Classification

PHYS591000 Spring 2021

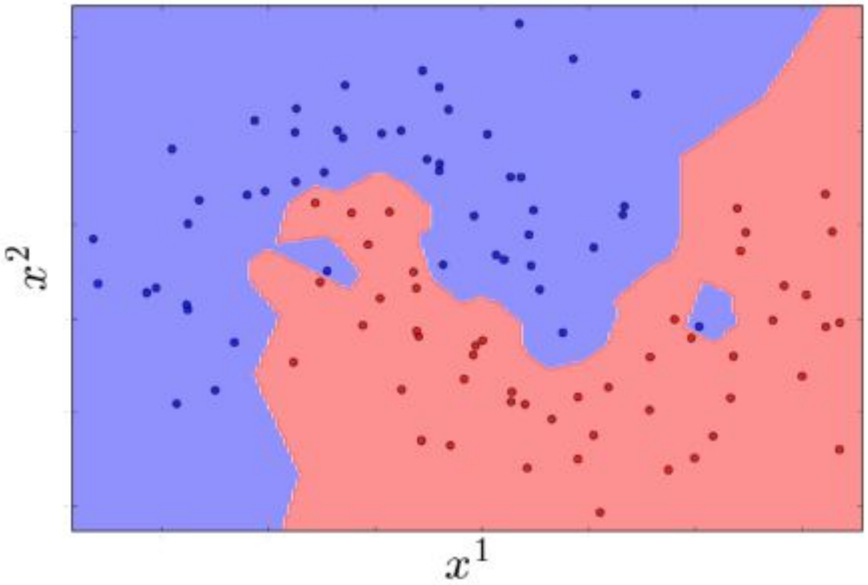
Reference: https://indico.cern.ch/event/619370/ Lecture by Michael Kagan

Outline

- Review: supervised and unsupervised learning
- Binary classification with linear discriminant
- Performance of the classification algorithm: ROC and AUC

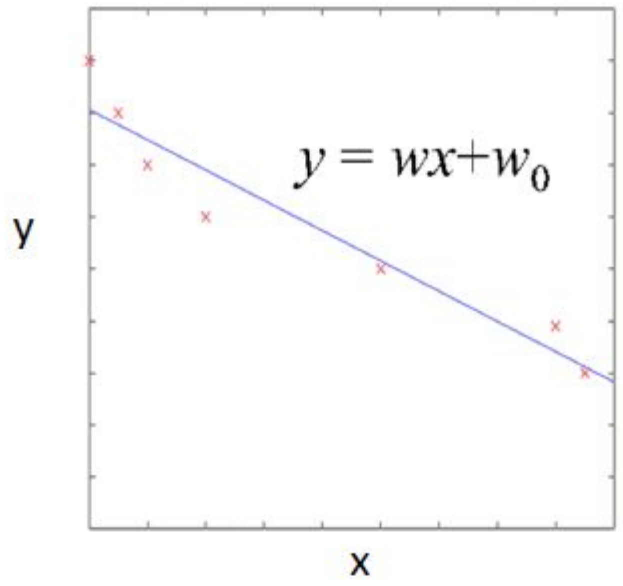
Supervised Learning

- Trained with the right answers given
- Typical task: **Classification**



Supervised Learning

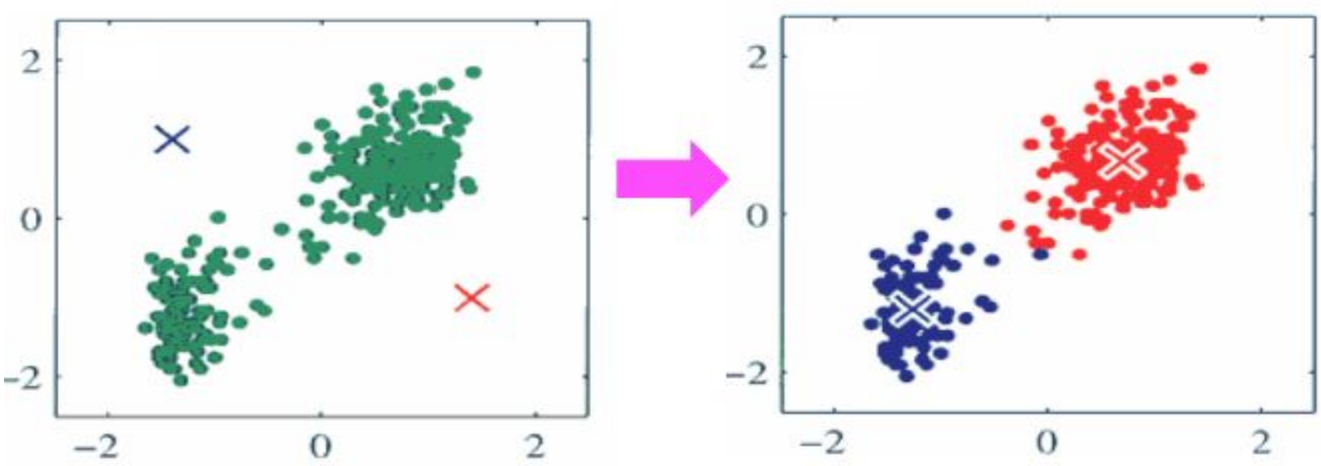
- Trained with the right answers given
- Typical task: **Regression**



Unsupervised Learning

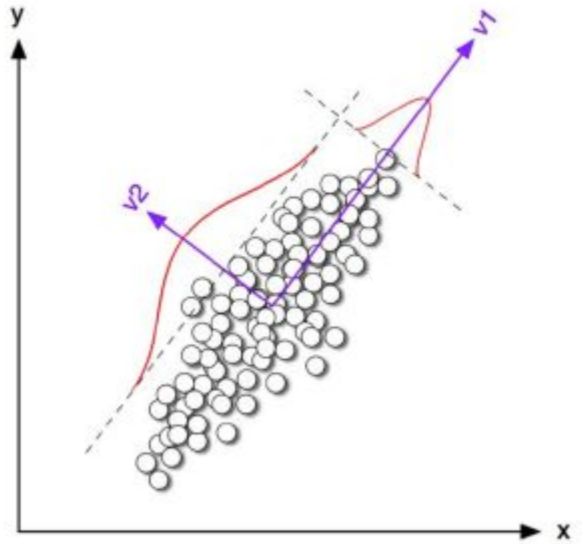
- Trained without knowing the right answers

- Typical task: **Clustering**



Unsupervised Learning

- Trained without knowing the right answers
- Typical task: **Dimensionality reduction**

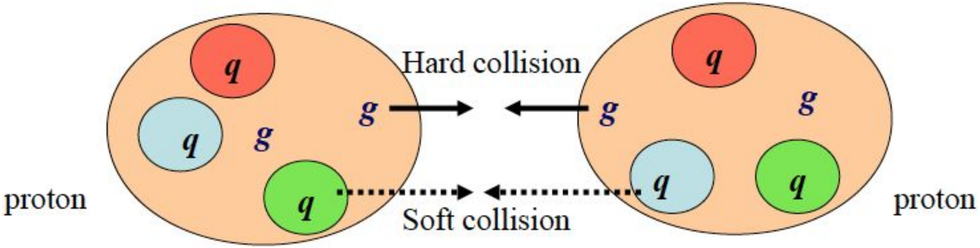


Classification

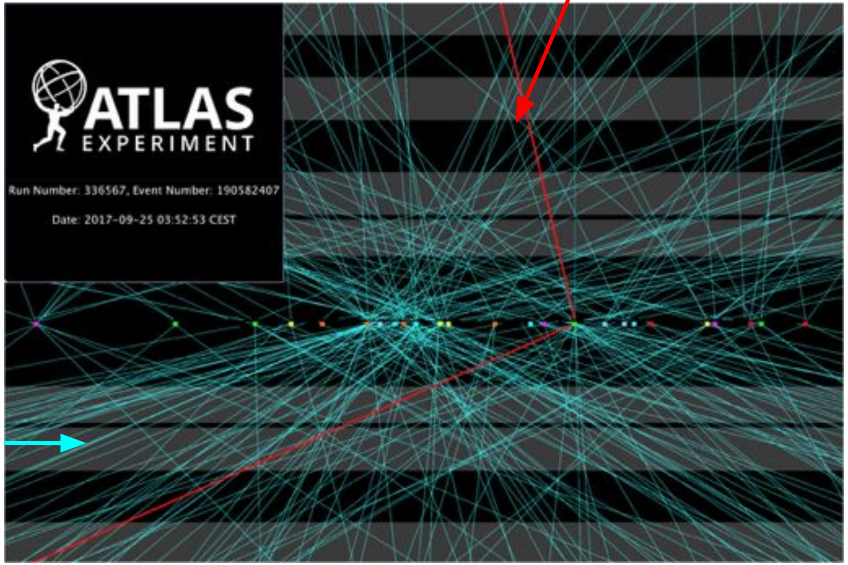
- Topic for the week
- Focus on **binary classification:** Distinguish signal from background
    - Example: Distinguish particles from hard (head-on) collisions (signal) and particles from soft collisions (background) at the LHC

Binary Classification

- At the LHC we collide bunches of protons



Signal

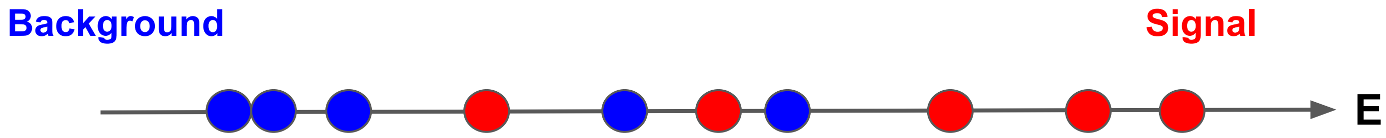


Background

Binary Classification

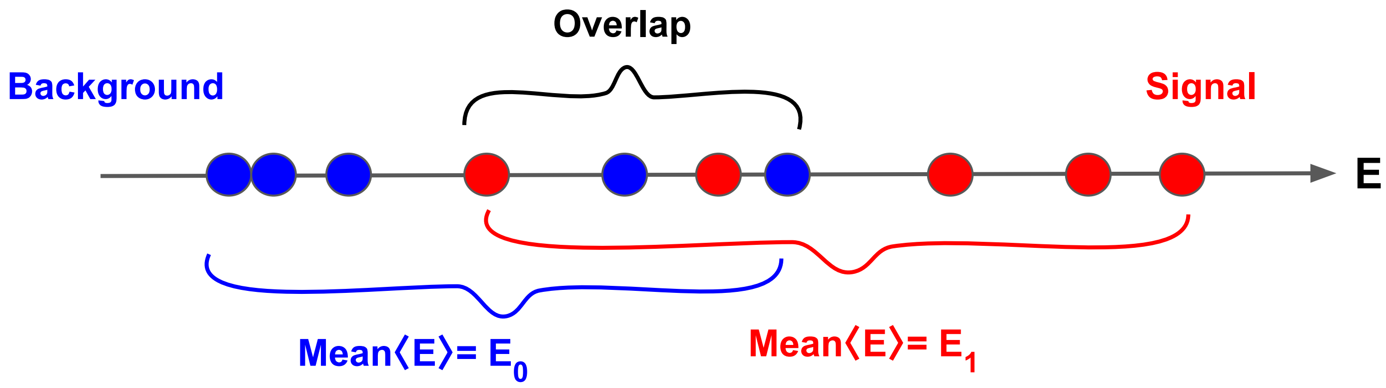
- Use physics to ‘guess’ a feature that can separate signals and backgrounds, e.g. energy of a particle

- Expect higher energy (E) for signals from hard collisions



Binary Classification

- Signals from hard collisions have higher *mean* (average) energy
- There is overlap of signal and background due to *spreads* of their energy distributions

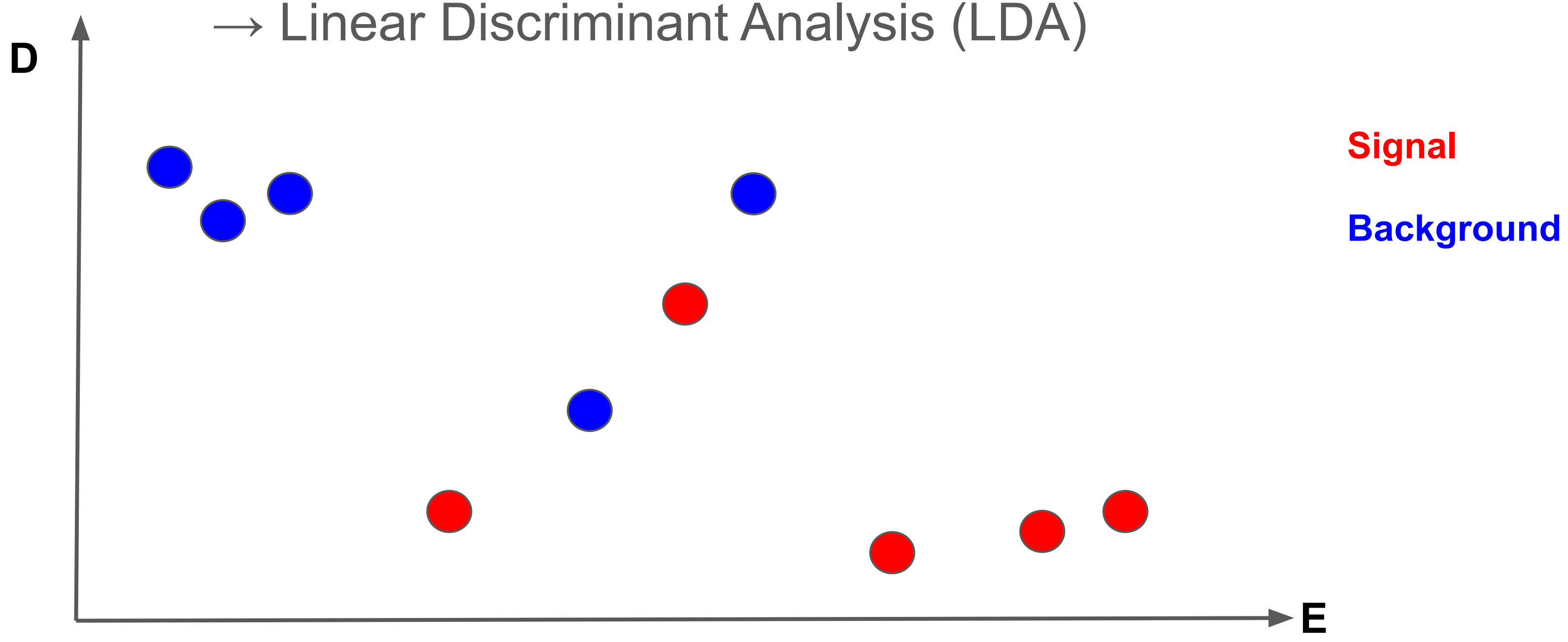


Binary Classification

- Using the energy to separate signals from backgrounds is OK.
- We may be able to do better by using more information, e.g.
  D = distance from the primary vertex (main collision point)
  -> Expect signals originate from tracks nearer the primary vertex.

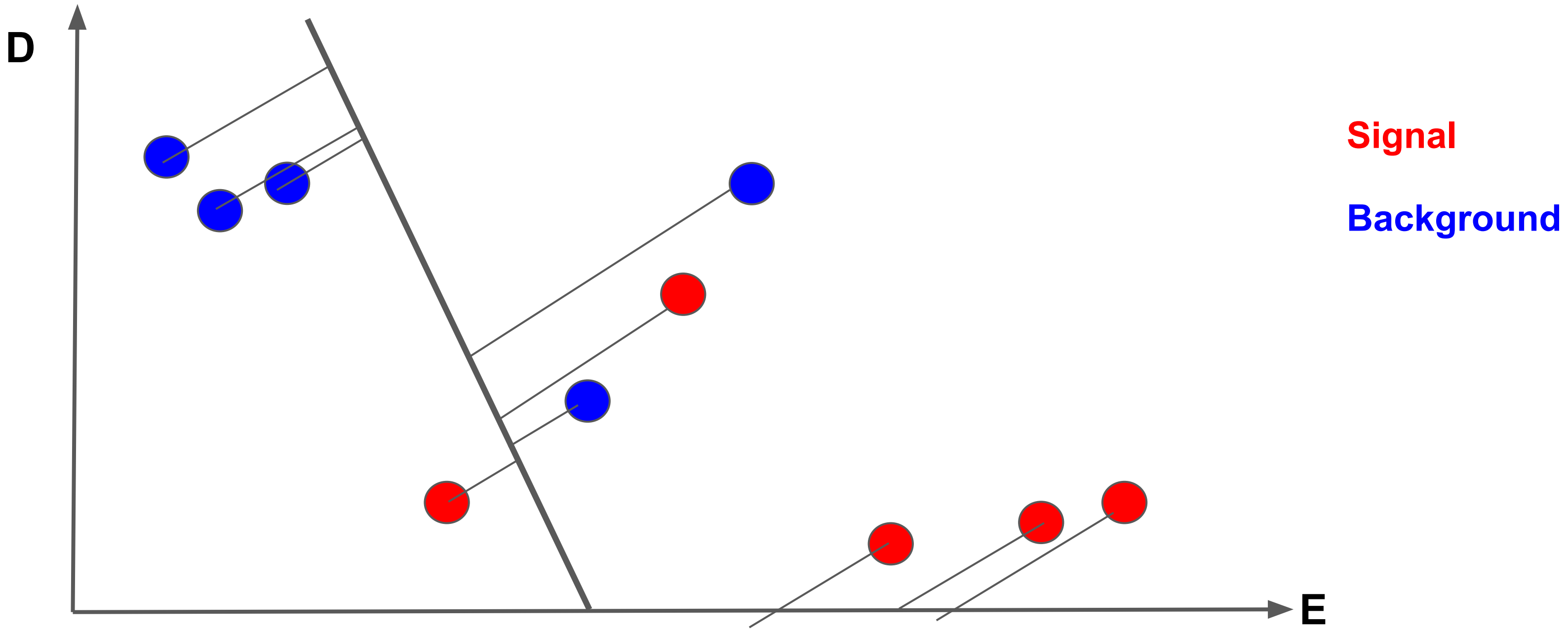
Binary Classification

- Want to transform the 2D information into a number (1D)
  → Linear Discriminant Analysis (LDA)



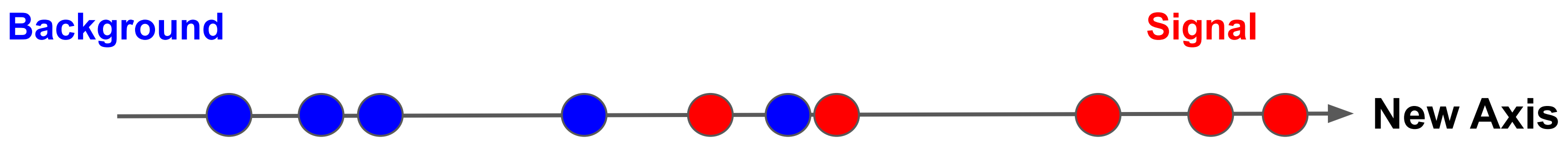
Linear Discriminant Analysis (LDA)

- LDA finds a new axis and projects all data onto the new axis



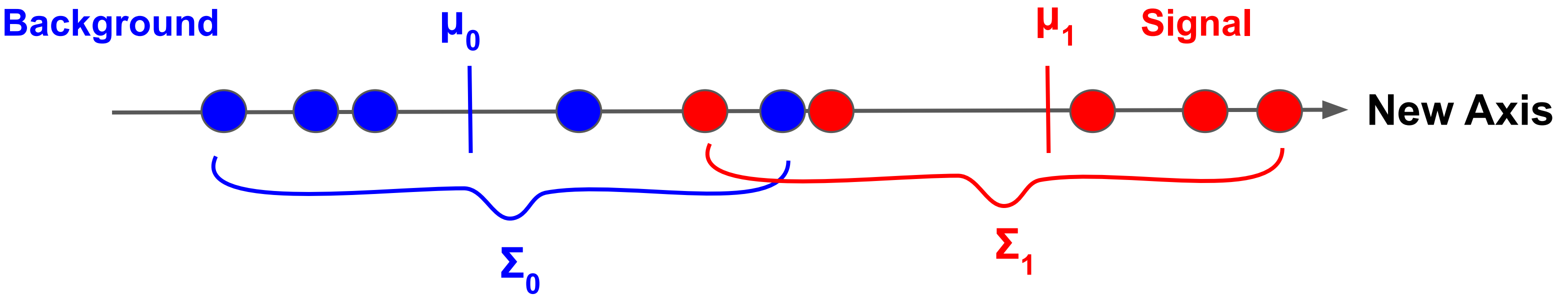
Linear Discriminant Analysis (LDA)

- The new axis can maximize the separation between the two categories when data are projected on it.



Linear Discriminant Analysis (LDA)

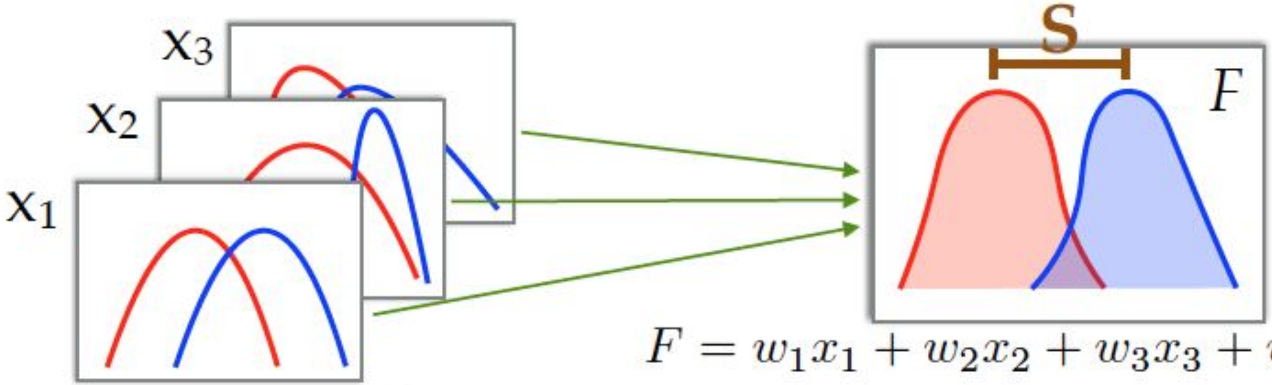
- The new axis is created by using optimized weights to combine all the information in order to
  - Maximize the distance between the means (μ)
  - Minimize the spreads (“covariance” Σ) within each category



Linear Discriminant Analysis (LDA)

Courtesy of Prof. Kai-Feng Chen (NTU)

- Now let's practice the easiest/simplest algorithm: **Linear discriminant analysis (LDA)**, or even simpler, the **Fisher's discriminant**, by combining the multiple features into one variable:



$$F = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + \cdots$$

Calculate the weights ($\boldsymbol{w_i}$) to maximize the separation $\boldsymbol{S}$.

Linear Discriminant Analysis (LDA)

Courtesy of Prof. Kai-Feng Chen (NTU)

- Consider a set of observables: $\vec{x} = (x_1, x_2, x_3, \cdots)$
- For 2 different event classes, the **mean** and **covariance** of the observables are: $\vec{\mu}_0, \vec{\mu}_1, \Sigma_0, \Sigma_1$

$$\vec{\mu} = \langle \vec{x} \rangle \qquad \Sigma = \langle (\vec{x} - \vec{\mu}) \cdot (\vec{x} - \vec{\mu})^T \rangle$$

- The separation ***S*** is given by

$$S = \frac{(\vec{w} \cdot \vec{\mu}_1 - \vec{w} \cdot \vec{\mu}_0)^2}{\vec{w}^T \Sigma_1 \vec{w} + \vec{w}^T \Sigma_0 \vec{w}}$$

distance of μ

covariance Σ

- The optimal weights can be determined by maximizing the ***S***:

$$\vec{w} \propto (\Sigma_0 + \Sigma_1)^{-1} (\vec{\mu}_1 - \vec{\mu}_0)$$

Linear Discriminant Analysis (LDA)

- It's straightforward to implement the calculation in numpy:

```
mu0 = var0.mean(axis=1)
mu1 = var1.mean(axis=1)          ⇐ mean values, in shape of (2, )
cov0 = np.cov(var0)
cov1 = np.cov(var1)              ⇐ covariance matrix, in shape of (2, 2)

weight = np.dot(linalg.inv(cov1+cov0),mu1-mu0)     ⇐ weight calculation
norm = np.sqrt((weight**2).sum())
weight /= norm
```

Courtesy of Prof. Kai-Feng Chen (NTU)

Linear Discriminant Analysis (LDA)

- Or with Scikit-learn:

```
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
```

```
clf = LinearDiscriminantAnalysis()
f_train = clf.fit_transform(x_train, y_train)
```

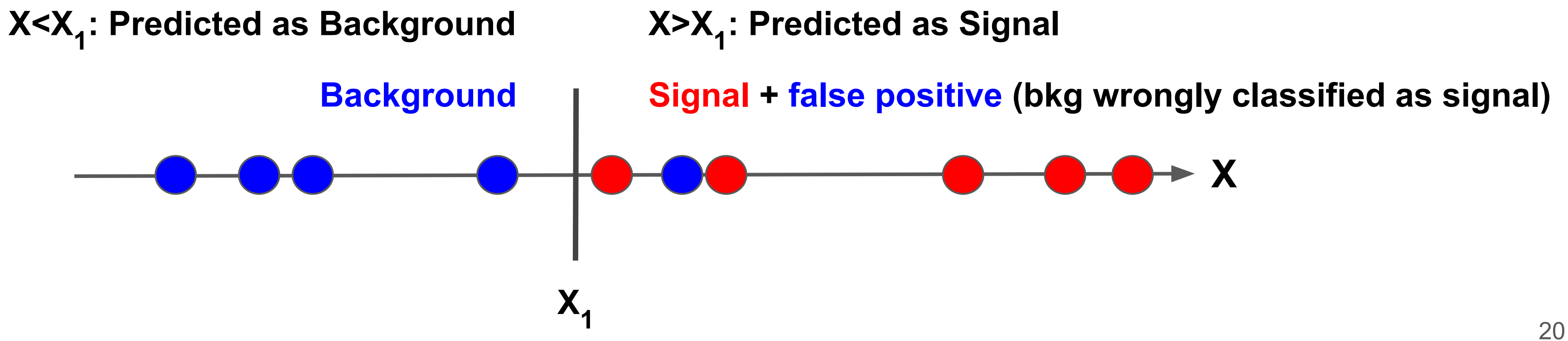
⇐ "training"

Courtesy of Prof. Kai-Feng Chen (NTU)

You'll learn more in the in-class sessions.

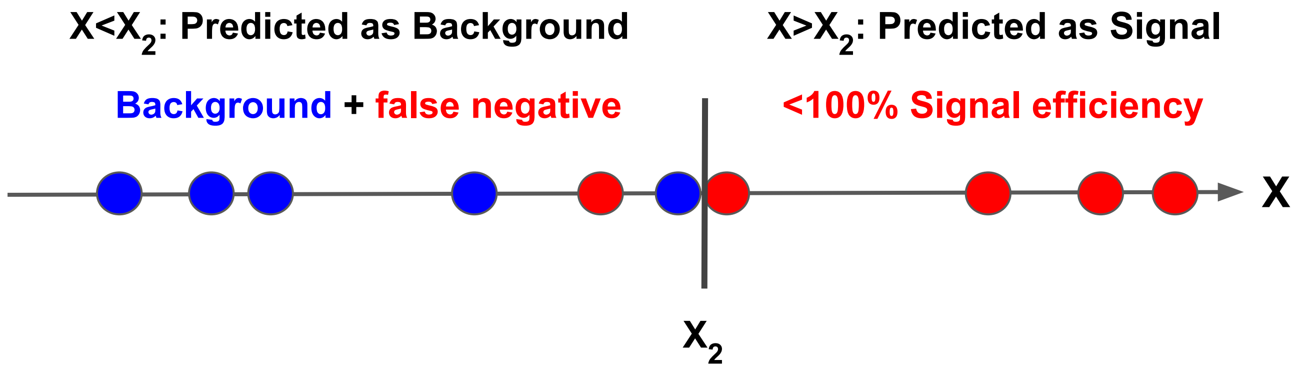
Performance of the Classifier

- The LDA maps all the information into one ‘score’.
  -> Where should we cut on to separate signals from backgrounds?
- The performance of this LDA depends on the cut value we choose.



Performance of the Classifier

- The LDA maps all the information into one ‘score’.
  -> Where should we cut on to separate signals from background?
- The performance of this LDA depends on the cut value we choose.



Performance of the Classifier

- One way to quantify the performance of the chosen cut is to construct the corresponding **confusion matrix**

	Actual Signal	**Actual** Background
Predicted as Signal	True Positives	False Positives
Predicted as Background	False Negatives	True Negatives

Performance of the Classifier

- One way to quantify the performance of the chosen cut is to construct the corresponding **confusion matrix**

X_1	**Actual** Signal	**Actual** Background
Predicted as Signal	5	1
Predicted as Background	0	4

X_2	**Actual** Signal	**Actual** Background
Predicted as Signal	4	0
Predicted as Background	1	5

Performance of the Classifier

- A good way to summarize all confusion matrices is the **ROC** curve (receiver operating characteristic curve)

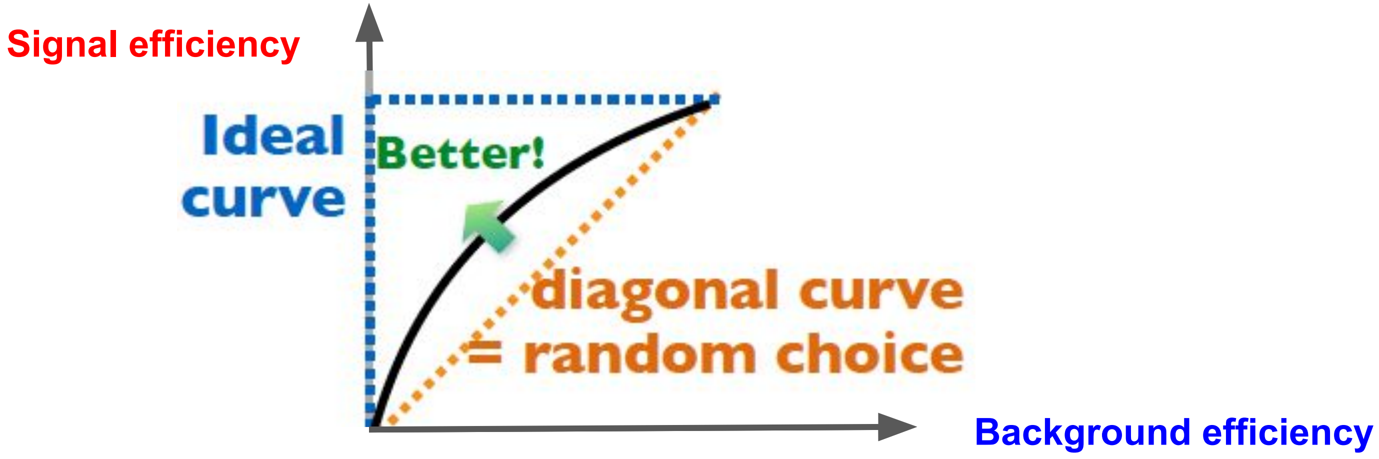
Signal efficiency
(True Positive Rate) =

$$\frac{\text{True Positives}}{\text{Actual Positives}}$$

Background efficiency
(False Positive Rate) = $\frac{\text{False Positives}}{\text{Actual Negatives}}$

Performance of the Classifier

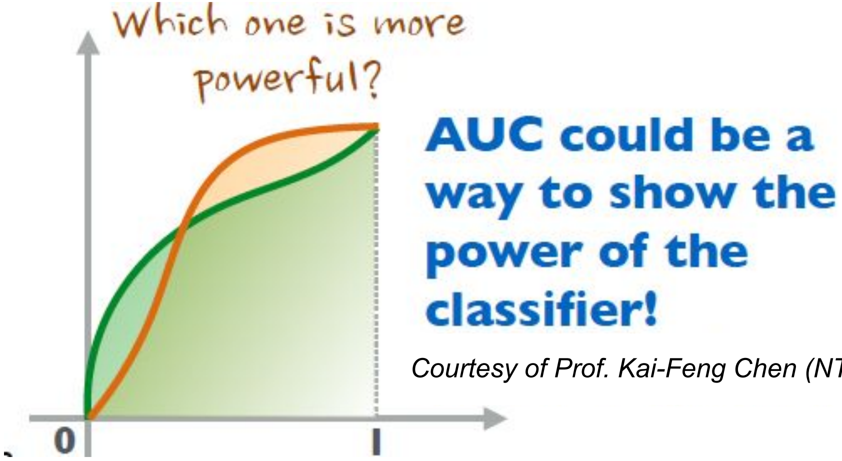
- The **ROC** curve illustrates the ability of the binary classifier when the discrimination threshold is varied.



Performance of the Classifier

- ROC curves can be used to compare two or more classifiers: The more it bends away from the diagonal line, the better its performance is.

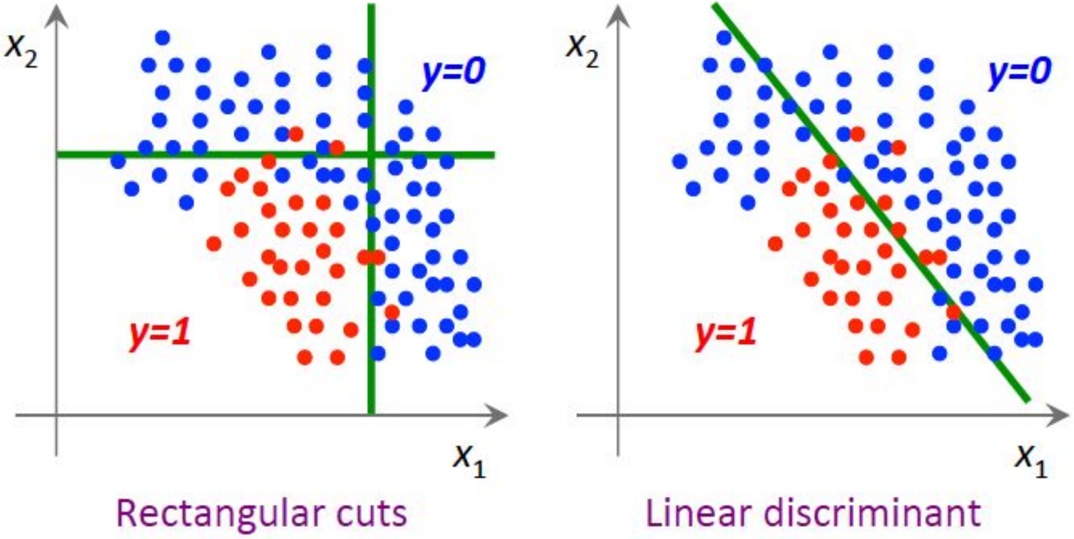
- Another way to estimate the performance is the **AUC** (area under the curve), which varies from 0.5 (diagonal line) to 1.0 (ideal case)



Courtesy of Prof. Kai-Feng Chen (NTU)

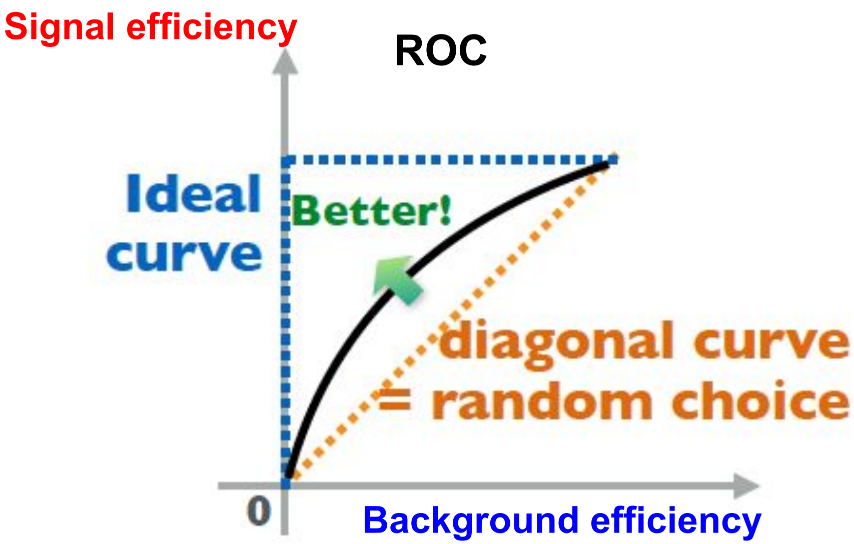
Summary-I

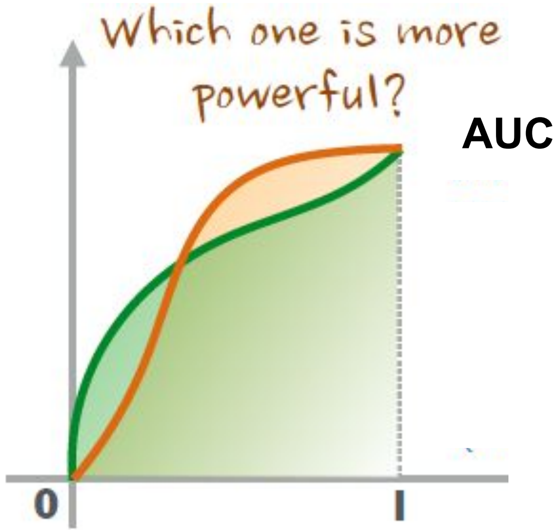
- The linear discriminant analysis (LDA) algorithm projects all information to a new axis which maximizes the separation of the two categories



Summary-II

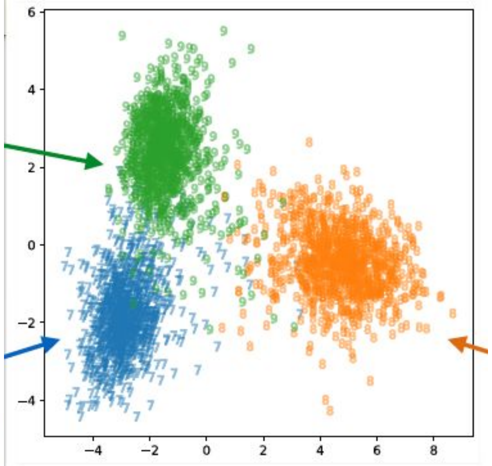
- The performance of a binary classifier can be estimated by the ROC curve or the AUC



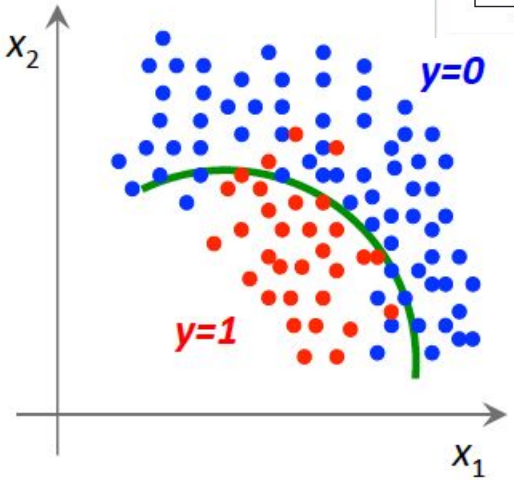


Outlook

- What if we want to do multiclass classification?



- Or if we need a Non-linear Discriminant?



Nonlinear discriminant

Outlook

- What if we want to do multiclass classification?
- Or if we need a Non-linear Discriminant?
- We will make use of more sophisticated algorithms such as support vector machines (SVM), decision trees, neural networks... next time!

LDA in Scikit-learn

- In Scikit-learn LDA makes predictions by estimating the *probability* of an event belongs to each class, assuming the probability of each class is Gaussian and shares the same covariance.

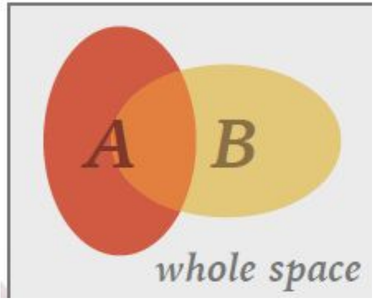
- The predicted class is the one with the highest (posterior) probability.

LDA in Scikit-learn

- The probability is calculated using Bayes' Theorem

- Then the conditional probability, **P(A | B)**, the probability that an elementary event, known to belong to the set ***B***, and is also a member of set ***A***:

$$P(A \text{ and } B) = P(A|B)P(B) = P(B|A)P(A)$$



Bayes theorem $P(A|B) = P(B|A) \cdot P(A)/P(B)$

Courtesy of Prof. Kai-Feng Chen (NTU)