Classification

PHYS591000 2022.02.23

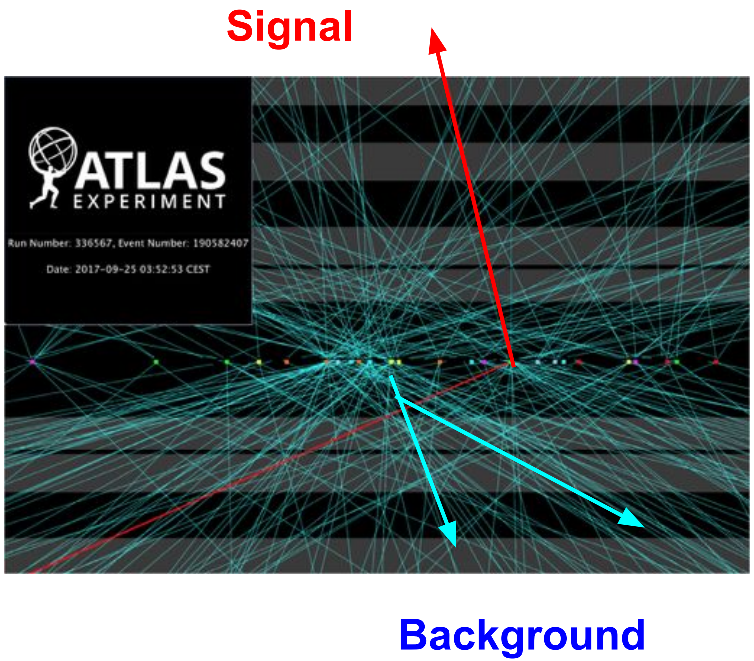
Announcement

- From this week (week 02) do *not* accept lab (homework) submission if you are absent from class without permission.

- As usual, take 3 mins to introduce yourself to your teammate for this week!
  – "How do you like the class last week?"
  – If your teammate just joined the class this week, tell your teammate what we did last week.

Classification

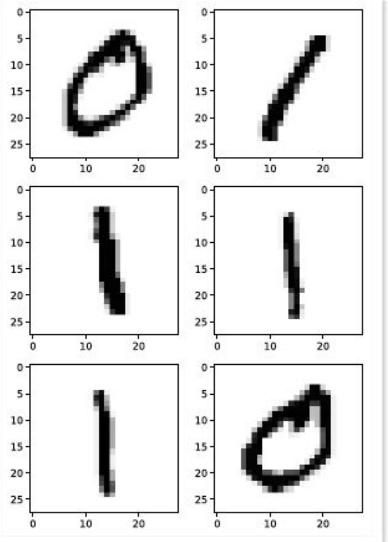
- Focus on **binary classification**
    - Example: Distinguish particles from hard collisions (**signal**) and particles from soft collisions (**background**) at the LHC



Binary Classification

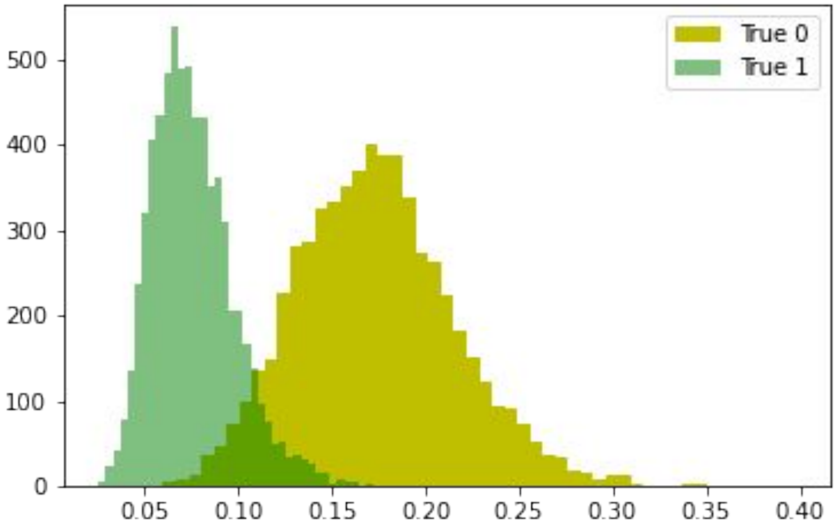
- In-class: Use the MNIST database
  - x_train: 28*28 images; each pixel is associated with a number from 0-255 (~ the amount of 'ink')

- Task: Separate 0 (background) and 1 (signal)



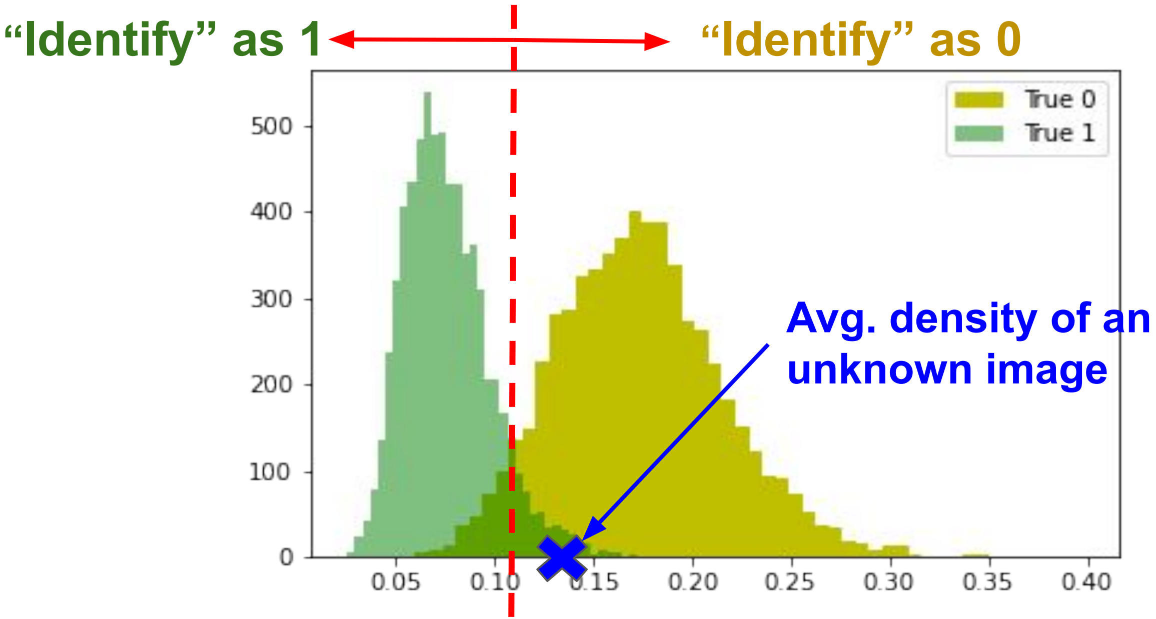
Binary Classification

- Week 01: A useful **feature** is the average pixel density (‘average amount of ink on each pixel’)



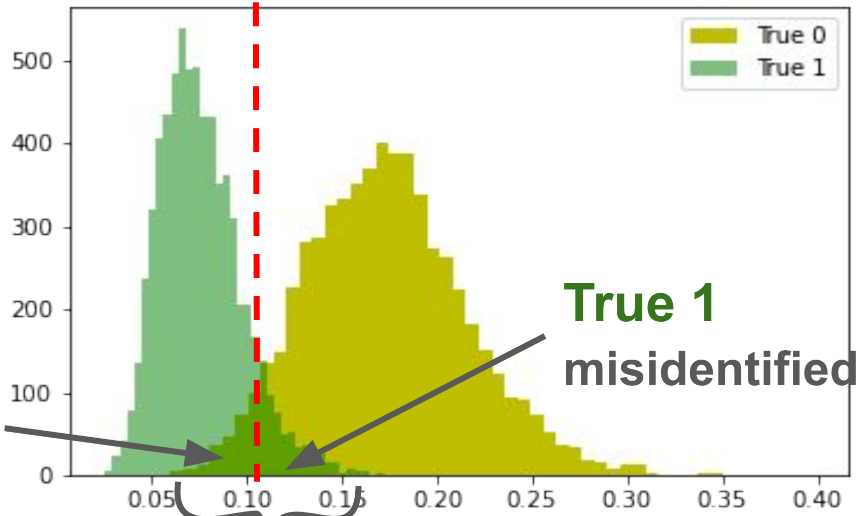
Binary Classification

- Given an (unknown) image, calculate its average pixel density



Binary Classification

- There is an overlap of signal and background due to ***spreads*** of their distributions



True 0
misidentified as 1

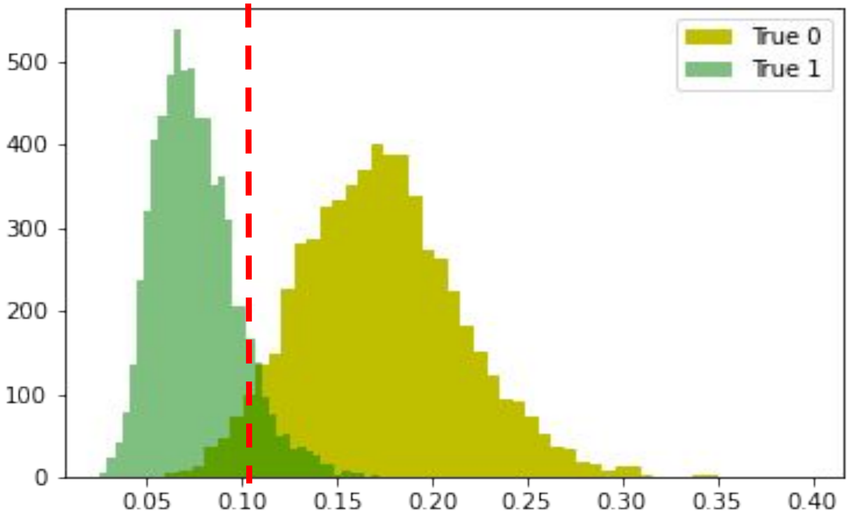
True 1
misidentified as 0

Overlap

Binary Classification

- The performance of the cut (the ‘**classifier**’) thus depends on the cut value (threshold) we choose

Courtesy of Prof. Kai-Feng Chen (NTU)

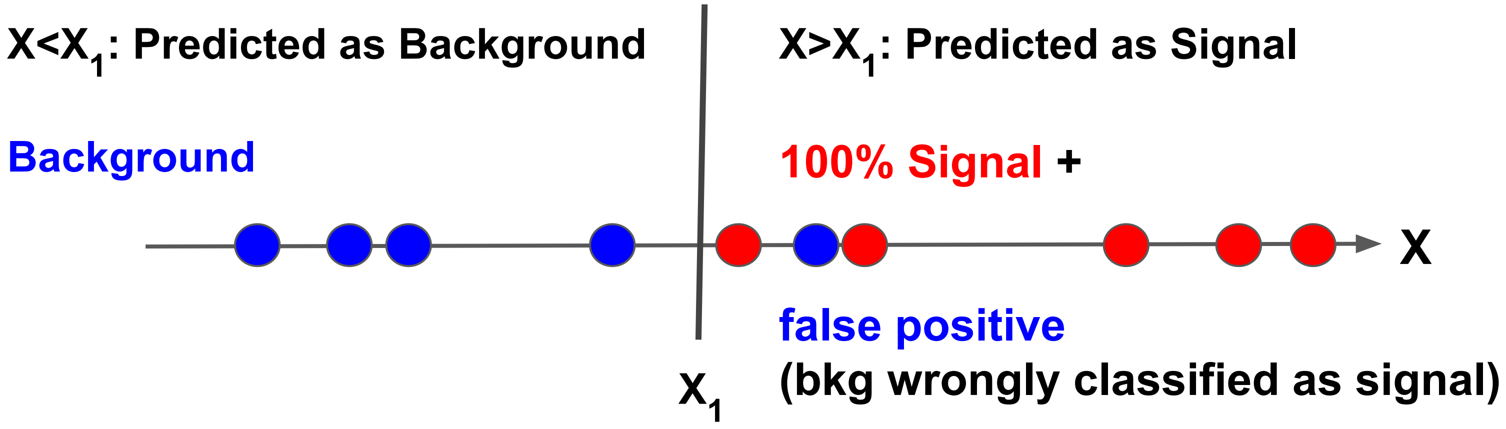


- ☐ If a threshold of **0.11** is set:
  93.0% of the “ones” are selected;
  94.5% of the “zeros” are rejected.
  (*or **5.5%** of the zeros are misidentified*)

- ☐ If a threshold of **0.16** is set:
  99.8% of the “ones” are selected;
  61.2% of the “zeros” are rejected.
  (*or **38.8%** of the zeros are misidentified*)

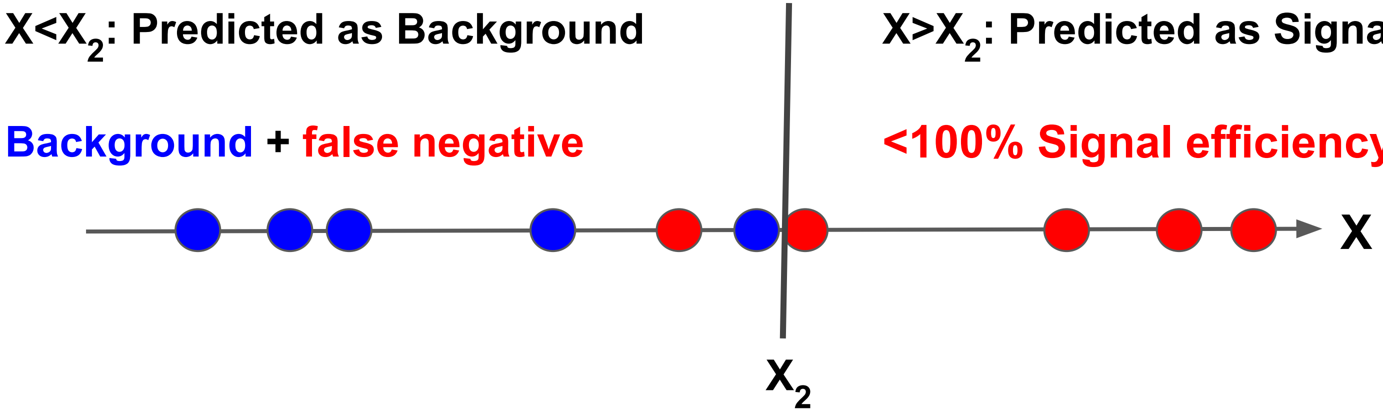
Performance of the Classifier

- The cut is a **linear** classifier which maps data into a ‘score’ X
- The performance depends on the cut value we choose, e.g.



Performance of the Classifier

- The cut is a **linear** classifier which maps data into a ‘score’ X
- The performance depends on the cut value we choose, e.g.



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

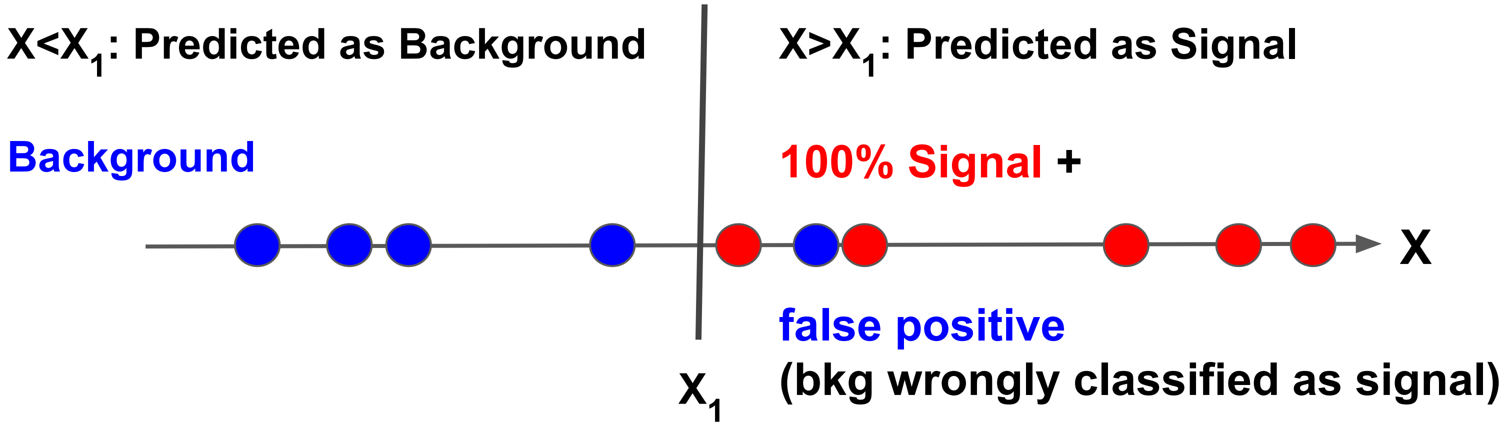
- One confusion matrix for each cut value

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

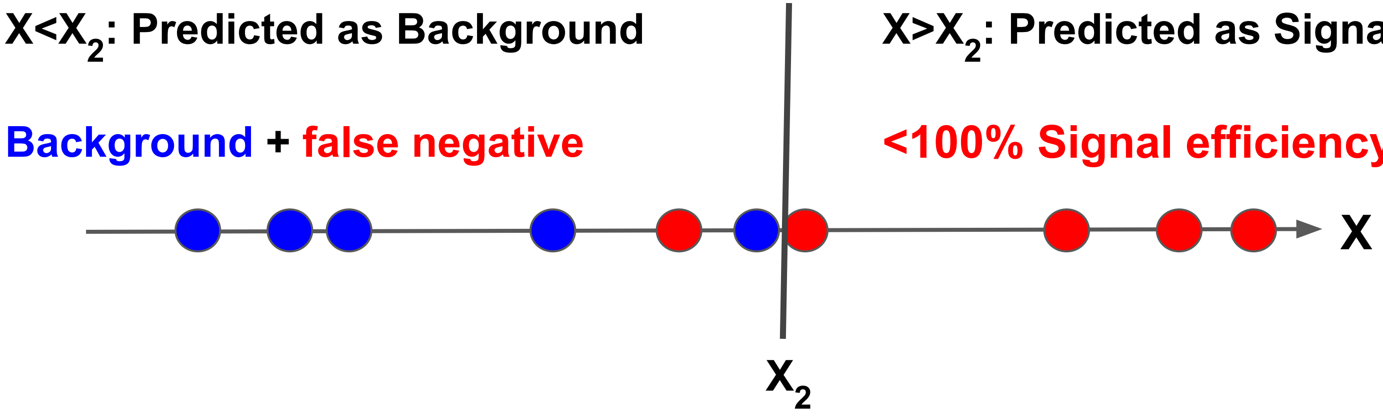
Performance of the Classifier

- Which cut to choose depends on what is more important to your goal: E.g. Searching for rare signals → Identify as many signals as possible → Choose X_1



Performance of the Classifier

- Which cut to choose depends on what is more important to your goal: E.g. Perform a precise measurement → Reject as many background as possible → Choose X_2



Performance of the Classifier

- There are many metrics for comparing the performances of the models based on the confusion matrix.

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

$$\text{sensitivity} = \frac{\text{number of true positives}}{\text{number of true positives} + \text{number of false negatives}}$$

Performance of the Classifier

- There are many metrics for comparing the performances of the models based on the confusion matrix.

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

$$\text{specificity} = \frac{\text{number of true negatives}}{\text{number of true negatives} + \text{number of false positives}}$$

Performance of the Classifier

X_1 **Better Sensitivity**	**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

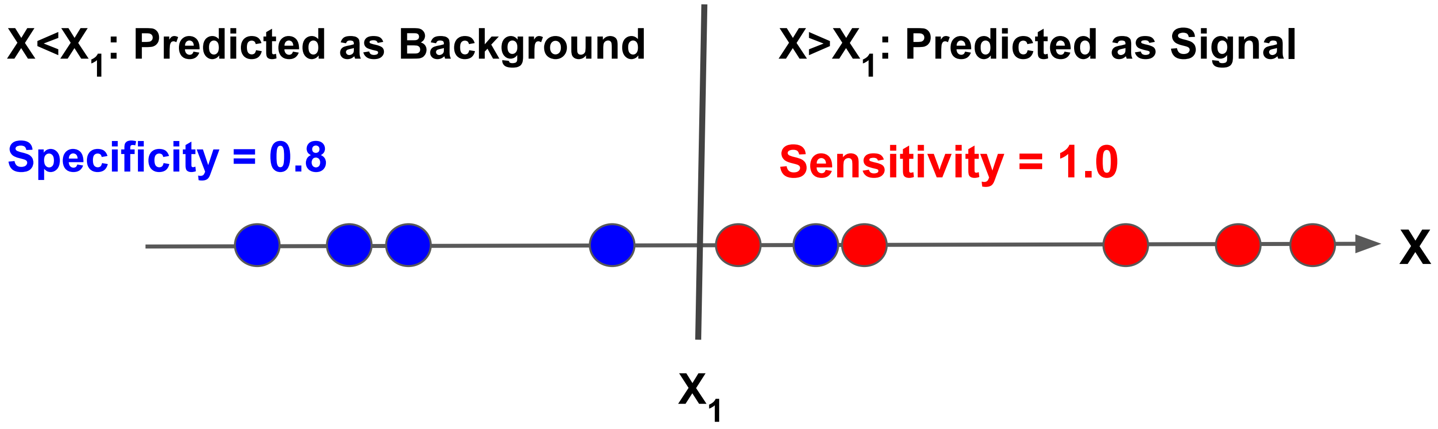
Performance of the Classifier

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

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

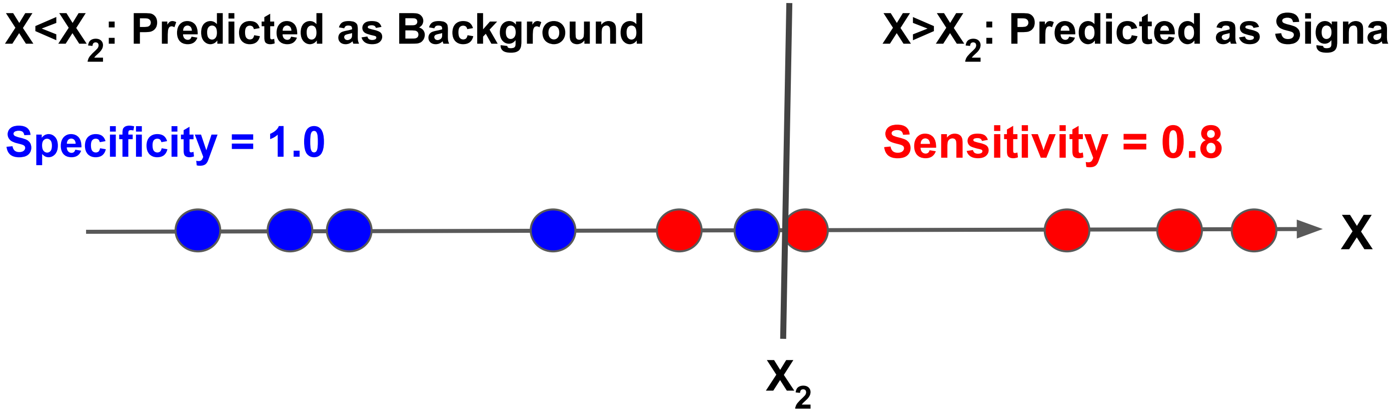
Performance of the Classifier

- E.g. Searching for rare signals → Correctly identify positives is (Sensitivity) more important→ Choose X_1



Performance of the Classifier

- E.g. Perform a precise measurement → Correctly identify negatives (Specificity) is more important → Choose X_2



From Wikipedia, the free encyclopedia

Sources: [1][2][3][4][5][6][7][8] **view** · talk · edit

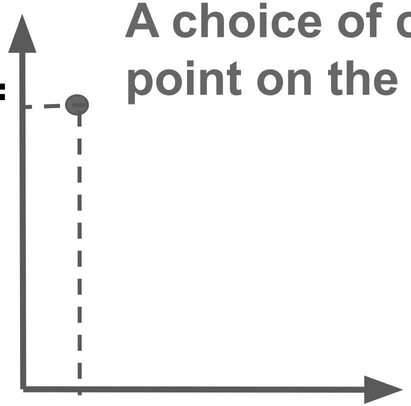
		Predicted condition			
	Total population $= P + N$	**Positive (PP)**	**Negative (PN)**	Informedness, bookmaker informedness (BM) $= TPR + TNR - 1$	Prevalence threshold (PT) $= \frac{\sqrt{TPR \times FPR} - FPR}{TPR - FPR}$
Actual condition	**Positive (P)**	**True positive (TP), hit**	**False negative (FN), type II error, miss, underestimation**	True positive rate (TPR), recall, sensitivity (SEN), probability of detection, hit rate, power $= \frac{TP}{P} = 1 - FNR$	False negative rate (FNR), miss rate $= \frac{FN}{P} = 1 - TPR$
	Negative (N)	**False positive (FP), type I error, false alarm, overestimation**	**True negative (TN), correct rejection**	False positive rate (FPR), probability of false alarm, fall-out $= \frac{FP}{N} = 1 - TNR$	True negative rate (TNR), specificity (SPC), selectivity $= \frac{TN}{N} = 1 - FPR$
	Prevalence $= \frac{P}{P + N}$	Positive predictive value (PPV), precision $= \frac{TP}{PP} = 1 - FDR$	False omission rate (FOR) $= \frac{FN}{PN} = 1 - NPV$	Positive likelihood ratio (LR+) $= \frac{TPR}{FPR}$	Negative likelihood ratio (LR−) $= \frac{FNR}{TNR}$
	Accuracy (ACC) $= \frac{TP + TN}{P + N}$	False discovery rate (FDR) $= \frac{FP}{PP} = 1 - PPV$	Negative predictive value (NPV) $= \frac{TN}{PN}$ $= 1 - FOR$	Markedness (MK), deltaP (Δp) $= PPV + NPV - 1$	Diagnostic odds ratio (DOR) $= \frac{LR+}{LR-}$
	Balanced accuracy (BA) $= \frac{TPR + TNR}{2}$	F_1 score $= \frac{2PPV \times TPR}{PPV + TPR} = \frac{2TP}{2TP + FP + FN}$	Fowlkes–Mallows index (FM) $= \sqrt{PPV \times TPR}$	Matthews correlation coefficient (MCC) $= \sqrt{TPR \times TNR \times PPV \times NPV} - \sqrt{FNR \times FPR \times FOR \times FDR}$	Threat score (TS), critical success index (CSI), Jaccard index $= \frac{TP}{TP + FN + FP}$

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) = $\dfrac{\textbf{True Positives}}{\textbf{Actual Positives}}$



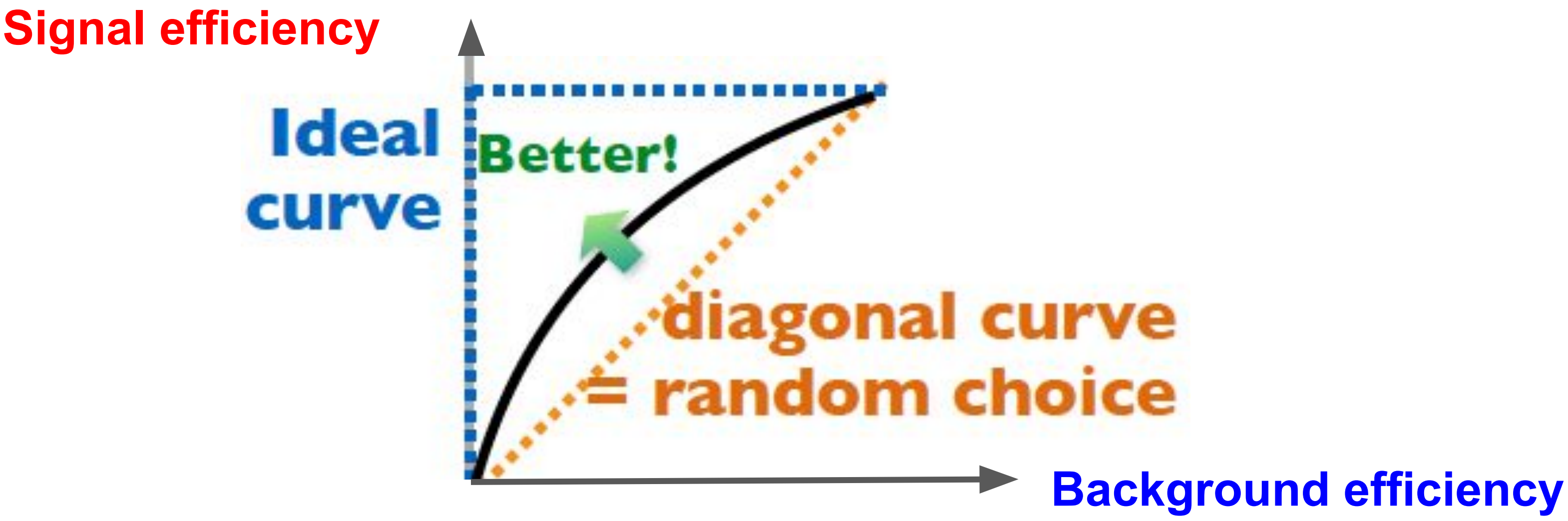
A choice of cut corresponds to a point on the plane

Background efficiency

(False Positive Rate) = $\dfrac{\textbf{False Positives}}{\textbf{Actual Negatives}}$

Performance of the Classifier

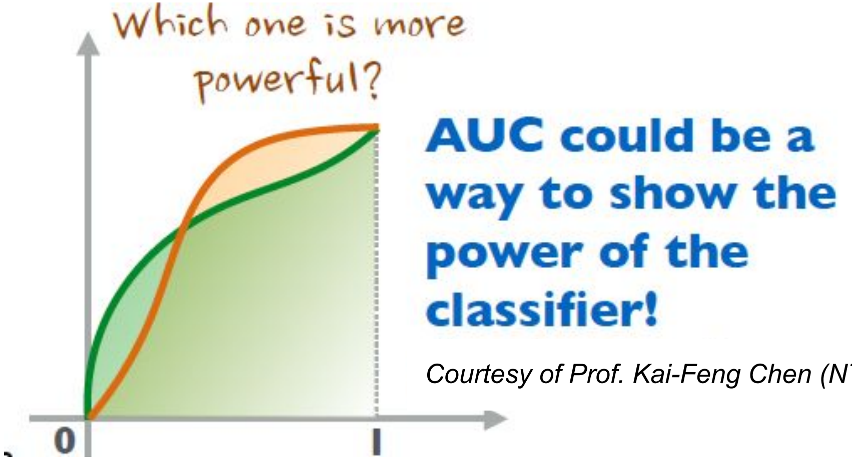
- The **ROC** curve illustrates the ability of the binary classifier when the discrimination threshold is varied.



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.

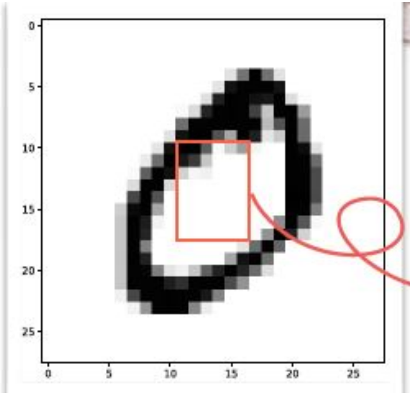
- Another way to estimate the performance is the **AUC** (area under the curve), which varies from 0.5 (diagonal line) to 1.0 (ideal case)



Courtesy of Prof. Kai-Feng Chen (NTU)

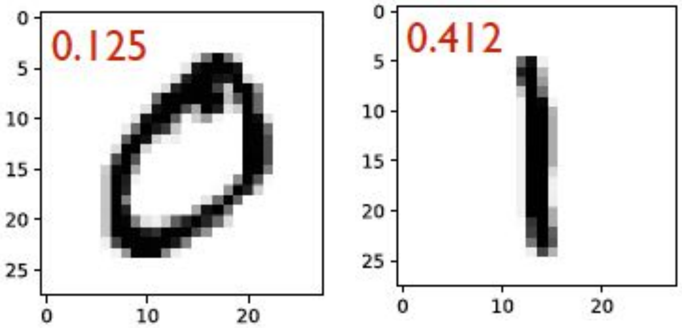
Binary Classification

- Can we do better by using more information, e.g. take the average density in the center of the image?



calculating the average of the centered 6×8 pixels, since there is obviously **a "hole" for the 0's**?

average here!



In-class exercise: Week-02

- Let's turn to this week's in-class exercise (Join Competition → code → inClass-exercise-02) and
  - Compute and plot the average center density
  - Compare the ROC curves of the two classifier (one with full average and another with center average)
  - 'Combine' the two distributions (full and center averages) and construct a more powerful classifier.

In-class exercise: Week-02

- We will use the Scikit-Learn, which is a machine learning library with Python: http://scikit-learn.org/

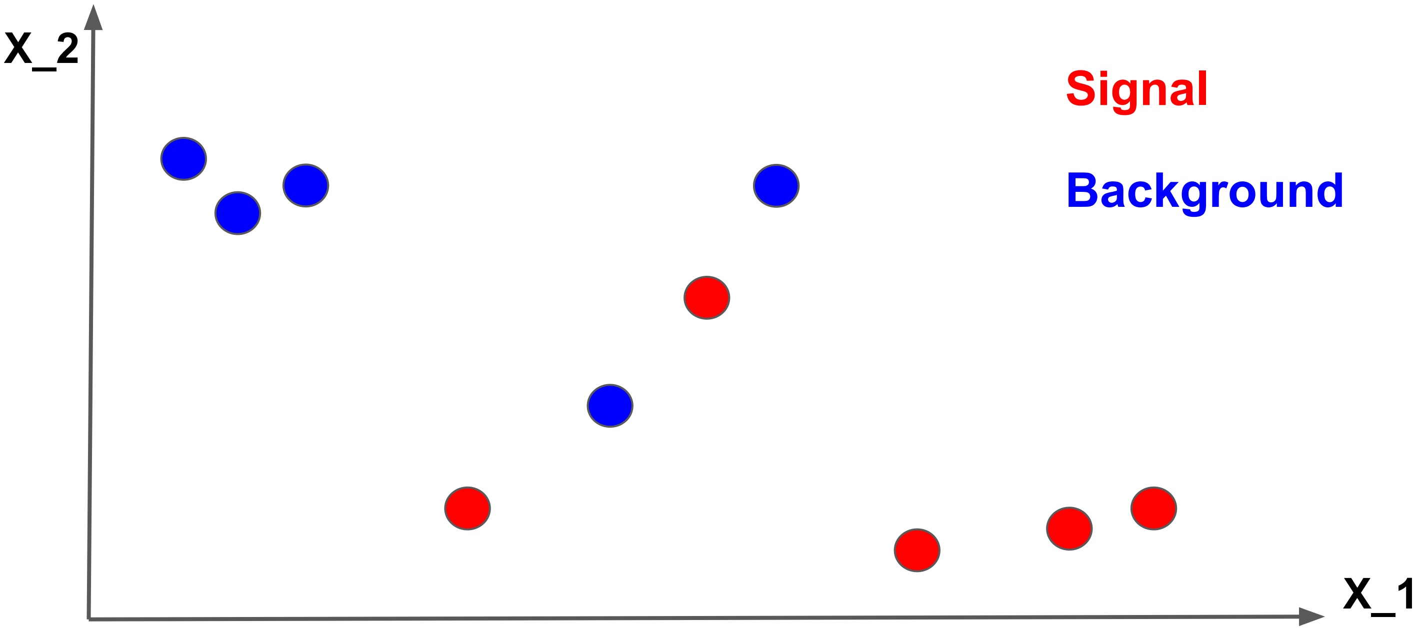
In-class exercise: Week-02

- How to combine the two distributions (full and center averages) and construct a new, more powerful **linear** classifier?

  → Linear Discriminant Analysis (LDA)

Linear Discriminant Analysis (LDA)

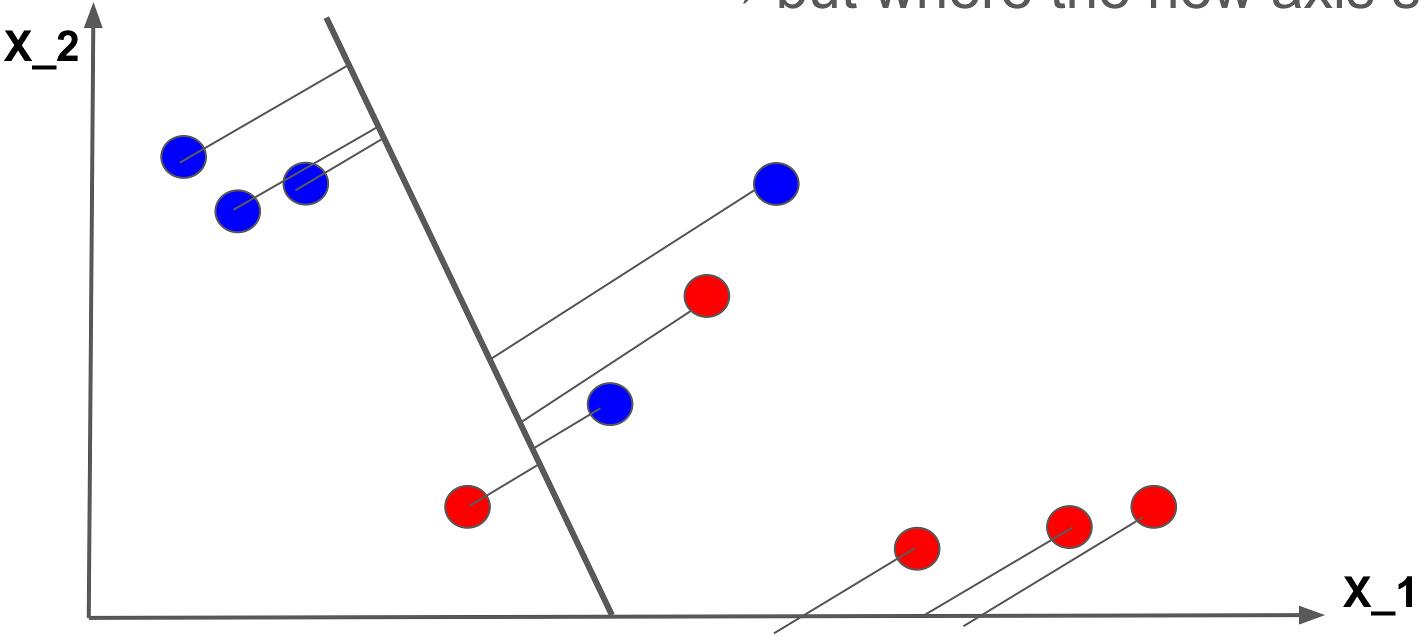
- Want to transform the 2D information into a number (1D)



Linear Discriminant Analysis (LDA)

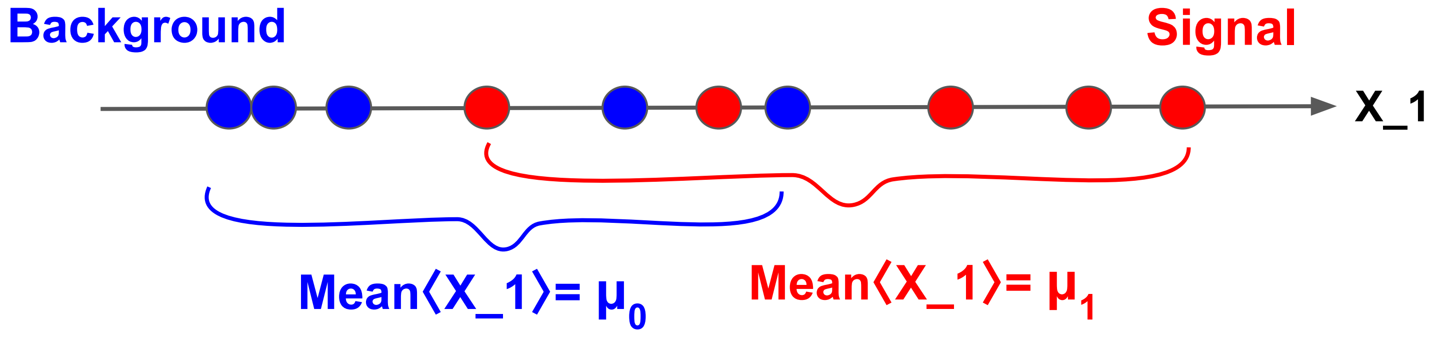
- LDA: find a new axis and projects all data onto the new axis

  → but where the new axis should be?



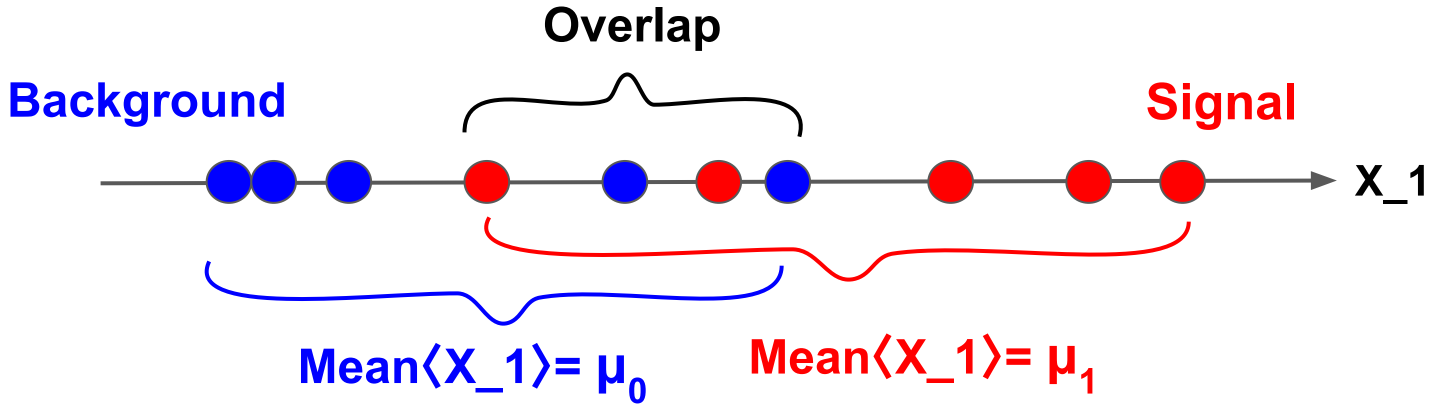
Linear Discriminant Analysis (LDA)

- Recall that a variable X_1 can be used to separate signals and backgrounds when they have different *means* of X_1 distributions



Linear Discriminant Analysis (LDA)

- There is an overlap of signals and backgrounds due to their *spreads* of X_1 distributions

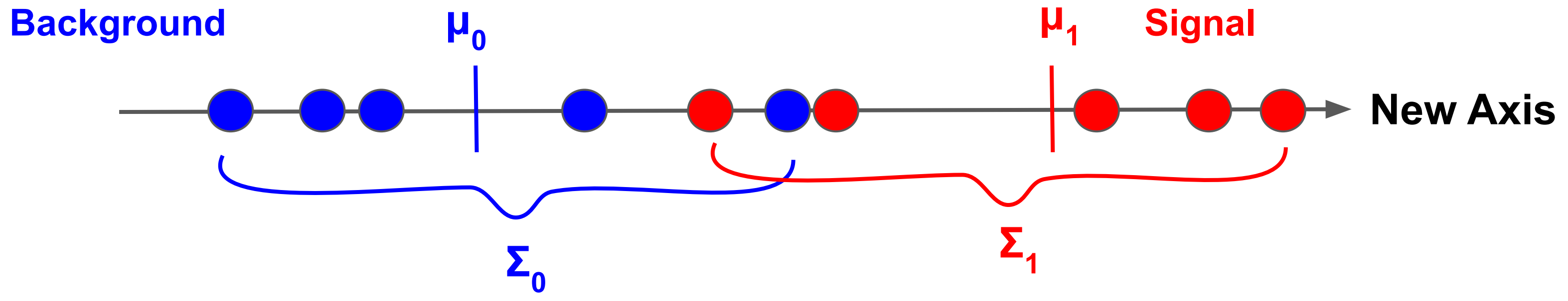


Linear Discriminant Analysis (LDA)

- Better separation power means
    - Larger difference of the means between the signal and the background
    - Smaller spreads of the distributions of the signal and the background

Linear Discriminant Analysis (LDA)

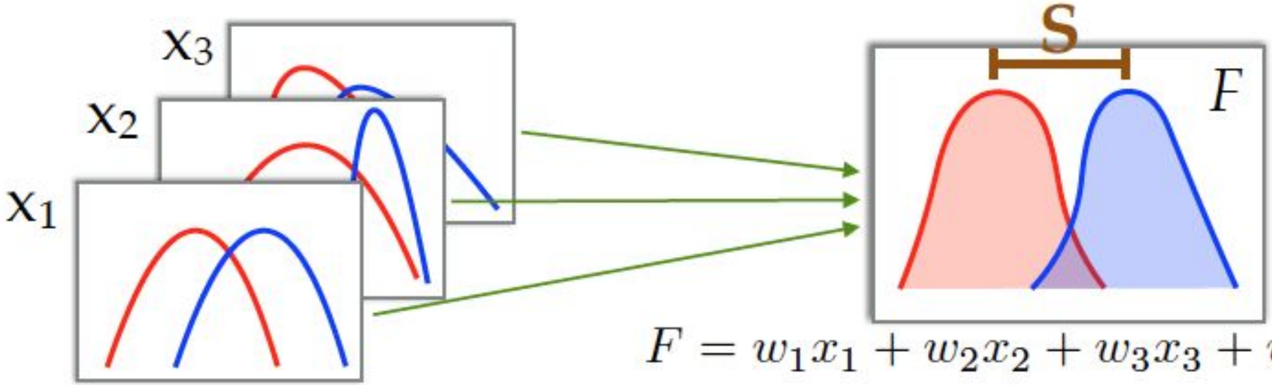
- LDA creates the new axis by combining all distributions with optimized weights which
  - Maximize the distance between the means (μ)
  - Minimize the spreads ("covariance" Σ) within each category



Performing LDA 'by hand': Fisher's discriminant

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 (***wi***) to maximize the separation ***S***.

LDA with Fisher's discriminant

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

distance of μ (large)

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

covariance Σ (small)

- 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)

- 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)

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.

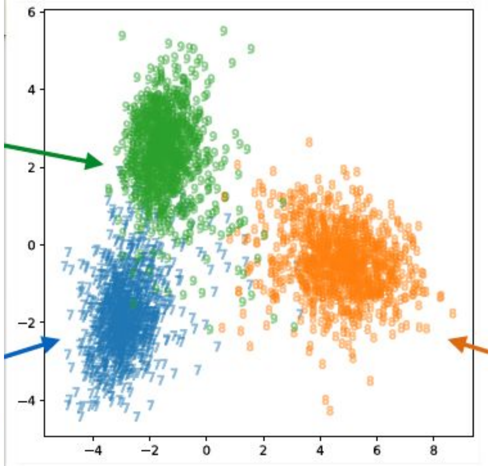
- The predicted class is the one with the highest probability.

In-class exercise: Week-02

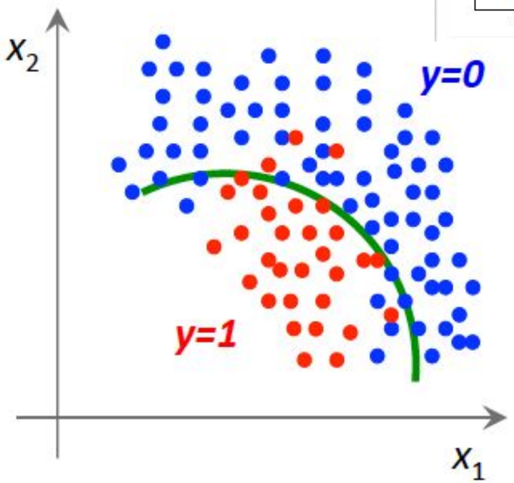
- Let’s go back to the in-class exercise and play with LDA using Scikit-learn!

Outlook

- What if we want to do multiclass classification?



- Or if we need a Non-linear Discriminant?



Nonlinear discriminant

Outlook

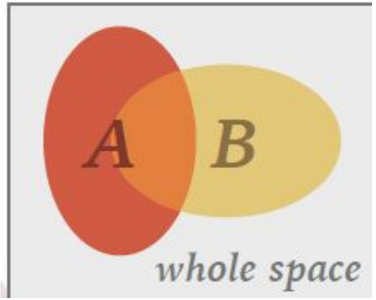
- What if we want to do multiclass classification?
- Or if we need a Non-linear Discriminant?
- We will make use of more sophisticated algorithms such as support vector machines (SVM), decision trees, neural networks... next time!

LDA in Scikit-learn

- The probability is calculated using Bayes' Theorem

- Then the conditional probability, $P(A \mid B)$, 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)