4.6.4 Quadratic Discriminant Analysis

May 18, 2018

In [35]:
```
# conventional way to import pandas
import pandas as pd
# conventional way to import seaborn
import seaborn as sns
# conventional way to import numpy
import numpy as np

from sklearn import metrics
import matplotlib.pyplot as plt

data = pd.read_csv("https://raw.github.com/vincentarelbundock/Rdatasets/master/csv/ISL

data.head()
```

Out[35]:

	Year	Lag1	Lag2	Lag3	Lag4	Lag5	Volume	Today	Direction
1	2001	0.381	-0.192	-2.624	-1.055	5.010	1.1913	0.959	Up
2	2001	0.959	0.381	-0.192	-2.624	-1.055	1.2965	1.032	Up
3	2001	1.032	0.959	0.381	-0.192	-2.624	1.4112	-0.623	Down
4	2001	-0.623	1.032	0.959	0.381	-0.192	1.2760	0.614	Up
5	2001	0.614	-0.623	1.032	0.959	0.381	1.2057	0.213	Up

We will split the data into data before 2005 and after. Next we will make our tranining data.

In [36]:
```
import statsmodels.api as sm
from scipy import stats
from patsy import dmatrices

MarketAfter_2005 = data.query('Year >= 2005')
MarketBefore_2005 = data.query('Year < 2005')

y_train, X_train = dmatrices('Direction~Lag1+Lag2', MarketBefore_2005, return_type =

y_test, X_test = dmatrices('Direction~Lag1+Lag2', MarketAfter_2005, return_type = 'da
```

Now we will use the sklearn lib to do our Quadratic Discriminant Analysis (QDA).

In [37]:
```
import sklearn.discriminant_analysis
from sklearn.discriminant_analysis import QuadraticDiscriminantAnalysis
```

The training process. Please note we will only use Lag1 and Lag2. Hint iloc[:,1:3] and that means. Get first and the secound column of data frame.

```
In [38]: sklearn_qda = QuadraticDiscriminantAnalysis(priors=None,store_covariance=True) #creat

         qda = sklearn_qda.fit(X_train.iloc[:,1:3], y_train.iloc[:,1]) #learning the projectio

         X_labels = qda.predict(X_train.iloc[:,1:3]) #gives you the predicted label for each s

         X_prob = qda.predict_proba(X_train.iloc[:,1:3]) #the probability of each sample to be
```

Testing step. Now we will test out model using the data.

```
In [39]: X_test_labels = qda.predict(X_test.iloc[:,1:3])
         X_test_prob   = qda.predict_proba(X_test.iloc[:,1:3])

         np.mean(y_test.iloc[:,1]==X_test_labels)

Out[39]: 0.59920634920634919
```

Interestingly, the QDA predictions are accurate almost 60% of the time, even though the 2005 data was not used to fit the model. This level of accuracy is quite impressive for stock market data, which is known to be quite hard to model accurately. This suggests that the quadratic form assumed by QDA may capture the true relationship more accurately than the linear forms assumed by LDA and logistic regression.

The output contains the group means. But it does not contain the coefficients of the linear discriminants, because the QDA classifier involves a quadratic, rather than a linear, function of the predictors.

```
In [40]: qda.means_

Out[40]: array([[ 0.04279022,  0.03389409],
                [-0.03954635, -0.03132544]])
```

	Lag1	Lag2
Down	0.04279022	0.03389409
Up	-0.03954635	-0.03132544

and the covariances

```
In [41]: qda.covariance_

Out[41]: [array([[ 1.50662277, -0.03924806],
                 [-0.03924806,  1.53559498]]), array([[ 1.51700576, -0.02787349],
                 [-0.02787349,  1.49026815]])]
```