In [1]:
```
1 import pandas as pd
2 %pylab inline
```

```
Populating the interactive namespace from numpy and matplotlib
```

The wine recognition data

This (https://archive.ics.uci.edu/ml/datasets/wine (https://archive.ics.uci.edu/ml/datasets/wine)) is a one of the classic datasets.

Each row corresponds to a particular bottle of wine. Each wine is characterized by 13 numericcal attributes

The attributes

1. Class = type of wine, can be 1,2 or 3, In this notebook we ignore class.
2. Alcohol
3. Malic acid
4. Ash
5. Alcalinity of ash
6. Magnesium
7. Total phenols
8. Flavanoids
9. Nonflavanoid phenols
10. Proanthocyanins
11. Color intensity
12. Hue
13. OD280/OD315 of diluted wines
14. Proline

In [2]:
```
1 !ls data
```

```
Icon?      wine.data  wine.names
```

In [3]:
```
df=pd.read_csv('data/wine.data',header=None)

df.columns=['class','Alcohol','Malic acid','Ash','Alcalinity of ash','M
'Nonflavanoid phenols','Proanthocyanins','Color intensity','Hue','OD280
df.head()
```

Out[3]:

	class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nonflavanoid phenols	Proanthoc
0	1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	
1	1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	
2	1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	
3	1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	
4	1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	

The `df.describe()` method in pandas provides summary statistics for each column in the dataframe `df`

In [5]:
```
desc=df.describe()
desc
```

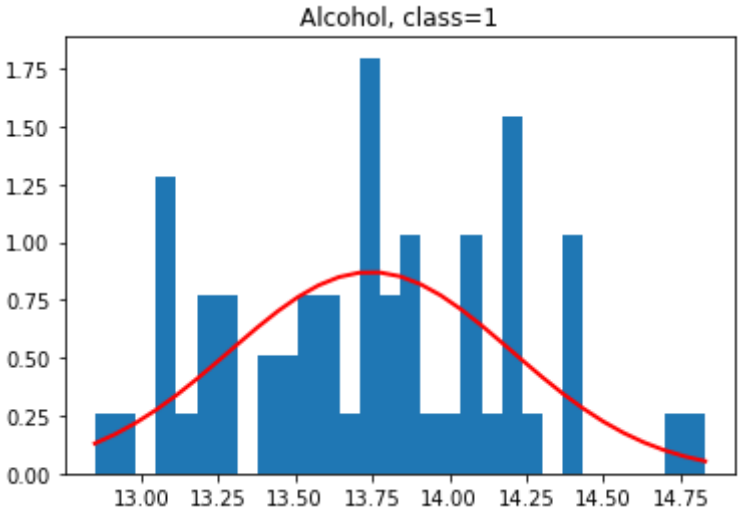
Out[5]:

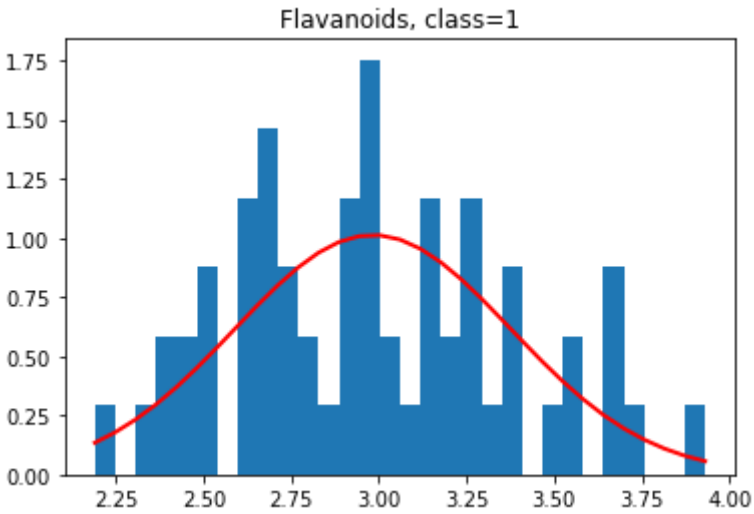
	class	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavar
count	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.000000	178.00
mean	1.938202	13.000618	2.336348	2.366517	19.494944	99.741573	2.295112	2.02
std	0.775035	0.811827	1.117146	0.274344	3.339564	14.282484	0.625851	0.99
min	1.000000	11.030000	0.740000	1.360000	10.600000	70.000000	0.980000	0.34
25%	1.000000	12.362500	1.602500	2.210000	17.200000	88.000000	1.742500	1.20
50%	2.000000	13.050000	1.865000	2.360000	19.500000	98.000000	2.355000	2.13
75%	3.000000	13.677500	3.082500	2.557500	21.500000	107.000000	2.800000	2.87
max	3.000000	14.830000	5.800000	3.230000	30.000000	162.000000	3.880000	5.08

In [6]:
```
def plot_stat(Col,_title='',_range=[],height=20):
    Col.hist(bins=20)
    _mean=Col.mean()
    _std=Col.std()
    for x in [_mean,_mean-_std,_mean+_std]:
        plot([x,x],[0,height])
        if len(_range)==2:
            xlim(_range)
    title(_title)
```

In [7]:
```
def plot_normal(s):
    count, bins, ignored = plt.hist(s, 30, density=True);
    sigma=std(s); mu=mean(s)
    plt.plot(bins, 1/(sigma * np.sqrt(2 * np.pi)) *
                   np.exp( - (bins - mu)**2 / (2 * sigma**2) ),
             linewidth=2, color='r');
```

In [8]:
```
j=1
_class=1
rdf=df[df['class']==_class]
for name in ['Alcohol','Flavanoids']:
    plot_normal(rdf[name])
    title('%s, class=%d'%(name,_class))
    figure()
```






```
<Figure size 432x288 with 0 Axes>
```

In [9]:
```
import matplotlib.pyplot as plt
import numpy as np
from   scipy.stats import multivariate_normal

```

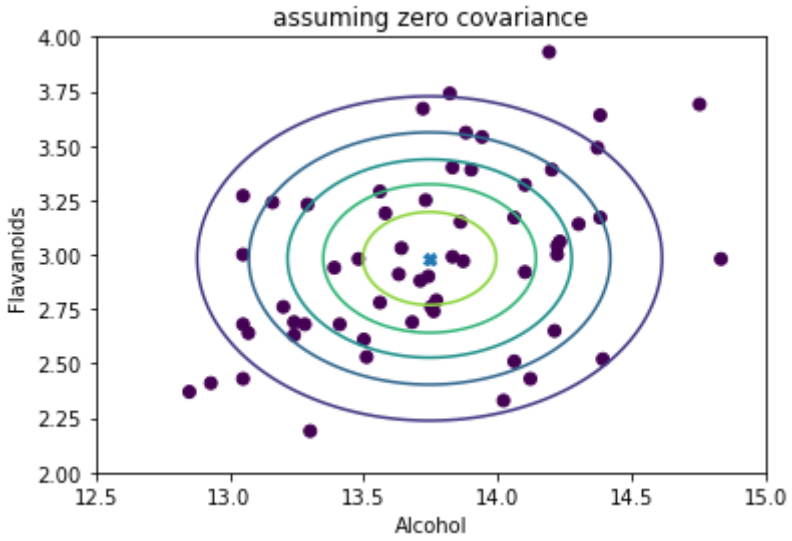
In [36]:

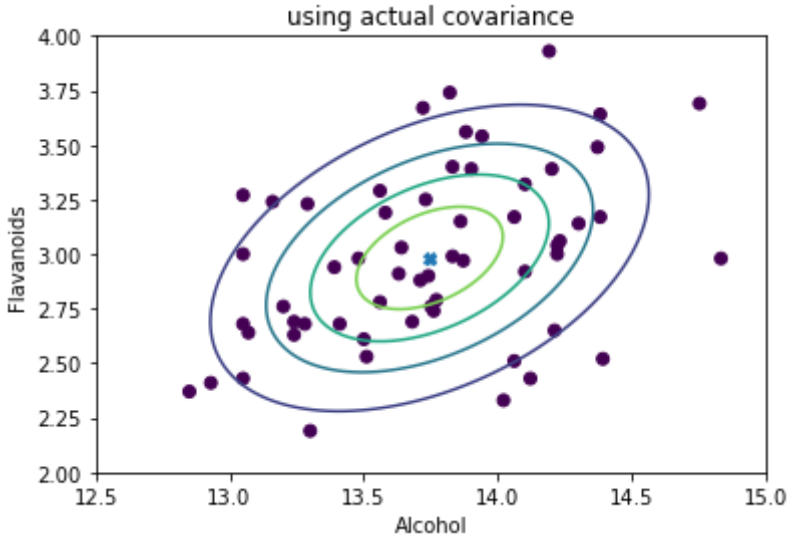
```
def scatterAndGauss(rdf,Feat1,Feat2,_cov,_mean):
    fig, ax = plt.subplots()
    scatter = ax.scatter(rdf[Feat1],rdf[Feat2],c=rdf['class'])
    xlabel(Feat1)
    ylabel(Feat2)

    plot(_mean[0],_mean[1],'X')
    N    = 200
    X    = np.linspace(12.5, 15, N)
    Y    = np.linspace(2, 4, N)
    X, Y = np.meshgrid(X, Y)
    pos  = np.dstack((X, Y))
    rv   = multivariate_normal(_mean, _cov)
    Z    = rv.pdf(pos)

    plt.contour(X, Y, Z,5)
#    plt.show()
```

In [37]:
```
Feat1='Alcohol'
Feat2='Flavanoids'
subset=rdf[[Feat1,Feat2]]
_cov=np.array(subset.cov())
_cov[1,0]=0
_cov[0,1]=0
_mean=np.array(subset.mean())
scatterAndGauss(rdf,Feat1,Feat2,_cov,_mean)
title('assuming zero covariance')
_cov=np.array(subset.cov())
scatterAndGauss(rdf,Feat1,Feat2,_cov,_mean)
title('using actual covariance');
```





In [35]:
```
1 rdf.iloc[:,1:].corr()
```

Out[35]:

	Alcohol	Malic acid	Ash	Alcalinity of ash	Magnesium	Total phenols	Flavanoids	Nor
Alcohol	1.000000	-0.040513	-0.148595	-0.318367	0.159361	0.420687	0.414904	
Malic acid	-0.040513	1.000000	0.026221	0.060031	0.079317	-0.083514	-0.191278	
Ash	-0.148595	0.026221	1.000000	0.549330	0.382549	0.004802	-0.070454	
Alcalinity of ash	-0.318367	0.060031	0.549330	1.000000	0.238337	-0.222999	-0.287103	
Magnesium	0.159361	0.079317	0.382549	0.238337	1.000000	0.307241	0.123332	
Total phenols	0.420687	-0.083514	0.004802	-0.222999	0.307241	1.000000	0.803784	
Flavanoids	0.414904	-0.191278	-0.070454	-0.287103	0.123332	0.803784	1.000000	
Nonflavanoid phenols	0.015659	-0.089366	0.465901	0.302262	0.237248	-0.016992	-0.089538	
Proanthocyanins	0.307571	-0.080798	-0.145471	-0.173628	-0.059042	0.373601	0.548624	
Color intensity	0.408291	-0.257587	-0.124220	-0.210951	0.184661	0.650086	0.741560	
Hue	0.080020	-0.419981	0.239214	0.092980	-0.111386	-0.224330	0.007863	
OD280/OD315	0.069818	0.173244	-0.081593	-0.117704	0.120651	0.053165	-0.088529	
Proline	0.360646	-0.372629	-0.029525	-0.122436	-0.147912	0.294994	0.380446	

HW ¶

- For each of the following three pairs of features compute
  - the mean and std of each feature,
  - the covariance and correlation coefficient of the pair.
  - Use `scatterAndGauss()` to generate scatter-figures and contours og the fitted Gaussian.

1. The pair of features (in `rdf`) is most correlated (excluding the correlation of a feature with itself which is always zero)
2. pair of features is most anti-correlated.
3. the pair of features that are least correlated(correlation coefficient closest to zero)

In [ ]:
```
1
```