# Classification: Part I

Statistical Analysis and Document Mining

Spring 2021

Vasilii Feofanov

Université Grenoble Alpes

vasilii.feofanov@univ-grenoble-alpes.fr

## Outline

## Introduction

- During first weeks of the course, you studied the regression task, where the target variable is continuous, which represents usually in practice some measurement.

## Introduction

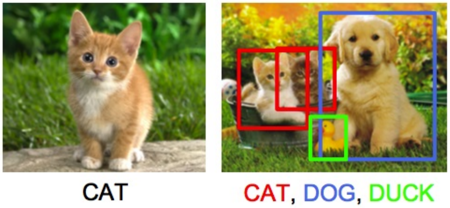
- During first weeks of the course, you studied the regression task, where the target variable is continuous, which represents usually in practice some measurement.
- However, in many applications, the target variable is discrete and indicates to which sub-category an observation belongs.

## Introduction

- During first weeks of the course, you studied the regression task, where the target variable is continuous, which represents usually in practice some measurement.
- However, in many applications, the target variable is discrete and indicates to which sub-category an observation belongs.
- For example:
  In pattern recognition, a task can be to recognize a handwritten digit;

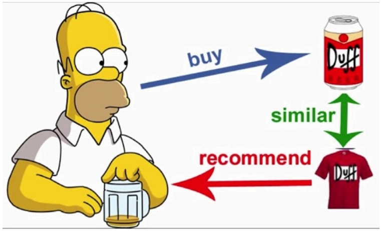
## Introduction

- During first weeks of the course, you studied the regression task, where the target variable is continuous, which represents usually in practice some measurement.
- However, in many applications, the target variable is discrete and indicates to which sub-category an observation belongs.
- For example:
  In visual tracking, it is important to recognize who (e.g. what animal) is on the image;



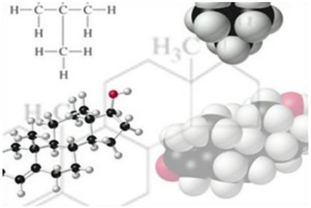
## Introduction

- During first weeks of the course, you studied the regression task, where the target variable is continuous, which represents usually in practice some measurement.
- However, in many applications, the target variable is discrete and indicates to which sub-category an observation belongs.
- For example:
  Some recommender systems are aimed to predict which product we recommend to a given customer;



## Introduction

- During first weeks of the course, you studied the regression task, where the target variable is continuous, which represents usually in practice some measurement.
- However, in many applications, the target variable is discrete and indicates to which sub-category an observation belongs.
- For example:
  In bioinformatics, one of the task is to predict a gene type given a DNA sequence.



## Classical Data Set: Iris Flower (1936)

(a) Iris Setosa

(b) Iris Versicolor

(c) Iris Virginica

Figure: The data set consists of 50 samples from each of 3 species of Iris. From each sample the length and the width of the sepals and petals were measured. Based on this, the goal is to build a model that distinguishes the species from each other.[1]

[1] `https://en.wikipedia.org/wiki/Iris_flower_data_set`

## Classification: Simple Formulation

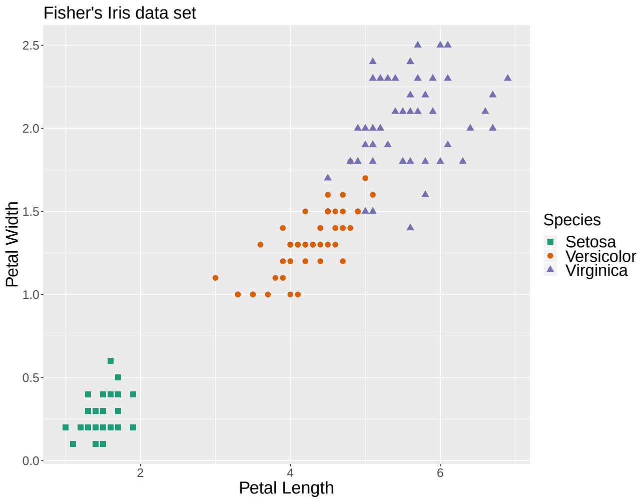
- Training set $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$: collected samples with class labels;
- A new data point $\mathbf{x}$ without any label;
- Build a classifier $h(\mathbf{x})$ that predicts $y$ as accurately as possible:

$$\begin{array}{cccccccc} & \text{Sepal.L} & \text{Sepal.W} & \text{Petal.L} & \text{Petal.W} & & \\ \mathbf{x} = & (6.2, & 2.8, & 5.6, & 2.4) & \xrightarrow{?} & y = \texttt{"Virginica"} \end{array}$$

*How to do that?*

## Iris Classification

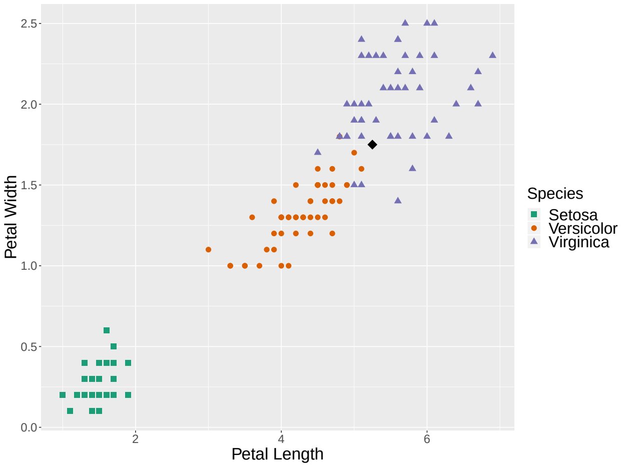
Let's try to predict the type of iris' species based on Petal Length and Petal Width. Below is the plot of our training data.



## Iris: New Data Point

Imagine we have a new observation (the black point). How we can predict its label given training observations?

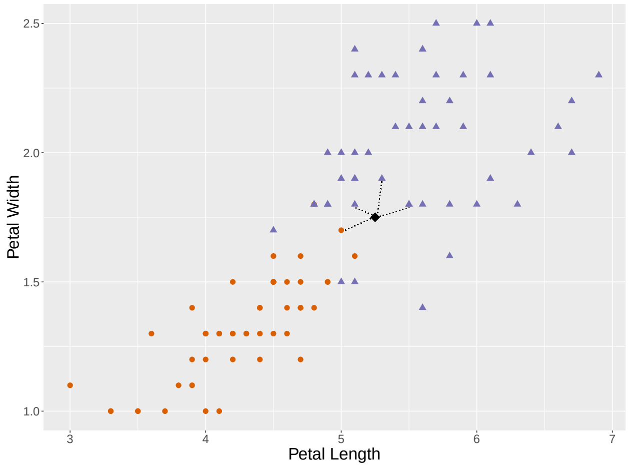
Fisher's Iris data set



## 4 Nearest Neighbours

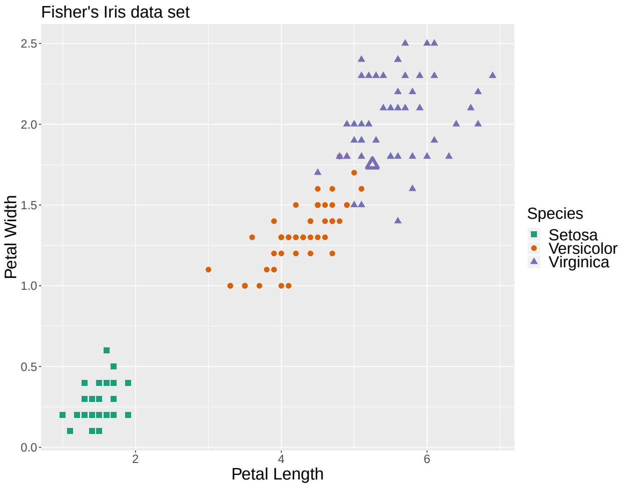
Assumption: examples from one class are close to each other in terms of distance. Find the 4 closest points to the black point.

Fisher's Iris data set



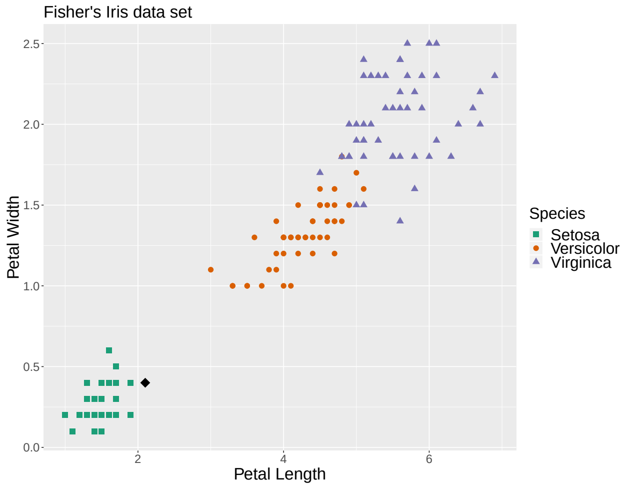
## 4NN Classification

We predict that our new example is Virginica, since the 3 of 4 neighbours were from this class.



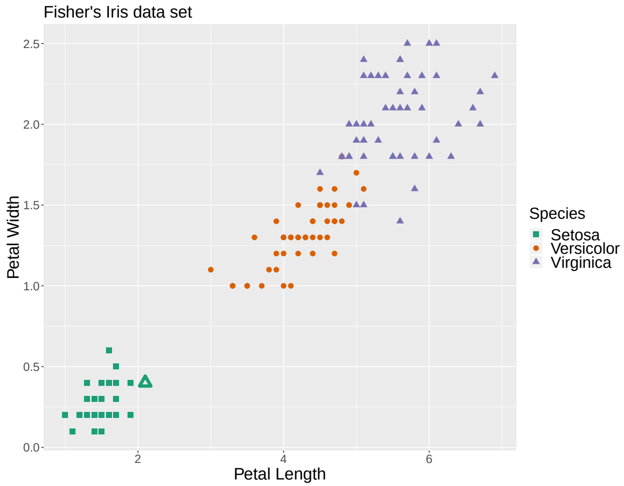
## Another Data Point

Another example: what class we predict this time?

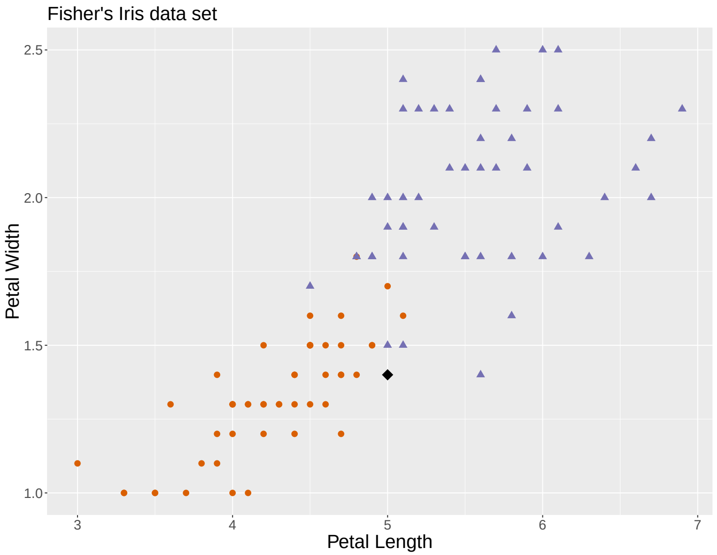


## And Its Prediction

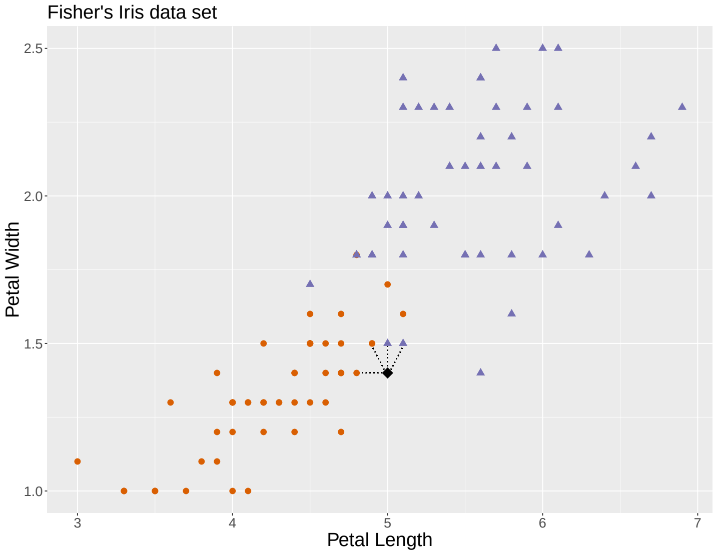
Yes, right! It belongs to Setosa.



# One More Example



# What to do in this case?



## kNN Algorithm

We can generalize this approach to $k$ nearest neighbours.

```
Algorithm k Nearest Neighbours (kNN)
Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$;
       Number of classes $K$;
       New data point $\mathbf{x}$.
1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \dots, n\}$.
2. Find $k$ closest training examples to $\mathbf{x}$:
   $J \subset \{1, \dots, n\}$  s.t.  $|J| = k$;
       $\forall j \in J, \forall t \in \{1, \dots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \leq \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.
Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\dots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

## kNN Algorithm

```
Algorithm k Nearest Neighbours (kNN)
    Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$;
           Number of classes $K$;
           New data point $\mathbf{x}$.
    1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \dots, n\}$.
    2. Find $k$ closest training examples to $\mathbf{x}$:

    $J \subset \{1, \dots, n\}$   s.t.   $|J| = k$;
                   $\forall j \in J, \forall t \in \{1, \dots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \leq \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.

    Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\dots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

## kNN Algorithm

```
Algorithm k Nearest Neighbours (kNN)
    Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$;
           Number of classes $K$;
           New data point $\mathbf{x}$.
    1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \dots, n\}$.
    2. Find $k$ closest training examples to $\mathbf{x}$:
       $J \subset \{1, \dots, n\}$  s.t.  $|J| = k$;
                                         $\forall j \in J, \forall t \in \{1, \dots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \le \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.
    Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\dots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

- What is the time complexity of the algorithm?

# kNN Algorithm

```
Algorithm k Nearest Neighbours (kNN)
  Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$;
         Number of classes $K$;
         New data point $\mathbf{x}$.
  1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \ldots, n\}$.
  2. Find $k$ closest training examples to $\mathbf{x}$:

  $J \subset \{1, \ldots, n\}$   s.t.   $|J| = k$;
                                  $\forall j \in J, \forall t \in \{1, \ldots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \leq \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.

  Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\ldots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

- How does the choice of $k$ affect kNN?

## kNN Algorithm

```
Algorithm k Nearest Neighbours (kNN)
  Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^{n}$;
         Number of classes $K$;
         New data point $\mathbf{x}$.
  1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \dots, n\}$.
  2. Find $k$ closest training examples to $\mathbf{x}$:
     $J \subset \{1, \dots, n\}$  s.t.  $|J| = k$;
         $\forall j \in J, \forall t \in \{1, \dots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \le \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.
  Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\dots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

- How does the choice of distance metric affect the algorithm?

## kNN Algorithm

```
Algorithm k Nearest Neighbours (kNN)
  Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$;
         Number of classes $K$;
         New data point $\mathbf{x}$.
  1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \dots, n\}$.
  2. Find $k$ closest training examples to $\mathbf{x}$:
     $J \subset \{1, \dots, n\}$ s.t. $|J| = k$;
         $\forall j \in J, \forall t \in \{1, \dots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \leq \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.
  Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\dots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

- Is the nearest neighbours approach applicable for the regression task?

## kNN Algorithm

```
Algorithm k Nearest Neighbours (kNN)
Input: Training set $\{\mathbf{x}_i, y_i\}_{i=1}^n$;
       Number of classes $K$;
       New data point $\mathbf{x}$.
1. Compute distance $\mathrm{d}(\mathbf{x}, \mathbf{x}_i)$ for $i = \{1, \dots, n\}$.
2. Find $k$ closest training examples to $\mathbf{x}$:
   $J \subset \{1, \dots, n\}$ s.t. $|J| = k$;
       $\forall j \in J, \forall t \in \{1, \dots, n\} \setminus J : \mathrm{d}(\mathbf{x}, \mathbf{x}_j) \leq \mathrm{d}(\mathbf{x}, \mathbf{x}_t)$.
Output: Majority class $h(\mathbf{x}) = \mathrm{argmax}_{c=1,\dots,K} \sum_{j \in J} \mathbb{I}(y_j = c)$.
```

- kNN is an *instance-based* learning algorithm. What is the main drawback of such methods?

## Outline

## Discriminant Analysis (Fisher, 1936)

A historically first classification algorithm was proposed by Ronald Fisher who studied the problem in the probabilistic way.

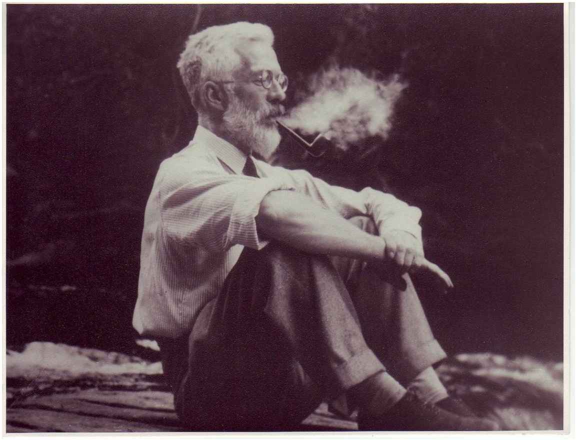


Figure: Ronald A. Fisher in 1946.

## Classification Task: More Formally

- *Input space:* $\mathcal{X} \subseteq \mathbb{R}^d$;
- Output space: $\mathcal{Y} = \{-1, +1\}$ (binary classification),
  $\mathcal{Y} = \{1, \dots, K\}$ (multi-class classification);

- *Assumption:* all $(\mathbf{X}, Y) \in \mathcal{X} \times \mathcal{Y}$ are **i.i.d.** with respect to a fixed unknown probability distribution $P(\mathbf{X}, Y)$;
- *Sample Data:* we observe $S = \{\mathbf{x}_i, y_i\}_{i=1}^n$;

## Classification Task: More Formally

- *Input space:* $\mathcal{X} \subseteq \mathbb{R}^d$;
- Output space: $\mathcal{Y} = \{-1, +1\}$ (binary classification),
  $\mathcal{Y} = \{1, \dots, K\}$ (multi-class classification);

- *Assumption:* all $(\mathbf{X}, Y) \in \mathcal{X} \times \mathcal{Y}$ are **i.i.d.** with respect to a fixed unknown probability distribution $P(\mathbf{X}, Y)$;
- *Sample Data:* we observe $S = \{\mathbf{x}_i, y_i\}_{i=1}^n$;

- *Loss Function:* $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}^+$;
- *Target*: minimise the *risk* $R^\ell(h) := \mathbb{E}_{P(\mathbf{X},Y)}\, \ell(h(\mathbf{X}), Y)$, where $h : \mathcal{X} \to \mathcal{Y}$ is a classifier.

## 0/1 Loss Function

0/1 loss function has the following view:

$$\ell^{0/1}(h(\mathbf{x}), y) = \mathbb{I}(h(\mathbf{x}) \neq y) = \begin{cases} 1, & \text{if } h(\mathbf{x}) \neq y; \\ 0, & \text{if } h(\mathbf{x}) = y. \end{cases}$$

## 0/1 Loss Function

0/1 loss function has the following view:

$$\ell^{0/1}(h(\mathbf{x}), y) = \mathbb{I}(h(\mathbf{x}) \neq y) = \begin{cases} 1, & \text{if } h(\mathbf{x}) \neq y; \\ 0, & \text{if } h(\mathbf{x}) = y. \end{cases}$$

In this case, the risk is written as:

$$\begin{aligned} R(h) &= P(h(\mathbf{X}) \neq Y) \qquad \leftarrow \text{Also called } \textit{misclassification probability} \\ &= \sum_{c \in \{1,\dots,K\}} P(Y = c) P(h(\mathbf{X}) \neq c | Y = c). \end{aligned}$$

## 0/1 Loss Function

0/1 loss function has the following view:

$$\ell^{0/1}(h(\mathbf{x}), y) = \mathbb{I}(h(\mathbf{x}) \neq y) = \begin{cases} 1, & \text{if } h(\mathbf{x}) \neq y; \\ 0, & \text{if } h(\mathbf{x}) = y. \end{cases}$$

In this case, the risk is written as:

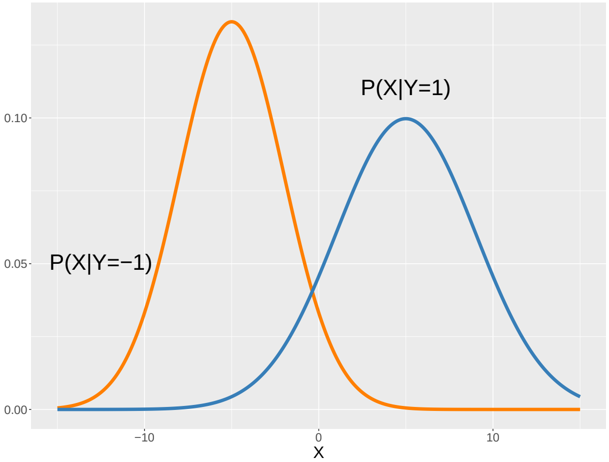
$$\begin{aligned} R(h) &= P(h(\mathbf{X}) \neq Y) \qquad \leftarrow \text{Also called } \textit{misclassification probability} \\ &= \sum_{c \in \{1,\dots,K\}} P(Y = c) P(h(\mathbf{X}) \neq c | Y = c). \end{aligned}$$

In the binary case, it is represented as:

$$R(h) = P(Y = -1)P(h(\mathbf{X}) = 1 | Y = -1) + P(Y = 1)P(h(\mathbf{X}) = -1 | Y = 1)$$

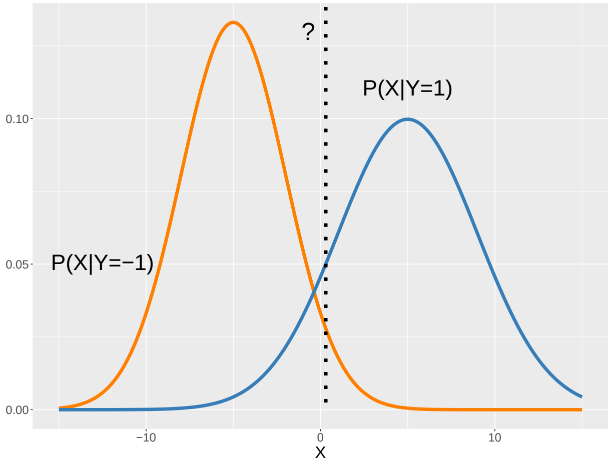
## Classification in Terms of Distributions

Our training data is drawn from a mixture of distributions, where $P(\mathbf{X}|Y = c)$ is a distribution of the class $c$.



## Classification in Terms of Distributions

*Question:* how to partition data so that the misclassification probability is small?



## Bayes' Rule

$$\overset{\text{Posterior}}{P(Y|\mathbf{X})} = \frac{\overset{\text{Likelihood}}{P(\mathbf{X}|Y)}\,\overset{\text{Class Prior}}{P(Y)}}{\underset{\text{Evidence}}{P(\mathbf{X})}}$$

## Maximum A Posteriori Rule

*Idea:* Classify $\mathbf{x}$ to a class with the highest posterior probability:

$$h_B(\mathbf{x}) := \operatorname*{argmax}_{y \in \mathcal{Y}} P(Y = y | \mathbf{X} = \mathbf{x}).$$

This is equivalent to:

$$h_B(\mathbf{x}) \propto \operatorname*{argmax}_{y \in \mathcal{Y}} P(\mathbf{X} = \mathbf{x} | Y = y) P(Y = y).$$

The method is often called the *Bayes* classifier.

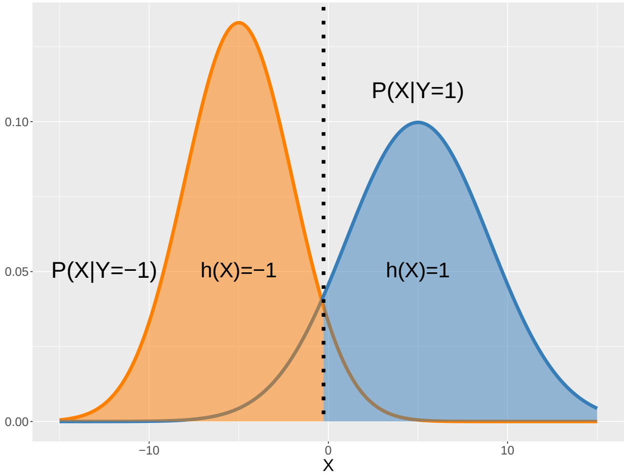
## Bayes Classifier in the Binary Case

In the binary case ($\mathcal{Y} = \{-1, +1\}$), the Bayes classifier is usually defined in the following way:

$$h_B(\mathbf{x}) = \begin{cases} +1, & \text{if } P(Y = +1|\mathbf{X} = \mathbf{x}) \geq P(Y = -1|\mathbf{X} = \mathbf{x}); \\ -1, & \text{if } P(Y = +1|\mathbf{X} = \mathbf{x}) < P(Y = -1|\mathbf{X} = \mathbf{x}). \end{cases}$$

## Particular Case: Maximum Likelihood Estimator

If we assume that priors are equal $(P(Y = +1) = P(Y = -1))$, the Bayes classifier separates the class distributions as follows:



## Bayes Classifier Is Optimal

**Theorem**

*Suppose $P(Y)$ and $P(\mathbf{X}|Y)$ are given. Then the Bayes classifier yields the minimum of the misclassification error.*

**Exercise:** Prove this theorem.

## Bayes Classifier Is Optimal

**Theorem**

*Suppose $P(Y)$ and $P(\mathbf{X}|Y)$ are given. Then the Bayes classifier yields the minimum of the misclassification error.*

**Exercise:** Prove this theorem.

1. $P(h(\mathbf{X}) \neq Y) = \int P(h(\mathbf{X}) \neq Y | \mathbf{X} = \mathbf{x}) dP(\mathbf{X} = \mathbf{x})$. Then, what is the value of $P(h_B(\mathbf{X}) \neq Y | \mathbf{X} = \mathbf{x})$?

## Bayes Classifier Is Optimal

**Theorem**

*Suppose $P(Y)$ and $P(\mathbf{X}|Y)$ are given. Then the Bayes classifier yields the minimum of the misclassification error.*

**Exercise:** Prove this theorem.

1. $P(h(\mathbf{X}) \neq Y) = \int P(h(\mathbf{X}) \neq Y|\mathbf{X} = \mathbf{x})dP(\mathbf{X} = \mathbf{x})$. Then, what is the value of $P(h_B(\mathbf{X}) \neq Y|\mathbf{X} = \mathbf{x})$?
2. $P(h_B(\mathbf{x}) \neq Y|\mathbf{X} = \mathbf{x}) = 1 - \max_{c\in\mathcal{Y}} P(Y = c|\mathbf{X} = \mathbf{x})$. Does this prove the theorem?

## Bayes Classifier: Discussion

In practice, we don't have any information about data distribution.

- Can we estimate $P(Y)$?

## Bayes Classifier: Discussion

In practice, we don't have any information about data distribution.

- Can we estimate $P(Y)$?
  - We can. $P(Y=c) \approx \frac{\sum_{i=1}^{n} \mathbb{I}(y_i = c)}{n}$.

## Bayes Classifier: Discussion

In practice, we don't have any information about data distribution.

- Can we estimate $P(Y)$?
  - We can. $P(Y = c) \approx \frac{\sum_{i=1}^{n} \mathbb{I}(y_i = c)}{n}$.
- Can we estimate $P(\mathbf{X}|Y)$?

## Bayes Classifier: Discussion

In practice, we don't have any information about data distribution.

- Can we estimate $P(Y)$?
  - We can. $P(Y=c) \approx \frac{\sum_{i=1}^{n} \mathbb{I}(y_i=c)}{n}$.
- Can we estimate $P(\mathbf{X}|Y)$?
  - Hmm... Not clear. Maybe we need to make additional assumptions.

## Additional Assumptions

- *Assumption 1:* Observations from a class $c \in \mathcal{Y}$ are normally distributed $[\mathbf{X}|Y = c] \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$.
    - Remember the formula of the multivariate normal distribution.

## Additional Assumptions

- *Assumption 1:* Observations from a class $c \in \mathcal{Y}$ are normally distributed $[\mathbf{X}|Y = c] \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$.
  - Remember the formula of the multivariate normal distribution.

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}\text{det}(\boldsymbol{\Sigma})^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^\intercal \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)}.$$

## Additional Assumptions

- *Assumption 1:* Observations from a class $c \in \mathcal{Y}$ are normally distributed $[\mathbf{X}|Y = c] \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$.
    - Remember the formula of the multivariate normal distribution.

$$f(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}\text{det}(\boldsymbol{\Sigma})^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^\intercal \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)}.$$

- *Assumption 2:* The covariance matrices of the classes are equal:

$$\boldsymbol{\Sigma}_1 = \cdots = \boldsymbol{\Sigma}_K = \boldsymbol{\Sigma}.$$

## Linear Discriminant Analysis

Due to the logarithm properties, the Bayes classifier can be written as follows:

$$h_B(\mathbf{x}) = \operatorname*{argmax}_{c \in \mathcal{Y}} [\ln P(\mathbf{X} = \mathbf{x} | Y = c) + \ln P(Y = c)].$$

## Linear Discriminant Analysis

Due to the logarithm properties, the Bayes classifier can be written as follows:

$$h_B(\mathbf{x}) = \operatorname*{argmax}_{c \in \mathcal{Y}} [\ln P(\mathbf{X} = \mathbf{x} | Y = c) + \ln P(Y = c)].$$

Taking into account Assumption 1 and Assumption 2, we derive the algorithm called *Linear Discriminant Analysis (LDA)*:

$$h_{LDA}(\mathbf{x}) = \operatorname*{argmax}_{c \in \mathcal{Y}} \delta_c(\mathbf{x}),$$
$$\delta_c(\mathbf{x}) = \boldsymbol{\mu}_c^\intercal \boldsymbol{\Sigma}^{-1} \mathbf{x} - \frac{1}{2} \boldsymbol{\mu}_c^\intercal \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu}_c + \ln P(Y = c);$$

$\delta_c$ is usually called the *discriminant function*.

## LDA in the Binary Case

In the binary case, the decision rule can be also written as:

$$h_{LDA}(\mathbf{x}) = \begin{cases} +1, & \text{if } \delta_{+1} - \delta_{-1} \geq 0; \\ -1, & \text{otherwise,} \end{cases}$$

where $\delta_{+1} - \delta_{-1}$ is:

$$(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_{+1} + \boldsymbol{\mu}_{-1}) + \ln \frac{P(Y=+1)}{P(Y=-1)}.$$

## LDA in the Binary Case

In the binary case, the decision rule can be also written as:

$$h_{LDA}(\mathbf{x}) = \begin{cases} +1, & \text{if } \delta_{+1} - \delta_{-1} \geq 0; \\ -1, & \text{otherwise,} \end{cases}$$

where $\delta_{+1} - \delta_{-1}$ is:

$$(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_{+1} + \boldsymbol{\mu}_{-1}) + \ln \frac{P(Y=+1)}{P(Y=-1)}.$$

- Why it is called "linear"?

## LDA in the Binary Case

In the binary case, the decision rule can be also written as:

$$h_{LDA}(\mathbf{x}) = \begin{cases} +1, & \text{if } \delta_{+1} - \delta_{-1} \geq 0; \\ -1, & \text{otherwise}, \end{cases}$$

where $\delta_{+1} - \delta_{-1}$ is:

$$\overbrace{(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}}^{\mathbf{a}^\intercal} \mathbf{x} - \overbrace{\frac{1}{2}(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_{+1} + \boldsymbol{\mu}_{-1}) + \ln \frac{P(Y=+1)}{P(Y=-1)}}^{\mathbf{b}}.$$

- Why it is called "linear"?
- How many parameters we need to estimate?

## LDA in the Binary Case

In the binary case, the decision rule can be also written as:

$$h_{LDA}(\mathbf{x}) = \begin{cases} +1, & \text{if } \delta_{+1} - \delta_{-1} \geq 0; \\ -1, & \text{otherwise,} \end{cases}$$

where $\delta_{+1} - \delta_{-1}$ is:

$$\overbrace{(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^{\intercal} \boldsymbol{\Sigma}^{-1}}^{\mathbf{a}^{\intercal}} \mathbf{x} - \overbrace{\frac{1}{2}(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^{\intercal} \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_{+1} + \boldsymbol{\mu}_{-1}) + \ln \frac{P(Y=+1)}{P(Y=-1)}}^{\mathbf{b}}.$$

- Why it is called "linear"?
- How many parameters we need to estimate?
- What happens when Assumption 1, 2 are violated?

## LDA in the Binary Case

In the binary case, the decision rule can be also written as:

$$h_{LDA}(\mathbf{x}) = \begin{cases} +1, & \text{if } \delta_{+1} - \delta_{-1} \geq 0; \\ -1, & \text{otherwise,} \end{cases}$$

where $\delta_{+1} - \delta_{-1}$ is:

$$\overbrace{(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}}^{\mathbf{a}^\intercal} \mathbf{x} - \overbrace{\frac{1}{2}(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_{+1} + \boldsymbol{\mu}_{-1}) + \ln \frac{P(Y=+1)}{P(Y=-1)}}^{\mathbf{b}}.$$

- Why it is called "linear"?
- How many parameters we need to estimate?
- What happens when Assumption 1, 2 are violated?
- What is the time complexity during training phase?

## LDA in the Binary Case

In the binary case, the decision rule can be also written as:

$$h_{LDA}(\mathbf{x}) = \begin{cases} +1, & \text{if } \delta_{+1} - \delta_{-1} \geq 0; \\ -1, & \text{otherwise,} \end{cases}$$

where $\delta_{+1} - \delta_{-1}$ is:

$$\overbrace{(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}}^{\mathbf{a}^\intercal} \mathbf{x} - \overbrace{\frac{1}{2}(\boldsymbol{\mu}_{+1} - \boldsymbol{\mu}_{-1})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_{+1} + \boldsymbol{\mu}_{-1}) + \ln \frac{P(Y=+1)}{P(Y=-1)}}^{\mathbf{b}}.$$

- Why it is called "linear"?
- How many parameters we need to estimate?
- What happens when Assumption 1, 2 are violated?
- What is the time complexity during training phase?
- What is the time complexity to predict a label for new $\mathbf{x}$?