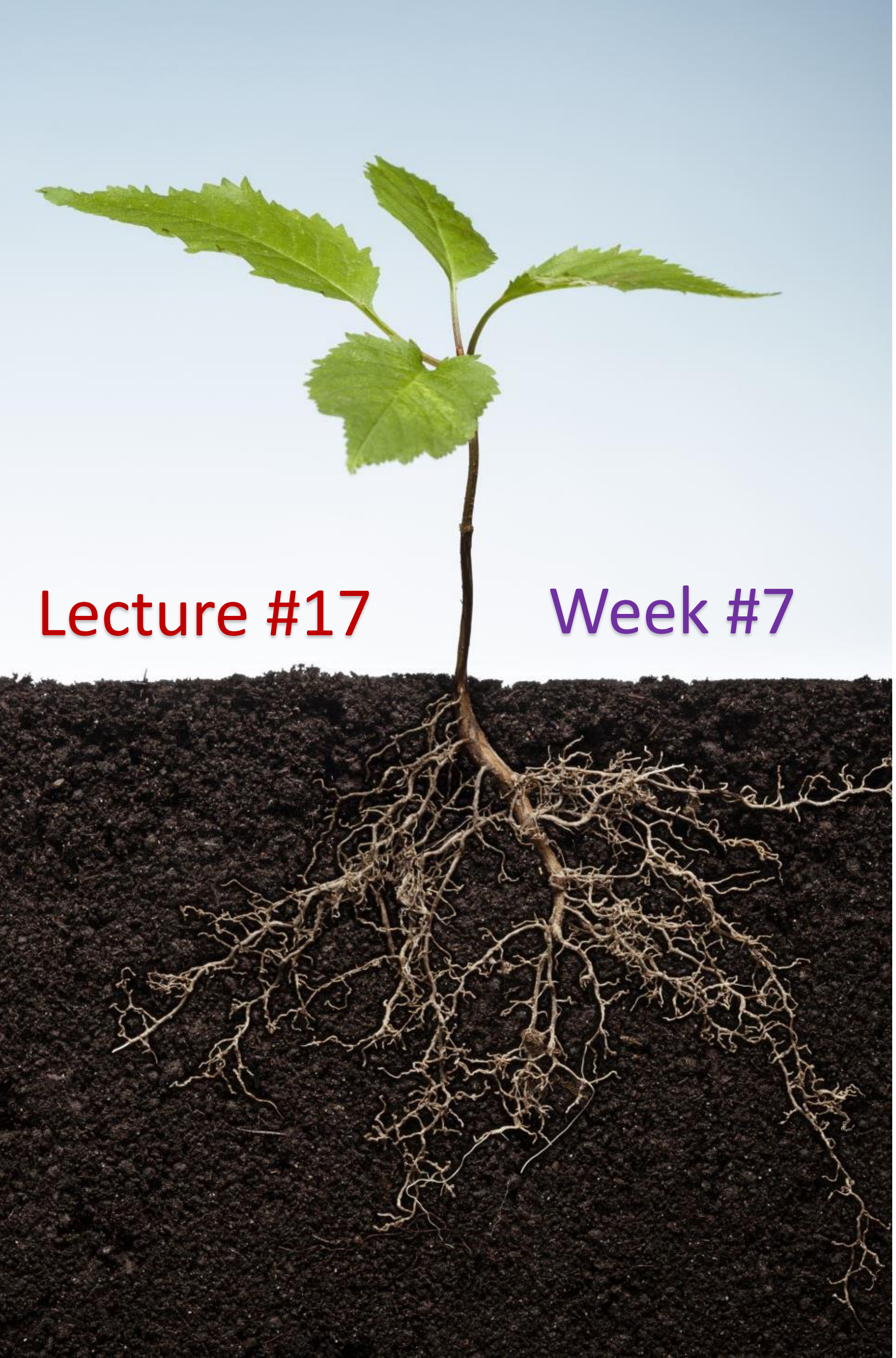


Pattern Recognition

ECSE 4410/6410 CAPA

Spring 2021

LDA +

Course Instructor - Thirimachos Bourlai

January to May 2021

Please Check the 2021 Syllabus

OVERVIEW

LDA +

- The limitations of logistic regression and the need for linear discriminant analysis
- The representation of the model that is learned from data and can be saved to file
- How the model is estimated from your data
- How to make predictions from a learned LDA model
- How to prepare your data to get the most from the LDA model
- The assumptions made by LDA about your training data
- How to calculate statistics required by the LDA model from your data
- How to make predictions using the learned LDA model

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

Linear Regression: Limitations

- LR → Simple and useful linear classification algorithm
- LR has limitations → thus, there is a need for alternate linear classification algorithms

LIMITATIONS

- **Better used on 2-Class Problems**:
  - It is intended for 2-class or binary classification problems
  - It can be extended for multiclass classification → downside: rarely used for this purpose
- **Unstable when working with <u>Well Separated Classes</u>**
  - Logistic regression can become unstable when the classes are well separated
- **Unstable with Few Examples**
  - With only a few examples from which to estimate the model parameters

Linear Regression: Limitations

LR - Unstable model →

- The LR model may converge on a set of coefficients but the outcome is "Poor Results" → poor performance
- The LR model may **not** reliably converge on a set of coefficients

LDA – potential solution:

- Address the above limitations
- It is the go-to linear method for multiclass classification problems
- Even with binary-classification problems, it is a good idea to try both LR and LDA

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

LDA Model Representation

The representation of LDA:

- It consists of statistical properties of our data
- These properties are calculated for each class
- For a **single input variable (x)** → this is the mean and the variance of the variable for each class
- For **multiple variables** → the same properties calculated over the multivariate Gaussian, i.e. the means and the covariance matrix Σ (multi-dimensional generalization of variance)

Process:

1. Compute statistical properties estimated from our data
2. Plug the statistical properties into the LDA equation
3. Make predictions

LDA Models - Learning

- LDA makes simplifying assumptions about our data:
  - That our data is Gaussian shaped
  - That **each variable** is shaped like a bell curve when plotted
  - That **each attribute** has the same variance
  - That **values of each variable** vary around the mean **by the same amount** on average.

With these assumptions, the **LDA model estimates** the **mean** and **variance** *from our data* for each class

LDA Models - Learning

- **For each input (x) and for each class (k)** →

The mean (mean) value can be estimated by dividing the sum of values by the total number of values

$$mean_k = \frac{1}{n_k} \times \sum_{i=1}^{n} x_i$$

- $mean_k$ = mean value of x for the class k
- n_k = number of instances with class k

The variance is calculated across all classes as the average squared difference of each value from the mean:

$$sigma^2 = \frac{1}{n-K} \times \sum_{i=1}^{n} (x_i - mean_k)^2$$

- $sigma^2$ = the variance across all inputs (x)
- n = number of instances
- K = number of classes
- $mean_k$ = mean of x for the class to which xi belongs

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

LDA: Making Predictions

- LDA **makes predictions** by estimating the probability that a new set of inputs belongs to each class
- **The class that gets the highest probability** is the output class and a prediction is made.

The Bayes Theorem can be used to estimate the probability of the **output class** (k) **given the input** (x) using:

1. The probability of each class and
2. The probability of the data belonging to each class:

$$P(Y = k|X = x) = \frac{P(k) \times P(x|k)}{\sum_{l=1}^{K} P(l) \times P(x|l)}$$

LDA: Making Predictions

$$P(Y = k|X = x) = \frac{P(k) \times P(x|k)}{\sum_{l=1}^{K} P(l) \times P(x|l)}$$

- $P(Y = k / X = x)$
  - ❖ Probability of the **class Y = k** given the input data x.
- $P(k)$
  - ❖ Base probability of a given class k we are considering (Y = k), e.g. the ratio of instances with this class in the training dataset.
- $P(x / k)$
  - ❖ Estimated probability of x, **given** that its is belonging the class k.
- The **denominator normalizes across for each class l**, e.g. the probability of the class P(l) and the probability of the input given the class $P(x / l)$

LDA: Making Predictions

A Gaussian distribution function can be used to estimate P(x / k).

Plugging the Gaussian into the above equation and simplifying we end up with:

$$D_k(x) = x \times \frac{mean_k}{sigma^2} - \frac{{mean_k}^2}{2 \times sigma^2} + ln(P(k))$$

It is no longer a probability as we discard some terms. Instead, it is called a **discriminate function for class k →**

- It is calculated for each class k and
- The class that has the **largest discriminant value** will make the output classification (Y = k):

Dk(x) is the discriminate function for class k given input x

LDA: Preparing Data

Suggestions to consider when preparing data for use with LDA

- **Classification Problems**
  - **LDA** is intended for classification problems where the **output variable** is **categorical**.
  - LDA supports both **binary** and **multiclass classification**
- **Gaussian Distribution**
  - The standard LDA model assumes input variables are Gaussian
  - Suggestion: **review** the univariate distributions of each attribute → use transforms to *make them more Gaussian-looking* (e.g. **log** and **root** for **EXP distributions** and **Box-Cox** for **skewed distributions**).

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

LDA: Preparing Data Cont.

Suggestions to consider when preparing data for use with LDA

- **Remove Outliers**
  - Outliers need to be removed → **can skew the basic statistics** (μ, σ) used to separate classes in LDA.
- **Same Variance**
  - LDA **assumes** that each input variable *has the same variance*.
  - **Standardize data (normalization)** before using LDA so that **mean=0** and **standard deviation=1**.

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

LDA-based Method Extensions

LDA → simple and effective method for classification → there are many extensions and variations to the method:

- **Quadratic Discriminant Analysis**: Each class uses its own estimate of variance
- **Flexible Discriminant Analysis**: nonlinear combination of inputs is used (e.g. splines)
- **Regularized Discriminant Analysis**: regularization is used into the estimate of the variance (or covariance) → this adjusts the influence of different LDA variables

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

LDA Step by Step (simplest case)

How to calculate an LDA model for simple dataset with 1 input and 1 output variable

We will show:

1. **Dataset**: what we are going to use for training and testing
2. **Learning the LDA Model**: learn all statistics needed to make predictions
3. **Making Predictions**: use the learned LDA model to make predictions for each instance in the training dataset

DATASET

- LDA data assumptions :
- **Input variables** → have a **Gaussian** distribution
- The **variance** - calculated for each input variable by class grouping - **is the same**
- **The mix of classes** in the *training set* is **representative of the problem**

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

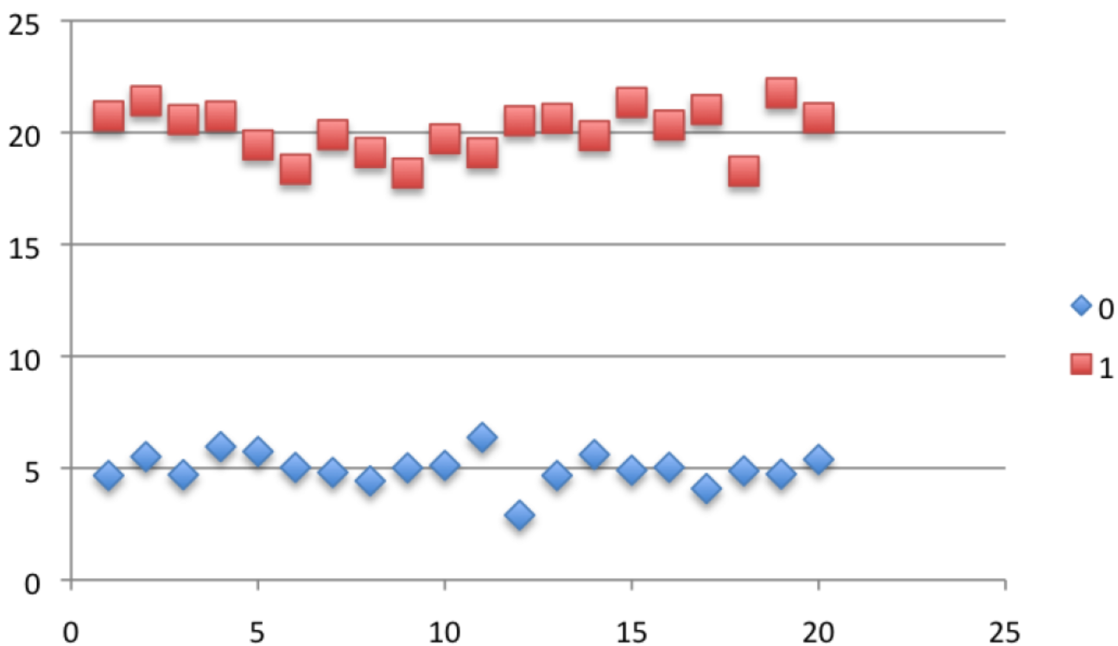
Dataset

Let's have a simple 2D dataset:

- **Variables**: X, Y; I/O
- **X values**: drawn from a Gaussian distribution
- **Y** = 0 or 1
- **Class 0**: All instances drawn from a Gaussian distribution with a μ=5, σ=1
- **Class 1**: same Gaussian, μ=20, σ=1
- The classes should be separable with a linear model like LDA

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

DATASET

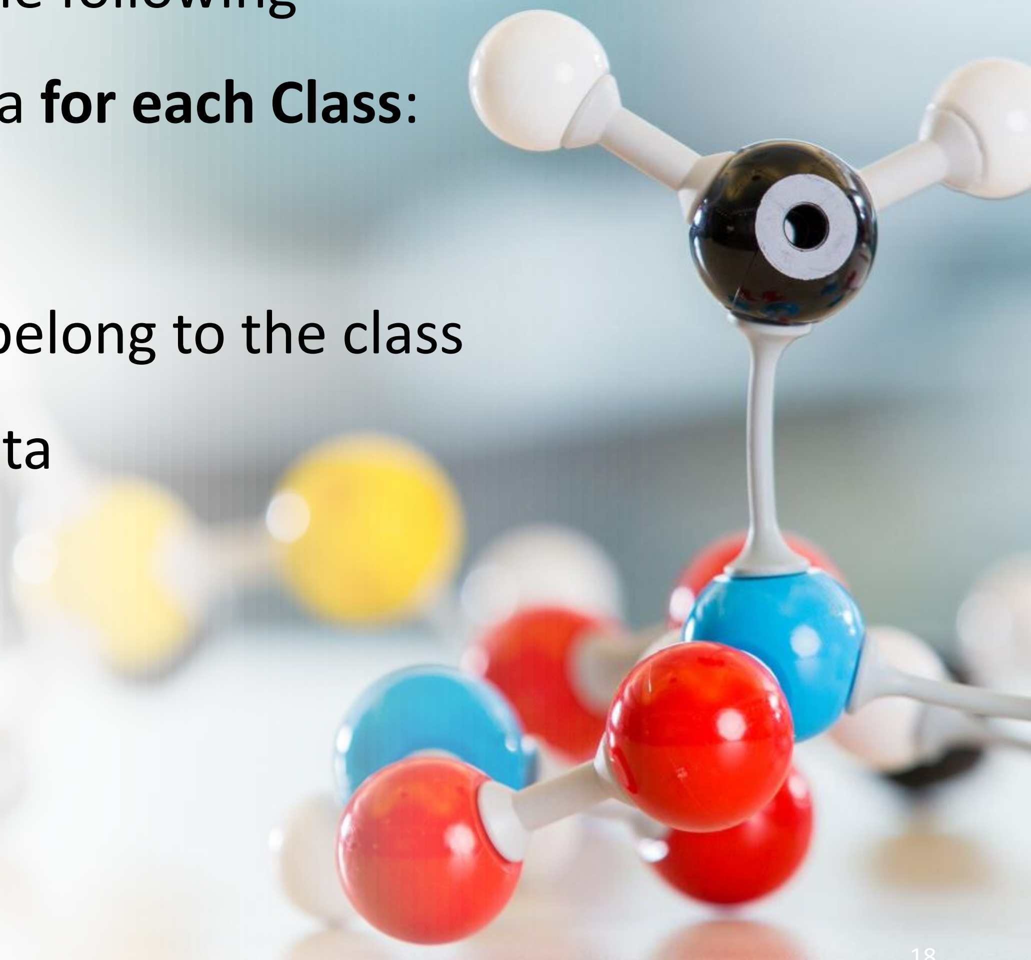


X	Y	20.74393514	1
4.667797637	0	21.41752855	1
5.509198779	0	20.57924186	1
4.702791608	0	20.7386947	1
5.956706641	0	19.44605384	1
5.738622413	0	18.36360265	1
5.027283325	0	19.90363232	1
4.805434058	0	19.10870851	1
4.425689143	0	18.18787593	1
5.009368635	0	19.71767611	1
5.116718815	0	19.09629027	1
6.370917709	0	20.52741312	1
2.895041947	0	20.63205608	1
4.666842365	0	19.86218119	1
5.602154638	0	21.34670569	1
4.902797978	0	20.333906	1
5.032652964	0	21.02714855	1
4.083972925	0	18.27536089	1
4.875524106	0	21.77371156	1
4.732801047	0	20.65953546	1
5.385993407	0		
x	Y		

Learning the LDA Model

It requires the estimation of the following statistics from the training data **for each Class**:

- **Mean per class**
- **Probability of an instance** belong to the class
- **Covariance** for the input data



Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

STEP 1 - MEAN

$$mean(x) = \frac{1}{n} \times \sum_{i=1}^{n} x_i$$

$$Y = 0:\ 4.975415507$$

$$Y = 1:\ 20.08706292$$

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

STEP 2 – Calculate Class Probabilities

$$P(y=0) = \frac{count(y=0)}{count(y=0) + count(y=1)}$$

$$P(y=1) = \frac{count(y=1)}{count(y=0) + count(y=1)}$$

$$P(y=0) = \frac{20}{20+20}$$

$$P(y=1) = \frac{20}{20+20}$$

- This is 0.5 for each class, as expected when creating the dataset.
- It is a good idea to work through each step of the model learning process.

STEP 3 - Variance

We need to calculate the variance for
the input variable for **each class**

Variance = the difference of each instance from the mean

We can calculate the variance for our dataset in two steps:

1. Calculate the **squared difference** for each input variable from the **group mean**
2. Calculate the **mean** of the **squared difference**

- Dataset is divided into 2 groups/classes as per the Y={0,1}
- Then, calculate the difference for each input value X from the mean from each class.

STEP 3 - Variance

We can calculate the difference of each input value from the mean using:

$$SquaredDifference = (x - mean_k)^2$$

X	Y	(x-mean_k)^2			
4.668	0	0.0946	20.744	1	0.4315
5.509	0	0.2849	21.418	1	1.7701
4.703	0	0.0743	20.579	1	0.2422
5.957	0	0.9629	20.739	1	0.4246
5.739	0	0.5825	19.446	1	0.4109
5.027	0	0.0027	18.364	1	2.9703
4.805	0	0.0289	19.904	1	0.0336
4.426	0	0.3022	19.109	1	0.9572
5.009	0	0.0012	18.188	1	3.6069
5.117	0	0.0200	19.718	1	0.1364
6.371	0	1.9474	19.096	1	0.9816
2.895	0	4.3280	20.527	1	0.1939
4.667	0	0.0952	20.632	1	0.2970
5.602	0	0.3928	19.862	1	0.0506
4.903	0	0.0053	21.347	1	1.5867
5.033	0	0.0033	20.334	1	0.0609
4.084	0	0.7947	21.027	1	0.8838
4.876	0	0.0100	18.275	1	3.2823
4.733	0	0.0589	21.774	1	2.8448
5.386	0	0.1686	20.660	1	0.3277

STEP 1	**MEAN**		
	Mean Y=0	**Mean Y=1**	
	4.975	20.087	
STEP 2	**VARIANCE**		
	P(Y=0)	**P(Y=1)**	
	0.5	0.5	
STEP 3	**Σ((x-mean_k=0)^2)**	**Σ((x-mean_k=1)^2)**	**k= Class 0 or 1**
	10.1582	21.4932	

STEP 3 - Variance

Next, we can calculate the variance as the average squared difference from the mean as:

$$variance = \frac{1}{count(x) - count(classes)} \times \sum_{i=1}^{n} SquaredDifference(x_i)$$

$$variance = \frac{1}{40 - 2} \times (10.15823013 + 21.49316708)$$

$$variance = 0.832931506$$

X	Y	(x-mean_k)^2
4.668	0	0.0946
5.509	0	0.2849
4.703	0	0.0743
5.957	0	0.9629
5.739	0	0.5825
5.027	0	0.0027
4.805	0	0.0289
4.426	0	0.3022
5.009	0	0.0012
5.117	0	0.0200
6.371	0	1.9474
2.895	0	4.3280
4.667	0	0.0952
5.602	0	0.3928
4.903	0	0.0053
5.033	0	0.0033
4.084	0	0.7947
4.876	0	0.0100
4.733	0	0.0589
5.386	0	0.1686
20.744	1	0.4315
21.418	1	1.7701
20.579	1	0.2422
20.739	1	0.4246
19.446	1	0.4109
18.364	1	2.9703
19.904	1	0.0336
19.109	1	0.9572
18.188	1	3.6069
19.718	1	0.1364
19.096	1	0.9816
20.527	1	0.1939
20.632	1	0.2970
19.862	1	0.0506
21.347	1	1.5867
20.334	1	0.0609
21.027	1	0.8838
18.275	1	3.2823
21.774	1	2.8448
20.660	1	0.3277

Class 1
Class sample count 20

Class 2
Class sample count 20

	Σ((x-mean_k=0)^2)	Σ((x-mean_k=1)^2)	k= Class 0 or 1
STEP 3	10.1582	21.4932	

	VARIANCE
STEP 4	=1/(COUNT(E2:E41)-2)*(L10+M10)

VARIANCE
0.832931506

STEP 4 - Predictions

- The discriminant function for a class given an input (x) is calculated using:

$$discriminant(x) = x \times \frac{mean}{variance} - \frac{mean^2}{2 \times variance} + ln(probability)$$

  - x is the input value
  - mean, variance and probability are calculated above for the class we are discriminating

Predictions are made by calculating the **discriminant function** <u>for each class</u> and predicting the class with the **largest value**.

STEP 4 - Calculations

Let's step through the calculation of the discriminate value of each class for the 1st instance.

The 1st instance in the dataset is:
X = 4:667797637 and Y = 0.

The discriminant value for Y = 0 is calculated as follows:

$$discriminant(Y = 0|x) = 4.667797637 \times \frac{4.975415507}{0.832931506} - \frac{4.975415507^2}{2 \times 0.832931506} + ln(0.5)$$

$$discriminant(Y = 0|x) = 12.3293558$$

We can also calculate the discriminant value for $Y = 1$:

$$discriminant(Y = 1|x) = 4.667797637 \times \frac{20.08706292}{0.832931506} - \frac{20.08706292^2}{2 \times 0.832931506} + ln(0.5)$$

$$discriminant(Y = 1|x) = -130.3349038$$

X	Y	(x-mean_k)^2
4.668	0	0.0946
5.509	0	0.2849
4.703	0	0.0743
5.957	0	0.9629
5.739	0	0.5825
5.027	0	0.0027
4.805	0	0.0289
4.426	0	0.3022
5.009	0	0.0012
	0	0.0200
	0	1.9474
	0	4.3280
	0	0.0952
	0	0.3928
	0	0.0053
	0	0.0033
	0	0.7947
	0	0.0100
4.733	0	0.0589
5.386	0	0.1686

STEP 4 – Decision Making

$$discriminant(Y = 0|x) = 4.667797637 \times \frac{4.975415507}{0.832931506} - \frac{4.975415507^2}{2 \times 0.832931506} + ln(0.5)$$

$$discriminant(Y = 0|x) = 12.3293558$$

We can also calculate the discriminant value for $Y = 1$:

$$discriminant(Y = 1|x) = 4.667797637 \times \frac{20.08706292}{0.832931506} - \frac{20.08706292^2}{2 \times 0.832931506} + ln(0.5)$$

$$discriminant(Y = 1|x) = -130.3349038$$

The discriminant value for Y = 0 (12.3293558)

> (LARGER)

The discriminate value for Y = 1 (-130.3349038)

Thus, the model predicts Y = 0

CORRECT
(as we already know)

STEP 4 – Repeat for ALL instances

We need to proceed to **calculate the Y values** for each instance in the dataset, as follows:

X	Y	(x-mean_k)^2	DISCRIMINANT Y=0	DISCRIMINANT Y=1	PREDICTION
4.668	0	0.0946	12.3293558	-130.2836105	0
5.509	0	0.2849	17.35536365	-109.992293	0
4.703	0	0.0743	12.53838805	-129.4396923	0
5.957	0	0.9629	20.02849769	-99.20014678	0
5.739	0	0.5825	18.72579795	1252.345751	1
5.027	0	0.0027	14.47670003	-121.6142162	0
4.805	0	0.0289	13.15151029	-126.9643563	0
4.426	0	0.3022	10.88315002	-136.1223242	0
5.009	0	0.0012	14.3696888	-122.0462488	0
5.117	0	0.0200	15.0109321	-119.4573807	0
6.371	0	1.9474	22.50273735	-89.21098953	0
2.895	0	4.3280	1.740014309	-173.0355713	0
4.667	0	0.0952	12.3236496	-130.306648	0
5.602	0	0.3928	17.91062422	-107.7505598	0
4.903	0	0.0053	13.73310188	-124.6163178	0
5.033	0	0.0033	14.50877492	-121.4847215	0
4.084	0	0.7947	8.841949563	-144.3631881	0
4.876	0	0.0100	13.57018471	-125.2740574	0
4.733	0	0.0589	12.7176458	-128.7159815	0
5.386	0	0.1686	16.61941128	-112.9635266	0

20.744	1	0.4315	108.3582168	257.4101954	1
21.418	1	1.7701	112.3818455	273.6546443	1
20.579	1	0.2422	107.3744415	253.4384351	1
20.739	1	0.4246	108.3269137	257.2838164	1
19.446	1	0.4109	100.60548	226.1103549	1
18.364	1	2.9703	94.1395889	200.0058493	1
19.904	1	0.0336	103.3387697	237.1453651	1
19.109	1	0.9572	98.59038855	217.9748998	1
18.188	1	3.6069	93.08990662	195.7680054	1
19.718	1	0.1364	102.2279828	232.6608258	1
19.096	1	0.9816	98.5162097	217.6754202	1
20.527	1	0.1939	107.0648488	252.1885279	1
20.632	1	0.2970	107.6899208	254.7121084	1
19.862	1	0.0506	103.0911664	236.1457254	1
21.347	1	1.5867	111.9587938	271.9466728	1
20.334	1	0.0609	105.9089574	247.52189	1
21.027	1	0.8838	110.0499579	264.2401995	1
18.275	1	3.2823	93.61248743	197.8778018	1
21.774	1	2.8448	114.5094616	282.2443908	1
20.660	1	0.3277	20.02849769	255.374804	1

- LDA has achieved an accuracy of 100% (no errors)
- This is not surprising given that the dataset was contrived so that the groups for Y = 0 and Y = 1 were clearly separable.

Create and Visualize Discriminant Analysis Classifier

This example shows how to perform linear and quadratic classification of Fisher iris data.

```
load fisheriris
```

The column vector, `species` , consists of iris flowers of three different species, setosa, versicolor, virginica. The double matrix `meas` consists of four types of measurements on the flowers, the length and width of sepals and petals in centimeters, respectively.

Use petal length (third column in `meas`) and petal width (fourth column in `meas`) measurements. Save these as variables PL and PW, respectively.

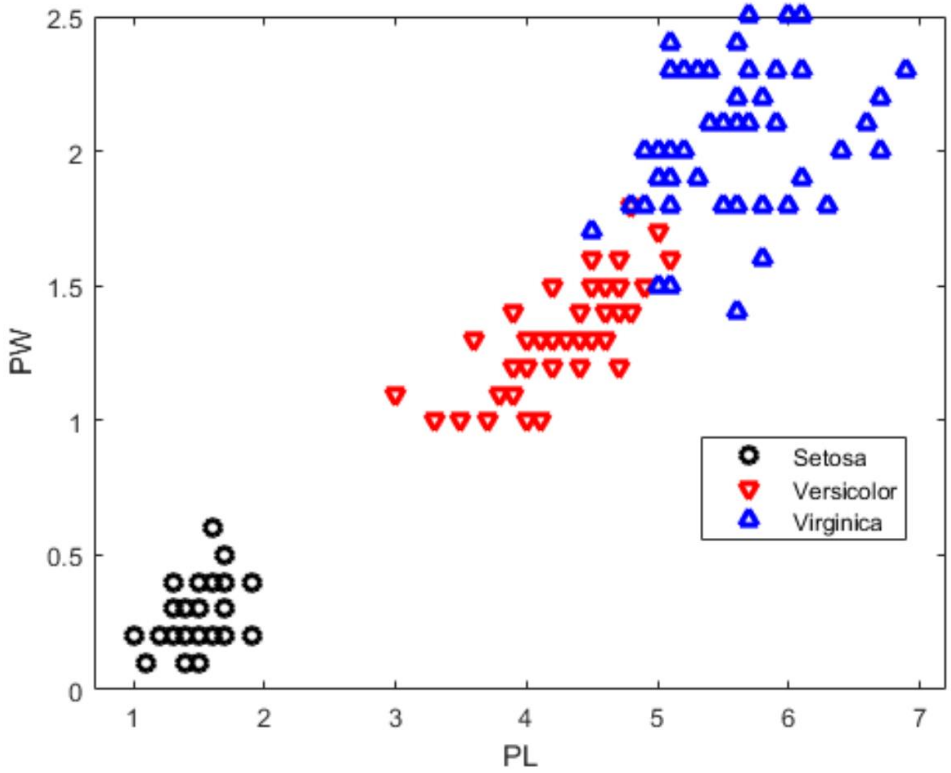
```
PL = meas(:,3);
PW = meas(:,4);
```

Plot the data, showing the classification, that is, create a scatter plot of the measurements, grouped by species.

```
h1 = gscatter(PL,PW,species,'krb','ov^',[],'off');
h1(1).LineWidth = 2;
h1(2).LineWidth = 2;
h1(3).LineWidth = 2;
legend('Setosa','Versicolor','Virginica','Location','best')
hold on
```

https://www.mathworks.com/help/stats/create-and-visualize-discriminant-analysis-classifier.html

Create and Visualize Discriminant Analysis Classifier



Create a linear classifier.

```
X = [PL,PW];
MdlLinear = fitcdiscr(X,species);
```

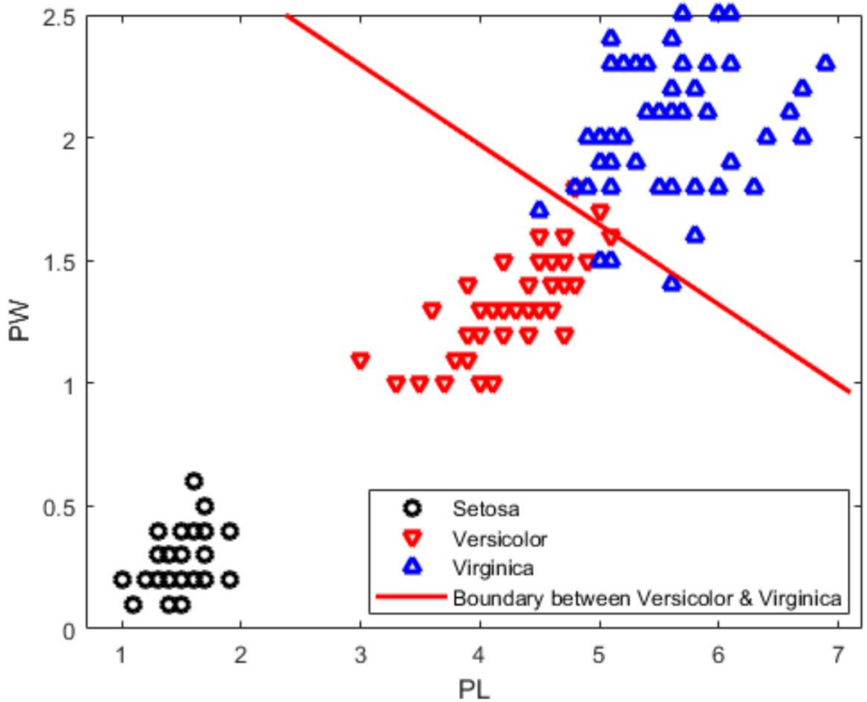
Retrieve the coefficients for the linear boundary between the second and third classes.

```
MdlLinear.ClassNames([2 3])
```

```
ans = 2x1 cell
    {'versicolor'}
    {'virginica' }
```

```
K = MdlLinear.Coeffs(2,3).Const;
L = MdlLinear.Coeffs(2,3).Linear;
```

Create and Visualize Discriminant Analysis Classifier



Plot the curve that separates the second and third classes

$$K + \begin{bmatrix} x_1 & x_2 \end{bmatrix} L = 0.$$

```
f = @(x1,x2) K + L(1)*x1 + L(2)*x2;
h2 = fimplicit(f,[.9 7.1 0 2.5]);
h2.Color = 'r';
h2.LineWidth = 2;
h2.DisplayName = 'Boundary between Versicolor & Virginica';
```

https://www.mathworks.com/help/stats/create-and-visualize-discriminant-analysis-classifier.html

Create and Visualize Discriminant Analysis Classifier

Retrieve the coefficients for the linear boundary between the first and second classes.

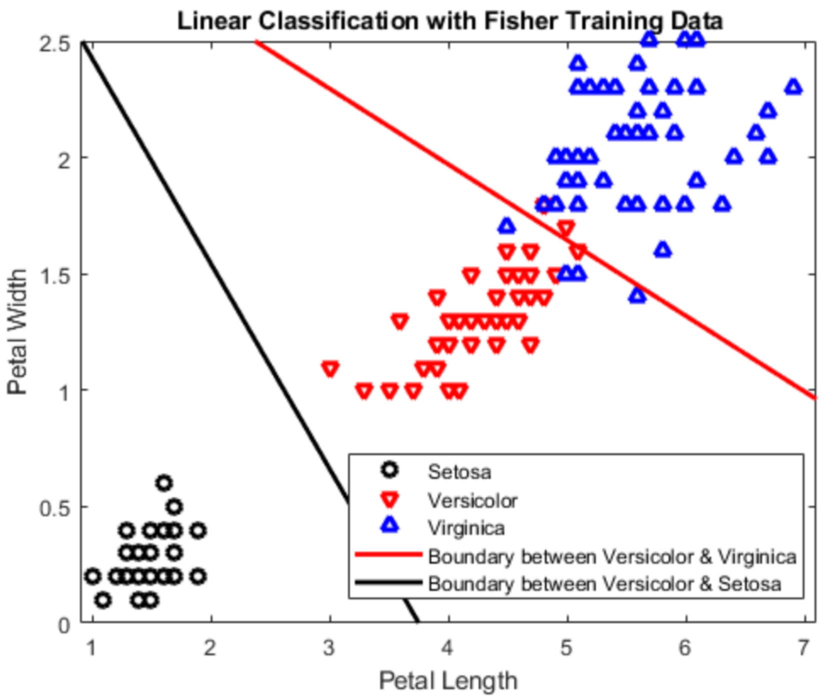
```
MdlLinear.ClassNames([1 2])
```

```
ans = 2x1 cell
    {'setosa'    }
    {'versicolor'}
```

```
K = MdlLinear.Coeffs(1,2).Const;
L = MdlLinear.Coeffs(1,2).Linear;
```

Plot the curve that separates the first and second classes.

```
f = @(x1,x2) K + L(1)*x1 + L(2)*x2;
h3 = fimplicit(f,[.9 7.1 0 2.5]);
h3.Color = 'k';
h3.LineWidth = 2;
h3.DisplayName = 'Boundary between Versicolor & Setosa';
axis([.9 7.1 0 2.5])
xlabel('Petal Length')
ylabel('Petal Width')
title('{\bf Linear Classification with Fisher Training Data}')
```



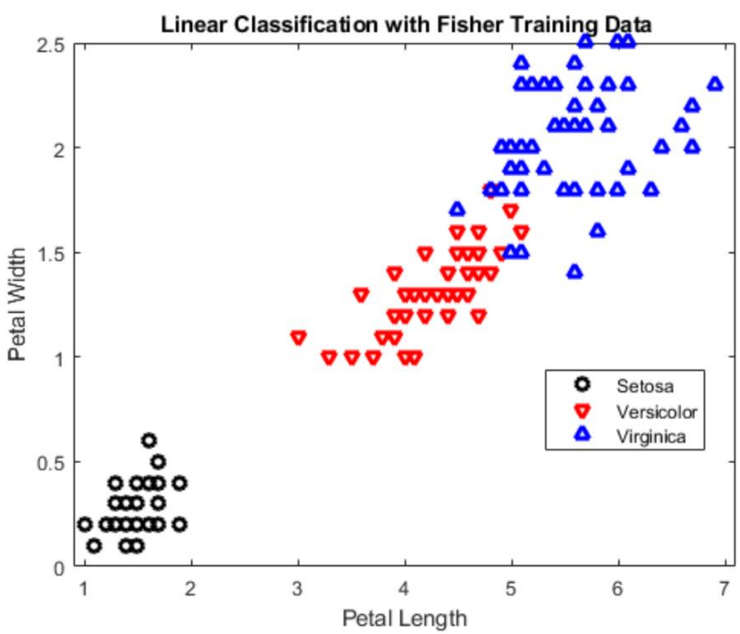
Create and Visualize Discriminant Analysis Classifier

Create a quadratic discriminant classifier.

```
MdlQuadratic = fitcdiscr(X,species,'DiscrimType','quadratic');
```

Remove the linear boundaries from the plot.

```
delete(h2);
delete(h3);
```



Retrieve the coefficients for the quadratic boundary between the second and third classes.

```
MdlQuadratic.ClassNames([2 3])
```

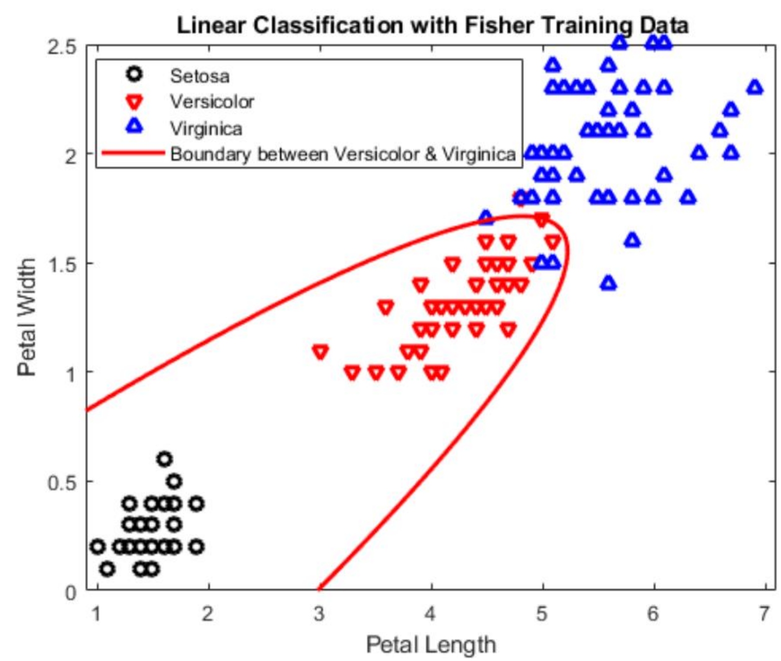
```
ans = 2x1 cell
    {'versicolor'}
    {'virginica' }
```

```
K = MdlQuadratic.Coeffs(2,3).Const;
L = MdlQuadratic.Coeffs(2,3).Linear;
Q = MdlQuadratic.Coeffs(2,3).Quadratic;
```

Plot the curve that separates the second and third classes

$$K + \begin{bmatrix} x_1 & x_2 \end{bmatrix} L + \begin{bmatrix} x_1 & x_2 \end{bmatrix} Q \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = 0.$$

```
f = @(x1,x2) K + L(1)*x1 + L(2)*x2 + Q(1,1)*x1.^2 + ...
    (Q(1,2)+Q(2,1))*x1.*x2 + Q(2,2)*x2.^2;
h2 = fimplicit(f,[.9 7.1 0 2.5]);
h2.Color = 'r';
h2.LineWidth = 2;
h2.DisplayName = 'Boundary between Versicolor & Virginica';
```



https://www.mathworks.com/help/stats/create-and-visualize-discriminant-analysis-classifier.html

Create and Visualize Discriminant Analysis Classifier

Retrieve the coefficients for the quadratic boundary between the first and second classes.

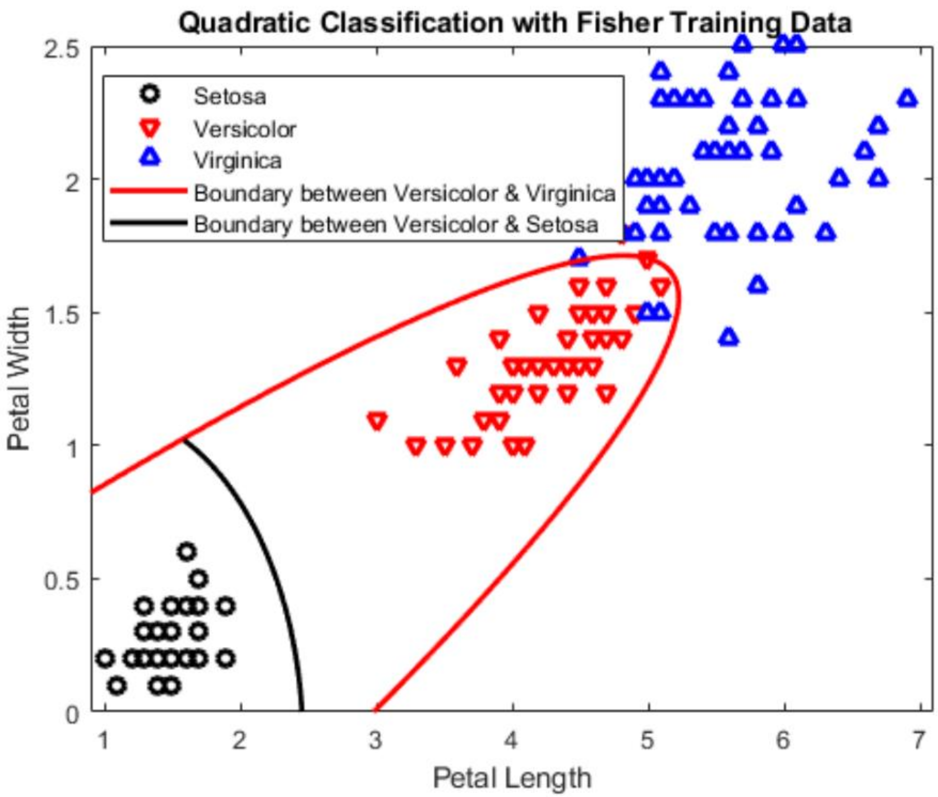
```
MdlQuadratic.ClassNames([1 2])
```

```
ans = 2x1 cell
    {'setosa'    }
    {'versicolor'}
```

```
K = MdlQuadratic.Coeffs(1,2).Const;
L = MdlQuadratic.Coeffs(1,2).Linear;
Q = MdlQuadratic.Coeffs(1,2).Quadratic;
```

Plot the curve that separates the first and second and classes.

```
f = @(x1,x2) K + L(1)*x1 + L(2)*x2 + Q(1,1)*x1.^2 + ...
    (Q(1,2)+Q(2,1))*x1.*x2 + Q(2,2)*x2.^2;
h3 = fimplicit(f,[.9 7.1 0 1.02]); % Plot the relevant portion of the curve.
h3.Color = 'k';
h3.LineWidth = 2;
h3.DisplayName = 'Boundary between Versicolor & Setosa';
axis([.9 7.1 0 2.5])
xlabel('Petal Length')
ylabel('Petal Width')
title('{\bf Quadratic Classification with Fisher Training Data}')
hold off
```



https://www.mathworks.com/help/stats/create-and-visualize-discriminant-analysis-classifier.html

Discriminant Analysis Classification

Discriminant analysis is a classification method. It **assumes** that different classes generate data based on different Gaussian distributions.

- To train (create) a classifier, the fitting function estimates the parameters of a Gaussian distribution for each class (see Creating Discriminant Analysis Model).
- To predict the classes of new data, the trained classifier finds the class with the smallest misclassification cost (see Prediction Using Discriminant Analysis Models)
- Linear discriminant analysis is also known as the Fisher discriminant, named for its inventor, Sir R. A. Fisher [1].

https://www.mathworks.com/help/stats/discriminant-analysis.html

Discriminant Analysis Classification

Create Discriminant Analysis Classifiers

This example shows how to train a basic discriminant analysis classifier to classify irises in Fisher's iris data.

Load the data.

```
load fisheriris
```

Create a default (linear) discriminant analysis classifier.

```
MdlLinear = fitcdiscr(meas,species);
```

To visualize the classification boundaries of a 2-D linear classification of the data, see Create and Visualize Discriminant Analysis Classifier.

Classify an iris with average measurements.

```
meanmeas = mean(meas);
meanclass = predict(MdlLinear,meanmeas)
```

```
meanclass = 1x1 cell array
    {'versicolor'}
```

https://www.mathworks.com/help/stats/discriminant-analysis.html

Discriminant Analysis Classification

Create a quadratic classifier.

```
MdlQuadratic = fitcdiscr(meas,species,'DiscrimType','quadratic');
```

To visualize the classification boundaries of a 2-D quadratic classification of the data, see Create and Visualize Discriminant Analysis Classifier.

Classify an iris with average measurements using the quadratic classifier.

```
meanclass2 = predict(MdlQuadratic,meanmeas)
```

```
meanclass2 = 1x1 cell array
    {'versicolor'}
```

References

[1] Fisher, R. A. The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics, Vol. 7, pp. 179–188, 1936. Available at https://digital.library.adelaide.edu.au/dspace/handle/2440/15227.

See Also → Functions → fitcdiscr

LDA – Face Recognition Solution

https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/35106/versions/1/previews/PhD_tool/features/perform_lda_PhD.m/index.html

-- Homework --

Use code and Celebrity Face Dataset > after face normalization

https://www.mathworks.com/help/stats/create-and-visualize-discriminant-analysis-classifier.html

What have we learned

- The model representation for LDA and what is distinct about a learned model
- How the parameters of the LDA model can be estimated from training data
- How the model can be used to make predictions on new data
- How to prepare your data to get the most from the method
- How to calculate the statistics from your dataset required by the LDA model
- How to use the LDA model to calculate a discriminant value for each class and make a prediction.
- MATLAB Examples

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

Questions?

THANK YOU!