Minería de Datos

La Técnica k-NN

Máster en Ciencia de Datos

Con la colaboración de:

Rodrigo García Manzanas (rodrigo.manzanas@unican.es)
Departamento de Matemática Aplicada y Ciencias de la Computación
Universidad de Cantabria

Contents

- k-NN for classification
- k-NN for regression
- The curse of dimensionality
- k-NN in the climate science

Aprendizaje supervisado

- Clasificación (Cla)
- Regresión/predicción (Reg)

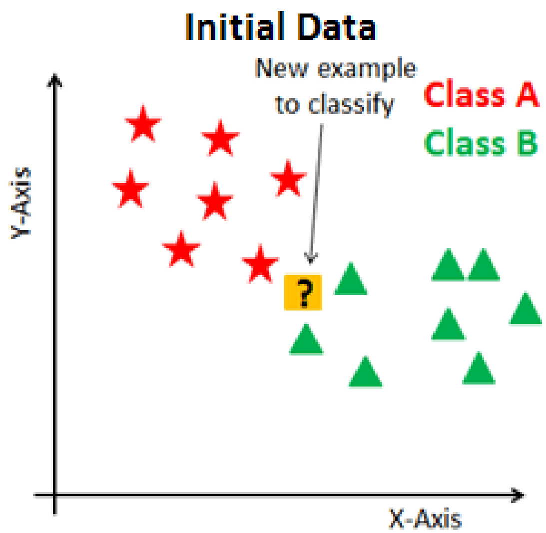
- Regresión logística (Cla)
- Regresión lineal (Reg)
- **k-NN (Cla, Reg)**
- Árboles de decisión (Cla, Reg)
- Métodos de ensembles: Random forests (Cla,Reg); AdaBoost (Cla); GBoost (Cla, Reg)
- Métodos de kernels (Cla, Reg)
- Máquinas de vector soporte (Cla, Reg)
- Redes neuronales (Cla, Reg)
- Redes probabilísticas (Cla, Reg)
- etc.

Aprendizaje no supervisado

- Asociación (Aso)
- Clustering (Clu)
- Reducción de la dimensionalidad (RDim)

- Reglas de asociación (Aso): Algoritmo Apriori, Algoritmo Eclat
- Clustering jerárquico (Clu): Dendograma
- Clustering no jerárquico (Clu): k-means
- Reducción de la dimensionalidad lineal (RDim): PCA, LDA
- Reducción de la dimensionalidad no lineal (RDim): MDS, MMF, Isomap, LLE, SNE
- Redes probabilísticas (Rdim)
- etc.

Minería de Datos Estadística Aprendizaje Automático Aprendizaje Automático II

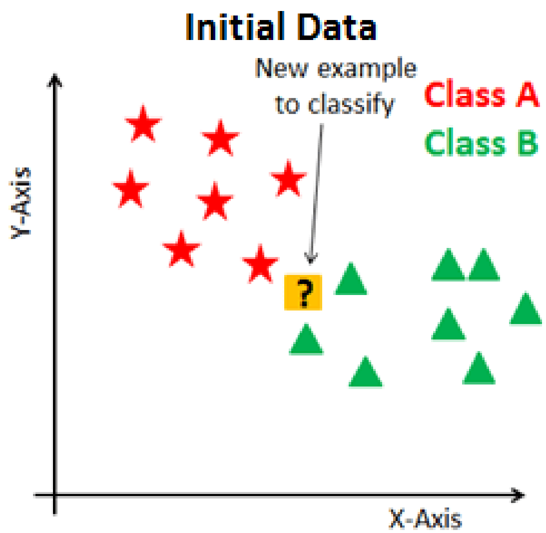


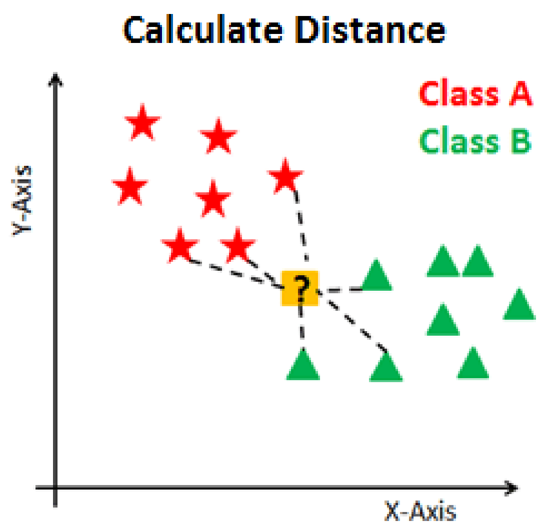
In **classification** problems...

Aim:
To **classify** a **discrete** target variable, based on some similarity measure in the predictors' space.

In **classification** problems...

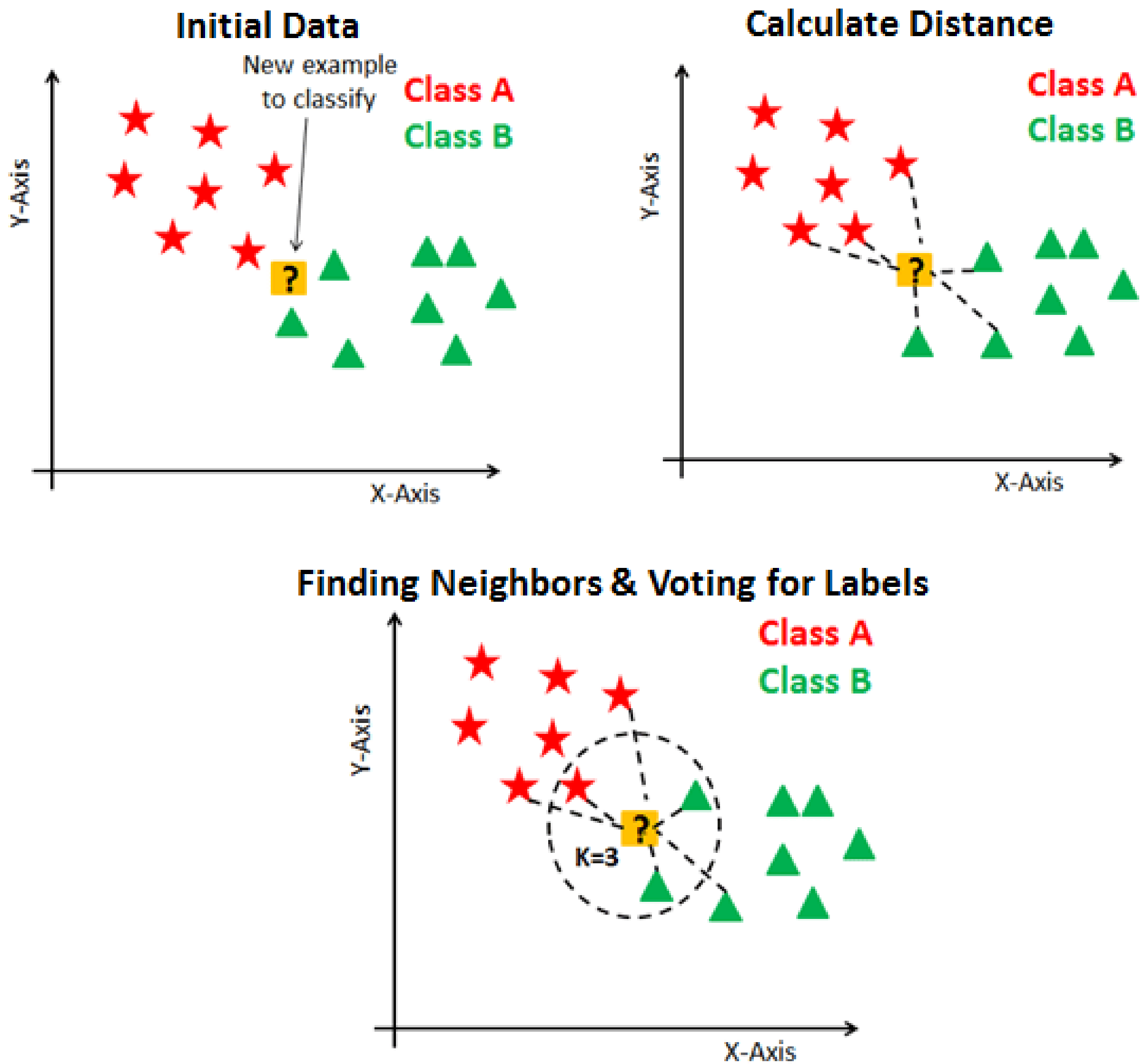
Aim:
To classify a discrete target variable, based on some **similarity measure in the predictors' space**.





In **classification** problems...

Aim:
To classify a discrete target variable, based on some similarity measure in the predictors' space.



The Core Idea

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width		Species
1	5.1	3.5	1.4	0.2		setosa
2	4.9	3.0	1.4	0.2		setosa
3	4.7	3.2	1.3	0.2		setosa
...						
55	6.5	2.8	4.6	1.5		versicolor
56	5.7	2.8	4.5	1.3		versicolor
57	6.3	3.3	4.7	1.6		versicolor
...						
148	6.5	3.0	5.2	2.0		virginica
149	6.2	3.4	5.4	2.3		virginica
150	5.9	3.0	5.1	1.8		virginica

The Core Idea

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
...					
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
...					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica
151	5.4	2.7	4.6	1.4	?

New instance to be classified

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
...					
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
...					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica
151	5.4	2.7	4.6	1.4	?

New instance to be classified

STEP 1: Computing distances

$$d_{151,1} = \sqrt{(5.4-5.1)^2+(2.7-3.5)^2+(4.6-1.4)^2+(1.4-0.2)^2} = 3.52$$

$$d_{151,2} = 3.47$$
$$d_{151,3} = 3.62$$
$$\dots$$
$$d_{151,55} = 1.11$$
$$d_{151,56} = 0.35$$
$$d_{151,57} = 1.10$$
$$\dots$$
$$d_{151,148} = 1.42$$
$$d_{151,149} = 1.61$$
$$d_{151,150} = 0.87$$

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
...					
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
...					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica
151	5.4	2.7	4.6	1.4	?

New instance to be classified

STEP 1: Computing distances

$$d_{151,1}=\sqrt{(5.4-5.1)^2+(2.7-3.5)^2+(4.6-1.4)^2+(1.4-0.2)^2}=3.52$$

$d_{151,2}=3.47$

$d_{151,3}=3.62$

...

$d_{151,55}=1.11$

$d_{151,56}=0.35$

$d_{151,57}=1.10$

...

$d_{151,148}=1.42$

$d_{151,149}=1.61$

$d_{151,150}=0.87$

STEP 2: Ordering distances

$d_{151,56}=0.35$

$d_{151,150}=0.87$

$d_{151,57}=1.10$

$d_{151,55}=1.11$

$d_{151,148}=1.42$

$d_{151,149}=1.61$

$d_{151,2}=3.47$

$d_{151,1}=3.52$

$d_{151,3}=3.62$

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
…					
55	6.5	2.8	4.6	1.5	versicolor
56	**5.7**	**2.8**	**4.5**	**1.3**	**versicolor**
57	**6.3**	**3.3**	**4.7**	**1.6**	**versicolor**
…					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	**5.9**	**3.0**	**5.1**	**1.8**	**virginica**
151	5.4	2.7	4.6	1.4	?

New instance to be classified

STEP 1: Computing distances

$$d_{151,1} = \sqrt{(5.4-5.1)^2 + (2.7-3.5)^2 + (4.6-1.4)^2 + (1.4-0.2)^2} = 3.52$$

$d_{151,2} = 3.47$
$d_{151,3} = 3.62$
…
$d_{151,55} = 1.11$
$d_{151,56} = 0.35$
$d_{151,57} = 1.10$
…
$d_{151,148} = 1.42$
$d_{151,149} = 1.61$
$d_{151,150} = 0.87$

STEP 2: Ordering distances

$\boxed{d_{151,56} = 0.35}$
$\boxed{d_{151,150} = 0.87}$
$\boxed{d_{151,57} = 1.10}$
$d_{151,55} = 1.11$
$d_{151,148} = 1.42$
$d_{151,149} = 1.61$
$d_{151,2} = 3.47$
$d_{151,1} = 3.52$
$d_{151,3} = 3.62$

k = 3

STEP 3: NN-based classification

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
...					
55	6.5	2.8	4.6	1.5	versicolor
56	**5.7**	**2.8**	**4.5**	**1.3**	**versicolor**
57	**6.3**	**3.3**	**4.7**	**1.6**	**versicolor**
...					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	**5.9**	**3.0**	**5.1**	**1.8**	**virginica**
151	5.4	2.7	4.6	1.4	**versicolor**

STEP 1: Computing distances

$$d_{151,1} = \sqrt{(5.4-5.1)^2 + (2.7-3.5)^2 + (4.6-1.4)^2 + (1.4-0.2)^2} = 3.52$$

$d_{151,2} = 3.47$
$d_{151,3} = 3.62$
...
$d_{151,55} = 1.11$
$d_{151,56} = 0.35$
$d_{151,57} = 1.10$
...
$d_{151,148} = 1.42$
$d_{151,149} = 1.61$
$d_{151,150} = 0.87$

STEP 2: Ordering distances

$\boxed{d_{151,56} = 0.35 \quad d_{151,150} = 0.87 \quad d_{151,57} = 1.10}$ $\xrightarrow{k=3}$ *151: versicolor*

$d_{151,55} = 1.11$
$d_{151,148} = 1.42$
$d_{151,149} = 1.61$
$d_{151,2} = 3.47$
$d_{151,1} = 3.52$
$d_{151,3} = 3.62$

STEP 3: NN-based classification

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
...					
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
...					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica
151	5.4	2.7	4.6	1.4	**versicolor**

Pros:

- Easy to understand
- Non-parametric: No assumption is made on the underlying data distribution
- Versatile: Classification and regression problems
- Good performance (benchmark)

STEP 1: Computing distances

$$d_{151,1}=\sqrt{(5.4-5.1)^2+(2.7-3.5)^2+(4.6-1.4)^2+(1.4-0.2)^2}=3.52$$

$d_{151,2}=3.47$

$d_{151,3}=3.62$

...

$d_{151,55}=1.11$

$d_{151,56}=0.35$

$d_{151,57}=1.10$

...

$d_{151,148}=1.42$

$d_{151,149}=1.61$

$d_{151,150}=0.87$

STEP 2: Ordering distances

$$\boxed{\begin{array}{l} d_{151,56}=0.35 \\ d_{151,150}=0.87 \\ d_{151,57}=1.10 \end{array}}$$

$d_{151,55}=1.11$

$d_{151,148}=1.42$

$d_{151,149}=1.61$

$d_{151,2}=3.47$

$d_{151,1}=3.52$

$d_{151,3}=3.62$

k = 3 → *151: versicolor*

STEP 3: NN-based classification

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
…					
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
…					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica
151	5.4	2.7	4.6	1.4	**versicolor**

STEP 1: Computing distances

$$d_{151,1} = \sqrt{(5.4-5.1)^2 + (2.7-3.5)^2 + (4.6-1.4)^2 + (1.4-0.2)^2} = 3.52$$

$d_{151,2} = 3.47$
$d_{151,3} = 3.62$
…
$d_{151,55} = 1.11$
$d_{151,56} = 0.35$
$d_{151,57} = 1.10$
…
$d_{151,148} = 1.42$
$d_{151,149} = 1.61$
$d_{151,150} = 0.87$

STEP 2: Ordering distances

$\boxed{d_{151,56} = 0.35 \quad d_{151,150} = 0.87 \quad d_{151,57} = 1.10}$ $\xrightarrow{k = 3}$ *151: versicolor*

$d_{151,55} = 1.11$
$d_{151,148} = 1.42$
$d_{151,149} = 1.61$
$d_{151,2} = 3.47$
$d_{151,1} = 3.52$
$d_{151,3} = 3.62$

STEP 3: NN-based classification

Pros:
- Easy to understand
- Non-parametric: No assumption is made on the underlying data distribution
- Versatile: Classification and regression problems
- Good performance (benchmark)

Cons:
- Non-generative: Computationally expensive
- Sensitive to scale of the data
- Sensitive to outliers
- Performance can be severely degraded in high dimensional problems

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
...					
55	6.5	2.8	4.6	1.5	versicolor
56	5.7	2.8	4.5	1.3	versicolor
57	6.3	3.3	4.7	1.6	versicolor
...					
148	6.5	3.0	5.2	2.0	virginica
149	6.2	3.4	5.4	2.3	virginica
150	5.9	3.0	5.1	1.8	virginica
151	5.4	2.7	4.6	1.4	**versicolor**

STEP 1: Computing distances

$$d_{151,1}=\sqrt{(5.4-5.1)^2+(2.7-3.5)^2+(4.6-1.4)^2+(1.4-0.2)^2}=3.52$$

$d_{151,2}=3.47$
$d_{151,3}=3.62$
…
$d_{151,55}=1.11$
$d_{151,56}=0.35$
$d_{151,57}=1.10$
…
$d_{151,148}=1.42$
$d_{151,149}=1.61$
$d_{151,150}=0.87$

STEP 2: Ordering distances

$d_{151,56}=0.35$
$d_{151,150}=0.87$
$d_{151,57}=1.10$
$d_{151,55}=1.11$
$d_{151,148}=1.42$
$d_{151,149}=1.61$
$d_{151,2}=3.47$
$d_{151,1}=3.52$
$d_{151,3}=3.62$

k = 3 → *151: versicolor*

STEP 3: NN-based classification

Pros:
- Easy to understand
- Non-parametric: No assumption is made on the underlying data distribution
- Versatile: Classification and regression problems
- Good performance (benchmark)

Cons:
- Non-generative: Computationally expensive
- Sensitive to scale of the data
- Sensitive to outliers
- Performance can be severely degraded in high dimensional problems

Applications:
- Economic sciences
- Political sciences
- Genetics
- Image recognition
- Climate

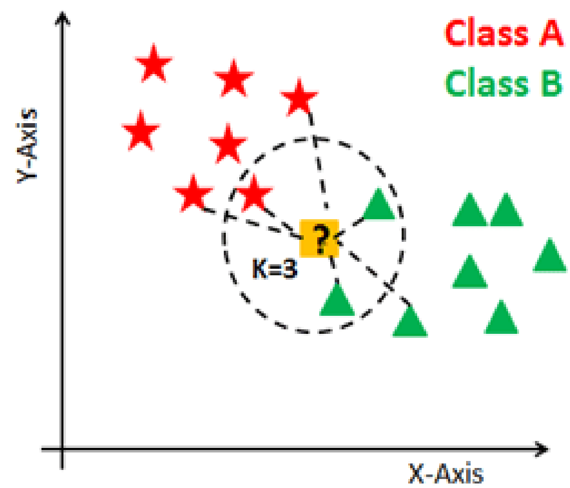
- **Distance metric**
  Different distance metrics can be used, depending on the application.

- **Number of neighbors (*k*)**
  This is the unique model parameter. Must be carefully chosen.

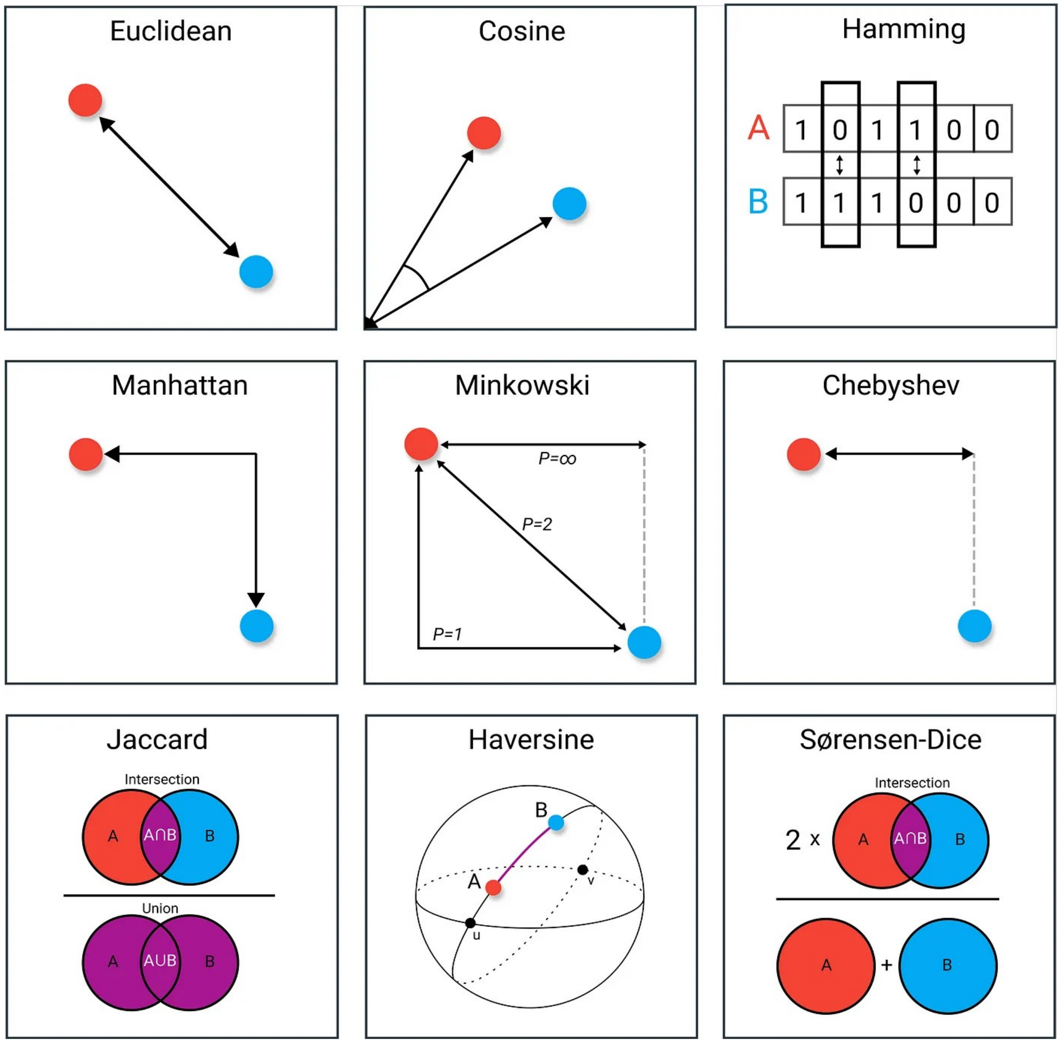
- **Inference criterion**
  - Majority vote
  - Random
  - etc.

Deterministic result

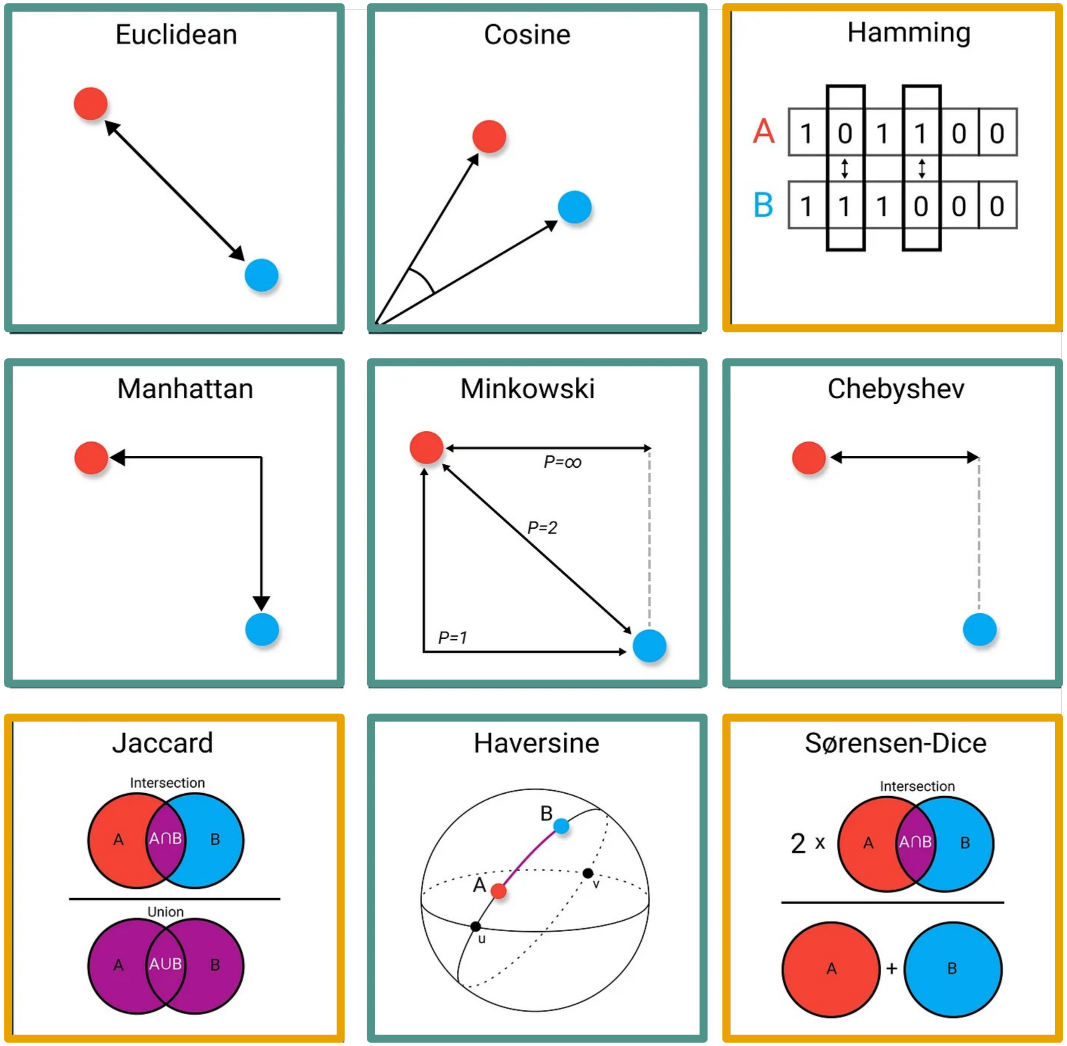
Stochastic result



There is a large number of available distance metrics: see Prasath et al. 2019 and this interesting post for details



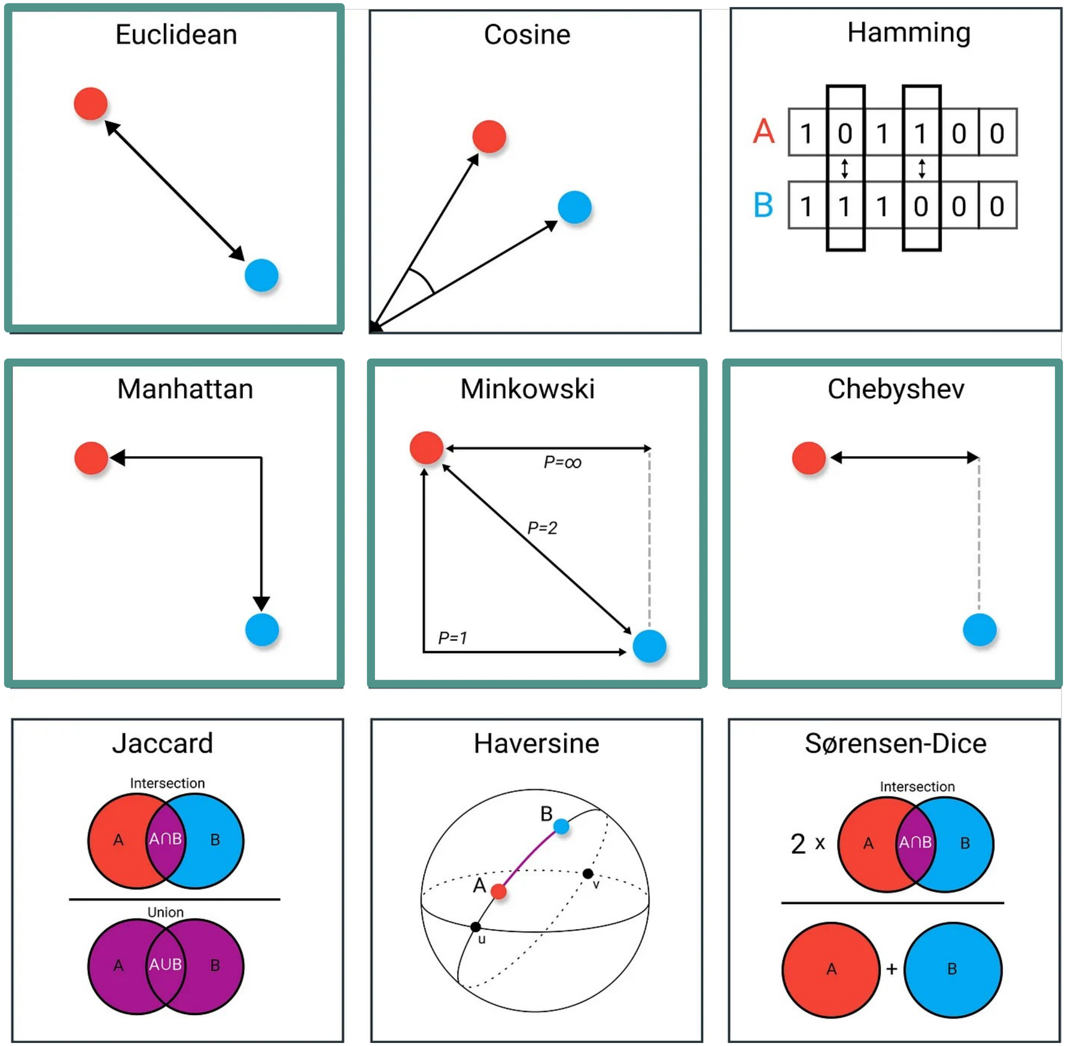
There is a large number of available distance metrics: see Prasath et al. 2019 and this interesting post for details



Geometric nature

Topological nature

There is a large number of available distance metrics: see Prasath et al. 2019 and this interesting post for details



Minkowsky-based geometric distances

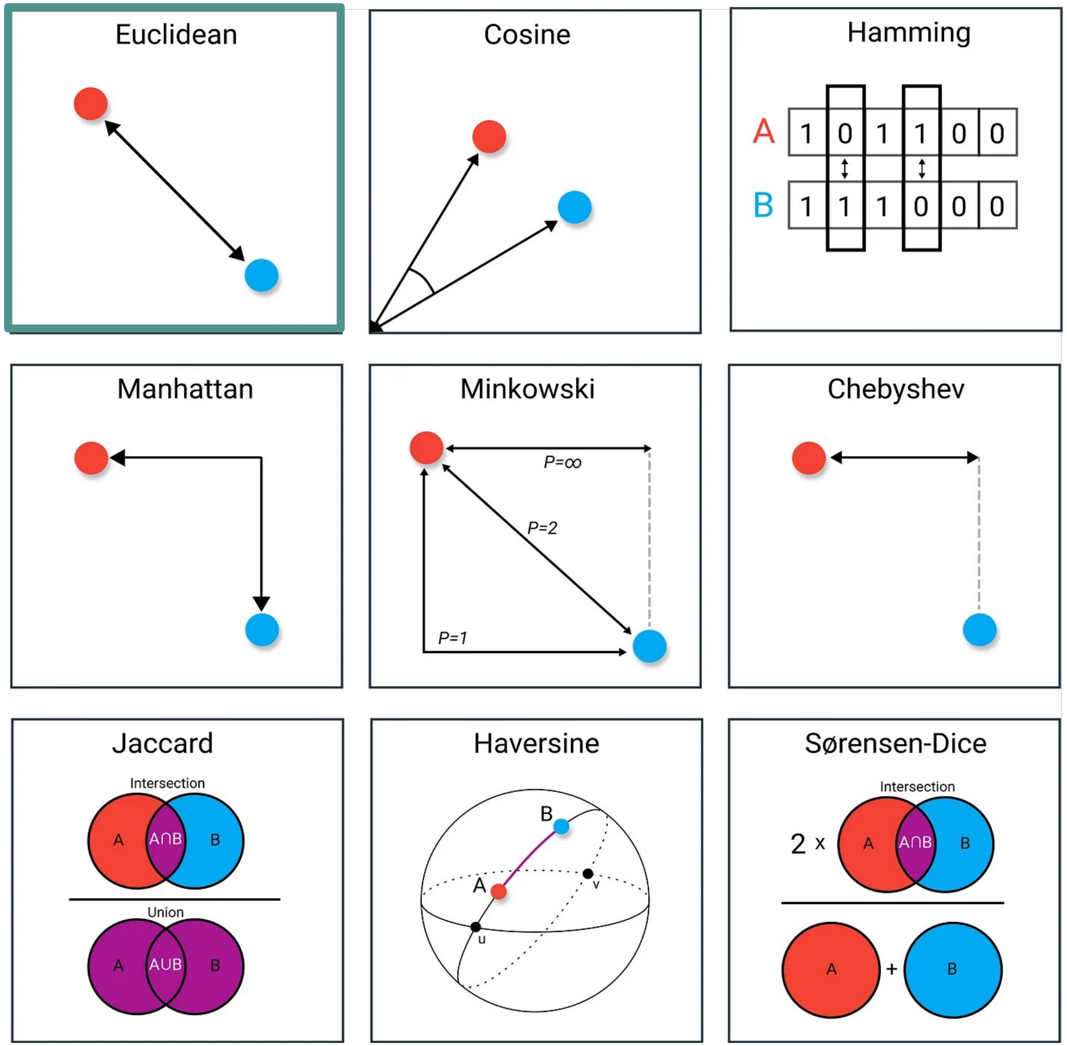
$$D_{Minkowsky}(x,y) = \left(\sum_{i=1}^{n}(x_i - y_i)^p\right)^{1/p}$$

$$D_{Manhattan}(x,y) = \sum_{i=1}^{n}|x_i - y_i|$$

$$D_{Euclidean}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

$$D_{Chebychev}(x,y) = \max_{i=1}^{n}|x_i - y_i|$$

There is a large number of available distance metrics: see Prasath et al. 2019 and this interesting post for details



$$D_{Minkowsky}(x,y) = \left(\sum_{i=1}^{n}(x_i - y_i)^p\right)^{1/p}$$

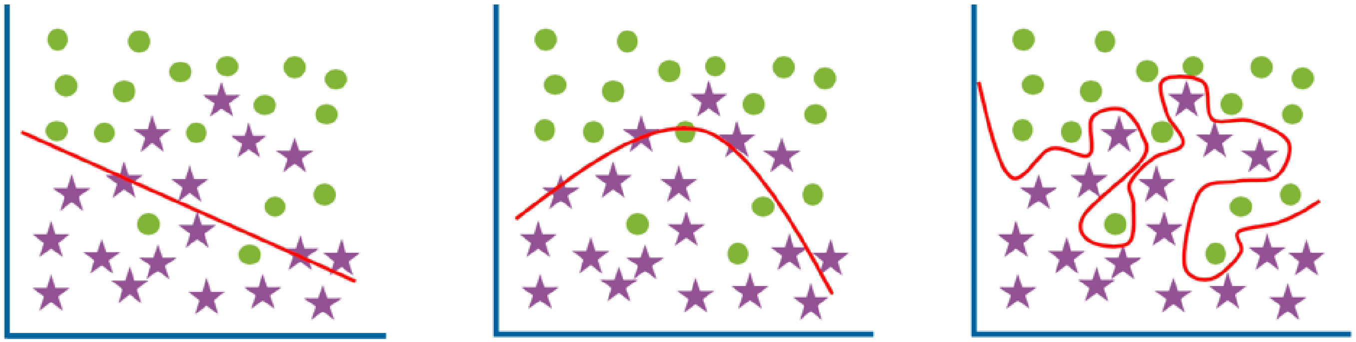
$$D_{Manhattan}(x,y) = \sum_{i=1}^{n}|x_i - y_i|$$

Most commonly used

$$\boxed{D_{Euclidean}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}}$$

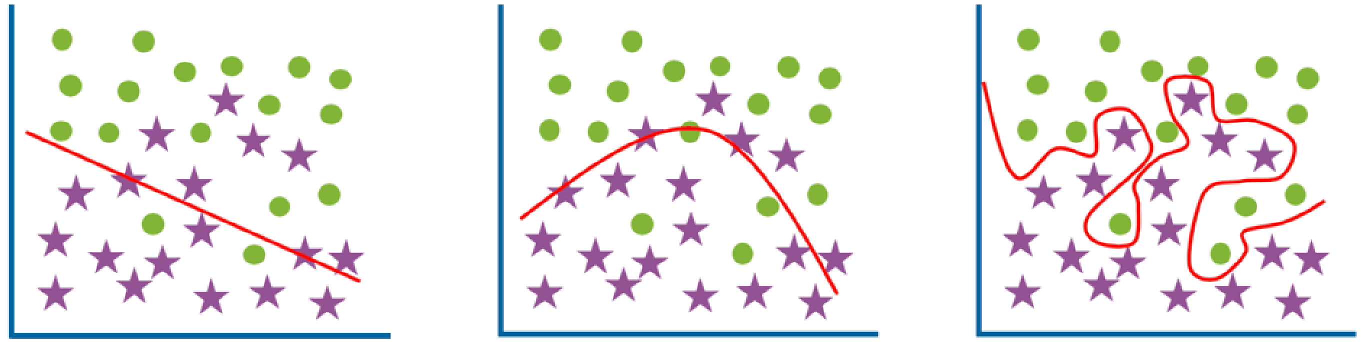
$$D_{Chebychev}(x,y) = \max_{i=1}^{n}|x_i - y_i|$$

Which model would you prefer?



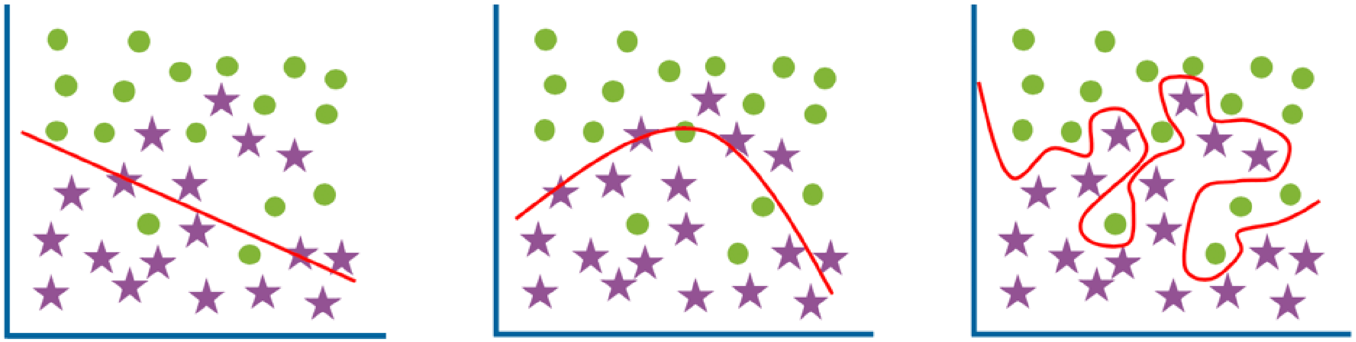
Classification borders obtained for different values of *k*

Which model would you prefer?



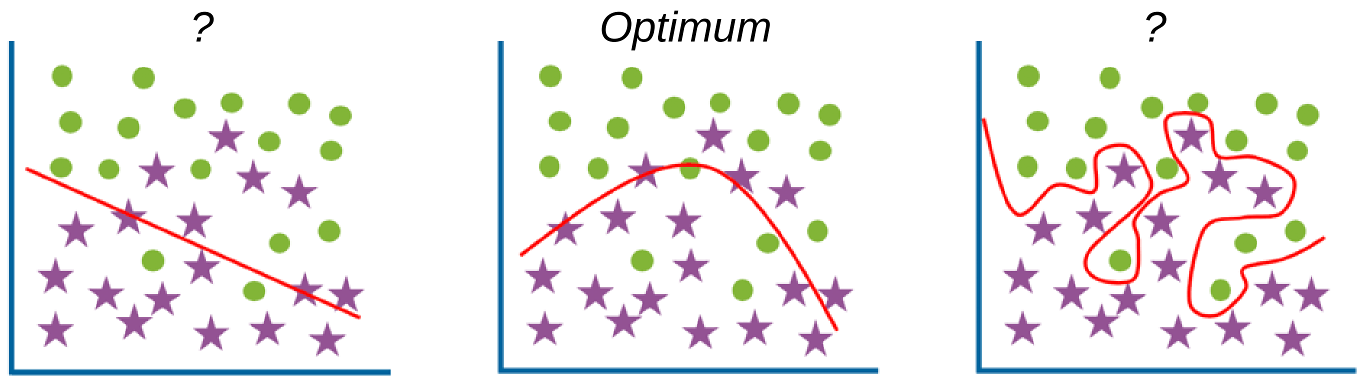
Resulting model	Value of *k*
Overfitted	*Too large k*
Underfitted	*Optimum k*
Optimum	*Too small k*

Which model would you prefer?



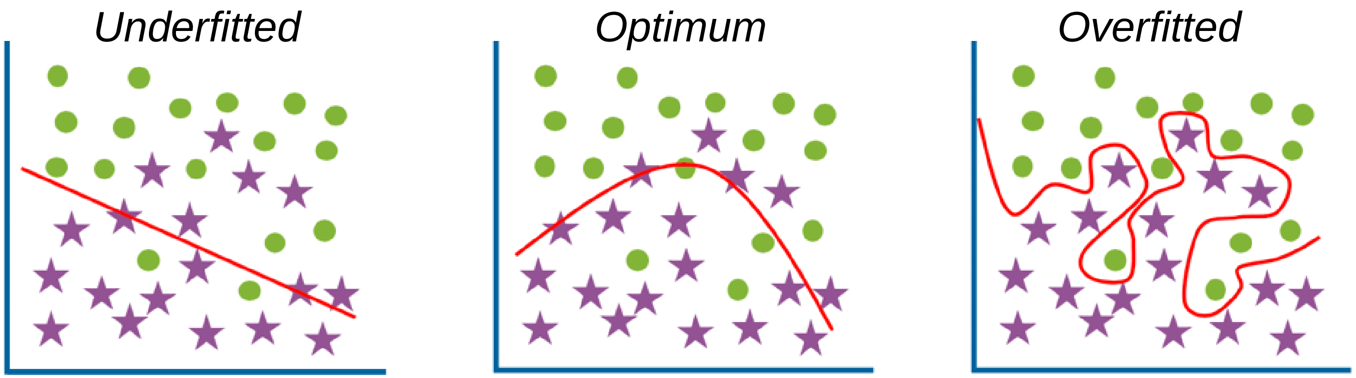
Resulting model	Value of *k*
Overfitted	*Too large k*
Underfitted	*Optimum k*
Optimum	*Too small k*

Which model would you prefer?



Resulting model	Value of *k*
Overfitted	*Too large k*
Underfitted	*Optimum k*
	Too small k

Which model would you prefer?



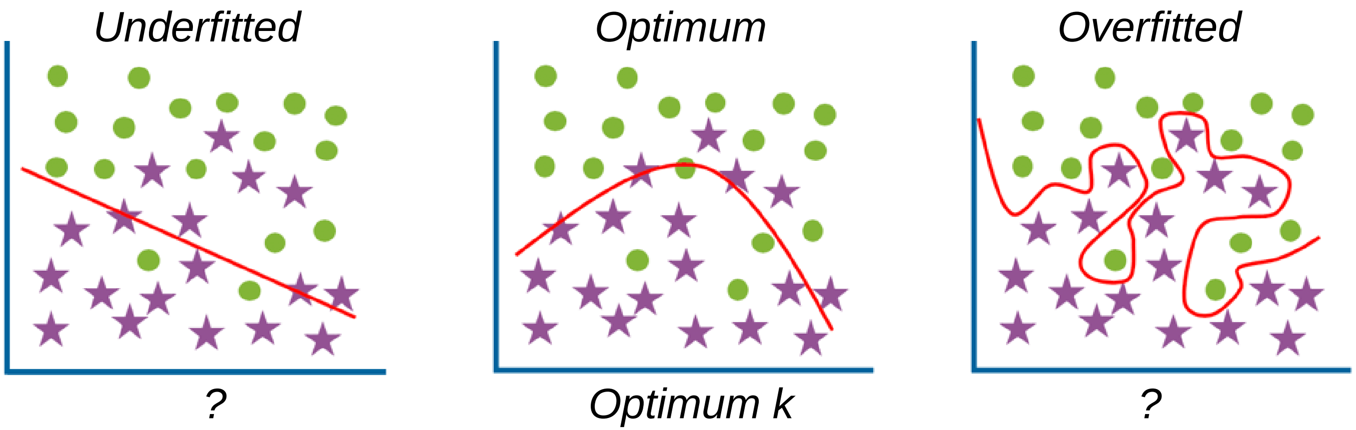
Value of k

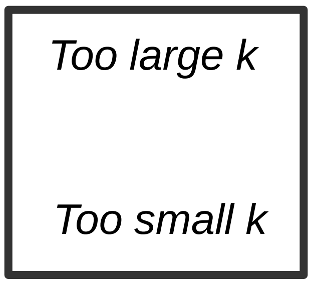
Too large k

Optimum k

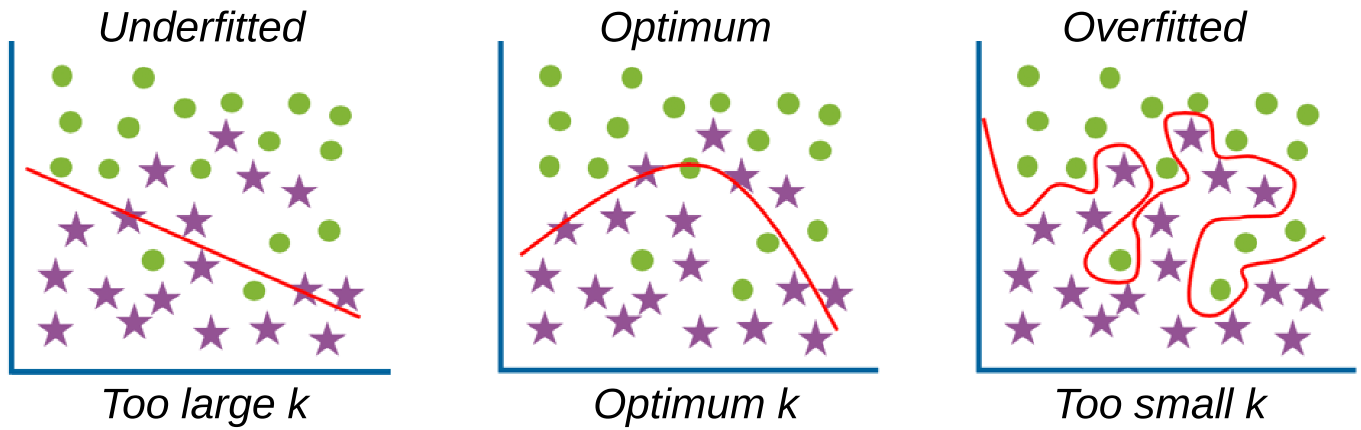
Too small k

Which model would you prefer?

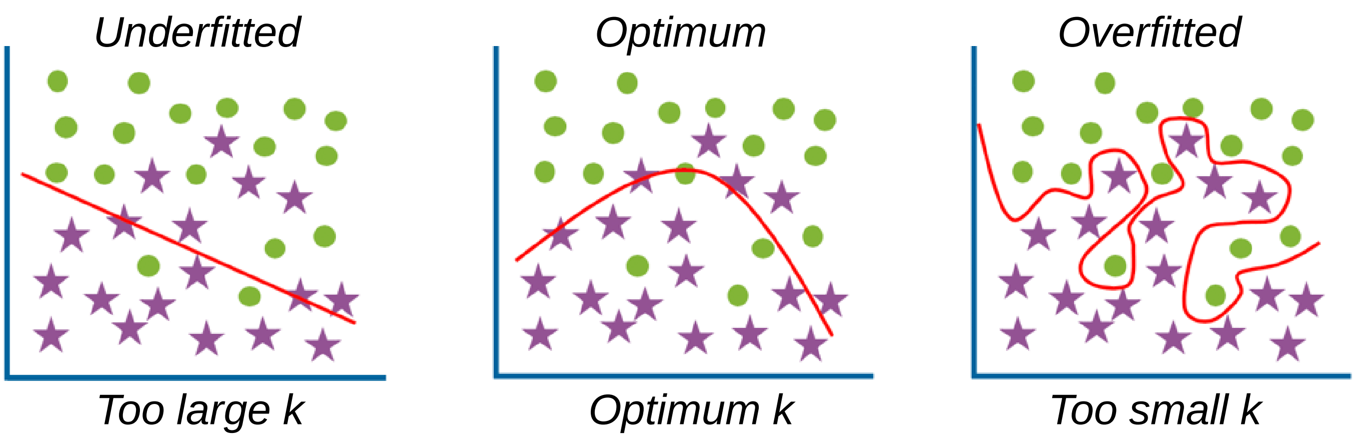




Which model would you prefer?

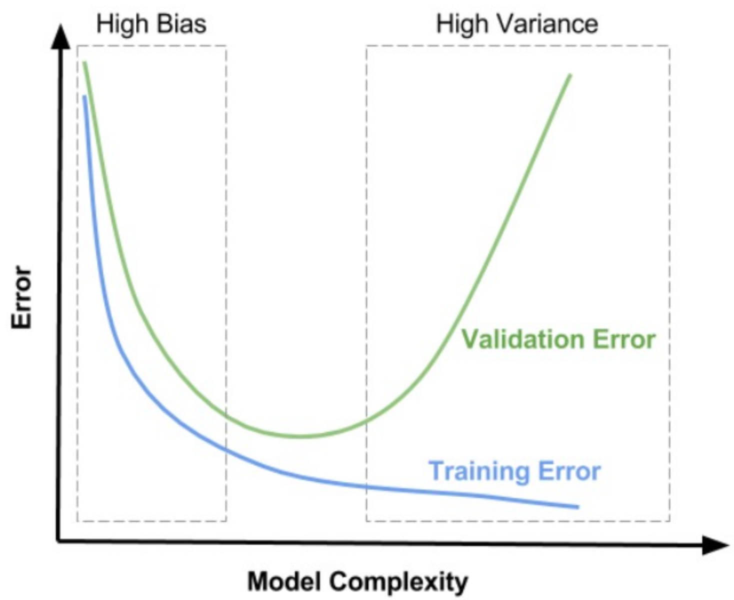


Which model would you prefer?



Cross-validation is needed to find the optimal *k*!

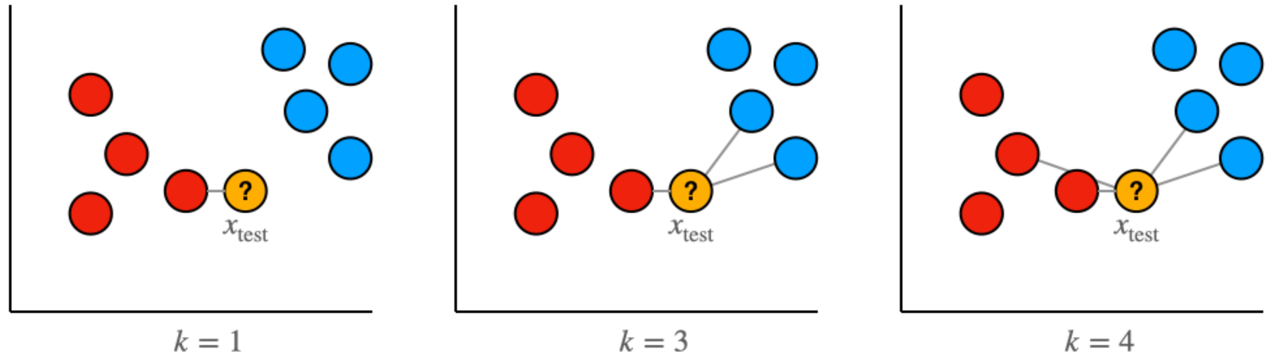
come si stabilisce la larghezza di high bias?



complexity ~ k

In classification problems, the choice of k can lead to **ties**

Binary classification



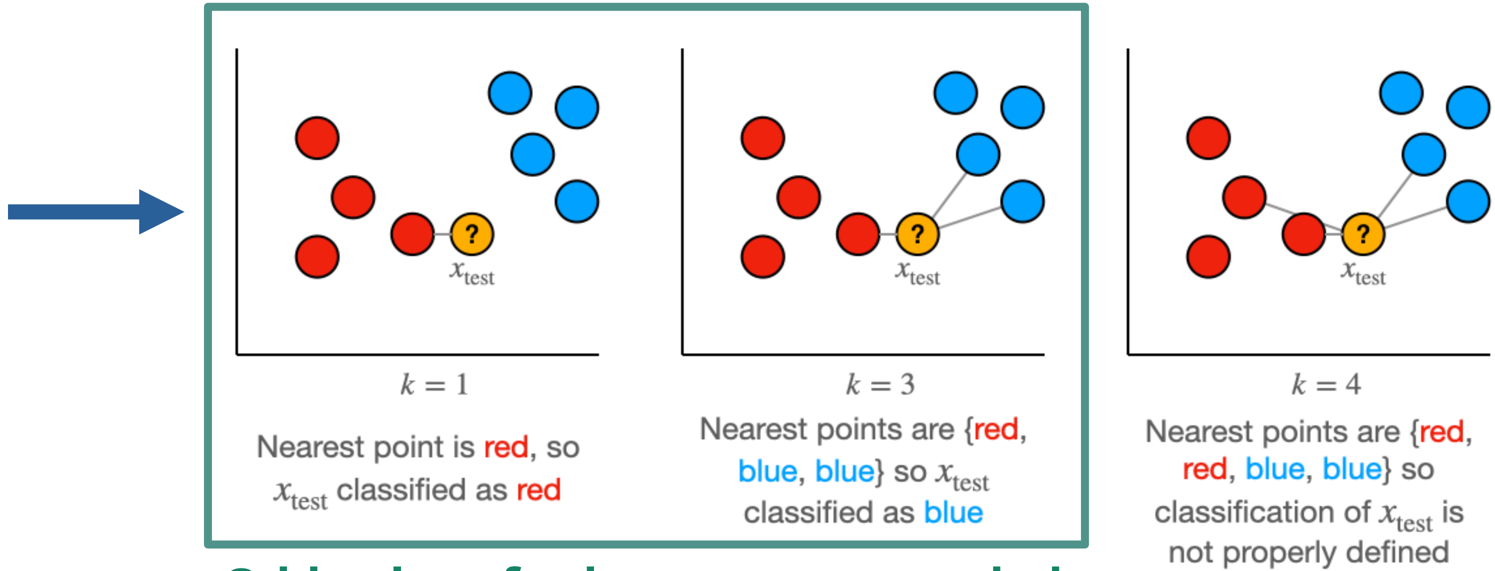
Nearest point is red, so x_{test} classified as red

Nearest points are {red, blue, blue} so x_{test} classified as blue

Nearest points are {red, red, blue, blue} so classification of x_{test} is not properly defined

In classification problems, the choice of k can lead to **ties**

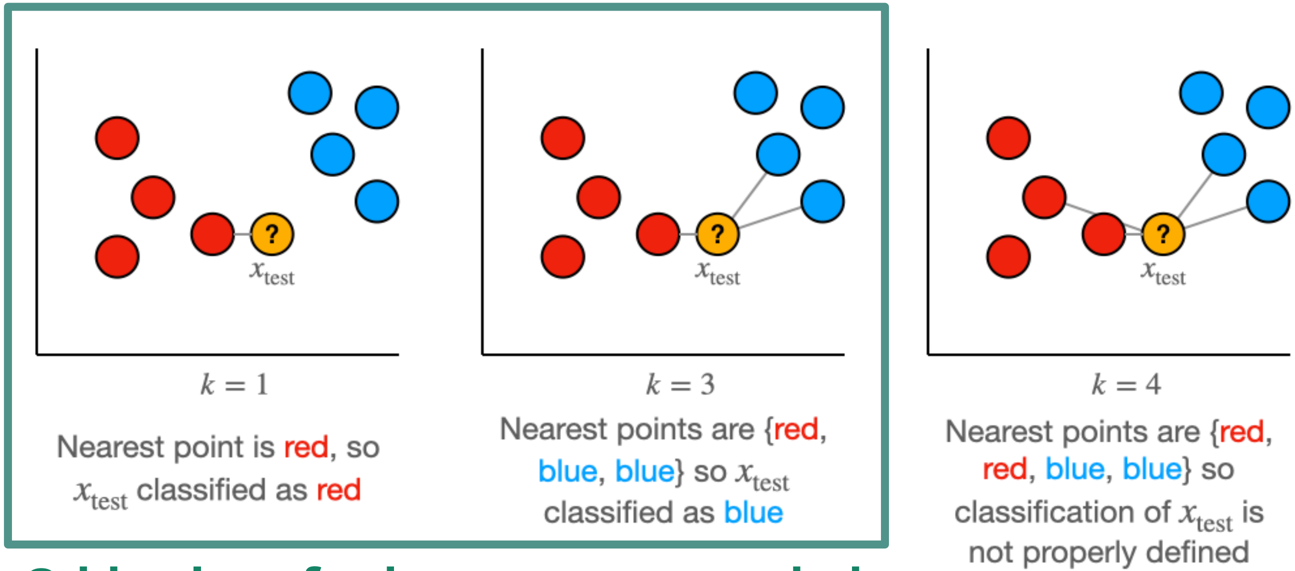
Binary classification



Odd values for k are recommended

In classification problems, the choice of k can lead to **ties**

Binary classification

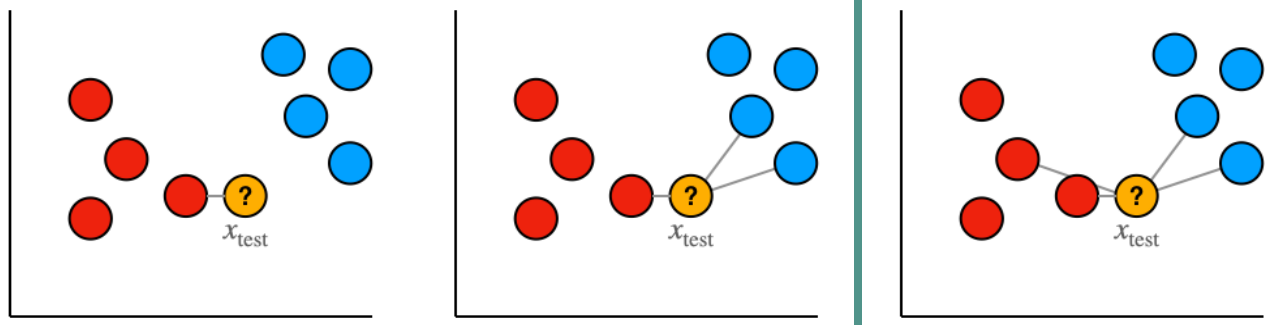


Odd values for k are recommended

Multinomial classification?

In classification problems, the choice of k can lead to **ties**

Binary classification



$k = 1$

Nearest point is red, so x_{test} classified as red

$k = 3$

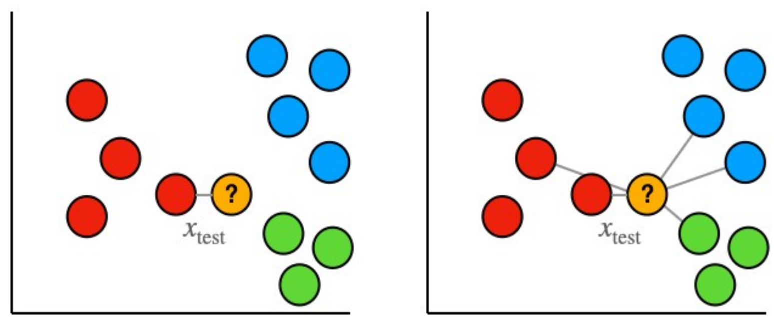
Nearest points are {red, blue, blue} so x_{test} classified as blue

$k = 4$

Nearest points are {red, red, blue, blue} so classification of x_{test} is not properly defined

Odd values for k are recommended

Multinomial classification



$k = 1$

Nearest point is red, so x_{test} classified as red

$k = 5$

Nearest points are {red, red, blue, blue, green}. The model cannot choose between blue and red because both appear twice

*Based on the **iris** dataset, **classify** the following **new instance**:*

(sepal l., sepal w., petal l., petal w.) = (5.4, 2.7, 4.6, 1.4)

```
## new instance
d.new = c(5.4, 2.7, 4.6, 1.4)
```

```
## euclidean distance between the new instance and all the
others
eucli = rep(***)
for (i in 1:nrow(iris)) {
  eucli[i] = ***
}
```

```
## ordering distances
ind.sort = sort(***)
```

```
## classifying based on the nearest
neighbor
pred.k1 = ***
```

```
## classifying based on the 20 nearest
neighbors
pred.k20 = ***
summary(pred.k20)
```

*Divide **iris** into train and test (75% and 25% of the total dataset, respectively) and find the **test error** for the nearest neighbor method (**k=1**). Use the function **knn** from package **class***

```
## train/test partition
n = nrow(iris)
indtrain = sample(1:n, round(0.75*n))
indtest = setdiff(1:n, indtrain)
iris.train = iris[indtrain,]
iris.test = iris[indtest,]
```

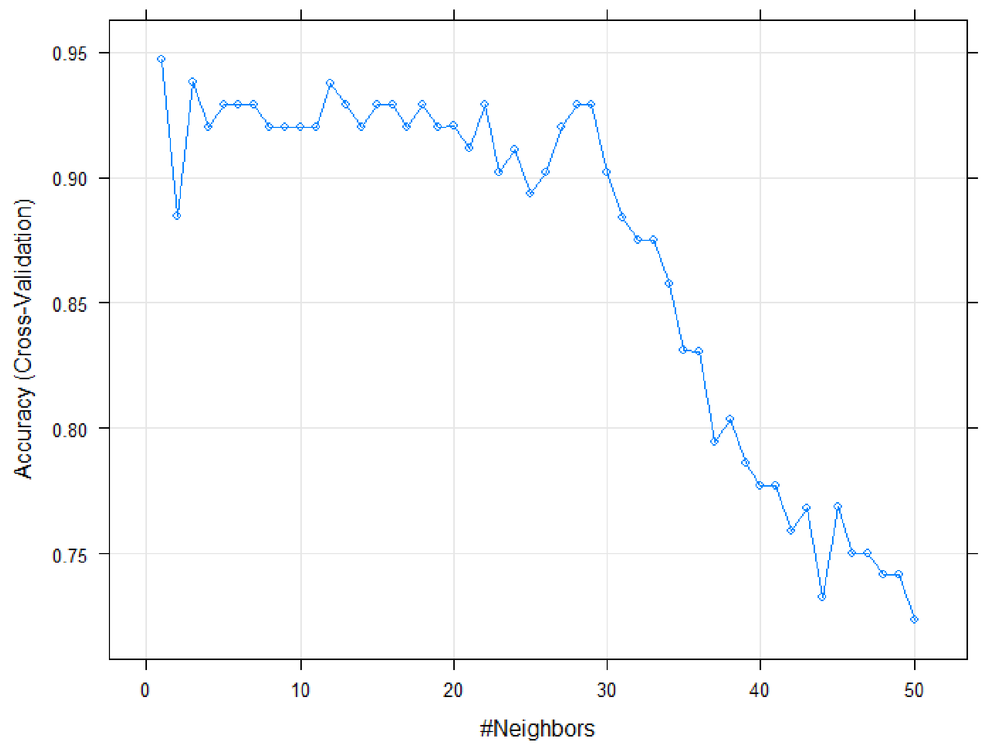
```
## classifying using the nearest neighbor method
library(class)
pred = knn(***)
```

```
## validation (accuracy)
table(***)


acc.class()
```

```
## evaluation function
acc.class = function(x, y) {
  stopifnot(length(x) == length(y))
  return(sum(diag(table(x, y))) / length(x))
}
```

*For the **iris** dataset, use the package **caret** (method knn) to **find the optimal k**. To do so, check **how the validation error varies with increasing k** (for values from **1 to 50**) under a **2-fold cross-validation** scheme. Then, check whether or not the optimal model is overfitted.*

```
library(caret)
## defining 2-fold cross-validation
trctrl = trainControl(method = "***", number =
***)
## searching the optimal k
knn.fit = train(Species ~ ., ***,
              method = "**",
              trControl = ***,
              tuneGrid = )
plot(knn.fit)
```




```
## predicting in test with the optimal k
pred = predict(knn.fit, iris.test)
acc = acc.class(pred, iris.test$Species)
0.9392613
```

Would you say your model is overfitted/underfitted?

In **regression** problems…

Aim:
To **predict** a **continuous** target variable, based on some **similarity measure in the predictors' space**.

	Wind (m/s)	Pressure (hPa)	Humidity (%)	Temperature (ºC)
1	4.1	1018	68	21
2	7.9	1020	64	23
3	1.6	1015	72	18
…				
498	12.3	1008	83	14
499	15.1	1010	80	16
500	4.3	1014	71	17
501	7.8	1013	74	?

New instance
to be predicted

In **regression** problems…

Aim:
To **predict** a **continuous** target variable, based on some **similarity measure in the predictors' space**.

What do we need?
As for classification:
- A distance metric (e.g. Euclidean)
- A value for *k*
- An inference criterion (e.g. the mean, a particular percentile, a random value, etc.)

	Wind (m/s)	Pressure (hPa)	Humidity (%)	Temperature (ºC)
1	4.1	1018	68	21
2	7.9	1020	64	23
3	1.6	1015	72	18
…				
498	12.3	1008	83	14
499	15.1	1010	80	16
500	4.3	1014	71	17
501	7.8	1013	74	?

New instance to be predicted

In **regression** problems…

Aim:
To **predict** a **continuous** target variable, based on some **similarity measure in the predictors' space.**

What do we need?
As for classification:
- A distance metric (e.g. Euclidean)
- A value for *k*
- An inference criterion (e.g. the mean, a particular percentile, a random value, etc.)

	Wind (m/s)	Pressure (hPa)	Humidity (%)	Temperature (ºC)
1	4.1	1018	68	21
2	7.9	1020	64	23
3	1.6	1015	72	18
…				
498	12.3	1008	83	14
499	15.1	1010	80	16
500	4.3	1014	71	17
501	7.8	1013	74	?

New instance to be predicted

To take into account:
- Predictor variables which are larger in magnitude and/or variability may have more weight in the search of neighbors. Thus, **standardizing the predictor data is highly recommended** to make the distance metric more meaningful $$Z = \frac{X - \mu}{\sigma}$$

In **regression** problems…

Aim:
To **predict** a **continuous** target variable, based on some **similarity measure in the predictors' space.**

What do we need?
As for classification:

- A distance metric (e.g. Euclidean)
- A value for *k*
- An inference criterion (e.g. the mean, a particular percentile, a random value, etc.)

	Wind (m/s)	Pressure (hPa)	Humidity (%)	Temperature (ºC)
1	4.1	1018	68	21
2	7.9	1020	64	23
3	1.6	1015	72	18
…				
498	12.3	1008	83	14
499	15.1	1010	80	16
500	4.3	1014	71	17
501	7.8	1013	74	?

New instance to be predicted

To take into account:

- Predictor variables which are larger in magnitude and/or variability may have more weight in the search of neighbors. Thus, **standardizing the predictor data is highly recommended** to make the distance metric more meaningful $Z = \frac{X - \mu}{\sigma}$

<u>*Note*</u>: In the preceding examples this was not important because all predictor variables in the iris dataset are similar in terms of magnitude and variability

In **regression** problems…

Aim:
To **predict** a **continuous** target variable, based on some **similarity measure in the predictors' space.**

What do we need?
As for classification:
- A distance metric (e.g. Euclidean)
- A value for *k*
- An inference criterion (e.g. the mean, a particular percentile, a random value, etc.)

	Wind (m/s)	Pressure (hPa)	Humidity (%)	Temperature (ºC)
1	4.1	1018	68	21
2	7.9	1020	64	23
3	1.6	1015	72	18
…				
498	12.3	1008	83	14
499	15.1	1010	80	16
500	4.3	1014	71	17
501	7.8	1013	74	?

New instance to be predicted

To take into account:
- Predictor variables which are larger in magnitude and/or variability may have more weight in the search of neighbors. Thus, **standardizing the predictor data is highly recommended** to make the distance metric more meaningful $Z = \frac{X - \mu}{\sigma}$
- Likewise, outlier values in the predictors' space may bias the search for neighbors. Thus, if needed, the **removal of outliers is recommended**

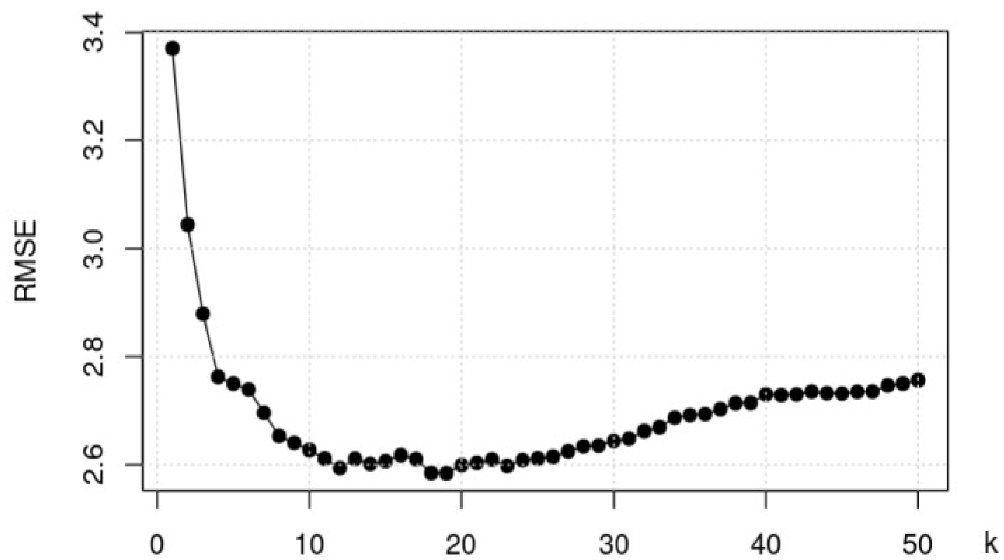
For regression, we will work with the dataset ***carseats*** *(included in the package* ***ISLR****). Our target variable will be* ***Sales****. First, we will remove all the categorical variables from the dataset, retaining only the continuous ones. We will use the function* ***knn.reg*** *from the package* ***FNN****. As you did for the case of classification, divide the total dataset in* ***75% for train and 25% for test*** *and see* ***how the test error (in terms of the RMSE) varies with k***

Examples in R

*For regression, we will work with the dataset **carseats** (included in the package **ISLR**). Our target variable will be **Sales**. First, we will remove all the categorical variables from the dataset, retaining only the continuous ones. We will use the function **knn.reg** from the package **FNN**. As you did for the case of classification, divide the total dataset in **75% for train and 25% for test** and see **how the test error (in terms of the RMSE) varies with k***

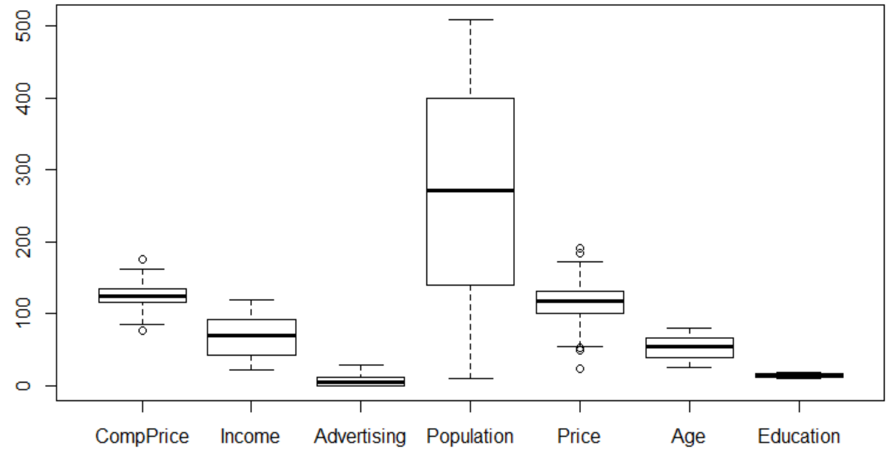
```
## prepare the data
library(ISLR)
attach(Carseats)
dataset = Carseats[, -c(7,10,11)]

## evaluation function
rmse <- function(x, y) {
  sqrt(mean((x - y)^2))
}

## train/test division
n = nrow(dataset)
indtrain = sample(1:n, round(0.75*n));
dataset.train = dataset[indtrain, ]
indtest = setdiff(1:n, indtrain);
dataset.test = dataset[indtest, ]
```

```
## test error as a function of k, from k=1 to k=50
library(FNN)
kmax = 50
test.err = rep(***)
for (k in 1:kmax) {
  pred = knn.reg(***)
  test.err[k] = rmse(***)
}
plot(1:kmax, test.err, type = "o", pch = 19,
xlab = "k", ylab = "RMSE"); grid()
```



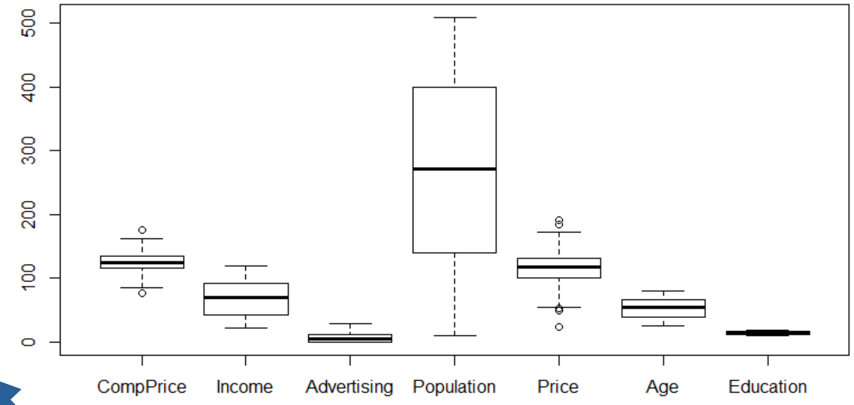
Continue with the same example. Let's now assess the ***effect of standardizing the predictor data****. Use the function* ***scale***

```
## predictor ranges
boxplot(dataset[,-1])
```



*Continue with the same example. Let's now assess the **effect of standardizing the predictor data**. Use the function **scale***

```
## predictor ranges
boxplot(dataset[,-1])
```




```
## predictor standardization
params.scaling = preProcess(***,
                method = ***)
x.train.scaled = scale(***,
              center = ***, scale = ***)
x.test.scaled = scale(***,
              center = ***, scale = ***)
```

```
## test error as a function of k (for standardized predictors)
test.err2 = rep(***)
for (k in 1:kmax) {
  ## prediction
  pred = knn.reg(***)
  ## validation
  test.err2[k] = rmse(***)
}
```

```
## plotting results
matplot(1:kmax, cbind(***),
        type = "o", pch = 19, lty = 1,
        col = c("black", "red"), xlab = "k", ylab = "RMSE")
legend("topright", c("without scaling", "with scaling"),
       lty = 1, col = c("black", "red"))
grid()
```

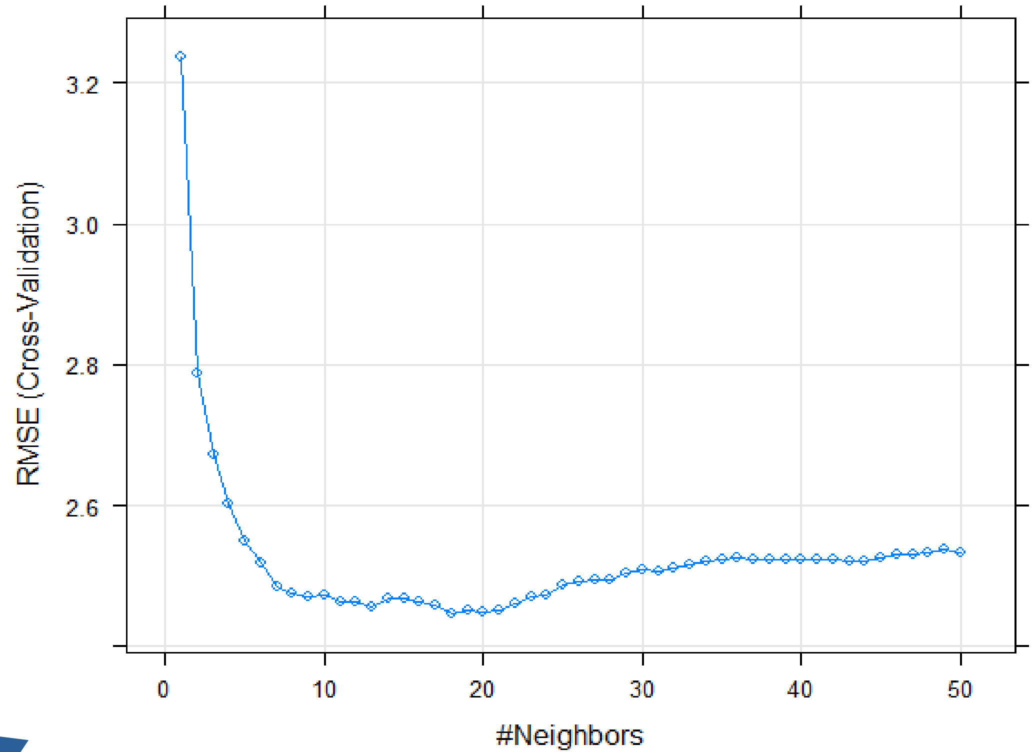
Be careful: Standardization must be properly done!

*Do the **same exercise**, but this time using **caret**, under a **2-fold cross-validation scheme**. Recall to standardize your predictor data to obtain meaningful results*

```
## defininig 2-fold cross-validation
trctrl <- trainControl(***)
```

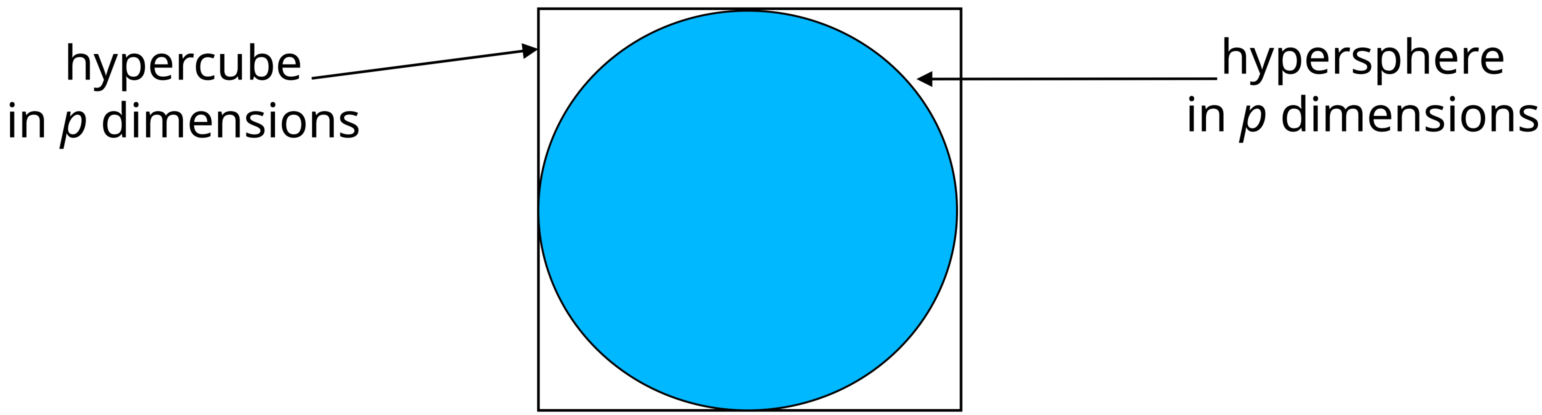
```
## searching the optimal k
(with standardized data)
knn.fit <- train(, data = ***,
              method = "***",
              trControl = ***,
              preProcess = ***,
              tuneGrid = ***)
plot(***)
```

```
## predicting in test with the optimal k
pred = predict(***)
## evaluation of test error
rmse(***)
```



Would you say your model is overfitted/underfitted?

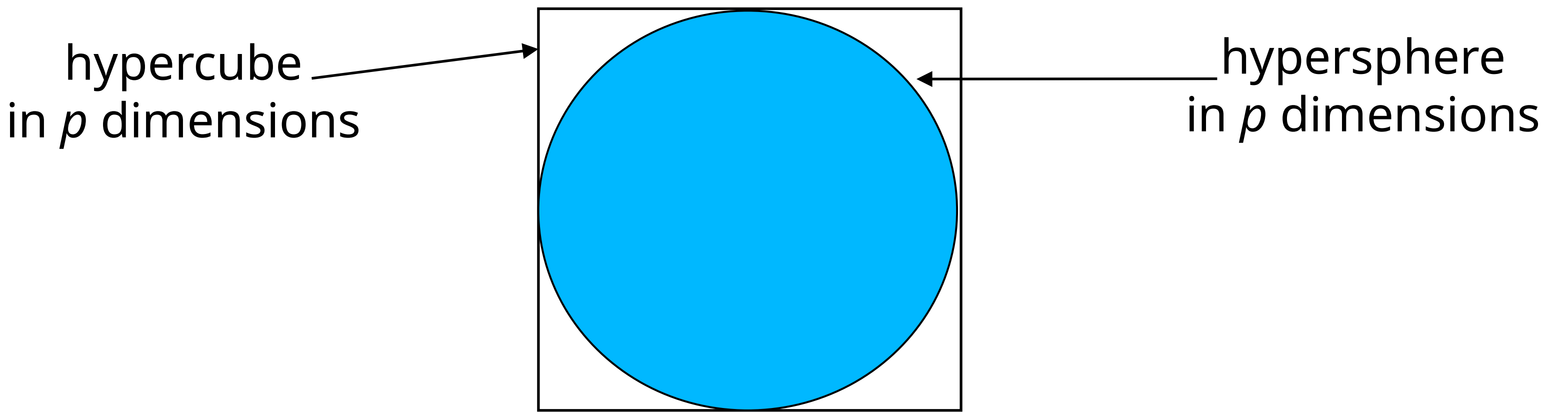
David Scott, Multivariate Density Estimation, Wiley, 1992



Volume of the sphere relative to that of the cube?

Dimension	**2**
Rel. vol.	**0.79**

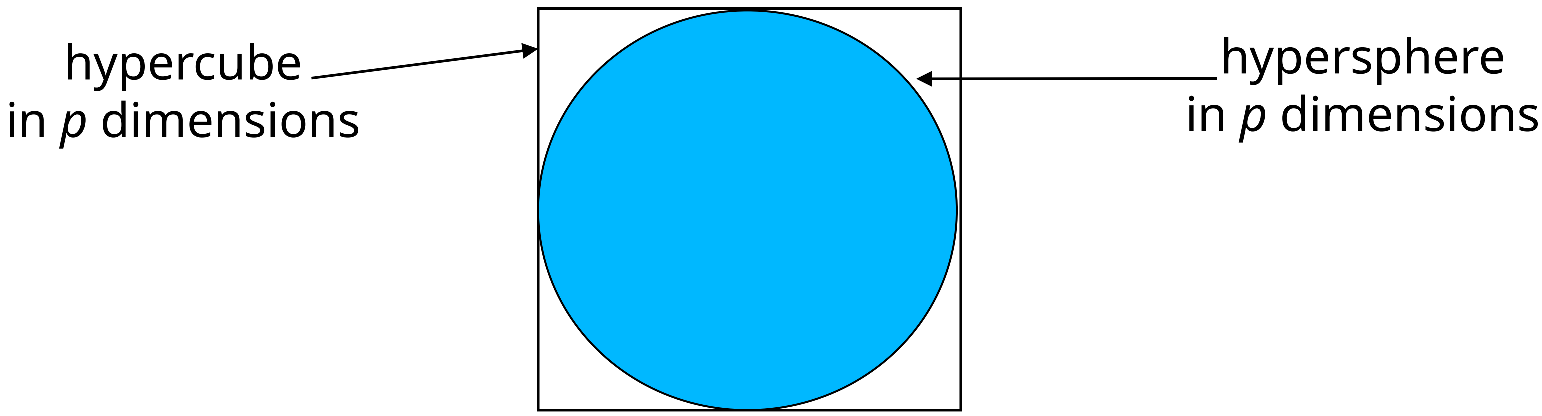
David Scott, Multivariate Density Estimation, Wiley, 1992



Volume of the sphere relative to that of the cube?

Dimension	**2**	**3**
Rel. vol.	**0.79**	**0.53**

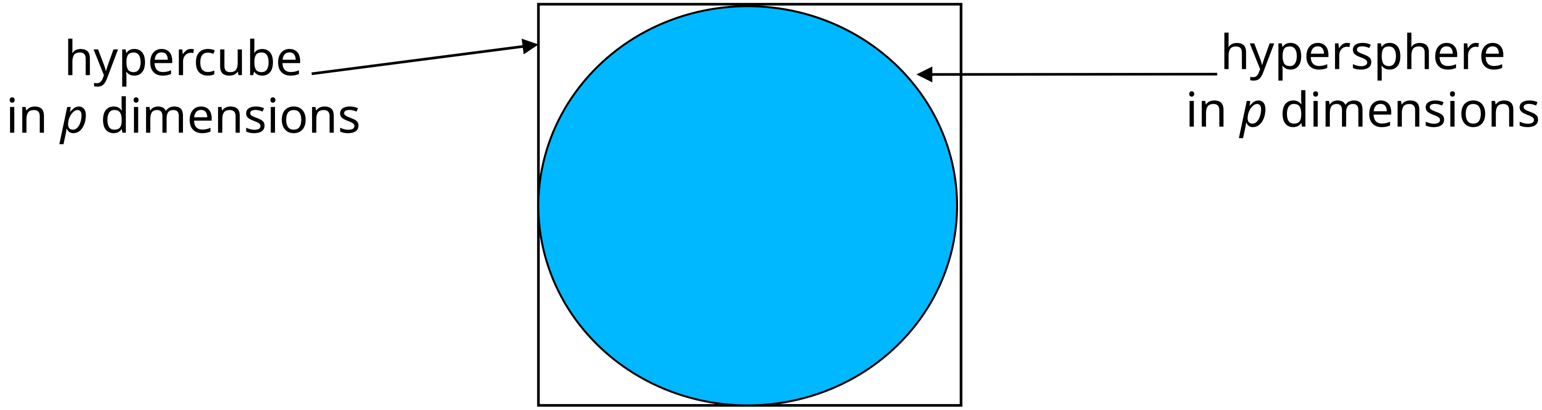
David Scott, Multivariate Density Estimation, Wiley, 1992



Volume of the sphere relative to that of the cube?

Dimension	**2**	**3**	**4**
Rel. vol.	**0.79**	**0.53**	**0.31**

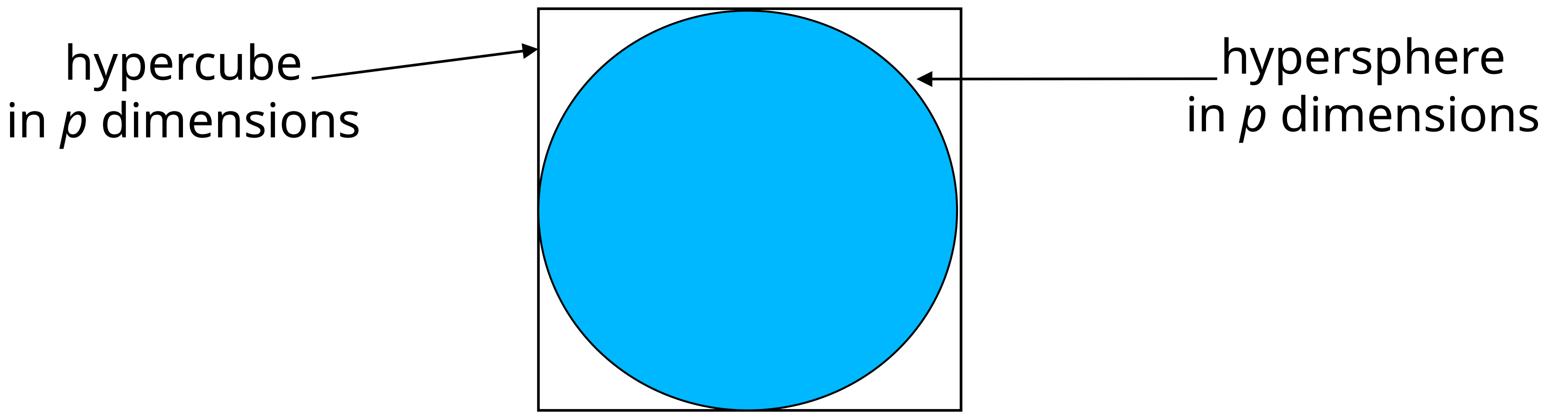
David Scott, Multivariate Density Estimation, Wiley, 1992



Volume of the sphere relative to that of the cube?

Dimension	**2**	**3**	**4**	**5**
Rel. vol.	**0.79**	**0.53**	**0.31**	**0.16**

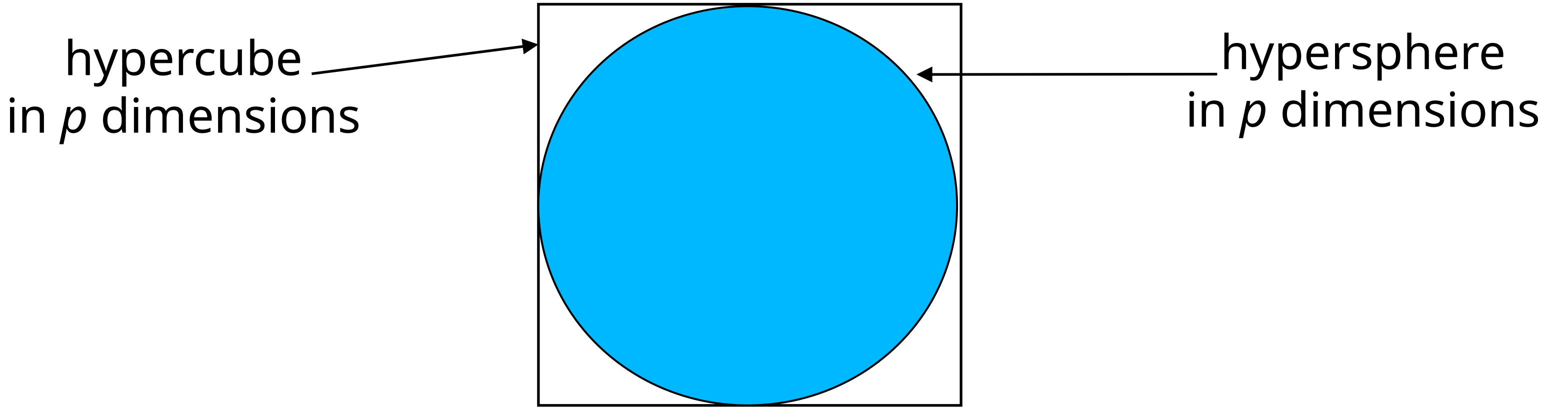
David Scott, Multivariate Density Estimation, Wiley, 1992



Volume of the sphere relative to that of the cube?

Dimension	**2**	**3**	**4**	**5**	**6**
Rel. vol.	**0.79**	**0.53**	**0.31**	**0.16**	**0.08**

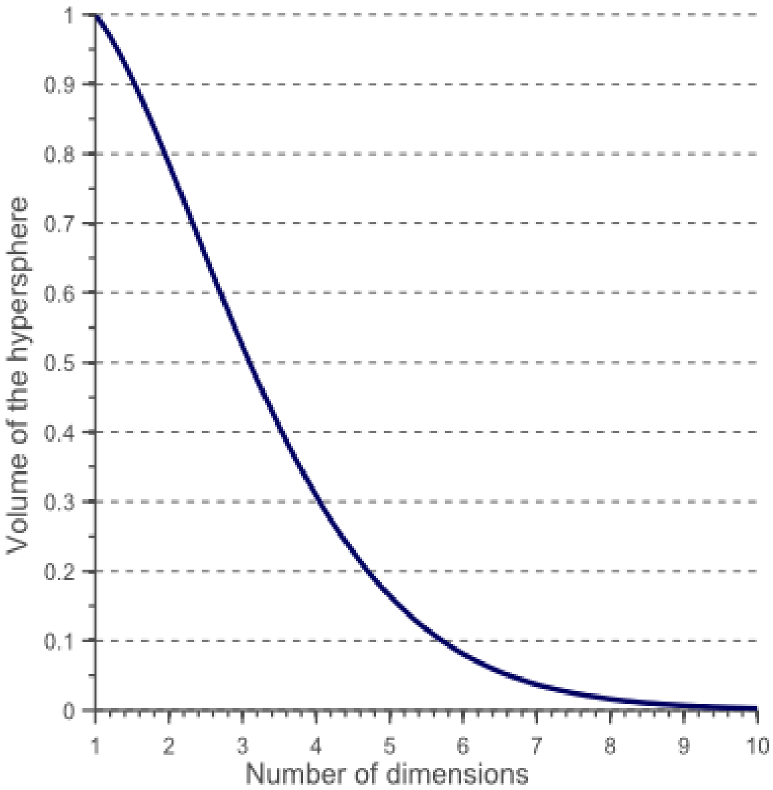
David Scott, Multivariate Density Estimation, Wiley, 1992



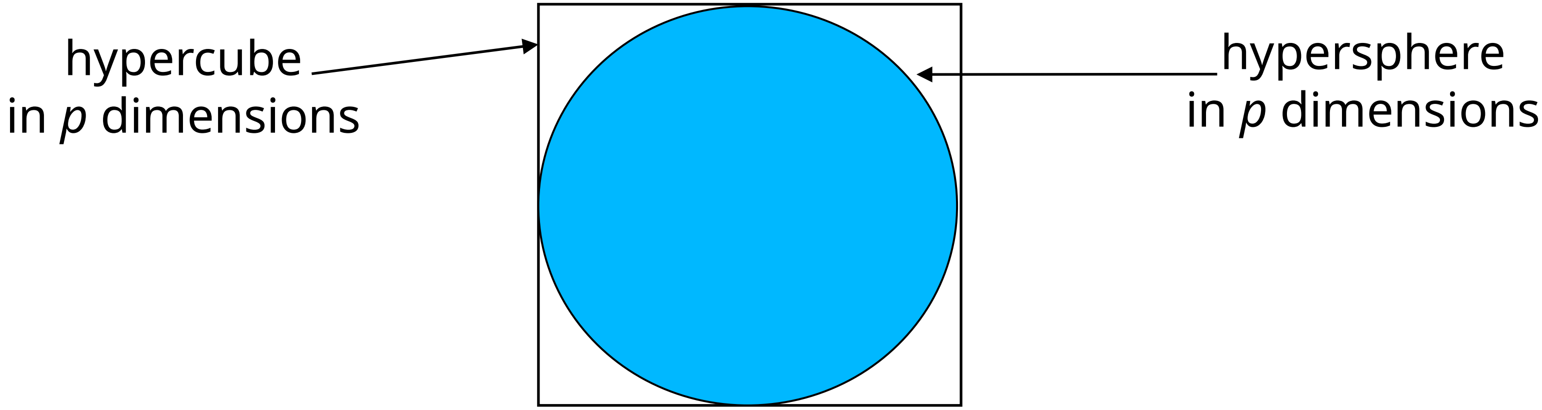
Volume of the sphere relative to that of the cube?

Dimension	2	3	4	5	6	7
Rel. vol.	**0.79**	**0.53**	**0.31**	**0.16**	**0.08**	**0.04**

Any thought?

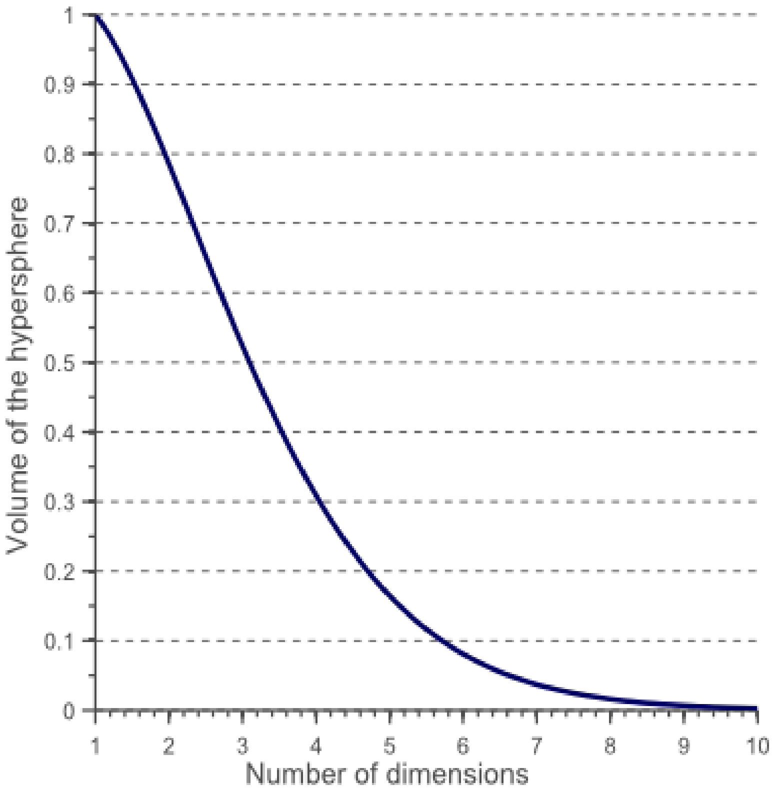


David Scott, Multivariate Density Estimation, Wiley, 1992



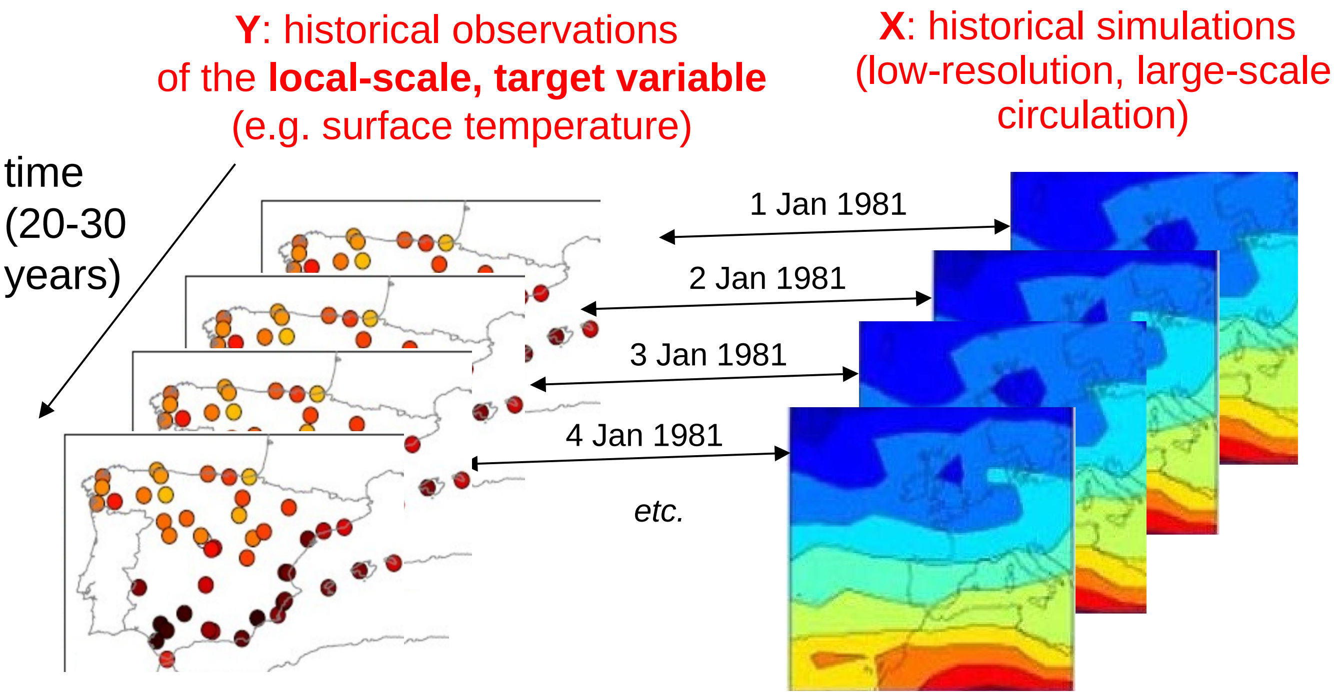
Volume of the sphere relative to that of the cube?

Dimension	**2**	**3**	**4**	**5**	**6**	**7**
Rel. vol.	**0.79**	**0.53**	**0.31**	**0.16**	**0.08**	**0.04**



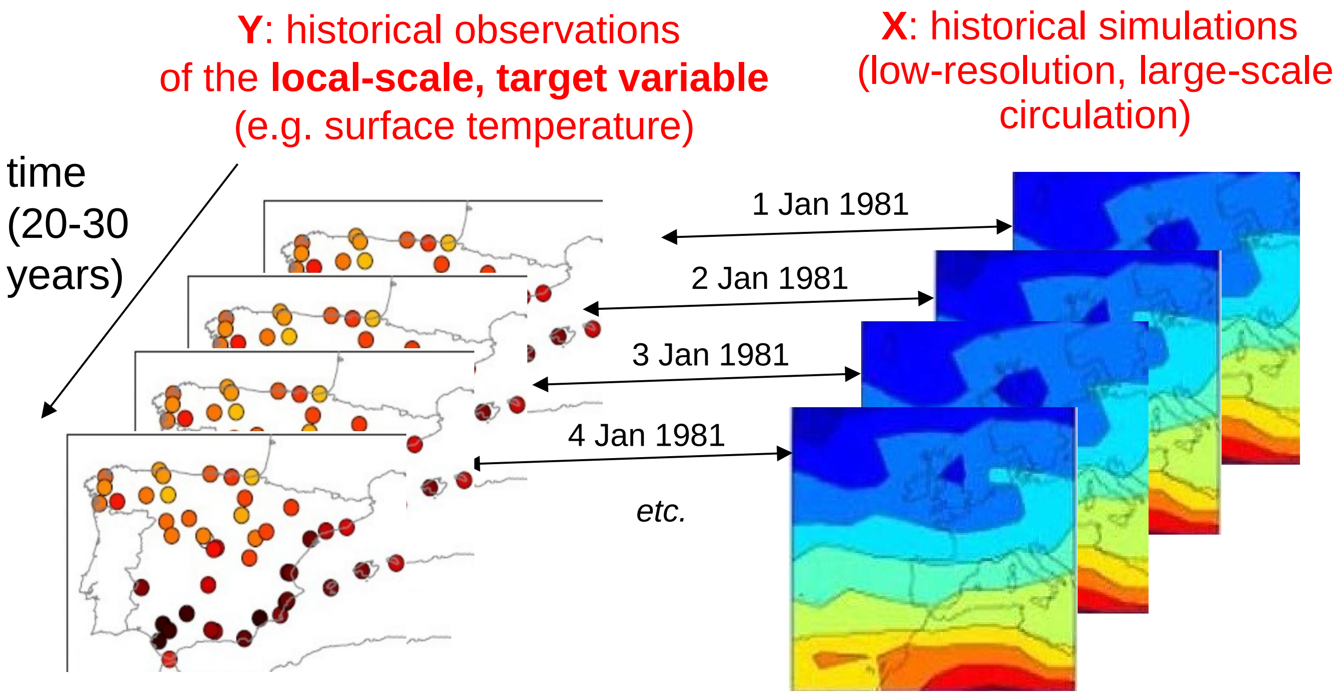
As dimensionality increases, a larger percentage of the training data resides in the corners of the predictors' space. Therefore, ***k*-NN is unhelpful in high dimensional problems** because distances are less meaningful.

The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions



The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

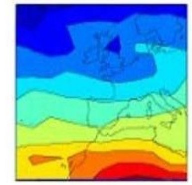
Problem: Y' (projection) for 26 Mar 2046?



The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

Problem: Y' (projection) for 26 Mar 2046?

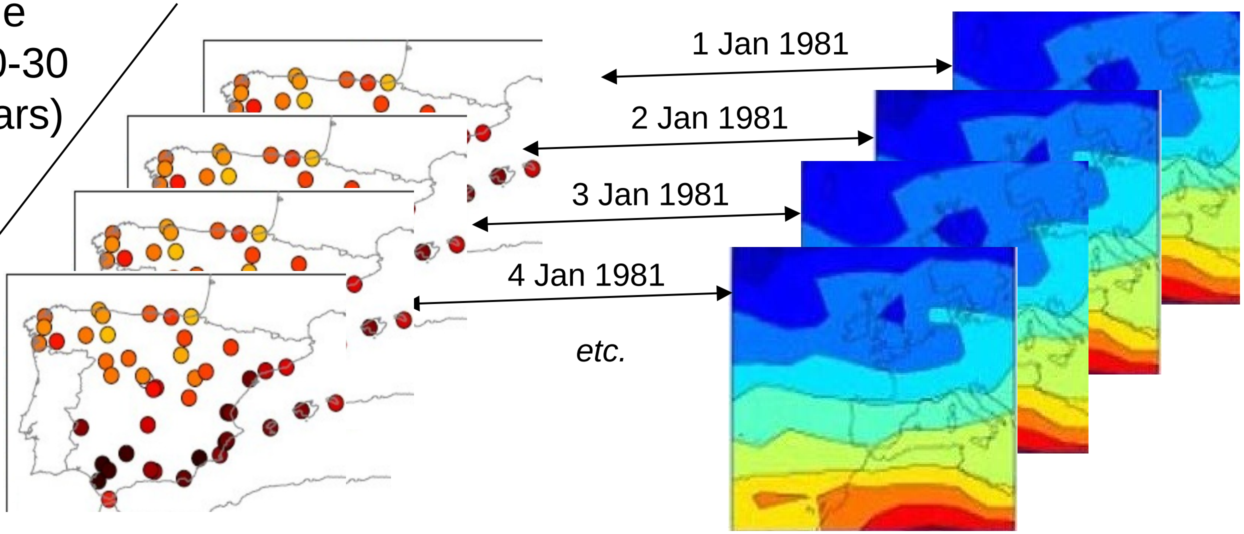
1) Take X' for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$



Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

X: historical simulations (low-resolution, large-scale circulation)

time (20-30 years)

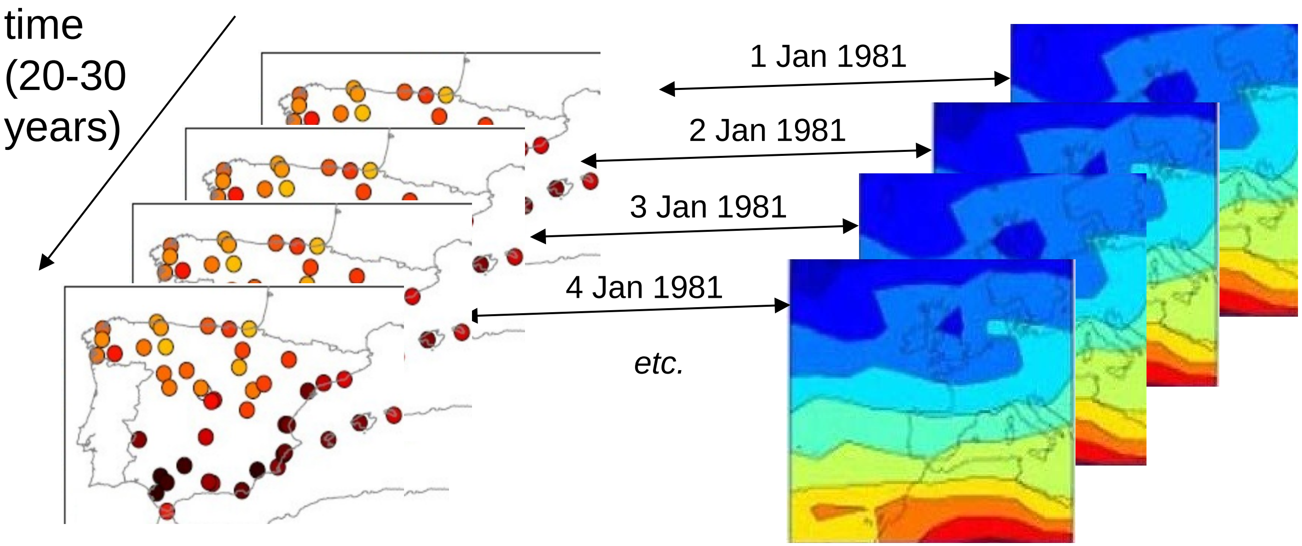


The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

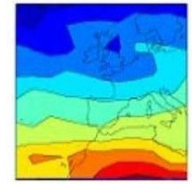
Problem: Y’ (projection) for 26 Mar 2046?

Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

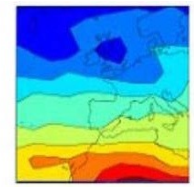
X: historical simulations (low-resolution, large-scale circulation)



1) Take X’ for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$



2) Search the nearest neighbor/s to $X'_{26\text{-}03\text{-}2046}$ within X. Let’s suppose k=1



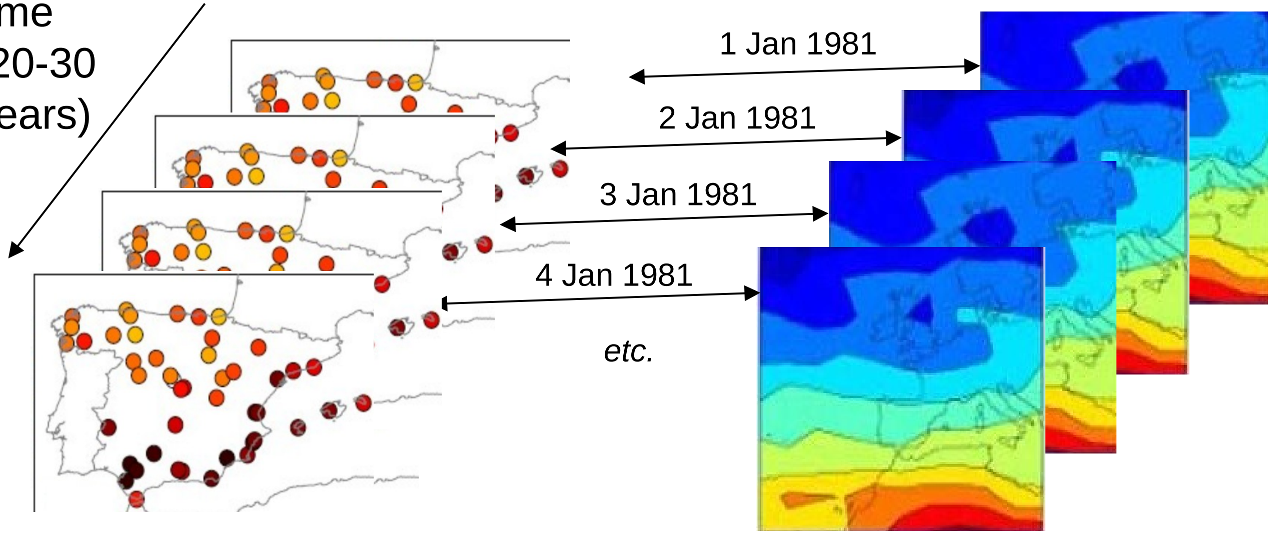
$X_{03\text{-}01\text{-}1981}$

The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

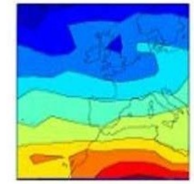
X: historical simulations (low-resolution, large-scale circulation)

time (20-30 years)

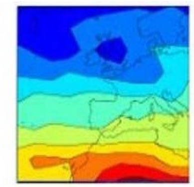


Problem: Y' (projection) for 26 Mar 2046?

1) Take X' for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$



2) Search the nearest neighbor/s to $X'_{26\text{-}03\text{-}2046}$ within X. Let's suppose k=1



$X_{03\text{-}01\text{-}1981}$

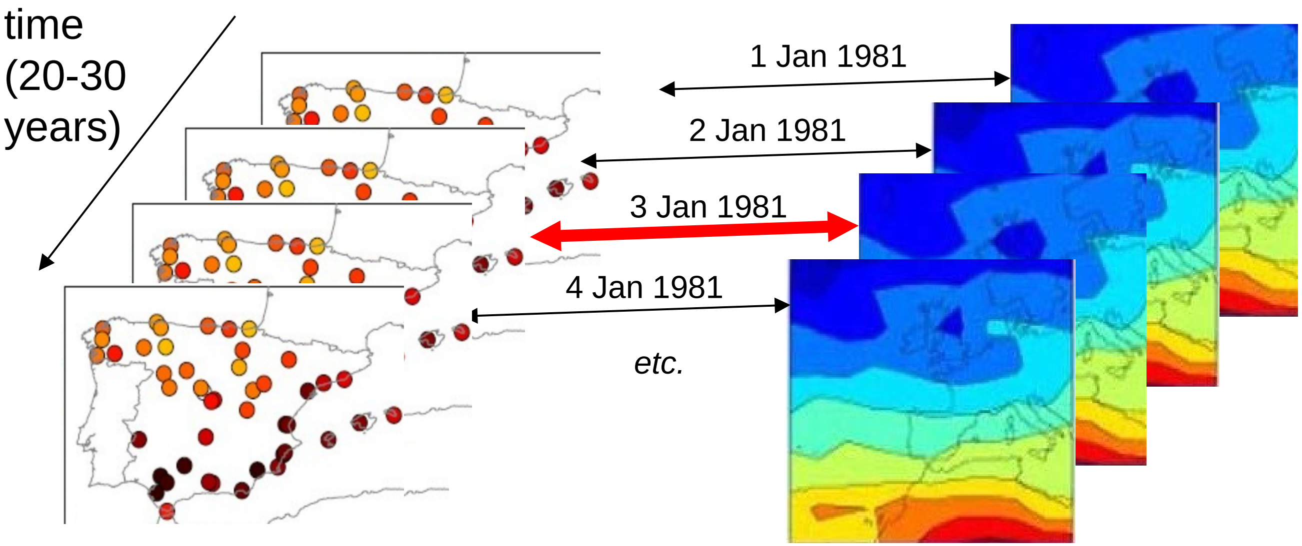
Can you imagine what the projection for 26 Mar 2046 will be?

The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

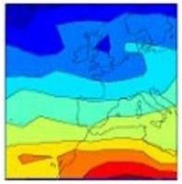
Problem: Y' (projection) for 26 Mar 2046?

Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

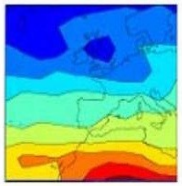
X: historical simulations (low-resolution, large-scale circulation)



1) Take X' for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$

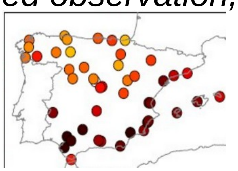


2) Search the nearest neighbor/s to $X'_{26\text{-}03\text{-}2046}$ within X. Let's suppose k=1



$X_{03\text{-}01\text{-}1981}$

3) The projection made, $Y'_{26\text{-}03\text{-}2046}$, is the recorded observation, $Y_{03\text{-}01\text{-}1981}$

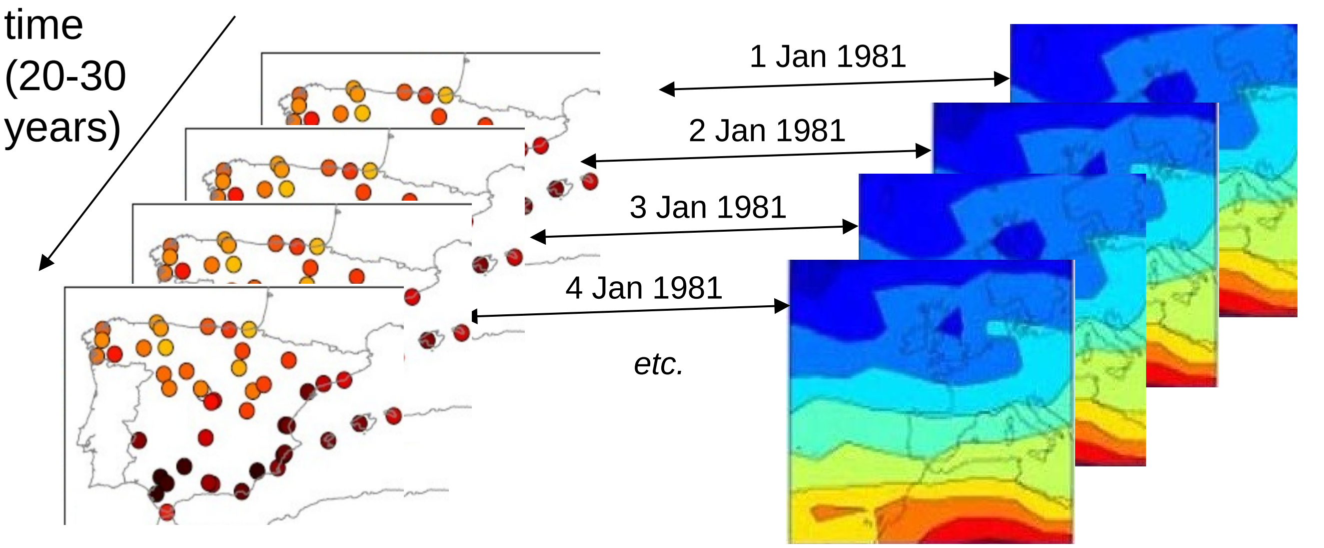


The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

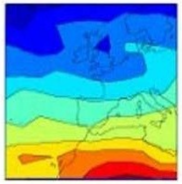
Problem: Y' (projection) for 26 Mar 2046?

Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

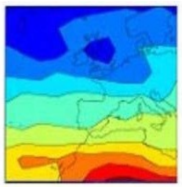
X: historical simulations (low-resolution, large-scale circulation)



1) Take X' for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$

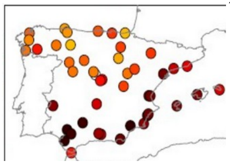


2) Search the nearest neighbor/s to $X'_{26\text{-}03\text{-}2046}$ within X. Let's suppose k=1



$X_{03\text{-}01\text{-}1981}$

3) The projection made, $Y'_{26\text{-}03\text{-}2046}$, is the recorded observation, $Y_{03\text{-}01\text{-}1981}$



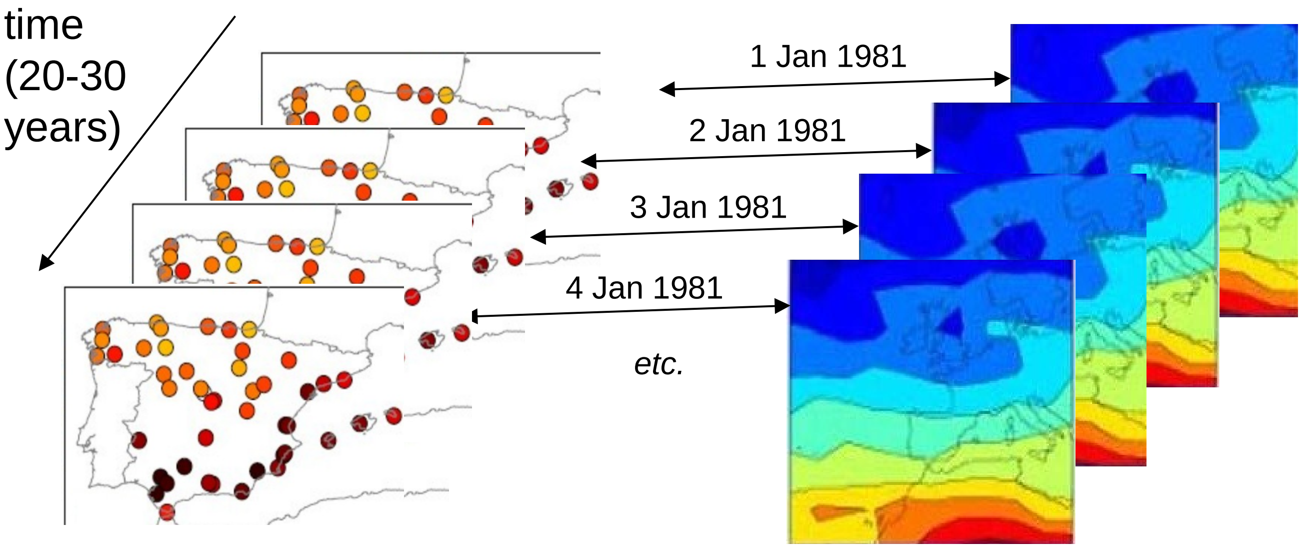
In the climate science, two important factors must be taken into account for the application of the k-NN technique:

- Large differences exist, in magnitude and variability, across predictors: **Scaling** must be applied
- The dimensionality of the predictors' space can be very large: **Dimensionality reduction** techniques (e.g. PCA) are commonly used

The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

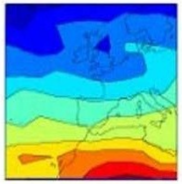
Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

X: historical simulations (low-resolution, large-scale circulation)

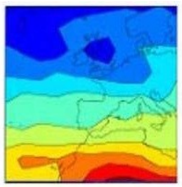


Problem: Y' (projection) for 26 Mar 2046?

1) Take X' for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$

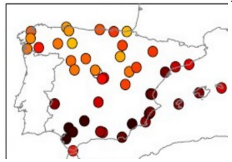


2) Search the nearest neighbor/s to $X'_{26\text{-}03\text{-}2046}$ within X. Let's suppose k=1



$X_{03\text{-}01\text{-}1981}$

3) The projection made, $Y'_{26\text{-}03\text{-}2046}$, is the recorded observation, $Y_{03\text{-}01\text{-}1981}$

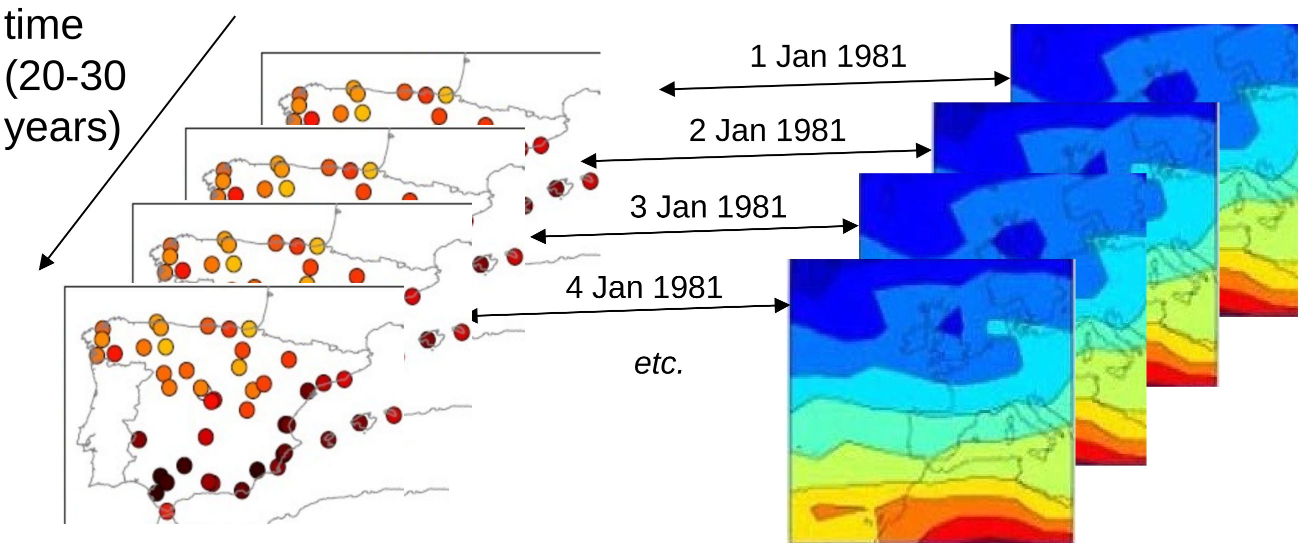


Is there any **limitation** you may think of in a **climate change** context?

The analog technique (Lorenz, 1969): Similar atmospheric patterns lead to similar meteorological conditions

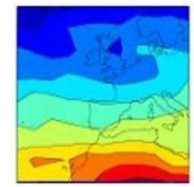
Y: historical observations of the **local-scale, target variable** (e.g. surface temperature)

X: historical simulations (low-resolution, large-scale circulation)

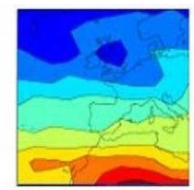


Problem: Y' (projection) for 26 Mar 2046?

1) Take X' for 26 March 2046: $X'_{26\text{-}03\text{-}2046}$

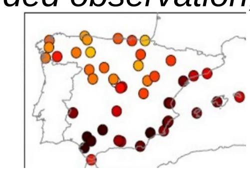


2) Search the nearest neighbor/s to $X'_{26\text{-}03\text{-}2046}$ within X. Let's suppose k=1



$X_{03\text{-}01\text{-}1981}$

3) The projection made, $Y'_{26\text{-}03\text{-}2046}$, is the recorded observation, $Y_{03\text{-}01\text{-}1981}$



k-NN has **no ability for extrapolation** beyond the learning space!