Not Graduate	No	2583	2358.0	120.0	360.0			
4	LP001008	Male	No	0.0	Graduate	No	6000	0.0	141.0	360.0			
5	LP001011	Male	Yes	2.0	Graduate	Yes	5417	4196.0	267.0	360.0			
6	LP001013	Male	Yes	0.0	Not Graduate	No	2333	1516.0	95.0	360.0			
7	LP001014	Male	Yes	3.0	Graduate	No	3036	2504.0	158.0	360.0			
8	LP001018	Male	Yes	2.0	Graduate	No	4006	1526.0	168.0	360.0			
9	LP001020	Male	Yes	1.0	Graduate	No	12841	10968.0	349.0	360.0			

```
3             1.0         Urban           Y
4             1.0         Urban           Y
5             1.0         Urban           Y
6             1.0         Urban           Y
7             0.0     Semiurban           N
8             1.0         Urban           Y
9             1.0     Semiurban           N
```

In [8]:
```
# The non-numerical values (e.g. Property_Area, Credit_History etc.), we can look at fre
df['Property_Area'].value_counts()
```

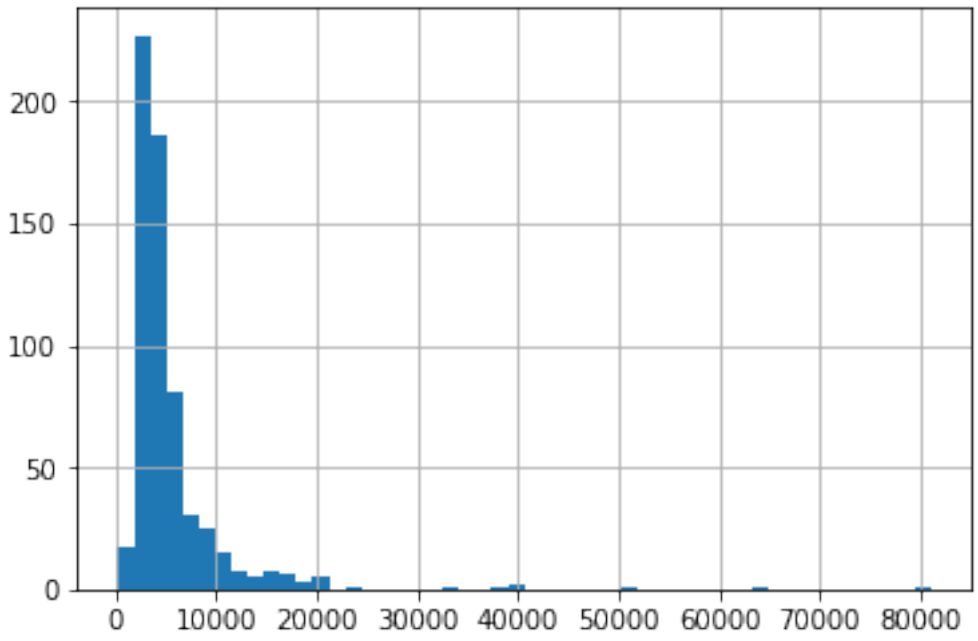
Out[8]:
```
Semiurban    236
Urban        205
Rural        183
Name: Property_Area, dtype: int64
```

In [9]:
```
df['ApplicantIncome'].hist(bins=50)
#Here we observe that there are few extreme values.
```

Out[9]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa06703bd30>
```

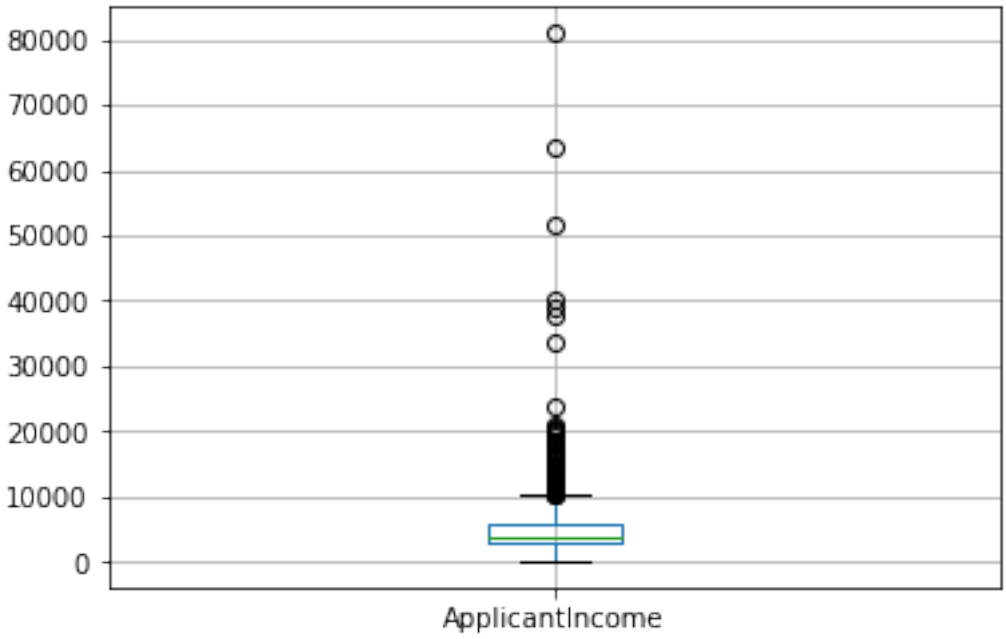
In [10]:
```
df['ApplicantIncome'].hist(bins=50)
 #Here we observe that there are few extreme values.
```

Out[10]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa06732b5f8>
```



In [11]:
```
df.boxplot(column='ApplicantIncome')
#we look at box plots to understand the distributions. Box plot for fare can be plotted
```

Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0xa067461a90>



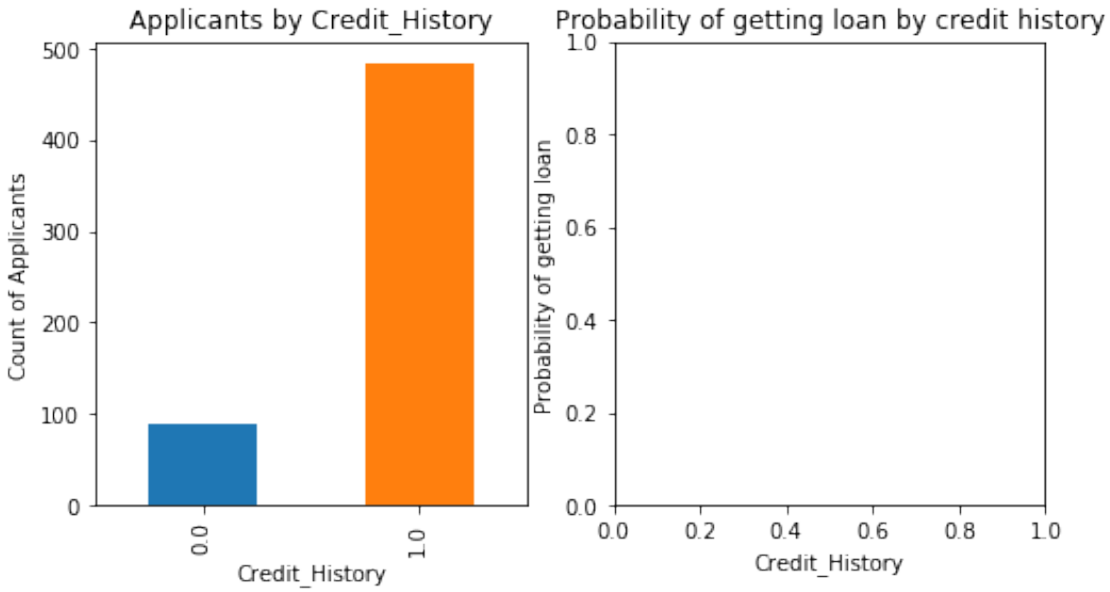
In [12]:
```
#We will look at the steps required to generate a similar insight using Python.
temp1 = df['Credit_History'].value_counts(ascending=True)
temp2 = df.pivot_table(values='Loan_Status',index=['Credit_History'],aggfunc=lambda x:
print ('Frequency Table for Credit History:')
print(temp1)

print('Probility of getting loan for each Credit History class:')
print(temp2)
```

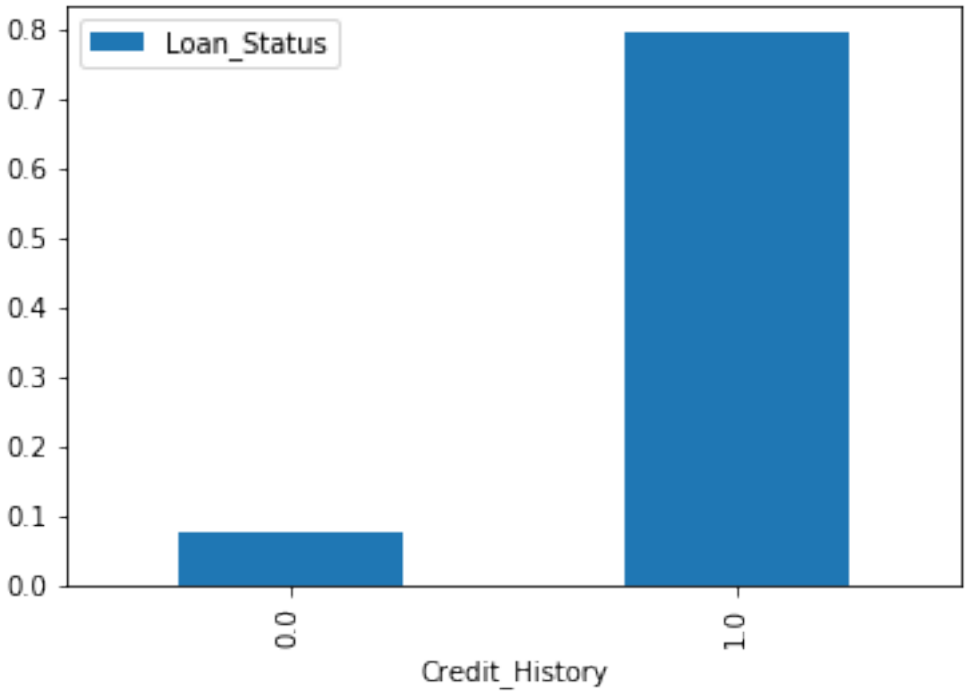
```
Frequency Table for Credit History:
0.0     90
1.0    484
Name: Credit_History, dtype: int64
Probility of getting loan for each Credit History class:
               Loan_Status
Credit_History
0.0                0.077778
1.0                0.797521
```

In [13]:
```
import matplotlib.pyplot as plt
fig = plt.figure(figsize=(8,4))
```

```
ax1 = fig.add_subplot(121)
ax1.set_xlabel('Credit_History')
ax1.set_ylabel('Count of Applicants')
ax1.set_title("Applicants by Credit_History")
temp1.plot(kind='bar')

ax2 = fig.add_subplot(122)
temp2.plot(kind = 'bar')
ax2.set_xlabel('Credit_History')
ax2.set_ylabel('Probability of getting loan')
ax2.set_title("Probability of getting loan by credit history")

#Now we can observe that we get a similar pivot_table like the MS Excel one.
#This can be plotted as a bar chart using the matplotlib library with following code:
```

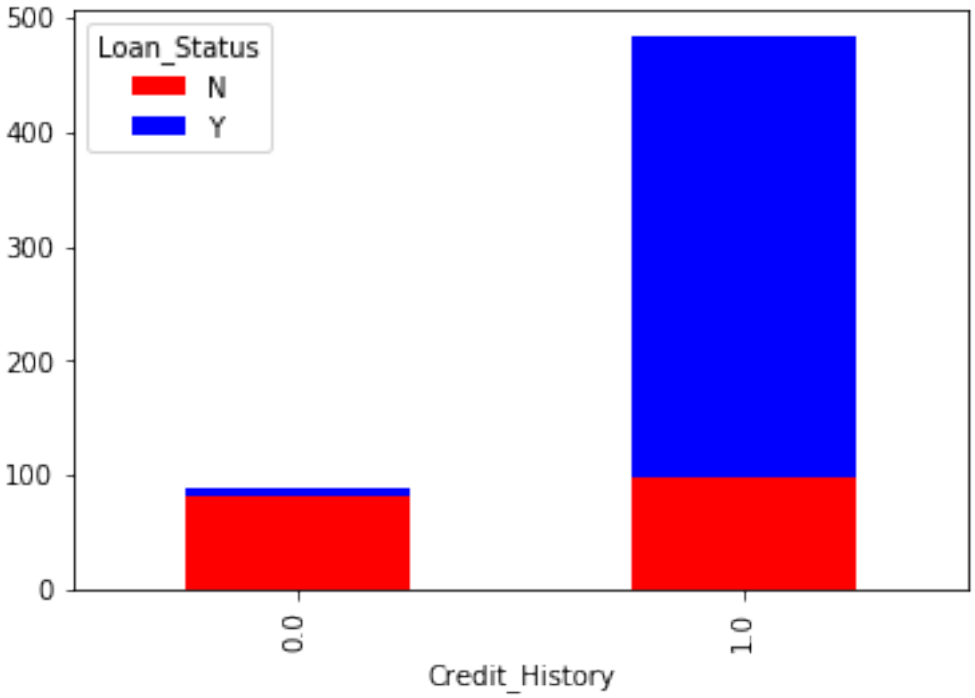
Out[13]: Text(0.5,1,'Probability of getting loan by credit history')





In [14]:
```
#This shows that the chances of getting a loan are eight-fold if the applicant has a va
#You can plot similar graphs by Married, Self-Employed, Property_Area, etc.
temp3 = pd.crosstab(df['Credit_History'], df['Loan_Status'])
temp3.plot(kind='bar', stacked=True, color=['red','blue'], grid=False)
```

Out[14]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa067557978>
```



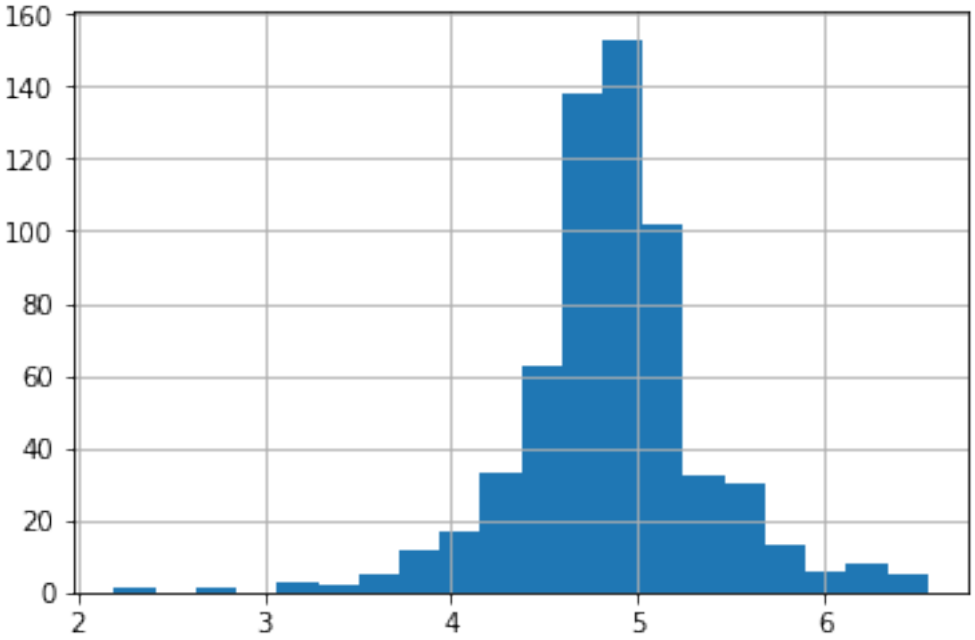
In [15]:
```
#Check missing values in the dataset
df.apply(lambda x: sum(x.isnull()),axis=0)
#This command should tell us the number of missing values in each column as isnull() re
```

Out[15]:
```
Loan_ID               0
Gender               13
Married               3
Dependents           15
Education             1
Self_Employed        32
ApplicantIncome       0
CoapplicantIncome     0
LoanAmount           23
Loan_Amount_Term     14
Credit_History       50
Property_Area         0
Loan_Status           0
dtype: int64
```

In [16]:
```
#The simplest being replacement by mean, which can be done by following code:
df['LoanAmount'].fillna(df['LoanAmount'].mean(), inplace=True)
```

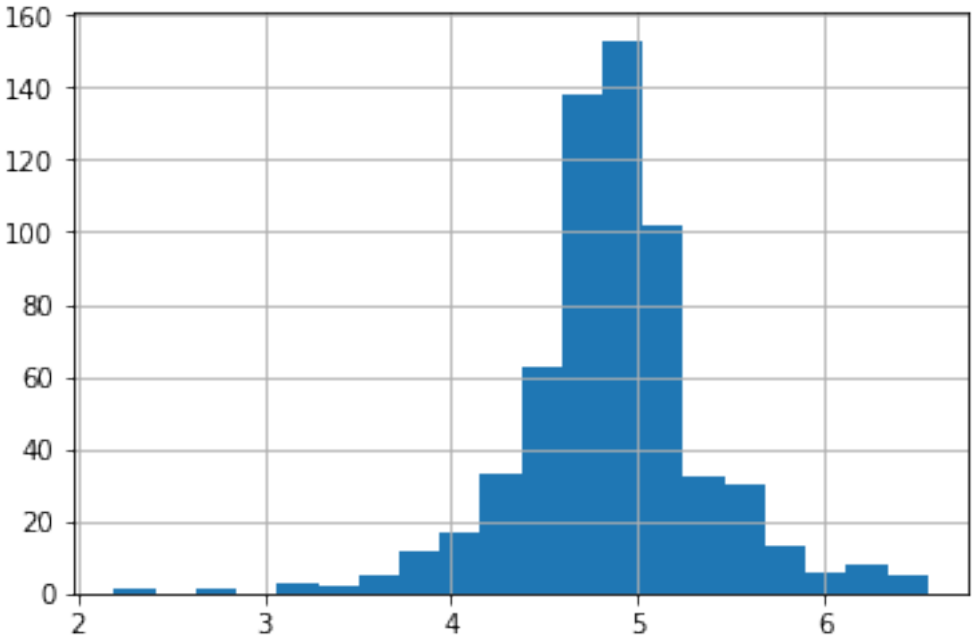
In [17]:
```
df['Self_Employed'].fillna('No',inplace=True)
#Since ~86% values are No, it is safe to impute the missing values as No as there is a
df['LoanAmount_log'] = np.log(df['LoanAmount'])
df['LoanAmount_log'].hist(bins=20)
```

Out[17]: <matplotlib.axes._subplots.AxesSubplot at 0xa068b17128>



In [18]:
```
df['TotalIncome'] = df['ApplicantIncome'] + df['CoapplicantIncome']
df['TotalIncome_log'] = np.log(df['TotalIncome'])
df['LoanAmount_log'].hist(bins=20)
```

Out[18]: <matplotlib.axes._subplots.AxesSubplot at 0xa068e86588>



In [19]:
```
#Then i try to do pandas with others  new dataset to observe

import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
%matplotlib inline
```

In [20]:
```
df = pd.read_csv("https://raw.githubusercontent.com/prasertcbs/tutorial/master/mpg.csv"
df.head()
# to observe the data with 5  rows by default
```

Out[20]:

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class
0	audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact
1	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
2	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
3	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact
4	audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact

In [21]:
```
df['cty_kml'] = df['cty'] * .425143707 #to create new column  just type nametable["name
df.head()
```

Out[21]:

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class	\
0	audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact	
1	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact	
2	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact	
3	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact	
4	audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact	

	cty_kml
0	7.652587
1	8.928018
2	8.502874
3	8.928018
4	6.802299

In [22]:
```
df['hwy_kml'] = df.hwy * 4.25143707
df.head()
```

Out[22]:

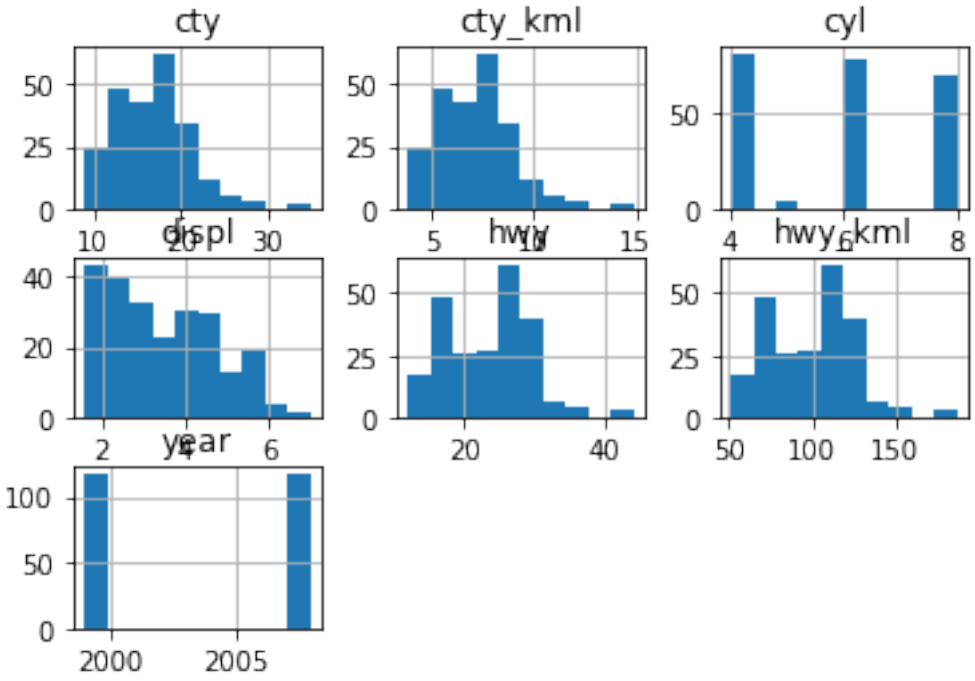
	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	class	\
0	audi	a4	1.8	1999	4	auto(l5)	f	18	29	p	compact	
1	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact	
2	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact	
3	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact	
4	audi	a4	2.8	1999	6	auto(l5)	f	16	26	p	compact	

	cty_kml	hwy_kml

```
0  7.652587  123.291675
1  8.928018  123.291675
2  8.502874  131.794549
3  8.928018  127.543112
4  6.802299  110.537364
```

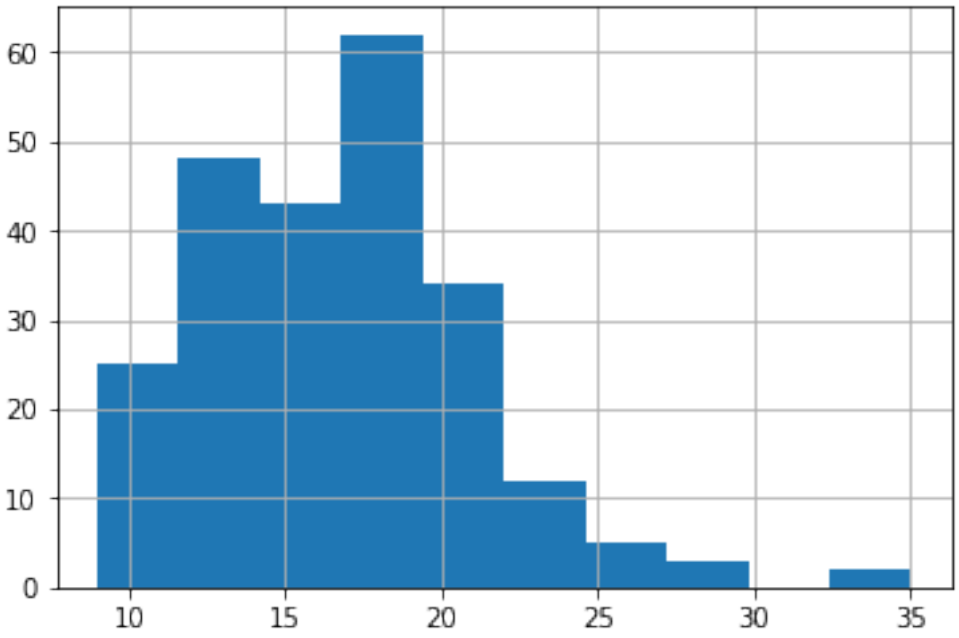
In [24]:
```
df.hist() #program will collect column to create histogram when data is numerical or in
```

Out[24]:
```
array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000000A06772B320>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000A068C80E80>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000A068D93E80>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000000A068DCEE80>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000A068E0AE80>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000A068E0AEB8>],
       [<matplotlib.axes._subplots.AxesSubplot object at 0x000000A069884940>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000A0698C1E10>,
        <matplotlib.axes._subplots.AxesSubplot object at 0x000000A069903390>]],
      dtype=object)
```



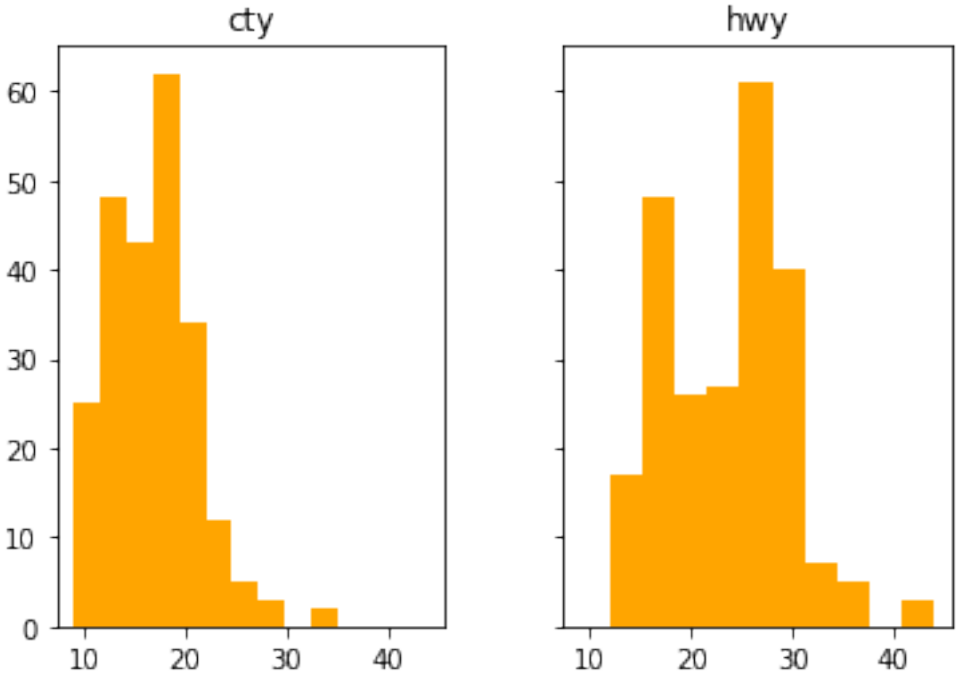
In [23]:
```
df.cty.hist() # to see in each column just code df.name_column.hist()
```

Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa067794d30>
```



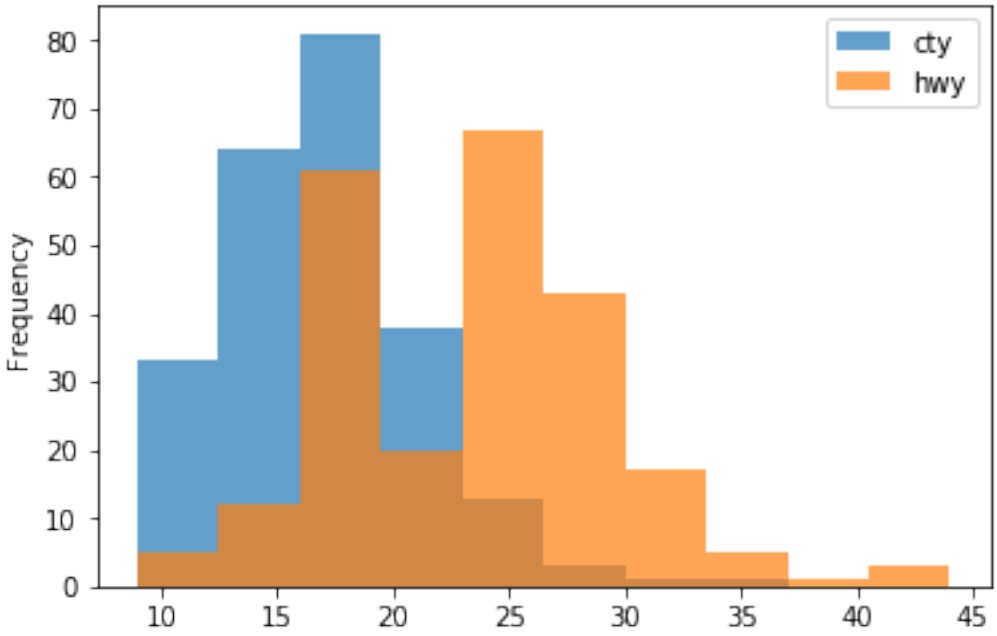

```
In [25]: df[["cty","hwy"]].hist(grid = False, color = "orange",sharex = True , sharey = True) #u
```

```
Out[25]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x000000A069A64C88>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x000000A069AAEB38>]],
               dtype=object)
```



In [26]:
```
df[["cty","hwy"]].plot.hist(alpha = .7)
```

Out[26]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa069b0b668>
```

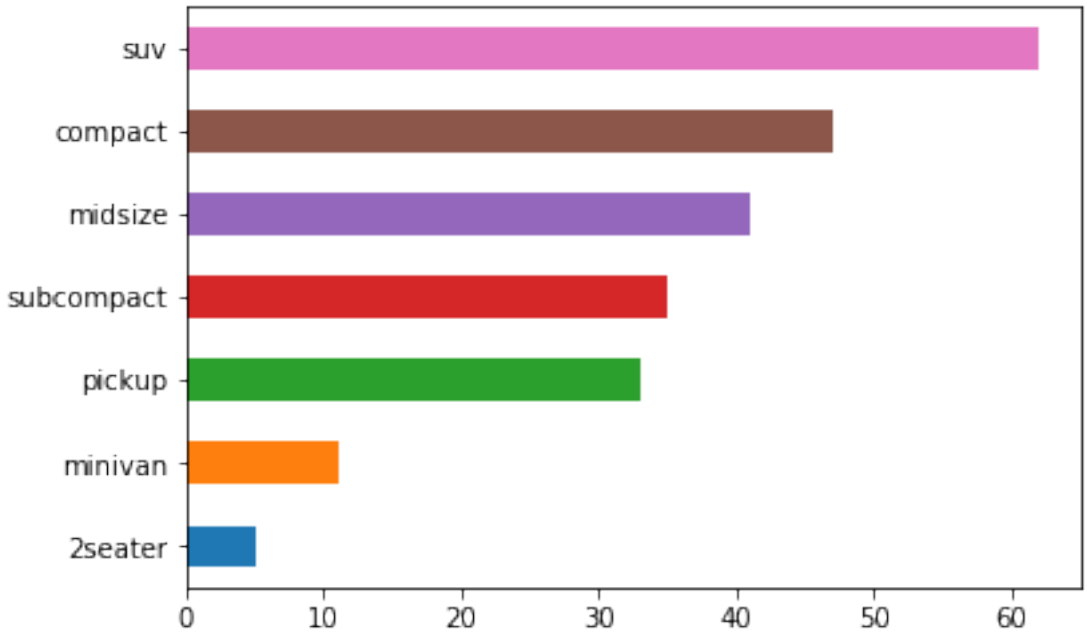


In [27]:
```
# if you want to see some column whose column name is class call  then we just code lik
df["class"].value_counts()
```

Out[27]:
```
suv           62
compact       47
midsize       41
subcompact    35
pickup        33
minivan       11
2seater        5
Name: class, dtype: int64
```

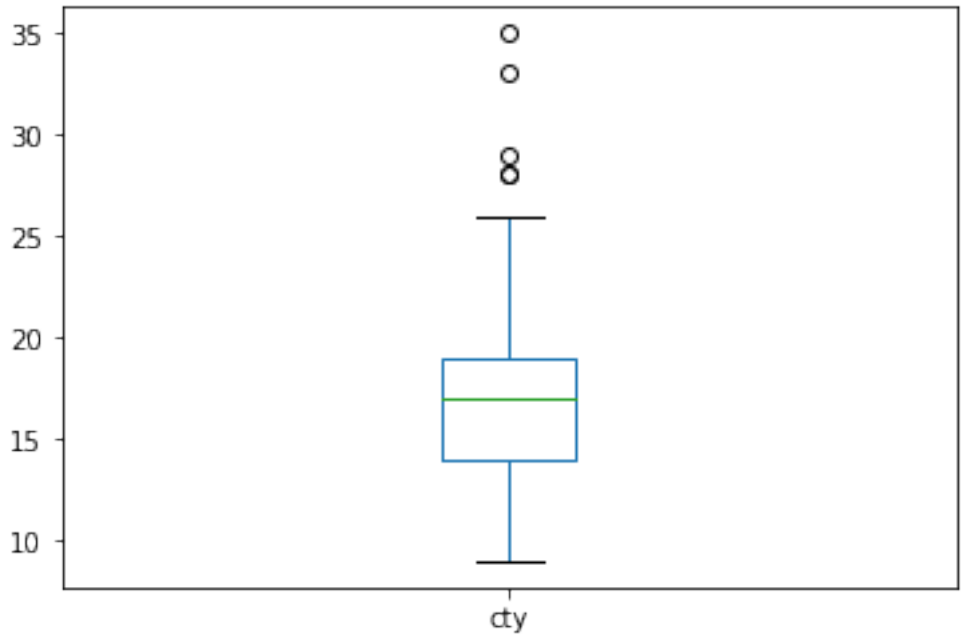
In [28]:
```
df["class"].value_counts().sort_values().plot.barh()
```

Out[28]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa069bd9278>
```



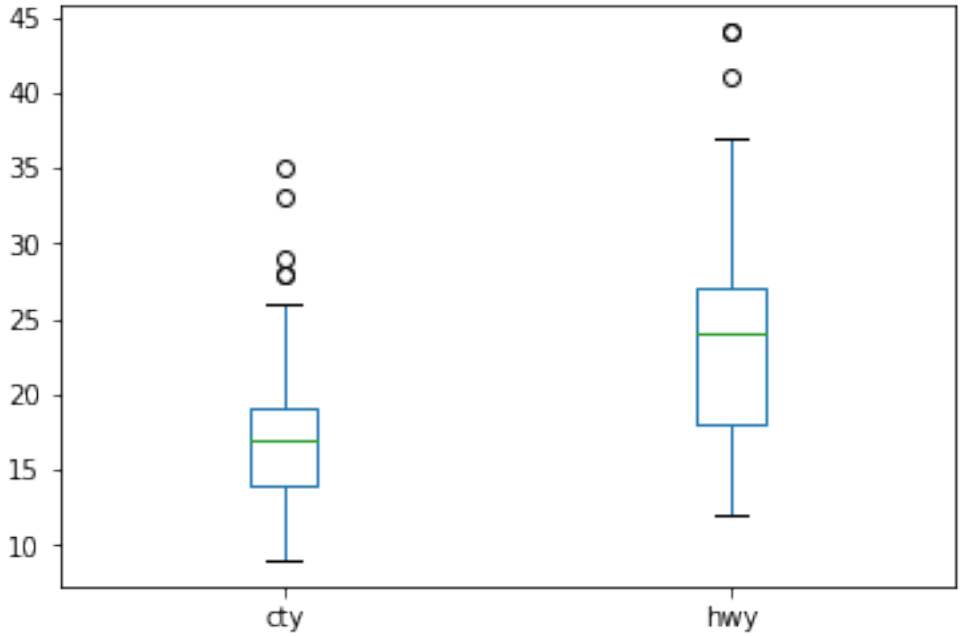
In [29]:
```
#Next is to create boxplot
df.cty.plot.box()
```

Out[29]:
```
<matplotlib.axes._subplots.AxesSubplot at 0xa069a535f8>
```



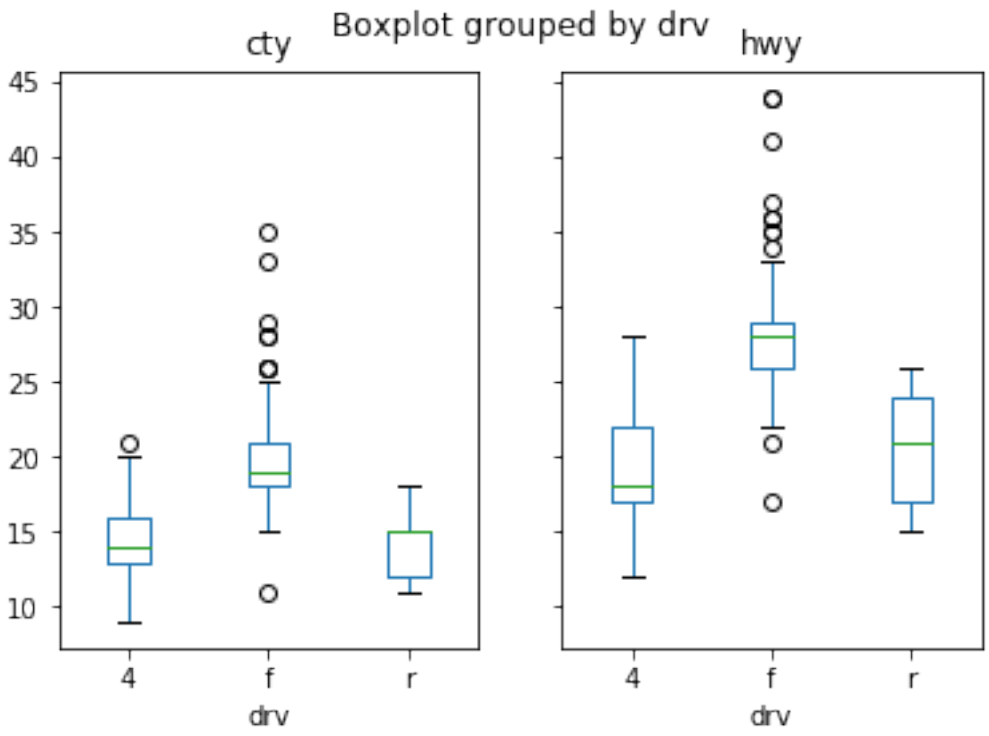
```
In [30]: #Another way to create boxplot
         df[['cty','hwy']].plot.box()
```

```
Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0xa06773ac88>
```




```
In [31]: df.boxplot(column=['cty','hwy'], by = 'drv', grid = False)
```

```
Out[31]: array([<matplotlib.axes._subplots.AxesSubplot object at 0x000000A069C74A58>,
                <matplotlib.axes._subplots.AxesSubplot object at 0x000000A069D7F6A0>],
               dtype=object)
```



In [32]:
```
#To use aggregate function
df.agg(['mean']) #to see only mean
```

Out[32]:

	displ	year	cyl	cty	hwy	cty_kml	hwy_kml
mean	3.471795	2003.5	5.888889	16.858974	23.440171	7.167487	99.654412

In [33]:
```
df.agg(['min', 'max' , 'mean'])
```

Out[33]:

	manufacturer	model	displ	year	cyl	trans	\
max	volkswagen	toyota tacoma 4wd	7.000000	2008.0	8.000000	manual(m6)	
mean	NaN	NaN	3.471795	2003.5	5.888889	NaN	
min	audi	4runner 4wd	1.600000	1999.0	4.000000	auto(av)	

	drv	cty	hwy	fl	class	cty_kml	hwy_kml
max	r	35.000000	44.000000	r	suv	14.880030	187.063231
mean	NaN	16.858974	23.440171	NaN	NaN	7.167487	99.654412
min	4	9.000000	12.000000	c	2seater	3.826293	51.017245

In [34]:
```
#This is to group only mean in each year
df.groupby('year').mean()
```

Out[34]:

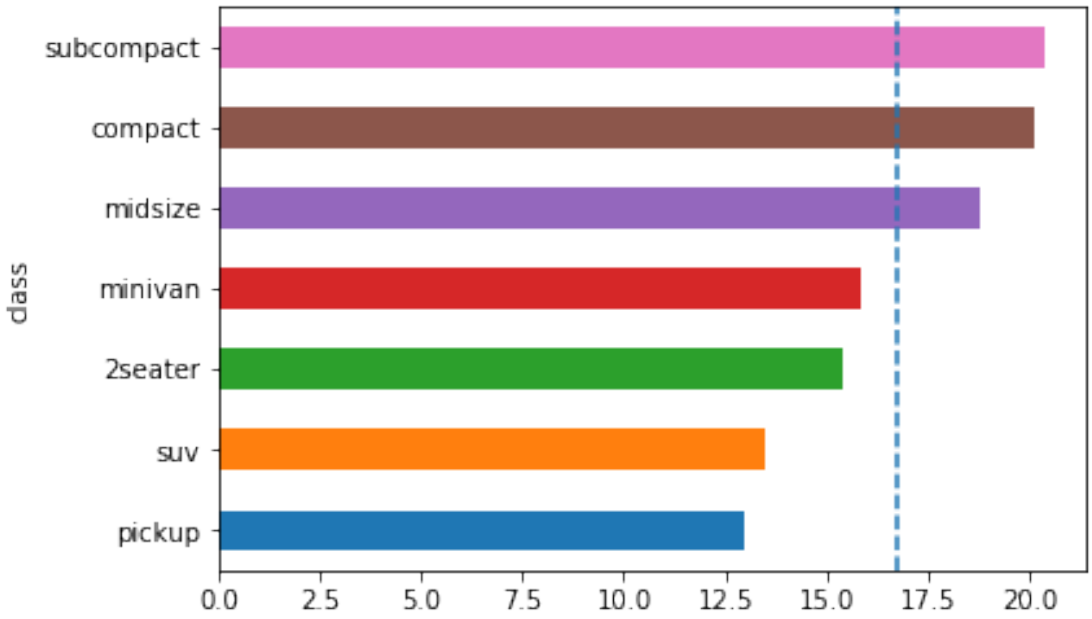
year	displ	cyl	cty	hwy	cty_kml	hwy_kml
1999	3.281197	5.692308	17.017094	23.427350	7.234710	99.599906
2008	3.662393	6.085470	16.700855	23.452991	7.100263	99.708917

In [35]:
```
#avg only class
avg = df.groupby('class').cty.mean().mean()
avg
```

Out[35]:
```
16.710481075007728
```

In [36]:
```
df.groupby('class').cty.mean().sort_values().plot(kind = 'barh')
plt.axvline(x=avg, linestyle = '--')
```

Out[36]:
```
<matplotlib.lines.Line2D at 0xa069e87940>
```



In [37]:
```
# How to handle duplicate values
df.drop_duplicates()
```

Out[37]:

	manufacturer	model	displ	year	cyl	trans	drv	cty	\
0	audi	a4	1.8	1999	4	auto(l5)	f	18	
1	audi	a4	1.8	1999	4	manual(m5)	f	21	
2	audi	a4	2.0	2008	4	manual(m6)	f	20	
3	audi	a4	2.0	2008	4	auto(av)	f	21	
4	audi	a4	2.8	1999	6	auto(l5)	f	16	
5	audi	a4	2.8	1999	6	manual(m5)	f	18	
6	audi	a4	3.1	2008	6	auto(av)	f	18	
7	audi	a4 quattro	1.8	1999	4	manual(m5)	4	18	
8	audi	a4 quattro	1.8	1999	4	auto(l5)	4	16	
9	audi	a4 quattro	2.0	2008	4	manual(m6)	4	20	
10	audi	a4 quattro	2.0	2008	4	auto(s6)	4	19	
11	audi	a4 quattro	2.8	1999	6	auto(l5)	4	15	

12	audi	a4 quattro	2.8	1999	6	manual(m5)	4	17
13	audi	a4 quattro	3.1	2008	6	auto(s6)	4	17
14	audi	a4 quattro	3.1	2008	6	manual(m6)	4	15
15	audi	a6 quattro	2.8	1999	6	auto(l5)	4	15
16	audi	a6 quattro	3.1	2008	6	auto(s6)	4	17
17	audi	a6 quattro	4.2	2008	8	auto(s6)	4	16
18	chevrolet	c1500 suburban 2wd	5.3	2008	8	auto(l4)	r	14
19	chevrolet	c1500 suburban 2wd	5.3	2008	8	auto(l4)	r	11
21	chevrolet	c1500 suburban 2wd	5.7	1999	8	auto(l4)	r	13
22	chevrolet	c1500 suburban 2wd	6.0	2008	8	auto(l4)	r	12
23	chevrolet	corvette	5.7	1999	8	manual(m6)	r	16
24	chevrolet	corvette	5.7	1999	8	auto(l4)	r	15
25	chevrolet	corvette	6.2	2008	8	manual(m6)	r	16
26	chevrolet	corvette	6.2	2008	8	auto(s6)	r	15
27	chevrolet	corvette	7.0	2008	8	manual(m6)	r	15
28	chevrolet	k1500 tahoe 4wd	5.3	2008	8	auto(l4)	4	14
29	chevrolet	k1500 tahoe 4wd	5.3	2008	8	auto(l4)	4	11
30	chevrolet	k1500 tahoe 4wd	5.7	1999	8	auto(l4)	4	11
..	...	...	...	...	...	...	..	...
204	toyota	toyota tacoma 4wd	3.4	1999	6	auto(l4)	4	15
205	toyota	toyota tacoma 4wd	4.0	2008	6	manual(m6)	4	15
206	toyota	toyota tacoma 4wd	4.0	2008	6	auto(l5)	4	16
207	volkswagen	gti	2.0	1999	4	manual(m5)	f	21
208	volkswagen	gti	2.0	1999	4	auto(l4)	f	19
209	volkswagen	gti	2.0	2008	4	manual(m6)	f	21
210	volkswagen	gti	2.0	2008	4	auto(s6)	f	22
211	volkswagen	gti	2.8	1999	6	manual(m5)	f	17
212	volkswagen	jetta	1.9	1999	4	manual(m5)	f	33
213	volkswagen	jetta	2.0	1999	4	manual(m5)	f	21
214	volkswagen	jetta	2.0	1999	4	auto(l4)	f	19
215	volkswagen	jetta	2.0	2008	4	auto(s6)	f	22
216	volkswagen	jetta	2.0	2008	4	manual(m6)	f	21
217	volkswagen	jetta	2.5	2008	5	auto(s6)	f	21
218	volkswagen	jetta	2.5	2008	5	manual(m5)	f	21
219	volkswagen	jetta	2.8	1999	6	auto(l4)	f	16
220	volkswagen	jetta	2.8	1999	6	manual(m5)	f	17
221	volkswagen	new beetle	1.9	1999	4	manual(m5)	f	35
222	volkswagen	new beetle	1.9	1999	4	auto(l4)	f	29
223	volkswagen	new beetle	2.0	1999	4	manual(m5)	f	21
224	volkswagen	new beetle	2.0	1999	4	auto(l4)	f	19
225	volkswagen	new beetle	2.5	2008	5	manual(m5)	f	20
226	volkswagen	new beetle	2.5	2008	5	auto(s6)	f	20
227	volkswagen	passat	1.8	1999	4	manual(m5)	f	21
228	volkswagen	passat	1.8	1999	4	auto(l5)	f	18
229	volkswagen	passat	2.0	2008	4	auto(s6)	f	19
230	volkswagen	passat	2.0	2008	4	manual(m6)	f	21
231	volkswagen	passat	2.8	1999	6	auto(l5)	f	16
232	volkswagen	passat	2.8	1999	6	manual(m5)	f	18

```
233   volkswagen              passat    3.6  2008    6    auto(s6)   f   17
```

	hwy	fl	class	cty_kml	hwy_kml
0	29	p	compact	7.652587	123.291675
1	29	p	compact	8.928018	123.291675
2	31	p	compact	8.502874	131.794549
3	30	p	compact	8.928018	127.543112
4	26	p	compact	6.802299	110.537364
5	26	p	compact	7.652587	110.537364
6	27	p	compact	7.652587	114.788801
7	26	p	compact	7.652587	110.537364
8	25	p	compact	6.802299	106.285927
9	28	p	compact	8.502874	119.040238
10	27	p	compact	8.077730	114.788801
11	25	p	compact	6.377156	106.285927
12	25	p	compact	7.227443	106.285927
13	25	p	compact	7.227443	106.285927
14	25	p	compact	6.377156	106.285927
15	24	p	midsize	6.377156	102.034490
16	25	p	midsize	7.227443	106.285927
17	23	p	midsize	6.802299	97.783053
18	20	r	suv	5.952012	85.028741
19	15	e	suv	4.676581	63.771556
21	17	r	suv	5.526868	72.274430
22	17	r	suv	5.101724	72.274430
23	26	p	2seater	6.802299	110.537364
24	23	p	2seater	6.377156	97.783053
25	26	p	2seater	6.802299	110.537364
26	25	p	2seater	6.377156	106.285927
27	24	p	2seater	6.377156	102.034490
28	19	r	suv	5.952012	80.777304
29	14	e	suv	4.676581	59.520119
30	15	r	suv	4.676581	63.771556
..	...	..	...	...	...
204	19	r	pickup	6.377156	80.777304
205	18	r	pickup	6.377156	76.525867
206	20	r	pickup	6.802299	85.028741
207	29	r	compact	8.928018	123.291675
208	26	r	compact	8.077730	110.537364
209	29	p	compact	8.928018	123.291675
210	29	p	compact	9.353162	123.291675
211	24	r	compact	7.227443	102.034490
212	44	d	compact	14.029742	187.063231
213	29	r	compact	8.928018	123.291675
214	26	r	compact	8.077730	110.537364
215	29	p	compact	9.353162	123.291675
216	29	p	compact	8.928018	123.291675
217	29	r	compact	8.928018	123.291675

```
218   29  r     compact   8.928018  123.291675
219   23  r     compact   6.802299   97.783053
220   24  r     compact   7.227443  102.034490
221   44  d  subcompact  14.880030  187.063231
222   41  d  subcompact  12.329168  174.308920
223   29  r  subcompact   8.928018  123.291675
224   26  r  subcompact   8.077730  110.537364
225   28  r  subcompact   8.502874  119.040238
226   29  r  subcompact   8.502874  123.291675
227   29  p     midsize   8.928018  123.291675
228   29  p     midsize   7.652587  123.291675
229   28  p     midsize   8.077730  119.040238
230   29  p     midsize   8.928018  123.291675
231   26  p     midsize   6.802299  110.537364
232   26  p     midsize   7.652587  110.537364
233   26  p     midsize   7.227443  110.537364

[225 rows x 13 columns]
```

In [38]:
```
df.drop_duplicates(subset = 'manufacturer') #to remove duplicate in each colum just use
```

Out[38]:

```
    manufacturer                model  displ  year  cyl       trans drv  cty  \
0           audi                   a4    1.8  1999    4    auto(l5)   f   18
18     chevrolet  c1500 suburban 2wd    5.3  2008    8    auto(l4)   r   14
37         dodge         caravan 2wd    2.4  1999    4    auto(l3)   f   18
74          ford      expedition 2wd    4.6  1999    8    auto(l4)   r   11
99         honda               civic    1.6  1999    4  manual(m5)   f   28
108      hyundai              sonata    2.4  1999    4    auto(l4)   f   18
122         jeep  grand cherokee 4wd    3.0  2008    6    auto(l5)   4   17
130   land rover         range rover    4.0  1999    8    auto(l4)   4   11
134      lincoln       navigator 2wd    5.4  1999    8    auto(l4)   r   11
137      mercury     mountaineer 4wd    4.0  1999    6    auto(l5)   4   14
141       nissan              altima    2.4  1999    4  manual(m5)   f   21
154      pontiac          grand prix    3.1  1999    6    auto(l4)   f   18
159       subaru        forester awd    2.5  1999    4  manual(m5)   4   18
173       toyota         4runner 4wd    2.7  1999    4  manual(m5)   4   15
207   volkswagen                 gti    2.0  1999    4  manual(m5)   f   21

     hwy fl       class    cty_kml     hwy_kml
0     29  p     compact   7.652587  123.291675
18    20  r         suv   5.952012   85.028741
37    24  r     minivan   7.652587  102.034490
74    17  r         suv   4.676581   72.274430
99    33  r  subcompact  11.904024  140.297423
108   26  r     midsize   7.652587  110.537364
122   22  d         suv   7.227443   93.531616
130   15  p         suv   4.676581   63.771556
134   17  r         suv   4.676581   72.274430
```

```
137   17  r         suv   5.952012   72.274430
141   29  r     compact   8.928018  123.291675
154   26  r     midsize   7.652587  110.537364
159   25  r         suv   7.652587  106.285927
173   20  r         suv   6.377156   85.028741
207   29  r     compact   8.928018  123.291675
```

In [39]:
```
# to create cross tab
pd.crosstab(df.drv,df['class']) #due rto class is key word in python .we must use blank
# it seems that compact cars that use 4 wheel  drv have 12 model.
```

Out[39]:
```
class  2seater  compact  midsize  minivan  pickup  subcompact  suv
drv
4            0       12        3        0      33           4   51
f            0       35       38       11       0          22    0
r            5        0        0        0       0           9   11
```

In [40]:
```
pd.crosstab(df.drv,df['class'], margins = True) #if you want to know total just add mar
```

Out[40]:
```
class  2seater  compact  midsize  minivan  pickup  subcompact  suv  All
drv
4            0       12        3        0      33           4   51  103
f            0       35       38       11       0          22    0  106
r            5        0        0        0       0           9   11   25
All          5       47       41       11      33          35   62  234
```

In [41]:
```
pd.crosstab(df.drv,df['class'], margins = True, normalize = 'all') #we try to use norma
```

Out[41]:
```
class   2seater   compact   midsize   minivan    pickup  subcompact       suv  \
drv
4      0.000000  0.051282  0.012821  0.000000  0.141026    0.017094  0.217949
f      0.000000  0.149573  0.162393  0.047009  0.000000    0.094017  0.000000
r      0.021368  0.000000  0.000000  0.000000  0.000000    0.038462  0.047009
All    0.021368  0.200855  0.175214  0.047009  0.141026    0.149573  0.264957

class       All
drv
4      0.440171
f      0.452991
r      0.106838
All    1.000000
```

In [42]:
```
(pd.crosstab(df.drv,df['class'], margins = True, normalize = 'all')*100).round(1) #we c
```

Out[42]:
```
class  2seater  compact  midsize  minivan  pickup  subcompact   suv    All
drv
4          0.0      5.1      1.3      0.0    14.1         1.7  21.8   44.0
f          0.0     15.0     16.2      4.7     0.0         9.4   0.0   45.3
r          2.1      0.0      0.0      0.0     0.0         3.8   4.7   10.7
All        2.1     20.1     17.5      4.7    14.1        15.0  26.5  100.0
```

In [88]:
```
#Newt we will do traing and test datase by using this datatype
url = "https://raw.githubusercontent.com/prasertcbs/tutorial/master/mtcars.csv"
df2 = pd.read_csv(url,nrows = 10)
df2
```

Out[88]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

In [89]:
```
#Next step i will show you how to do training and test data set
#normally training data wille be 70% and 30 % will be test set
df2.shape
```

Out[89]: (10, 12)

In [90]:
```
np.iinfo(np.int32).max
```

Out[90]: 2147483647

In [93]:
```
np.random.randint(np.iinfo(np.int32).min,np.iinfo(np.int32).max)
#now you will get the random number that every time you run ,it will random new number
```

Out[93]: 865473121

In [110]:
```
np.random.randint(np.iinfo(np.int32).min,np.iinfo(np.int32).max,5)
 #this will random 5 number that every time you run
```

Out[110]: array([752588946, 1750824090, 843263716, 1343895736, 1714292689])

In [111]:
```
df2['uid'] =np.random.randint(np.iinfo(np.int32).min,np.iinfo(np.int32).max,df2.shape[
df2
#now we wis ill create column that  is usi
```

Out[111]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	uid
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	2096065320
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	-106090991
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	-847626544
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	-1643140384
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2	400715154
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	-1432445439
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	-400941851
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	-107469547
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2	846756041
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	-366665134

In [115]:
```
df2.sort_values(['uid'], inplace =True)
df2
```

Out[115]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	uid
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	-1643140384
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	-1432445439
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	-847626544
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	-400941851
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	-366665134
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	-107469547
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	-106090991
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2	400715154
8	Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2	846756041
0	Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4	2096065320

```
9     4  -366665134
7     2  -107469547
1     4  -106090991
4     2   400715154
8     2   846756041
0     4  2096065320
```

In [117]:
```
train_ratio = 0.7
df_train = df2[:int(df2.shape[0] * train_ratio)]
df_train
#
```

Out[117]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	uid
3	Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1	-1643140384
5	Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1	-1432445439
2	Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1	-847626544
6	Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4	-400941851
9	Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4	-366665134
7	Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2	-107469547
1	Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4	-106090991

In [118]:
```
df_train.shape# it means that the fist 7 row will be training set that is 70%
```

Out[118]:
```
(7, 13)
```

In [120]:
```
df_test = df2[int(df2.shape[0] * train_ratio):]
df_test
```

Out[120]:

	model	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb	uid
4	Hornet Sportabout	18.7	8	360.0	175	3.15	3.44	17.02	0	0	3	2	400715154
8	Merc 230	22.8	4	140.8	95	3.92	3.15	22.90	1	0	4	2	846756041
0	Mazda RX4	21.0	6	160.0	110	3.90	2.62	16.46	0	1	4	4	2096065320

In [121]:
```
df_test.shape# test set will be 30%
# this is the example how to split data in to 70% and 30%
```

Out[121]: (3, 13)

In [114]:
```
#This is to do cluster analysis by getting the dataset from the link below to show you
#'well e clustering companies using their stock market prices, and distinguishing diff
df = pd.read_csv("https://assets.datacamp.com/production/course_2072/datasets/company-
df.head()
```

Out[114]:

	Unnamed: 0	2010-01-04	2010-01-05	2010-01-06	2010-01-07	\
0	Apple	0.580000	-0.220005	-3.409998	-1.170000	
1	AIG	-0.640002	-0.650000	-0.210001	-0.420000	
2	Amazon	-2.350006	1.260009	-2.350006	-2.009995	
3	American express	0.109997	0.000000	0.260002	0.720002	
4	Boeing	0.459999	1.770000	1.549999	2.690003	

	2010-01-08	2010-01-11	2010-01-12	2010-01-13	2010-01-14	...	\
0	1.680011	-2.689994	-1.469994	2.779997	-0.680003	...	
1	0.710001	-0.200001	-1.130001	0.069999	-0.119999	...	
2	2.960006	-2.309997	-1.640007	1.209999	-1.790001	...	
3	0.190003	-0.270001	0.750000	0.300004	0.639999	...	
4	0.059997	-1.080002	0.360000	0.549999	0.530002	...	

	2013-10-16	2013-10-17	2013-10-18	2013-10-21	2013-10-22	2013-10-23	\
0	0.320008	4.519997	2.899987	9.590019	-6.540016	5.959976	
1	0.919998	0.709999	0.119999	-0.480000	0.010002	-0.279998	
2	2.109985	3.699982	9.570008	-3.450013	4.820008	-4.079986	
3	0.680001	2.290001	0.409996	-0.069999	0.100006	0.069999	
4	1.559997	2.480003	0.019997	-1.220001	0.480003	3.020004	

	2013-10-24	2013-10-25	2013-10-28	2013-10-29
0	6.910011	-5.359962	0.840019	-19.589981
1	-0.190003	-0.040001	-0.400002	0.660000
2	2.579986	4.790009	-1.760009	3.740021
3	0.130005	1.849999	0.040001	0.540001
4	-0.029999	1.940002	1.130005	0.309998

[5 rows x 964 columns]

In [53]:
```
#points of size 300x2, where each row gives the (x, y) co-ordinates of a point on a map
points = np.array([[ 0.06544649, -0.76866376],
       [-1.52901547, -0.42953079],
       [ 1.70993371,  0.69885253],
       [ 1.16779145,  1.01262638],
       [-1.80110088, -0.31861296],
       [-1.63567888, -0.02859535],
       [ 1.21990375,  0.74643463],
       [-0.26175155, -0.62492939],
       [-1.61925804, -0.47983949],
       [-1.84329582, -0.16694431],
```

```
[ 1.35999602,  0.94995827],
[ 0.42291856, -0.7349534 ],
[-1.68576139,  0.10686728],
[ 0.90629995,  1.09105162],
[-1.56478322, -0.84675394],
[-0.0257849 , -1.18672539],
[ 0.83027324,  1.14504612],
[ 1.22450432,  1.35066759],
[-0.15394596, -0.71704301],
[ 0.86358809,  1.06824613],
[-1.43386366, -0.2381297 ],
[ 0.03844769, -0.74635022],
[-1.58567922,  0.08499354],
[ 0.6359888 , -0.58477698],
[ 0.24417242, -0.53172465],
[-2.19680359,  0.49473677],
[ 1.0323503 , -0.55688   ],
[-0.28858067, -0.39972528],
[ 0.20597008, -0.80171536],
[-1.2107308 , -0.34924109],
[ 1.33423684,  0.7721489 ],
[ 1.19480152,  1.04788556],
[ 0.9917477 ,  0.89202008],
[-1.8356219 , -0.04839732],
[ 0.08415721, -0.71564326],
[-1.48970175, -0.19299604],
[ 0.38782418, -0.82060119],
[-0.01448044, -0.9779841 ],
[-2.0521341 , -0.02129125],
[ 0.10331194, -0.82162781],
[-0.44189315, -0.65710974],
[ 1.10390926,  1.02481182],
[-1.59227759, -0.17374038],
[-1.47344152, -0.02202853],
[-1.35514704,  0.22971067],
[ 0.0412337 , -1.23776622],
[ 0.4761517 , -1.13672124],
[ 1.04335676,  0.82345905],
[-0.07961882, -0.85677394],
[ 0.87065059,  1.08052841],
[ 1.40267313,  1.07525119],
[ 0.80111157,  1.28342825],
[-0.16527516, -1.23583804],
[-0.33779221, -0.59194323],
[ 0.80610749, -0.73752159],
[-1.43590032, -0.56384446],
[ 0.54868895, -0.95143829],
[ 0.46803131, -0.74973907],
```

```
[-1.5137129 , -0.83914323],
[ 0.9138436 ,  1.51126532],
[-1.97233903, -0.41155375],
[ 0.5213406 , -0.88654894],
[ 0.62759494, -1.18590477],
[ 0.94163014,  1.35399335],
[ 0.56994768,  1.07036606],
[-1.87663382,  0.14745773],
[ 0.90612186,  0.91084011],
[-1.37481454,  0.28428395],
[-1.80564029, -0.96710574],
[ 0.34307757, -0.79999275],
[ 0.70380566,  1.00025804],
[-1.68489862, -0.30564595],
[ 1.31473221,  0.98614978],
[ 0.26151216, -0.26069251],
[ 0.9193121 ,  0.82371485],
[-1.21795929, -0.20219674],
[-0.17722723, -1.02665245],
[ 0.64824862, -0.66822881],
[ 0.41206786, -0.28783784],
[ 1.01568202,  1.13481667],
[ 0.67900254, -0.91489502],
[-1.05182747, -0.01062376],
[ 0.61306599,  1.78210384],
[-1.50219748, -0.52308922],
[-1.72717293, -0.46173916],
[-1.60995631, -0.1821007 ],
[-1.09111021, -0.0781398 ],
[-0.01046978, -0.80913034],
[ 0.32782303, -0.80734754],
[ 1.22038503,  1.1959793 ],
[-1.33328681, -0.30001937],
[ 0.87959517,  1.11566491],
[-1.14829098, -0.30400762],
[-0.58019755, -1.19996018],
[-0.01161159, -0.78468854],
[ 0.17359724, -0.63398145],
[ 1.32738556,  0.67759969],
[-1.93467327,  0.30572472],
[-1.57761893, -0.27726365],
[ 0.47639   ,  1.21422648],
[-1.65237509, -0.6803981 ],
[-0.12609976, -1.04327457],
[-1.89607082, -0.70085502],
[ 0.57466899,  0.74878369],
[-0.16660312, -0.83110295],
[ 0.8013355 ,  1.22244435],
```

```
[ 1.18455426,  1.4346467 ],
[ 1.08864428,  0.64667112],
[-1.61158505,  0.22805725],
[-1.57512205, -0.09612576],
[ 0.0721357 , -0.69640328],
[-1.40054298,  0.16390598],
[ 1.09607713,  1.16804691],
[-2.54346204, -0.23089822],
[-1.34544875,  0.25151126],
[-1.35478629, -0.19103317],
[ 0.18368113, -1.15827725],
[-1.31368677, -0.376357  ],
[ 0.09990129,  1.22500491],
[ 1.17225574,  1.30835143],
[ 0.0865397 , -0.79714371],
[-0.21053923, -1.13421511],
[ 0.26496024, -0.94760742],
[-0.2557591 , -1.06266022],
[-0.26039757, -0.74774225],
[-1.91787359,  0.16434571],
[ 0.93021139,  0.49436331],
[ 0.44770467, -0.72877918],
[-1.63802869, -0.58925528],
[-1.95712763, -0.10125137],
[ 0.9270337 ,  0.88251423],
[ 1.25660093,  0.60828073],
[-1.72818632,  0.08416887],
[ 0.3499788 , -0.30490298],
[-1.51696082, -0.50913109],
[ 0.18763605, -0.55424924],
[ 0.89609809,  0.83551508],
[-1.54968857, -0.17114782],
[ 1.2157457 ,  1.23317728],
[ 0.20307745, -1.03784906],
[ 0.84589086,  1.03615273],
[ 0.53237919,  1.47362884],
[-0.05319044, -1.36150553],
[ 1.38819743,  1.11729915],
[ 1.00696304,  1.0367721 ],
[ 0.56681869, -1.09637176],
[ 0.86888296,  1.05248874],
[-1.16286609, -0.55875245],
[ 0.27717768, -0.83844015],
[ 0.16563267, -0.80306607],
[ 0.38263303, -0.42683241],
[ 1.14519807,  0.89659026],
[ 0.81455857,  0.67533667],
[-1.8603152 , -0.09537561],
```

```
[ 0.965641  ,  0.90295579],
[-1.49897451, -0.33254044],
[-0.1335489 , -0.80727582],
[ 0.12541527, -1.13354906],
[ 1.06062436,  1.28816358],
[-1.49154578, -0.2024641 ],
[ 1.16189032,  1.28819877],
[ 0.54282033,  0.75203524],
[ 0.89221065,  0.99211624],
[-1.49932011, -0.32430667],
[ 0.3166647 , -1.34482915],
[ 0.13972469, -1.22097448],
[-1.5499724 , -0.10782584],
[ 1.23846858,  1.37668804],
[ 1.25558954,  0.72026098],
[ 0.25558689, -1.28529763],
[ 0.45168933, -0.55952093],
[ 1.06202057,  1.03404604],
[ 0.67451908, -0.54970299],
[ 0.22759676, -1.02729468],
[-1.45835281, -0.04951074],
[ 0.23273501, -0.70849262],
[ 1.59679589,  1.11395076],
[ 0.80476105,  0.544627  ],
[ 1.15492521,  1.04352191],
[ 0.59632776, -1.19142897],
[ 0.02839068, -0.43829366],
[ 1.13451584,  0.5632633 ],
[ 0.21576204, -1.04445753],
[ 1.41048987,  1.02830719],
[ 1.12289302,  0.58029441],
[ 0.25200688, -0.82588436],
[-1.28566081, -0.07390909],
[ 1.52849815,  1.11822469],
[-0.23907858, -0.70541972],
[-0.25792784, -0.81825035],
[ 0.59367818, -0.45239915],
[ 0.07931909, -0.29233213],
[-1.27256815,  0.11630577],
[ 0.66930129,  1.00731481],
[ 0.34791546, -1.20822877],
[-2.11283993, -0.66897935],
[-1.6293824 , -0.32718222],
[-1.53819139, -0.01501972],
[-0.11988545, -0.6036339 ],
[-1.54418956, -0.30389844],
[ 0.30026614, -0.77723173],
[ 0.00935449, -0.53888192],
```

```
[-1.33424393, -0.11560431],
[ 0.47504489,  0.78421384],
[ 0.59313264,  1.232239  ],
[ 0.41370369, -1.35205857],
[ 0.55840948,  0.78831053],
[ 0.49855018, -0.789949  ],
[ 0.35675809, -0.81038693],
[-1.86197825, -0.59071305],
[-1.61977671, -0.16076687],
[ 0.80779295, -0.73311294],
[ 1.62745775,  0.62787163],
[-1.56993593, -0.08467567],
[ 1.02558561,  0.89383302],
[ 0.24293461, -0.6088253 ],
[ 1.23130242,  1.00262186],
[-1.9651013 , -0.15886289],
[ 0.42795032, -0.70384432],
[-1.58306818, -0.19431923],
[-1.57195922,  0.01413469],
[-0.98145373,  0.06132285],
[-1.48637844, -0.5746531 ],
[ 0.98745828,  0.69188053],
[ 1.28619721,  1.28128821],
[ 0.85850596,  0.95541481],
[ 0.19028286, -0.82112942],
[ 0.26561046, -0.04255239],
[-1.61897897,  0.00862372],
[ 0.24070183, -0.52664209],
[ 1.15220993,  0.43916694],
[-1.21967812, -0.2580313 ],
[ 0.33412533, -0.86117761],
[ 0.17131003, -0.75638965],
[-1.19828397, -0.73744665],
[-0.12245932, -0.45648879],
[ 1.51200698,  0.88825741],
[ 1.10338866,  0.92347479],
[ 1.30972095,  0.59066989],
[ 0.19964876,  1.14855889],
[ 0.81460515,  0.84538972],
[-1.6422739 , -0.42296206],
[ 0.01224351, -0.21247816],
[ 0.33709102, -0.74618065],
[ 0.47301054,  0.72712075],
[ 0.34706626,  1.23033757],
[-0.00393279, -0.97209694],
[-1.64303119,  0.05276337],
[ 1.44649625,  1.14217033],
[-1.93030087, -0.40026146],
```

```
[-2.37296135, -0.72633645],
[ 0.45860122, -1.06048953],
[ 0.4896361 , -1.18928313],
[-1.02335902, -0.17520578],
[-1.32761107, -0.93963549],
[-1.50987909, -0.09473658],
[ 0.02723057, -0.79870549],
[ 1.0169412 ,  1.26461701],
[ 0.47733527, -0.9898471 ],
[-1.27784224, -0.547416  ],
[ 0.49898802, -0.6237259 ],
[ 1.06004731,  0.86870008],
[ 1.00207501,  1.38293512],
[ 1.31161394,  0.62833956],
[ 1.13428443,  1.18346542],
[ 1.27671346,  0.96632878],
[-0.63342885, -0.97768251],
[ 0.12698779, -0.93142317],
[-1.34510812, -0.23754226],
[-0.53162278, -1.25153594],
[ 0.21959934, -0.90269938],
[-1.78997479, -0.12115748],
[ 1.23197473, -0.07453764],
[ 1.4163536 ,  1.21551752],
[-1.90280976, -0.1638976 ],
[-0.22440081, -0.75454248],
[ 0.59559412,  0.92414553],
[ 1.21930773,  1.08175284],
[-1.99427535, -0.37587799],
[-1.27818474, -0.52454551],
[ 0.62352689, -1.01430108],
[ 0.14024251, -0.428266  ],
[-0.16145713, -1.16359731],
[-1.74795865, -0.06033101],
[-1.16659791,  0.0902393 ],
[ 0.41110408, -0.8084249 ],
[ 1.14757168,  0.77804528],
[-1.65590748, -0.40105446],
[-1.15306865,  0.00858699],
[ 0.60892121,  0.68974833],
[-0.08434138, -0.97615256],
[ 0.19170053, -0.42331438],
[ 0.29663162, -1.13357399],
[-1.36893628, -0.25052124],
[-0.08037807, -0.56784155],
[ 0.35695011, -1.15064408],
[ 0.02482179, -0.63594828],
[-1.49075558, -0.2482507 ],
```

```
                [-1.408588  ,  0.25635431],
                [-1.98274626, -0.54584475]])

In [51]: new_points = np.array([[  4.00233332e-01,  -1.26544471e+00],
                [  8.03230370e-01,   1.28260167e+00],
                [ -1.39507552e+00,   5.57292921e-02],
                [ -3.41192677e-01,  -1.07661994e+00],
                [  1.54781747e+00,   1.40250049e+00],
                [  2.45032018e-01,  -4.83442328e-01],
                [  1.20706886e+00,   8.88752605e-01],
                [  1.25132628e+00,   1.15555395e+00],
                [  1.81004415e+00,   9.65530731e-01],
                [ -1.66963401e+00,  -3.08103509e-01],
                [ -7.17482105e-02,  -9.37939700e-01],
                [  6.82631927e-01,   1.10258160e+00],
                [  1.09039598e+00,   1.43899529e+00],
                [ -1.67645414e+00,  -5.04557049e-01],
                [ -1.84447804e+00,   4.52539544e-02],
                [  1.24234851e+00,   1.02088661e+00],
                [ -1.86147041e+00,   6.38645811e-03],
                [ -1.46044943e+00,   1.53252383e-01],
                [  4.98981817e-01,   8.98006058e-01],
                [  9.83962244e-01,   1.04369375e+00],
                [ -1.83136742e+00,  -1.63632835e-01],
                [  1.30622617e+00,   1.07658717e+00],
                [  3.53420328e-01,  -7.51320218e-01],
                [  1.13957970e+00,   1.54503860e+00],
                [  2.93995694e-01,  -1.26135005e+00],
                [ -1.14558225e+00,  -3.78709636e-02],
                [  1.18716105e+00,   6.00240663e-01],
                [ -2.23211946e+00,   2.30475094e-01],
                [ -1.28320430e+00,  -3.93314568e-01],
                [  4.94296696e-01,  -8.83972009e-01],
                [  6.31834930e-02,  -9.11952228e-01],
                [  9.35759539e-01,   8.66820685e-01],
                [  1.58014721e+00,   1.03788392e+00],
                [  1.06304960e+00,   1.02706082e+00],
                [ -1.39732536e+00,  -5.05162249e-01],
                [ -1.09935240e-01,  -9.08113619e-01],
                [  1.17346758e+00,   9.47501092e-01],
                [  9.20084511e-01,   1.45767672e+00],
                [  5.82658956e-01,  -9.00086832e-01],
                [  9.52772328e-01,   8.99042386e-01],
                [ -1.37266956e+00,  -3.17878215e-02],
                [  2.12706760e-02,  -7.07614194e-01],
                [  3.27049052e-01,  -5.55998107e-01],
                [ -1.71590267e+00,   2.15222266e-01],
                [  5.12516209e-01,  -7.60128245e-01],
```

```
[  1.13023469e+00,   7.22451122e-01],
[ -1.43074310e+00,  -3.42787511e-01],
[ -1.82724625e+00,   1.17657775e-01],
[  1.41801350e+00,   1.11455080e+00],
[  1.26897304e+00,   1.41925971e+00],
[  8.04076494e-01,   1.63988557e+00],
[  8.34567752e-01,   1.09956689e+00],
[ -1.24714732e+00,  -2.23522320e-01],
[ -1.29422537e+00,   8.18770024e-02],
[ -2.27378316e-01,  -4.13331387e-01],
[  2.18830387e-01,  -4.68183120e-01],
[ -1.22593414e+00,   2.55599147e-01],
[ -1.31294033e+00,  -4.28892070e-01],
[ -1.33532382e+00,   6.52053776e-01],
[ -3.01100233e-01,  -1.25156451e+00],
[  2.02778356e-01,  -9.05277445e-01],
[  1.01357784e+00,   1.12378981e+00],
[  8.18324394e-01,   8.60841257e-01],
[  1.26181556e+00,   1.46613744e+00],
[  4.64867724e-01,  -7.97212459e-01],
[  3.60908898e-01,   8.44106720e-01],
[ -2.15098310e+00,  -3.69583937e-01],
[  1.05005281e+00,   8.74181364e-01],
[  1.06580074e-01,  -7.49268153e-01],
[ -1.73945723e+00,   2.52183577e-01],
[ -1.12017687e-01,  -6.52469788e-01],
[  5.16618951e-01,  -6.41267582e-01],
[  3.26621787e-01,  -8.80608015e-01],
[  1.09017759e+00,   1.10952558e+00],
[  3.64459576e-01,  -6.94215622e-01],
[ -1.90779318e+00,   1.87383674e-01],
[ -1.95601829e+00,   1.39959126e-01],
[  3.18541701e-01,  -4.05271704e-01],
[  7.36512699e-01,   1.76416255e+00],
[ -1.44175162e+00,  -5.72320429e-02],
[  3.21757168e-01,  -5.34283821e-01],
[ -1.37317305e+00,   4.64484644e-02],
[  6.87225910e-02,  -1.10522944e+00],
[  9.59314218e-01,   6.52316210e-01],
[ -1.62641919e+00,  -5.62423280e-01],
[  1.06788305e+00,   7.29260482e-01],
[ -1.79643547e+00,  -9.88307418e-01],
[ -9.88628377e-02,  -6.81198092e-02],
[ -1.05135700e-01,   1.17022143e+00],
[  8.79964699e-01,   1.25340317e+00],
[  9.80753407e-01,   1.15486539e+00],
[ -8.33224966e-02,  -9.24844368e-01],
[  8.48759673e-01,   1.09397425e+00],
```

```
[  1.32941649e+00,   1.13734563e+00],
[  3.23788068e-01,  -7.49732451e-01],
[ -1.52610970e+00,  -2.49016929e-01],
[ -1.48598116e+00,  -2.68828608e-01],
[ -1.80479553e+00,   1.87052700e-01],
[ -2.01907347e+00,  -4.49511651e-01],
[  2.87202402e-01,  -6.55487415e-01],
[  8.22295102e-01,   1.38443234e+00],
[ -3.56997036e-02,  -8.01825807e-01],
[ -1.66955440e+00,  -1.38258505e-01],
[ -1.78226821e+00,   2.93353033e-01],
[  7.25837138e-01,  -6.23374024e-01],
[  3.88432593e-01,  -7.61283497e-01],
[  1.49002783e+00,   7.95678671e-01],
[  6.55423228e-04,  -7.40580702e-01],
[ -1.34533116e+00,  -4.75629937e-01],
[ -8.03845106e-01,  -3.09943013e-01],
[ -2.49041295e-01,  -1.00662418e+00],
[ -1.41095118e+00,  -7.06744127e-02],
[ -1.75119594e+00,  -3.00491336e-01],
[ -1.27942724e+00,   1.73774600e-01],
[  3.35028183e-01,   6.24761151e-01],
[  1.16819649e+00,   1.18902251e+00],
[  7.15210457e-01,   9.26077419e-01],
[  1.30057278e+00,   9.16349565e-01],
[ -1.21697008e+00,   1.10039477e-01],
[ -1.70707935e+00,  -5.99659536e-02],
[  1.20730655e+00,   1.05480463e+00],
[  1.86896009e-01,  -9.58047234e-01],
[  8.03463471e-01,   3.86133140e-01],
[ -1.73486790e+00,  -1.49831913e-01],
[  1.31261499e+00,   1.11802982e+00],
[  4.04993148e-01,  -5.10900347e-01],
[ -1.93267968e+00,   2.20764694e-01],
[  6.56004799e-01,   9.61887161e-01],
[ -1.40588215e+00,   1.17134403e-01],
[ -1.74306264e+00,  -7.47473959e-02],
[  5.43745412e-01,   1.47209224e+00],
[ -1.97331669e+00,  -2.27124493e-01],
[  1.53901171e+00,   1.36049081e+00],
[ -1.48323452e+00,  -4.90302063e-01],
[  3.86748484e-01,  -1.26173400e+00],
[  1.17015716e+00,   1.18549415e+00],
[ -8.05381721e-02,  -3.21923627e-01],
[ -6.82273156e-02,  -8.52825887e-01],
[  7.13500028e-01,   1.27868520e+00],
[ -1.85014378e+00,  -5.03490558e-01],
[  6.36085266e-02,  -1.41257040e+00],
```

```
[  1.52966062e+00,   9.66056572e-01],
[  1.62165714e-01,  -1.37374843e+00],
[ -3.23474497e-01,  -7.06620269e-01],
[ -1.51768993e+00,   1.87658302e-01],
[  8.88895911e-01,   7.62237161e-01],
[  4.83164032e-01,   8.81931869e-01],
[ -5.52997766e-02,  -7.11305016e-01],
[ -1.57966441e+00,  -6.29220313e-01],
[  5.51308645e-02,  -8.47206763e-01],
[ -2.06001582e+00,   5.87697787e-02],
[  1.11810855e+00,   1.30254175e+00],
[  4.87016164e-01,  -9.90143937e-01],
[ -1.65518042e+00,  -1.69386383e-01],
[ -1.44349738e+00,   1.90299243e-01],
[ -1.70074547e-01,  -8.26736022e-01],
[ -1.82433979e+00,  -3.07814626e-01],
[  1.03093485e+00,   1.26457691e+00],
[  1.64431169e+00,   1.27773115e+00],
[ -1.47617693e+00,   2.60783872e-02],
[  1.00953067e+00,   1.14270181e+00],
[ -1.45285636e+00,  -2.55216207e-01],
[ -1.74092917e+00,  -8.34443177e-02],
[  1.22038299e+00,   1.28699961e+00],
[  9.16925397e-01,   7.32070275e-01],
[ -1.60754185e-03,  -7.26375571e-01],
[  8.93841238e-01,   8.41146643e-01],
[  6.33791961e-01,   1.00915134e+00],
[ -1.47927075e+00,  -6.99781936e-01],
[  5.44799374e-02,  -1.06441970e+00],
[ -1.51935568e+00,  -4.89276929e-01],
[  2.89939026e-01,  -7.73145523e-01],
[ -9.68154061e-03,  -1.13302207e+00],
[  1.13474639e+00,   9.71541744e-01],
[  5.36421406e-01,  -8.47906388e-01],
[  1.14759864e+00,   6.89915205e-01],
[  5.73291902e-01,   7.90802710e-01],
[  2.12377397e-01,  -6.07569808e-01],
[  5.26579548e-01,  -8.15930264e-01],
[ -2.01831641e+00,   6.78650740e-02],
[ -2.35512624e-01,  -1.08205132e+00],
[  1.59274780e-01,  -6.00717261e-01],
[  2.28120356e-01,  -1.16003549e+00],
[ -1.53658378e+00,   8.40798808e-02],
[  1.13954609e+00,   6.31782001e-01],
[  1.01119255e+00,   1.04360805e+00],
[ -1.42039867e-01,  -4.81230337e-01],
[ -2.23120182e+00,   8.49162905e-02],
[  1.25554811e-01,  -1.01794793e+00],
```

```
[ -1.72493509e+00,  -6.94426177e-01],
[ -1.60434630e+00,   4.45550868e-01],
[  7.37153979e-01,   9.26560744e-01],
[  6.72905271e-01,   1.13366030e+00],
[  1.20066456e+00,   7.26273093e-01],
[  7.58747209e-02,  -9.83378326e-01],
[  1.28783262e+00,   1.18088601e+00],
[  1.06521930e+00,   1.00714746e+00],
[  1.05871698e+00,   1.12956519e+00],
[ -1.12643410e+00,   1.66787744e-01],
[ -1.10157218e+00,  -3.64137806e-01],
[  2.35118217e-01,  -1.39769949e-01],
[  1.13853795e+00,   1.01018519e+00],
[  5.31205654e-01,  -8.81990792e-01],
[  4.33085936e-01,  -7.64059042e-01],
[ -4.48926156e-03,  -1.30548411e+00],
[ -1.76348589e+00,  -4.97430739e-01],
[  1.36485681e+00,   5.83404699e-01],
[  5.66923900e-01,   1.51391963e+00],
[  1.35736826e+00,   6.70915318e-01],
[  1.07173397e+00,   6.11990884e-01],
[  1.00106915e+00,   8.93815326e-01],
[  1.33091007e+00,   8.79773879e-01],
[ -1.79603740e+00,  -3.53883973e-02],
[ -1.27222979e+00,   4.00156642e-01],
[  8.47480603e-01,   1.17032364e+00],
[ -1.50989129e+00,  -7.12318330e-01],
[ -1.24953576e+00,  -5.57859730e-01],
[ -1.27717973e+00,  -5.99350550e-01],
[ -1.81946743e+00,   7.37057673e-01],
[  1.19949867e+00,   1.56969386e+00],
[ -1.25543847e+00,  -2.33892826e-01],
[ -1.63052058e+00,   1.61455865e-01],
[  1.10611305e+00,   7.39698224e-01],
[  6.70193192e-01,   8.70567001e-01],
[  3.69670156e-01,  -6.94645306e-01],
[ -1.26362293e+00,  -6.99249285e-01],
[ -3.66687507e-01,  -1.35310260e+00],
[  2.44032147e-01,  -6.59470793e-01],
[ -1.27679142e+00,  -4.85453412e-01],
[  3.77473612e-02,  -6.99251605e-01],
[ -2.19148539e+00,  -4.91199500e-01],
[ -2.93277777e-01,  -5.89488212e-01],
[ -1.65737397e+00,  -2.98337786e-01],
[  7.36638861e-01,   5.78037057e-01],
[  1.13709081e+00,   1.30119754e+00],
[ -1.44146601e+00,   3.13934680e-02],
[  5.92360708e-01,   1.22545114e+00],
```

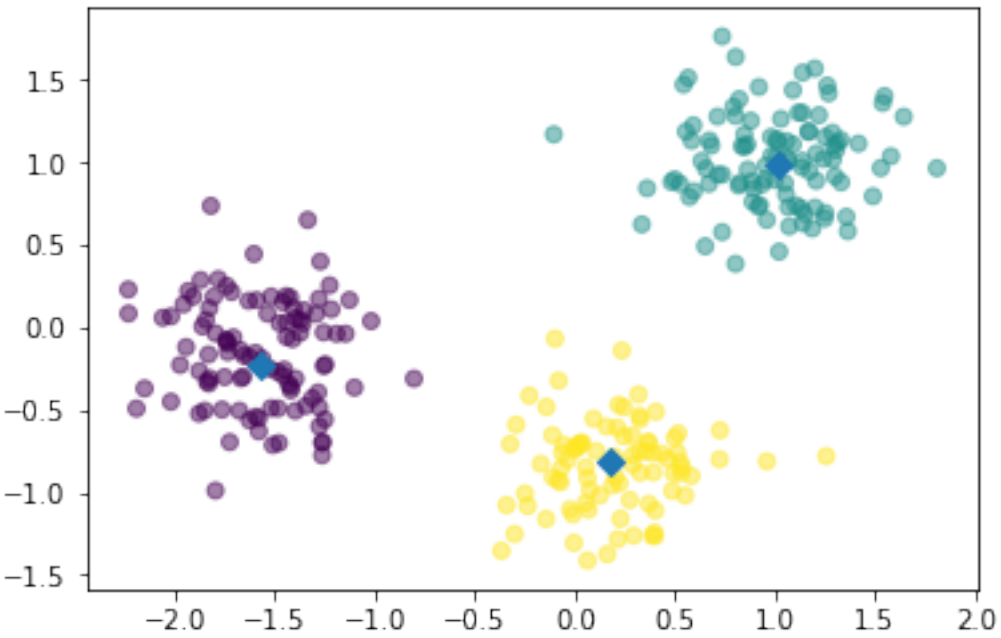
```
[  6.51719414e-01,   4.92674894e-01],
[  5.94559139e-01,   8.25637315e-01],
[ -1.87900722e+00,  -5.21899626e-01],
[  2.15225041e-01,  -1.28269851e+00],
[  4.99145965e-01,  -6.70268634e-01],
[ -1.82954176e+00,  -3.39269731e-01],
[  7.92721403e-01,   1.33785606e+00],
[  9.54363372e-01,   9.80396626e-01],
[ -1.35359846e+00,   1.03976340e-01],
[  1.05595062e+00,   8.07031927e-01],
[ -1.94311010e+00,  -1.18976964e-01],
[ -1.39604137e+00,  -3.10095976e-01],
[  1.28977624e+00,   1.01753365e+00],
[ -1.59503139e+00,  -5.40574609e-01],
[ -1.41994046e+00,  -3.81032569e-01],
[ -2.35569801e-02,  -1.10133702e+00],
[ -1.26038568e+00,  -6.93273886e-01],
[  9.60215981e-01,  -8.11553694e-01],
[  5.51803308e-01,  -1.01793176e+00],
[  3.70185085e-01,  -1.06885468e+00],
[  8.25529207e-01,   8.77007060e-01],
[ -1.87032595e+00,   2.87507199e-01],
[ -1.56260769e+00,  -1.89196712e-01],
[ -1.26346548e+00,  -7.74725237e-01],
[ -6.33800421e-02,  -7.59400611e-01],
[  8.85298280e-01,   8.85620519e-01],
[ -1.43324686e-01,  -1.16083678e+00],
[ -1.83908725e+00,  -3.26655515e-01],
[  2.74709229e-01,  -1.04546829e+00],
[ -1.45703573e+00,  -2.91842036e-01],
[ -1.59048842e+00,   1.66063031e-01],
[  9.25549284e-01,   7.41406406e-01],
[  1.97245469e-01,  -7.80703225e-01],
[  2.88401697e-01,  -8.32425551e-01],
[  7.24141618e-01,  -7.99149200e-01],
[ -1.62658639e+00,  -1.80005543e-01],
[  5.84481588e-01,   1.13195640e+00],
[  1.02146732e+00,   4.59657799e-01],
[  8.65050554e-01,   9.57714887e-01],
[  3.98717766e-01,  -1.24273147e+00],
[  8.62234892e-01,   1.10955561e+00],
[ -1.35999430e+00,   2.49942654e-02],
[ -1.19178505e+00,  -3.82946323e-02],
[  1.29392424e+00,   1.10320509e+00],
[  1.25679630e+00,  -7.79857582e-01],
[  9.38040302e-02,  -5.53247258e-01],
[ -1.73512175e+00,  -9.76271667e-02],
[  2.23153587e-01,  -9.43474351e-01],
```

```
       [  4.01989100e-01,  -1.10963051e+00],
       [ -1.42244158e+00,   1.81914703e-01],
       [  3.92476267e-01,  -8.78426277e-01],
       [  1.25181875e+00,   6.93614996e-01],
       [  1.77481317e-02,  -7.20304235e-01],
       [ -1.87752521e+00,  -2.63870424e-01],
       [ -1.58063602e+00,  -5.50456344e-01],
       [ -1.59589493e+00,  -1.53932892e-01],
       [ -1.01829770e+00,   3.88542370e-02],
       [  1.24819659e+00,   6.60041803e-01],
       [ -1.25551377e+00,  -2.96172009e-02],
       [ -1.41864559e+00,  -3.58230179e-01],
       [  5.25758326e-01,   8.70500543e-01],
       [  5.55599988e-01,   1.18765072e+00],
       [  2.81344439e-02,  -6.99111314e-01]])
```

In [54]:
```
# Import KMeans
from sklearn.cluster import KMeans

# Create a KMeans instance with 3 clusters: model
model = KMeans(n_clusters=3)

# Fit model to points
model.fit(points)

# Determine the cluster labels of new_points: labels
#After the model has been fit, you'll obtain the cluster labels for some new points usi

labels = model.predict(new_points)

# Print cluster labels of new_points
print(labels)

# performed k-Means clustering and predicted the labels of new points. But it is not ea
#A visualization would be far more useful. you'll inspect your clustering with a scatte



# Import pyplot
from matplotlib import pyplot as plt

# Assign the columns of new_points: xs and ys
xs = new_points[:,0]
ys = new_points[:,1]

# Make a scatter plot of xs and ys, using labels to define the colors
plt.scatter(xs, ys, c=labels, alpha=0.5)
```

```
        # Assign the cluster centers: centroids
        centroids = model.cluster_centers_

        # Assign the columns of centroids: centroids_x, centroids_y
        centroids_x = centroids[:,0]
        centroids_y = centroids[:,1]

        # Make a scatter plot of centroids_x and centroids_y
        plt.scatter(centroids_x, centroids_y, marker='D', s=50)
        plt.show()

        #The clustering looks better
```

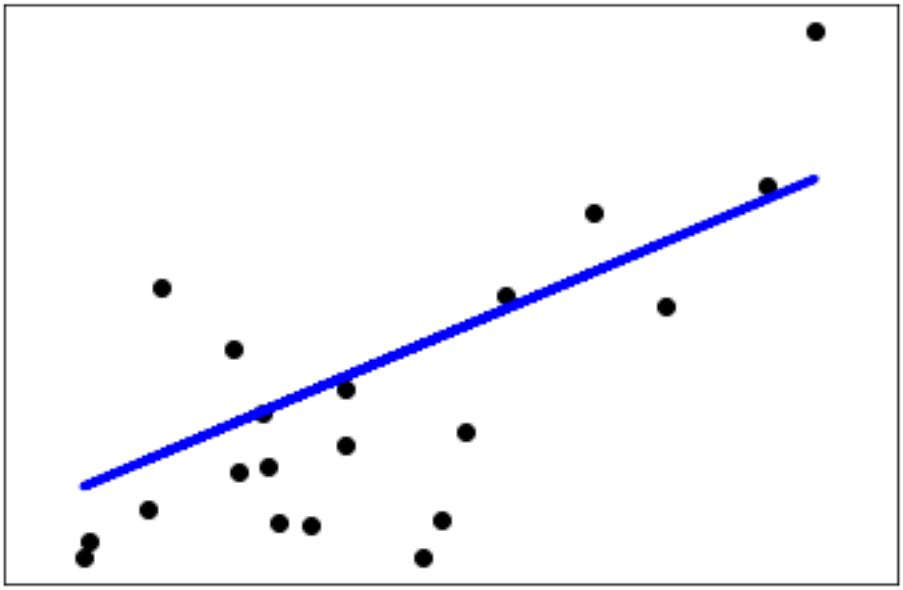
```
[2 1 0 2 1 2 1 1 1 0 2 1 1 0 0 1 0 0 1 1 0 1 2 1 2 0 1 0 0 2 2 1 1 1 0 2 1
 1 2 1 0 2 2 0 2 1 0 0 1 1 1 1 0 0 2 2 0 0 0 2 2 1 1 1 2 1 0 1 2 0 2 2 2 1
 2 0 0 2 1 0 2 0 2 1 0 1 0 2 1 1 1 2 1 1 2 0 0 0 0 2 1 2 0 0 2 2 1 2 0 0 2
 0 0 0 1 1 1 1 0 0 1 2 1 0 1 2 0 1 0 0 1 0 1 0 2 1 2 2 1 0 2 1 2 2 0 1 1 2
 0 2 0 1 2 0 0 2 0 1 1 0 1 0 0 1 1 2 1 1 0 2 0 2 2 1 2 1 1 2 2 0 2 2 2 0 1
 1 2 0 2 0 0 1 1 1 2 1 1 1 0 0 2 1 2 2 2 0 1 1 1 1 1 1 0 0 1 0 0 0 0 1 0 0
 1 1 2 0 2 2 0 2 0 2 0 1 1 0 1 1 1 0 2 2 0 1 1 0 1 0 0 1 0 0 2 0 2 2 2 1 0
 0 0 2 1 2 0 2 0 0 1 2 2 2 0 1 1 1 2 1 0 0 1 2 2 0 2 2 0 2 1 2 0 0 0 0 1 0
 0 1 1 2]
```




```
In [55]: import matplotlib.pyplot as plt
         import numpy as np
```

```
        from sklearn import datasets, linear_model
        from sklearn.metrics import mean_squared_error, r2_score

        # Load the diabetes dataset
        diabetes = datasets.load_diabetes()

In [56]: # Use only one feature
        diabetes_X = diabetes.data[:, np.newaxis, 2]

        # Split the data into training/testing sets
        diabetes_X_train = diabetes_X[:-20]
        diabetes_X_test = diabetes_X[-20:]

        # Split the targets into training/testing sets
        diabetes_y_train = diabetes.target[:-20]
        diabetes_y_test = diabetes.target[-20:]

        # Create linear regression object
        regr = linear_model.LinearRegression()

        # Train the model using the training sets
        regr.fit(diabetes_X_train, diabetes_y_train)

        # Make predictions using the testing set
        diabetes_y_pred = regr.predict(diabetes_X_test)

        # The coefficients
        print('Coefficients: \n', regr.coef_)
        # The mean squared error
        print("Mean squared error: %.2f"
              % mean_squared_error(diabetes_y_test, diabetes_y_pred))
        # Explained variance score: 1 is perfect prediction
        print('Variance score: %.2f' % r2_score(diabetes_y_test, diabetes_y_pred))

        # Plot outputs
        plt.scatter(diabetes_X_test, diabetes_y_test,  color='black')
        plt.plot(diabetes_X_test, diabetes_y_pred, color='blue', linewidth=3)

        plt.xticks(())
        plt.yticks(())

        plt.show()

        #The coefficients, the residual sum of squares and the variance score are also calculat

Coefficients:
 [938.23786125]
Mean squared error: 2548.07
```

```
Variance score: 0.47
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: