Faculty of SET / School of Computer and Mathematical Sciences

COMP SCI 3007/7059/7659
Artificial Intelligence
Forms of Learning

Acknowledgement of Country

We acknowledge and pay our respects to the Kaurna people, the traditional custodians whose ancestral lands we gather on.

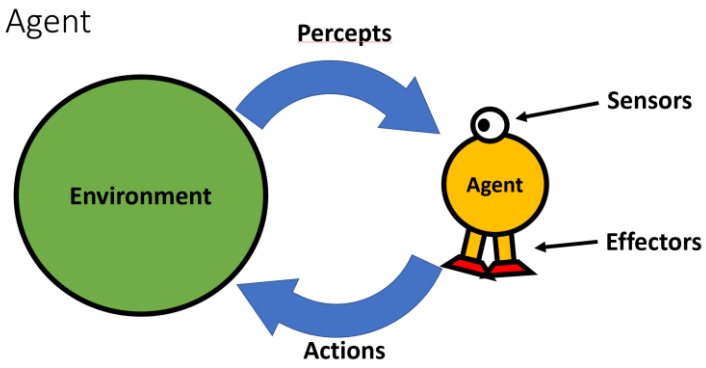
We acknowledge the deep feelings of attachment and relationship of the Kaurna people to the country and we respect and value their past, present and ongoing connection to the land and cultural beliefs.

Forms of Learning

AIMA C18.1-18.2

What is learning and why?

- Percepts should not only be used for acting



- Agent with a learning element
  - Receive sensory signal as input
  - Generate output
  - Receive feedback
  - Adjust the parameter by using feedback signals

- Learning allows an agent to adapt to new circumstances and to detect and extrapolate patterns

What is learning and why?

- Learning depends on four major factors
  - which component(s) are to be improved -> objectives
  - what prior knowledge the agent already has (i.e, are we building the agent from scratch, or just updating it?).

    X or $f(\mathbf{X})$

  - the representation of the percept, e.g., numeric, character strings, sets of objects.

    'artificial' or [0.26, 0.45, …0.83]
    'intelligence' [0.14, 0.67, …0.55]

  - what feedback is available to learn from.

    y or *rewards*

What is learning and why?

- Three main forms of learning in intelligent agents (depending on the feedback type)
  - Supervised Learning

    (**X, y**) ~ **y** = f(**X**)

  - Unsupervised Learning

    (**X,**) ~ *patterns*

    ~ *groups*

    ~ *representations*

  - Reinforcement Learning

    (actions**,** rewards) -> best actions

Supervised Learning

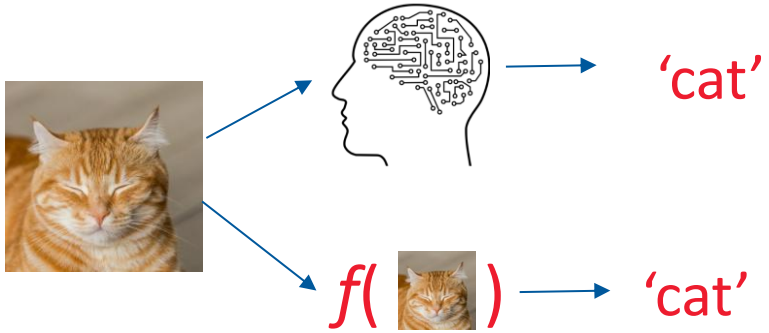
Given a **training set** of N example **input-output pairs**

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N),$$

where each y_i was generated by an unknown function $y = f(x)$, **discover a function** h that approximates f.

The goal is to **predict the output value** y^* for a new unseen before input x^*.

Example: f (x) is the function used in human brain for mapping image to its class category



Supervised Learning

Given a **training set** of N example **input-output pairs**

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N),$$

where each y_i was generated by an unknown function $y = f(x)$, **discover a function** h that approximates f.

The goal is to **predict the output value** y^* for a new unseen before input x^*.

The availability of a **target output value** y_i for each input x_i differentiates supervised learning from other forms of learning.

The function h is called a **hypothesis**. Supervised learning can be viewed as searching in the space of hypotheses for the best h.

$h \neq f$, *but* $h \approx f$

Supervised Learning

Given a **training set** of N example **input-output pairs**

$$(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N),$$

The availability of a **target output value** y_i for each input x_i differentiates supervised learning from other forms of learning.

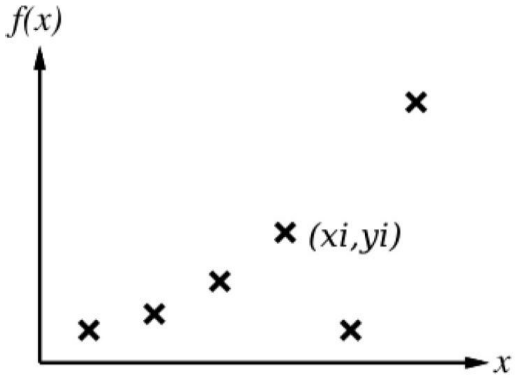
The function h is called a **hypothesis**. Supervised learning can be viewed as searching in the space of hypotheses for the best h.

h can be a family of functions with unknown parameters, e.g., $h = \boldsymbol{W}^T \boldsymbol{x} + \boldsymbol{b}$
Supervised learning => find best h => find best $\boldsymbol{W}$ and $\mathbf{b}$

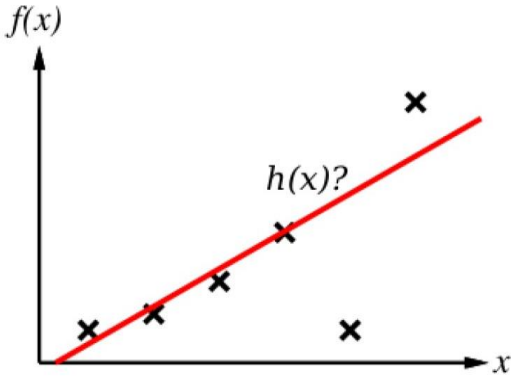
Supervised Learning (Cont.)

The values for x_i, y_i can be numeric, discrete, continuous, non-numeric (nominal, symbols, text), etc.

When the target outputs are **continuous** values, we have a **regression** problem.



(a) Training set of input-output pairs. $f(x)$ is unknown.



(b) Candidate hypothesis $h(x)$.

We want

$h(x_i) \approx y_i$

Supervised Learning (Cont.)

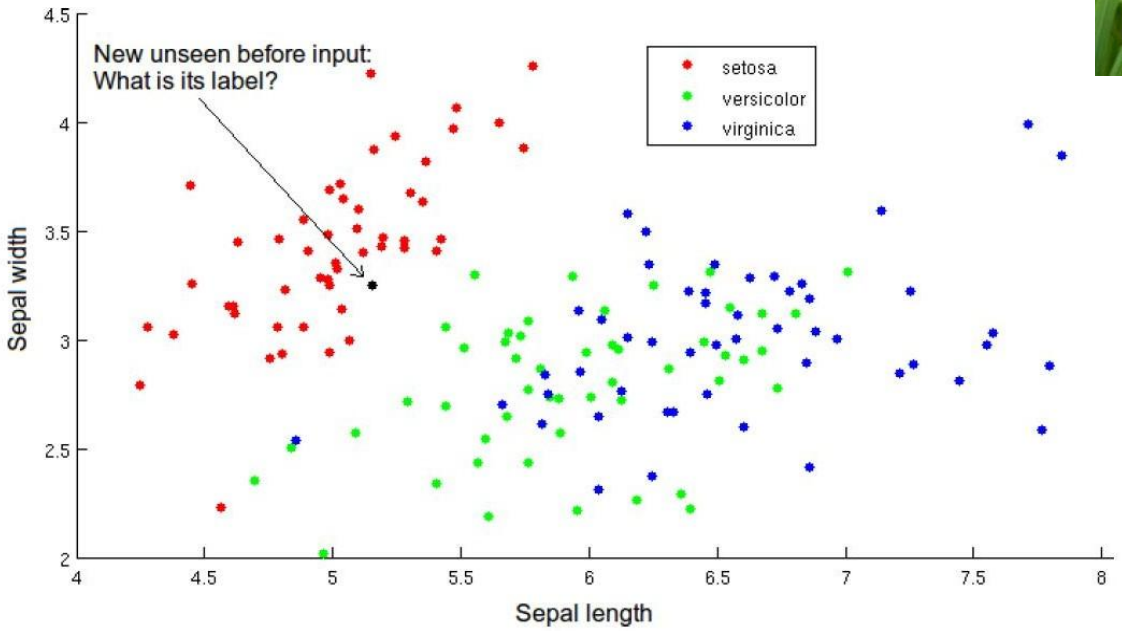
setosa

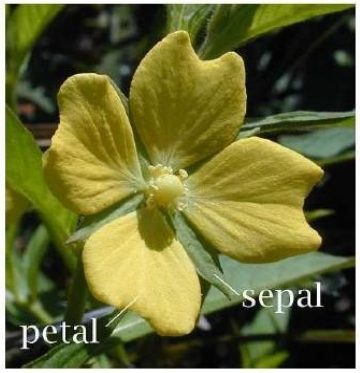
versicolor

virginica

When the target outputs are nominal values or **class labels**, we have a **classification** problem.

Example: Given measurements of sepal length and sepal width, identify the **type** of flower.





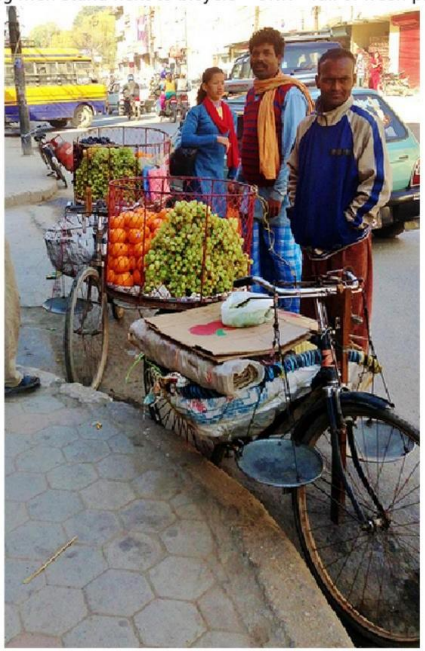
Here each input value $\mathbf{x}_i$ is a two-dimensional vector, while each target value y_i can be setosa, versicolor or virginica.

Supervised Learning (Cont.)

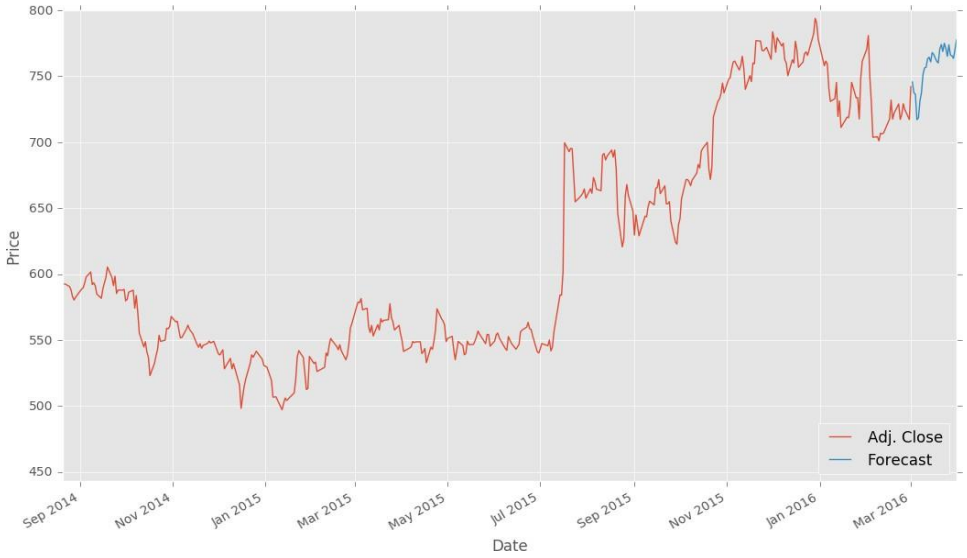
More examples

Image captioning

<START> young men stand next to bicycle <UNK> full of fresh produce <END>



Predicting stock price



Supervised Learning (Cont.)

More examples

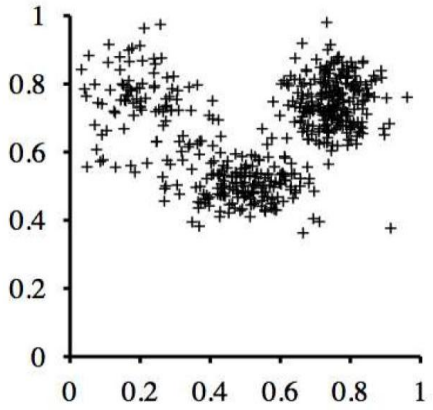
Learning to count

(a)

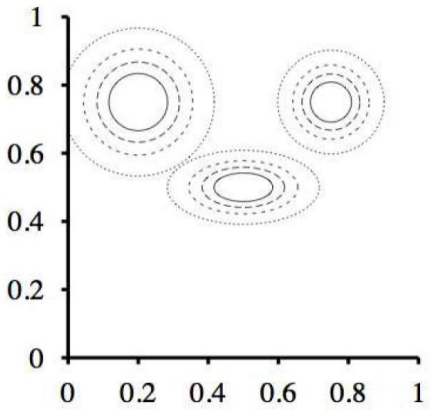
Unsupervised Learning

Learning **patterns** in the input when no specific target output values are supplied.

Example: Finding clusters in 2D input data.

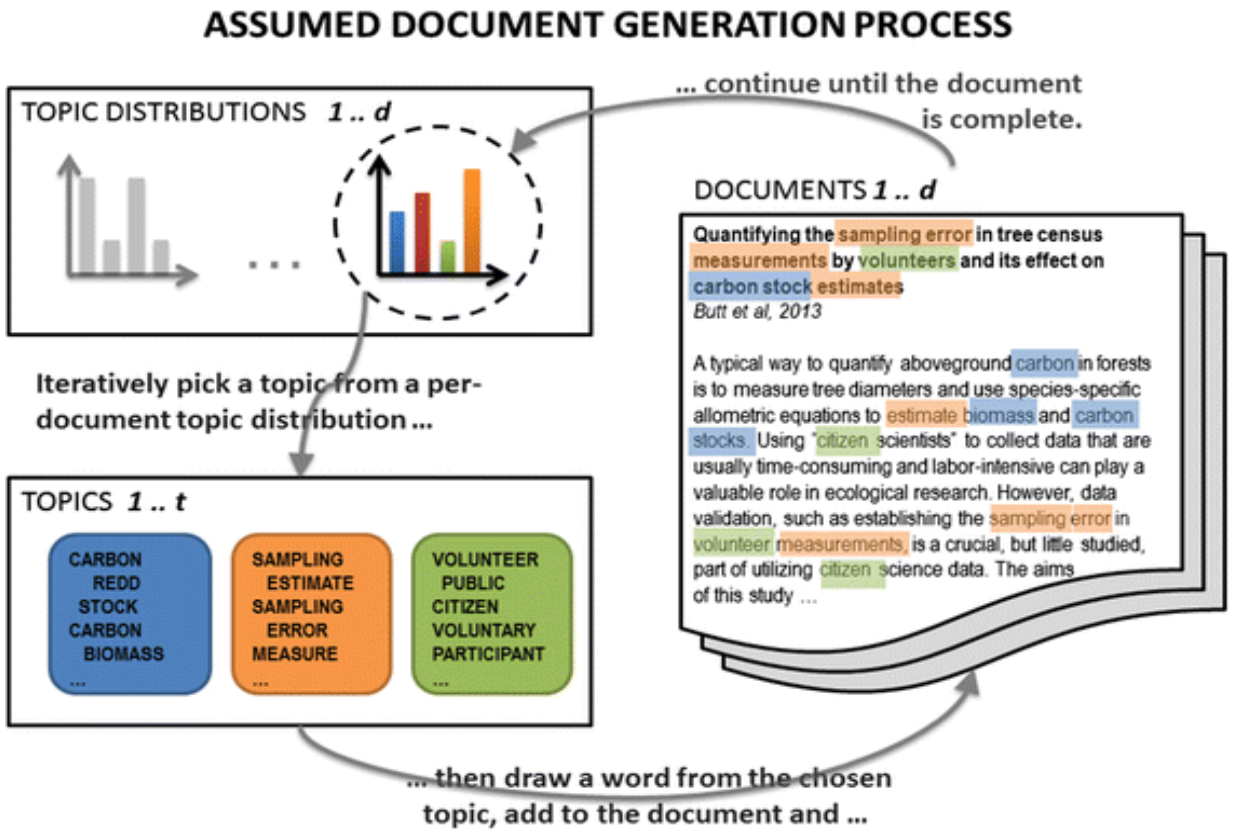


(a) Input data with unknown cluster memberships.



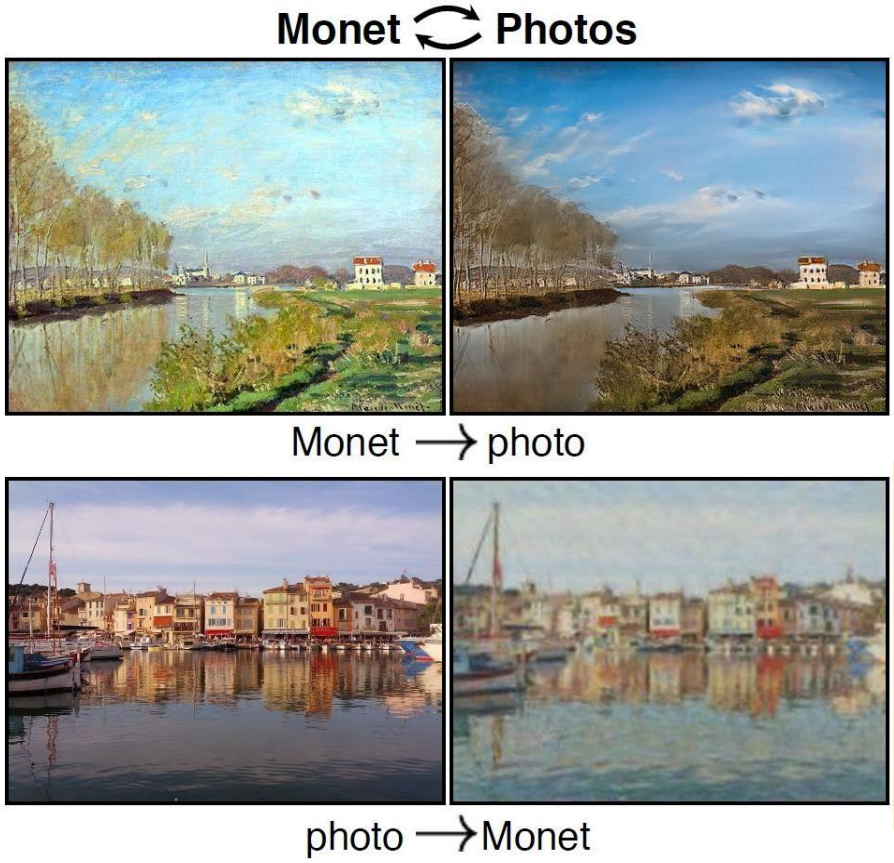
(b) A hypothesis clustering.

Unsupervised Learning (Cont.)

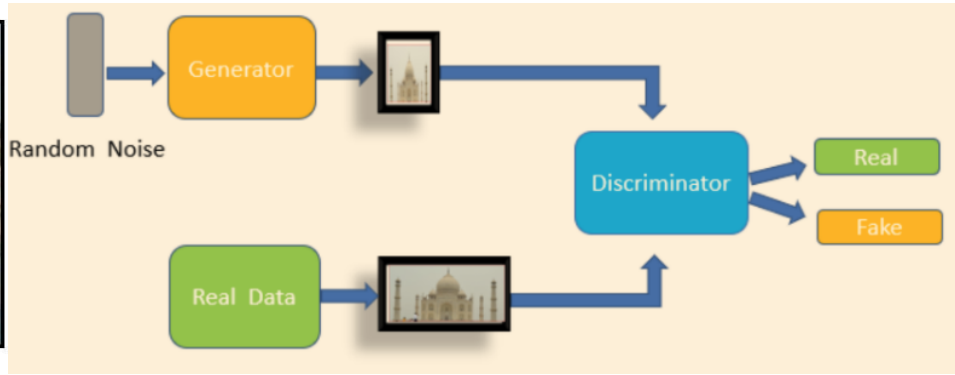


Topic modelling - Latent Dirichlet Allocation (LDA)

Unsupervised Learning (Cont.)



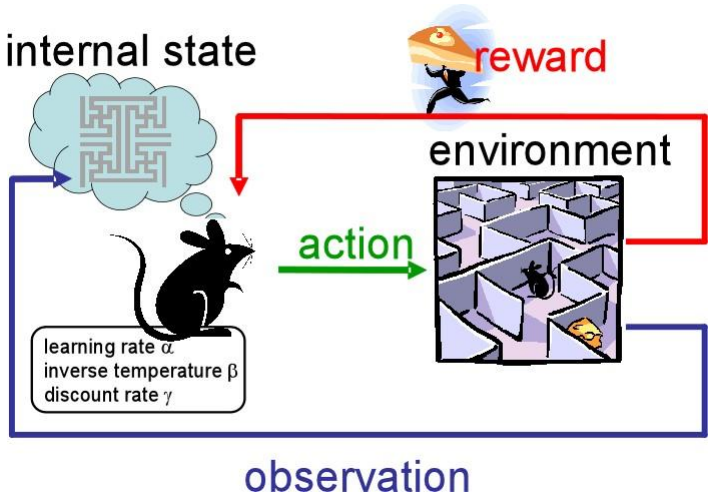
Models are learned from unlabelled data



Reinforcement Learning

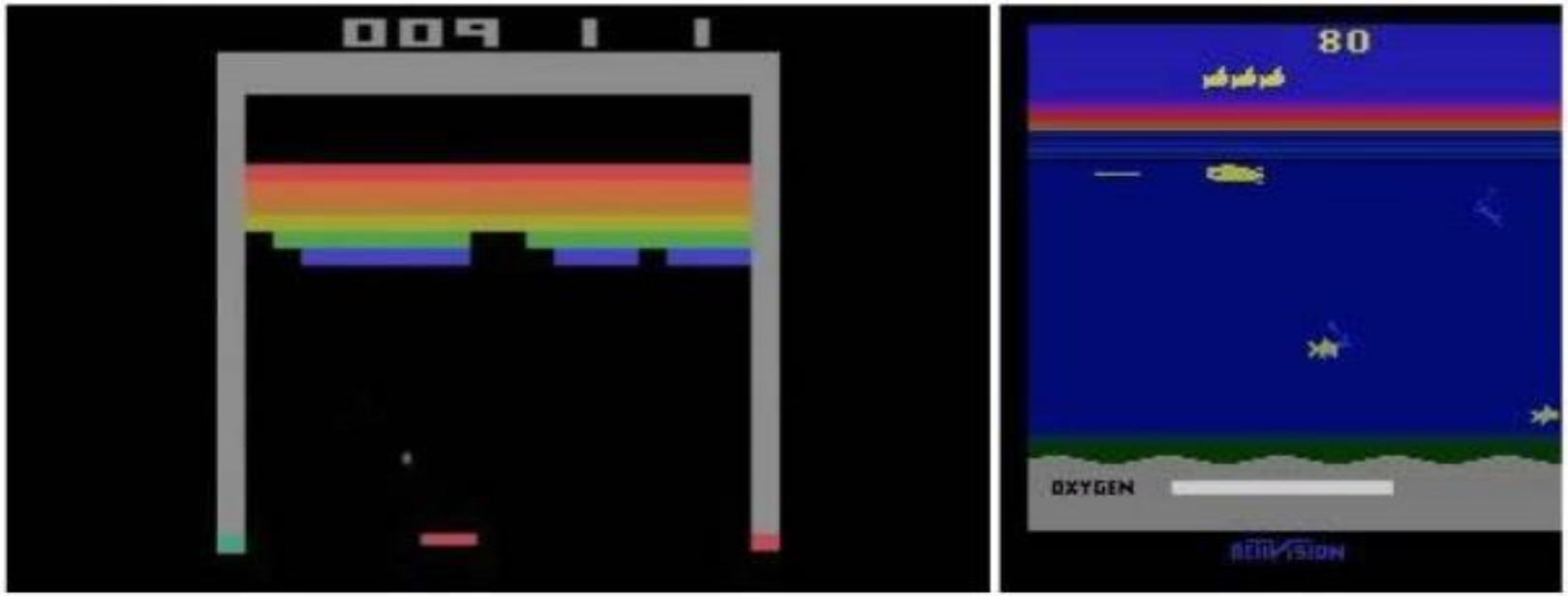
Learning what actions to take in order to maximise some **reward** or **utility**.

The reward (or penalty) is given after a **sequence of actions**. This differs from supervised learning in that explicit input-output examples are not available.



Reinforcement Learning

Playing Atari with deep reinforcement learning



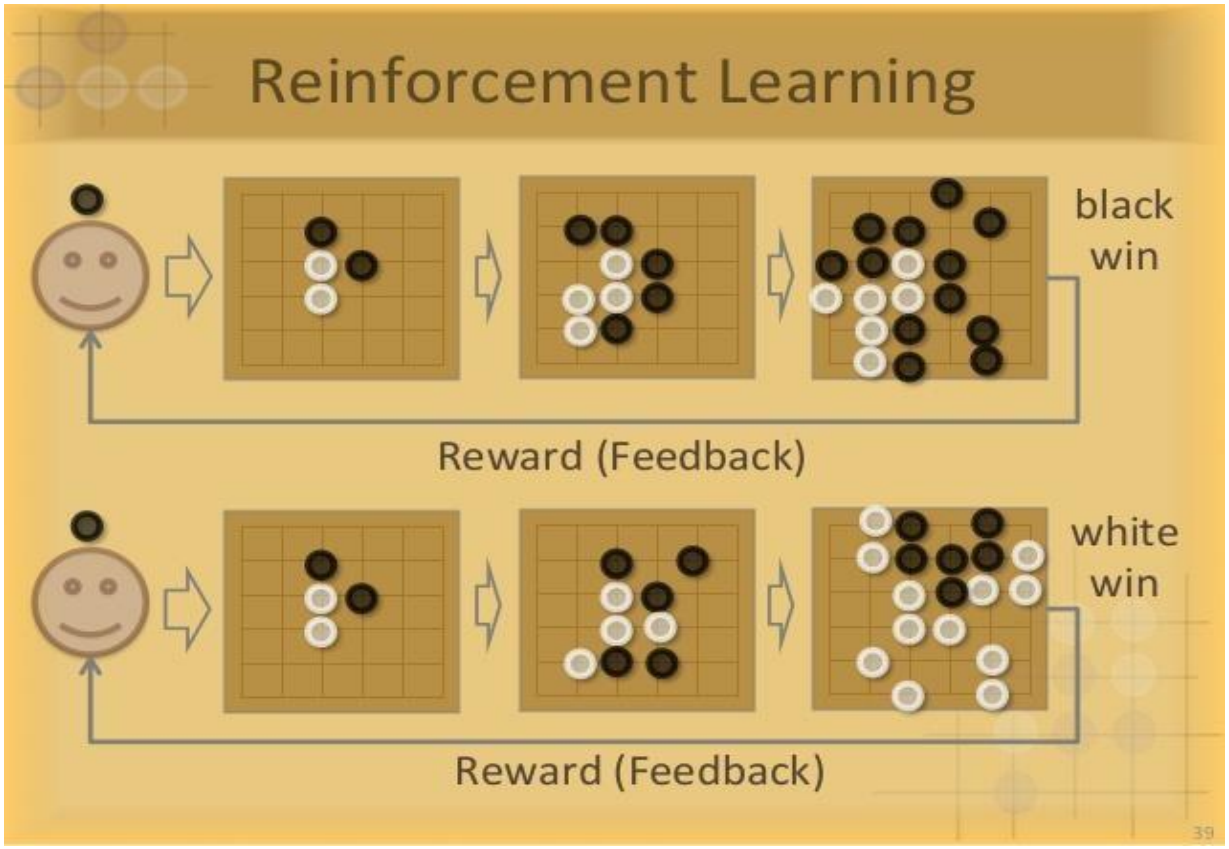
At Google Deepmind

At ULg

Human-level control through deep reinforcement learning. Nature, 2015.

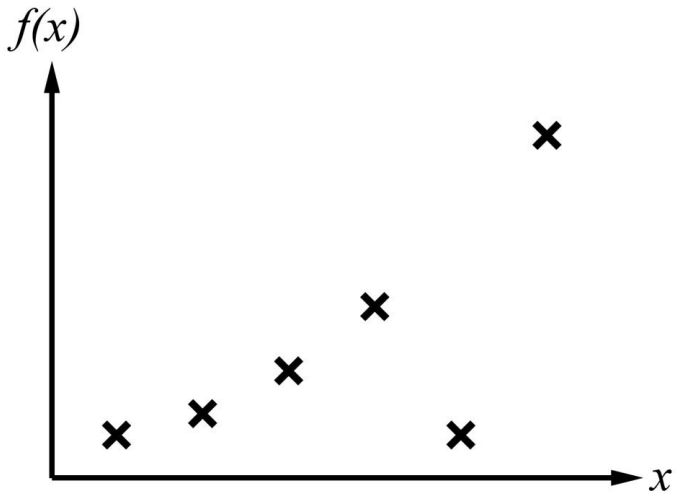
Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg & Demis Hassabis

Reinforcement Learning



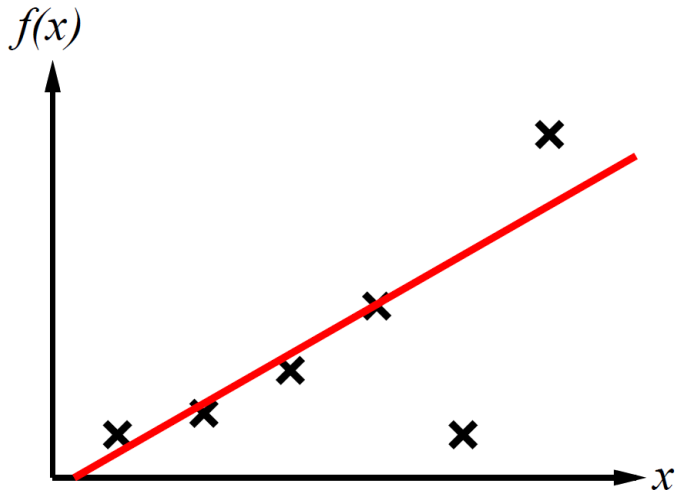
Complexity of hypothesis

Back to the regression problem. What is the best hypothesis?



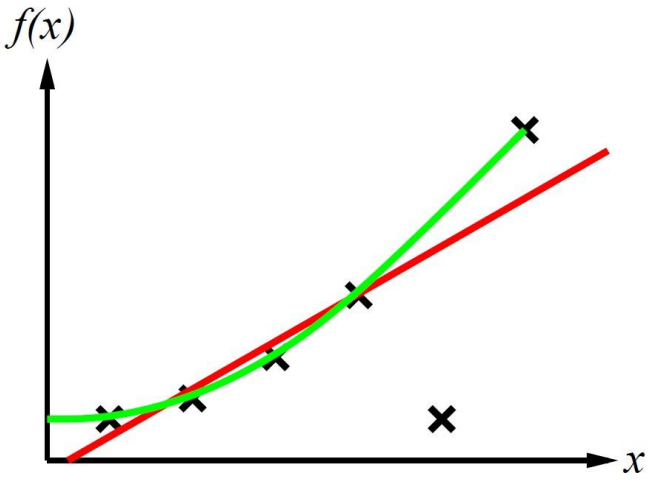
Complexity of hypothesis

Use a linear hypothesis? It doesn't agree with all the points...



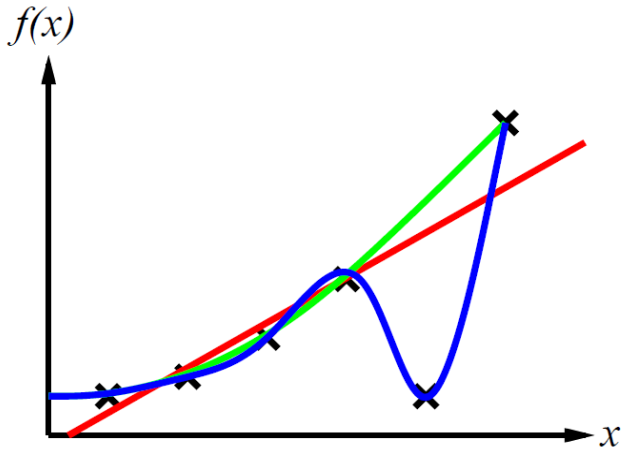
Complexity of hypothesis

Use a non-linear polynomial hypothesis? Slightly better...



Complexity of hypothesis

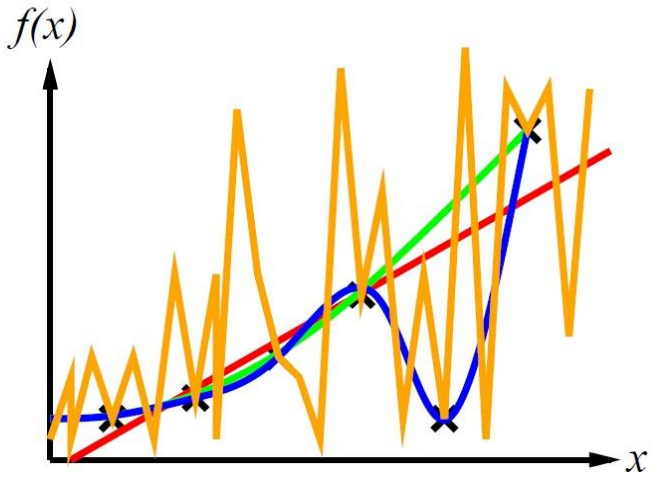
Increase the degree of the polynomial to fit all points.



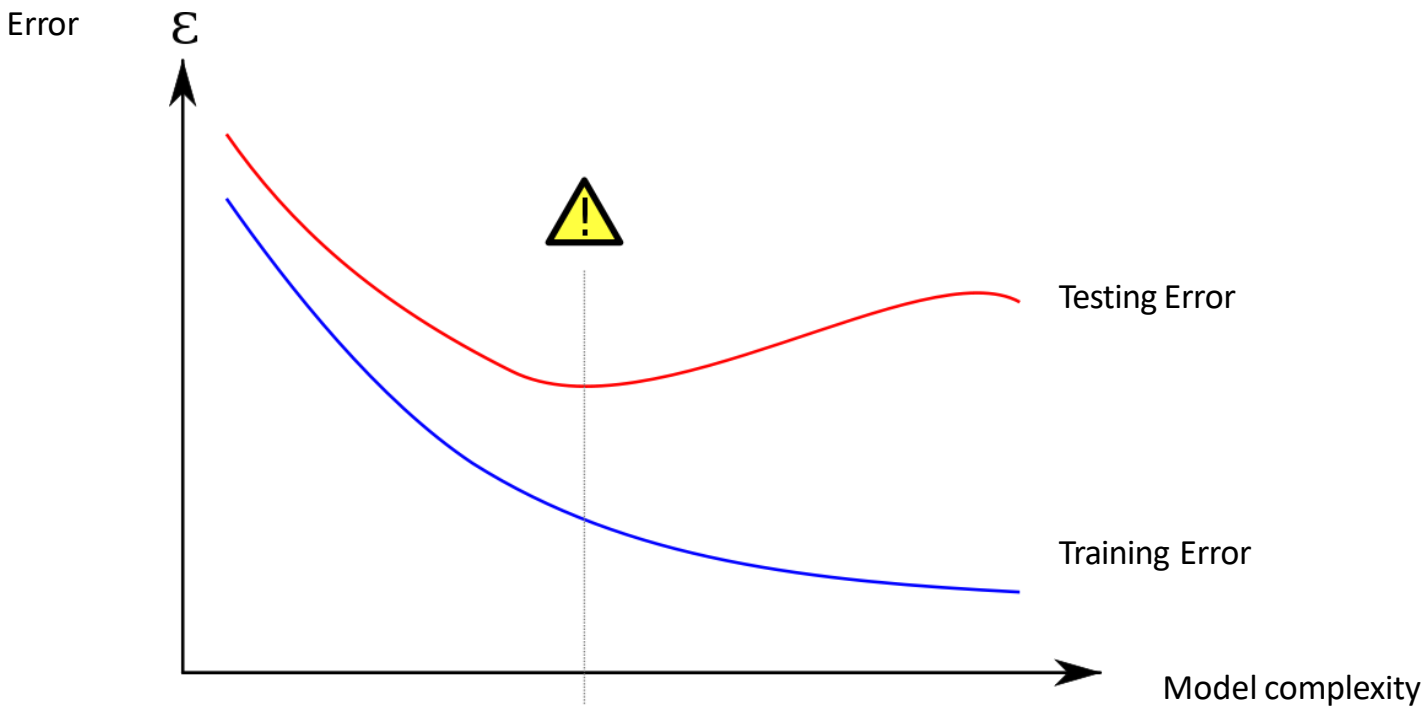
A hypothesis is **consistent** if it agrees with all the data.

Complexity of hypothesis

What about this one? Which hypothesis should we choose?



Complexity of hypothesis



Occam's Razor

"Plurality should not be posited without necessity."

William of Ockham

"Everything should be made as simple as possible, but not simpler."

Albert Einstein

A hypothesis can be regarded as an explanation of the data.

Occam's Razor states that we should prefer the simplest hypothesis consistent with the data.

If we accept that the data are noisy versions of the true data generated from f, it might be better to fit a simple hypothesis which is not exactly consistent with the data.

This **tradeoff** between hypothesis complexity and goodness-of-fit will crop up again in the learning algorithms we study later.

Nearest Neighbour Method

Introduction

Given a testing/query point, Nearest Neighbours performs a search for the point(s) closest to the testing point in the training data.

Nearest Neighbours is one of the simplest methods for statistical learning. It belongs to a group of techniques called "non-parametric learning methods" because it does not make use of any assumed model for the data.

When properly harnessed Nearest Neighbours can provide excellent classification accuracy despite its simplicity.

K-NN

Nearest Neighbours is primarily a **supervised learning** algorithm.

Let $X = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ be a set of N training data, where $\mathbf{x}_i$ is the i-th feature point, and y_i is the class label of $\mathbf{x}_i$. Note that $\mathbf{x}_i$ can be **multi-dimensional**.

Given a testing point $\mathbf{z}$ we wish to search among X the K points

$$\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_K^*$$

that are closest to $\mathbf{z}$, i.e. the K **nearest neighbours** of $\mathbf{z}$. It is almost always the case that $K < N$.

The testing point $\mathbf{z}$ is then assigned the **mode label** (label with highest count) among its K nearest neighbours:

$$mode(y_1^*, y_2^*, \ldots, y_K^*)$$

K-NN (Cont.)

The concept of "nearness" requires a way of measuring the distance between any two pairs of points. Under K-nn the distance used is frequently the **Euclidean** distance:

$$d(\mathbf{z}, \mathbf{x}) = \sqrt{\sum_{j=1}^{M} (\mathbf{z}[j] - \mathbf{x}[j])^2}$$

This is essentially the **straight line distance** between two points.

Example 1: If $\mathbf{x} = \begin{bmatrix} 0.5 \\ 2 \end{bmatrix}$ and $\mathbf{z} = \begin{bmatrix} 2.3 \\ -1.5 \end{bmatrix}$ then

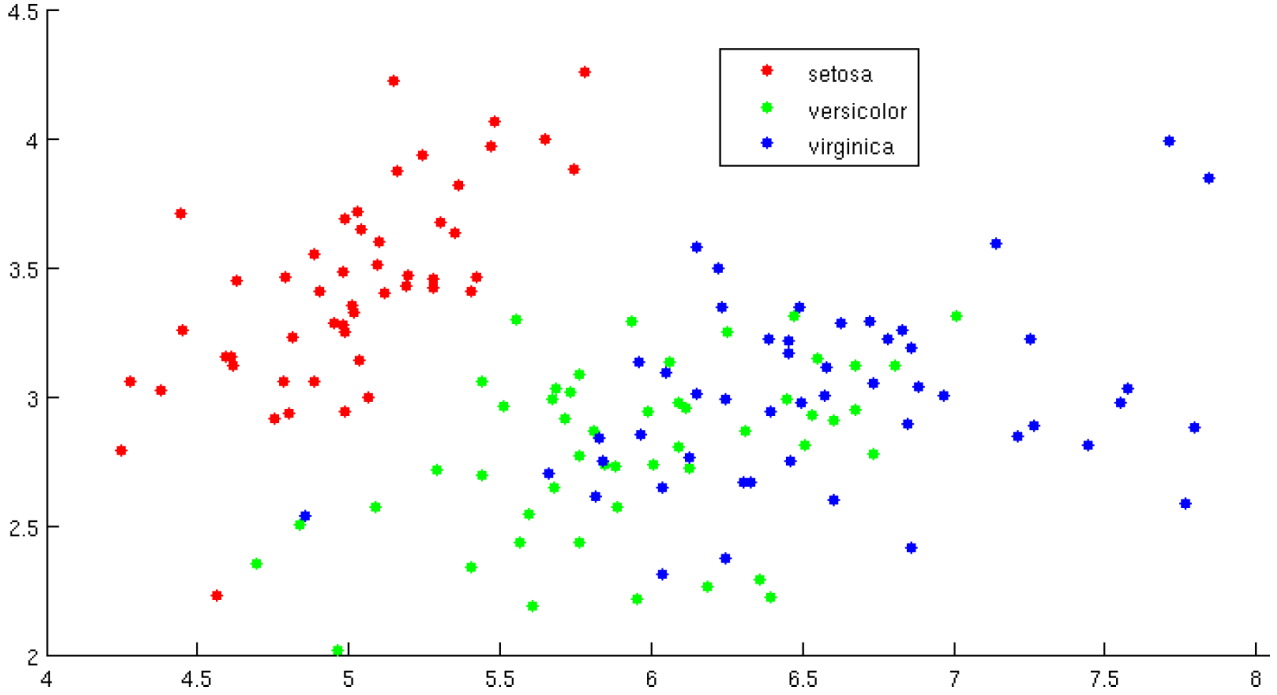
$$d(\mathbf{z}, \mathbf{x}) = \sqrt{(0.5 - 2.3)^2 + (2 - (-1.5))^2} = \sqrt{15.49}$$

Example 2: If $\mathbf{x} = \begin{bmatrix} -12 \\ 0 \\ -9.8 \end{bmatrix}$ and $\mathbf{z} = \begin{bmatrix} 13 \\ -1.5 \\ -7 \end{bmatrix}$ then

$$d(\mathbf{z}, \mathbf{x}) = \sqrt{(-12 - 13)^2 + (0 - (-1.5))^2 + (-9.8 - (-7))^2} = \sqrt{635.09}$$

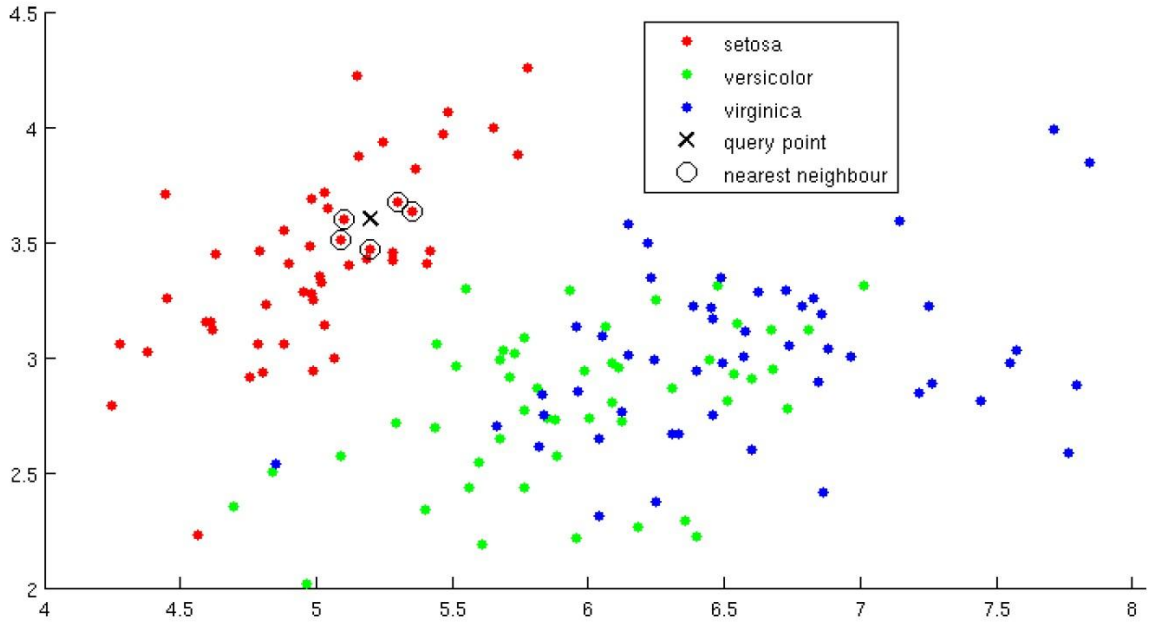
Iris data again

We use the Iris data again. This time we have 3 types of irises: Setosa, Versicolour and Virginica.



Iris data again

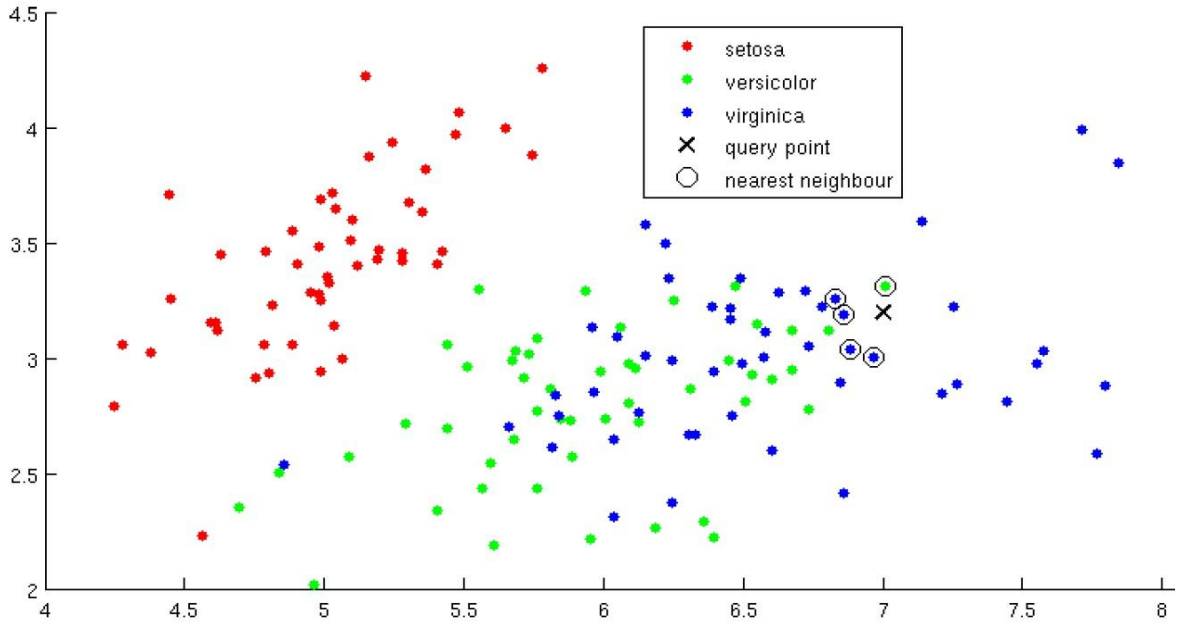
Given a testing point $\mathbf{z} = [\ 5.2\ \ 3.6\ ]^T$, we find its 5 nearest neighbours:



Observe that all the nearest neighbours are labelled Setosas. Thus $\mathbf{z}$ is assigned the label Setosa as well.

Iris data again

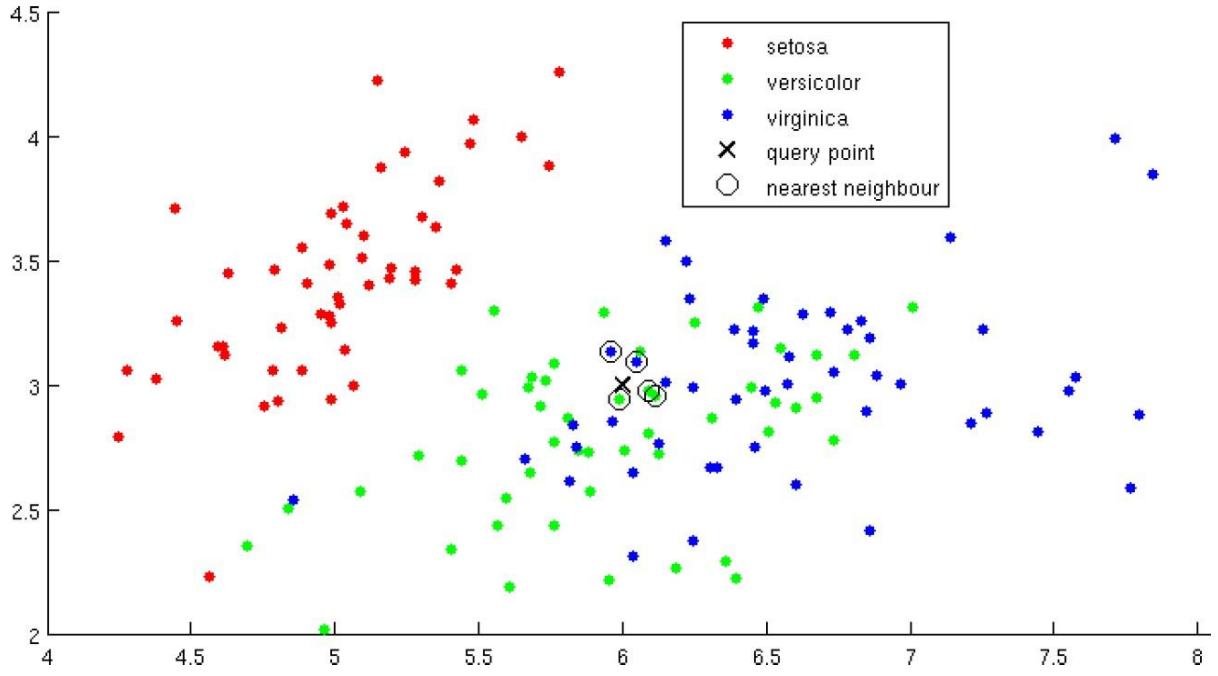
Given another testing point $\mathbf{z} = [\ 7.0 \quad 3.2\ ]^T$, the 5 nearest neighbours are:



The label for $\mathbf{z}$ is taken as the mode among 1 Versicolour and 4 Virginicas, thus $\mathbf{z}$ is assigned the label of Virginica.

Iris data again

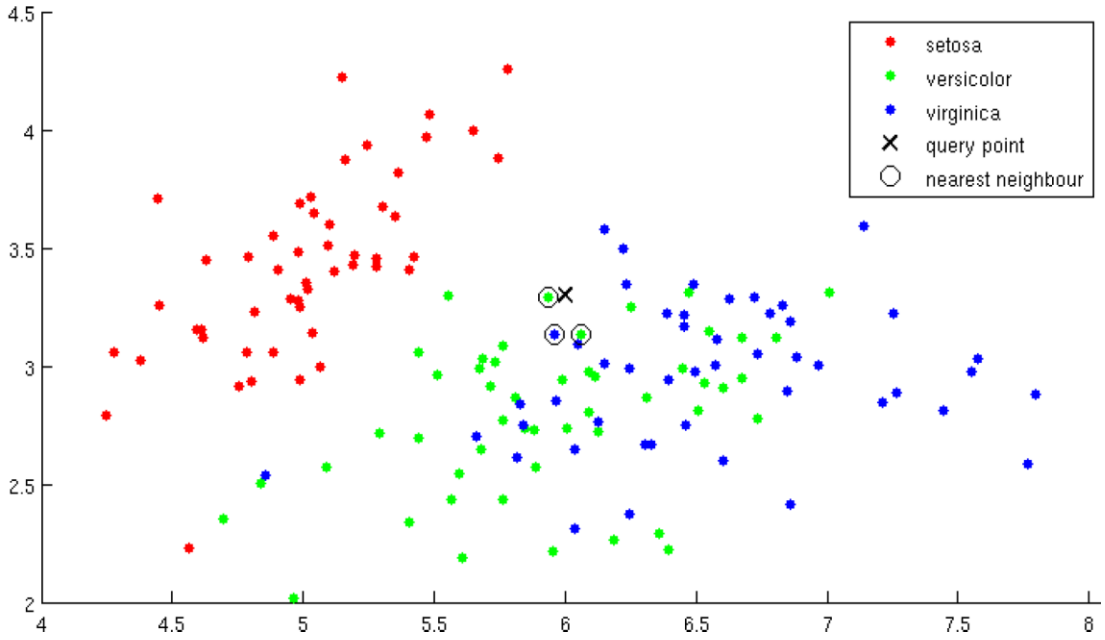
Similarly, the testing point $\mathbf{z} = [\ 6.0 \quad 3.0\ ]^T$ with 5 nearest neighbours is then labelled as Versicolour:



K is a free parameter

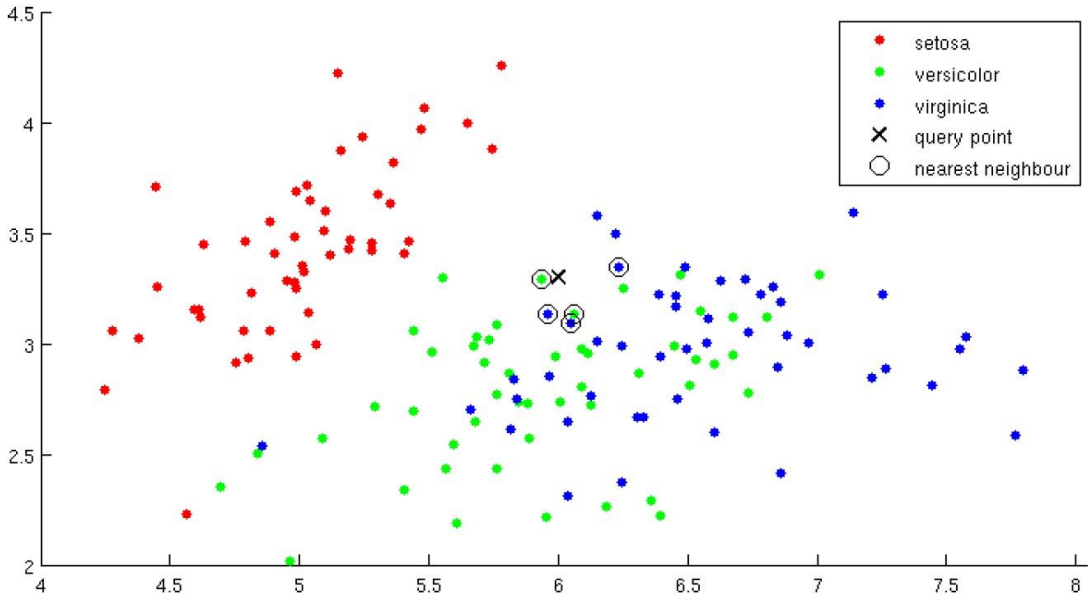
It is obvious that using different values of K will give different labelling results, especially at the boundaries between classes.

For example, for the testing point $\mathbf{z} = [\ 6.0 \quad 3.3\ ]^T$ with $k = 3$ gives the label of Versicolour.



K is a free parameter (Cont.)

Testing the same point of $\mathbf{z} = [\ 6.0 \quad 3.3\ ]^T$ with $k = 5$ however gives the label of Virginica.

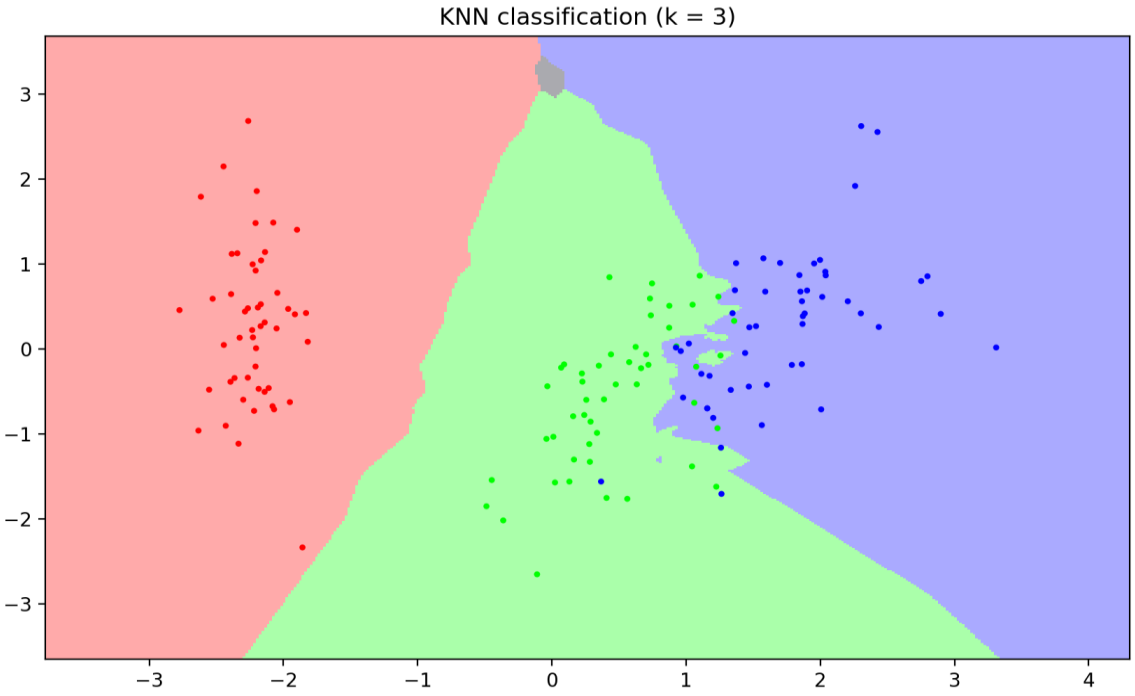


For most problems, setting the appropriate K is crucial. One often tunes K in a **trial-and-error** fashion (another drawback of K-nn).

K is a free parameter (Cont.)

One way of visualising the effects of different K is to plot the **decision areas** for the data. We subject every location (at suitable increments) to the K-NN classifier and set the colour accordingly.
Here are the decision areas for K=3.

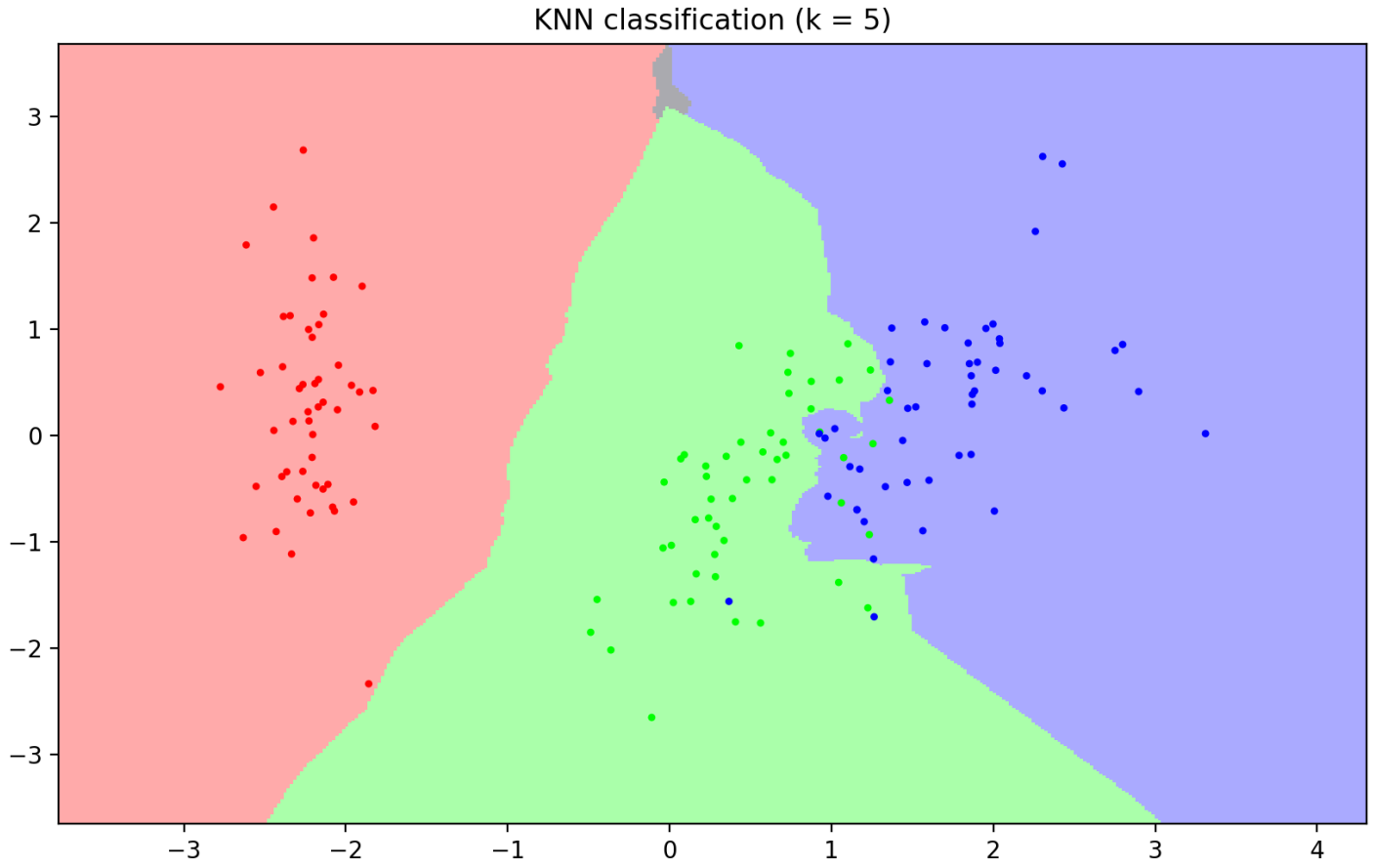
Note that in order to show real performance, we use all the 4 features in iris data and plot after a dimension reduction.



Coloured grey areas are with no classification, i.e., **no unique mode labels.**

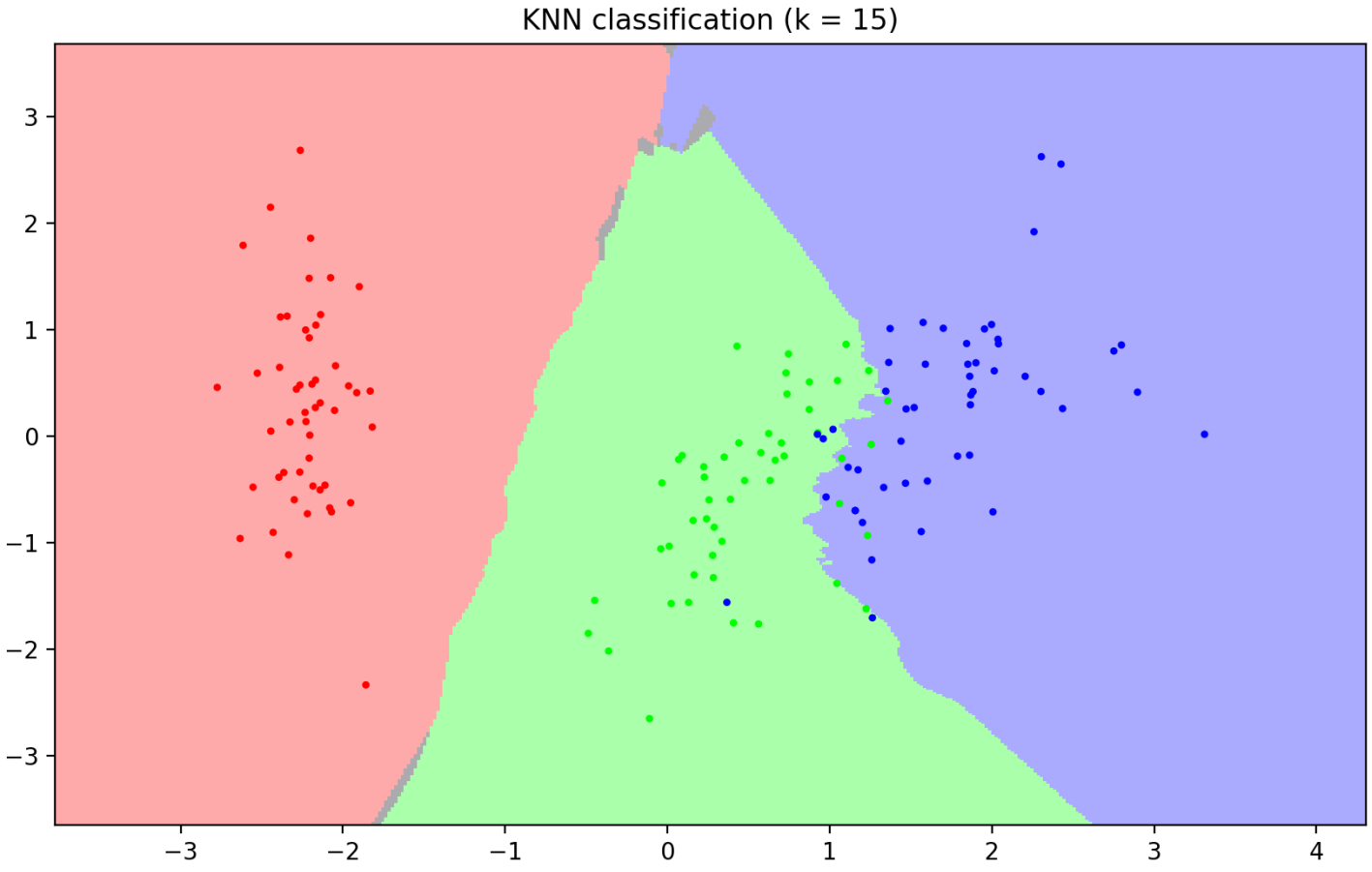
K is a free parameter (Cont.)

Here are the decision areas for K=5.



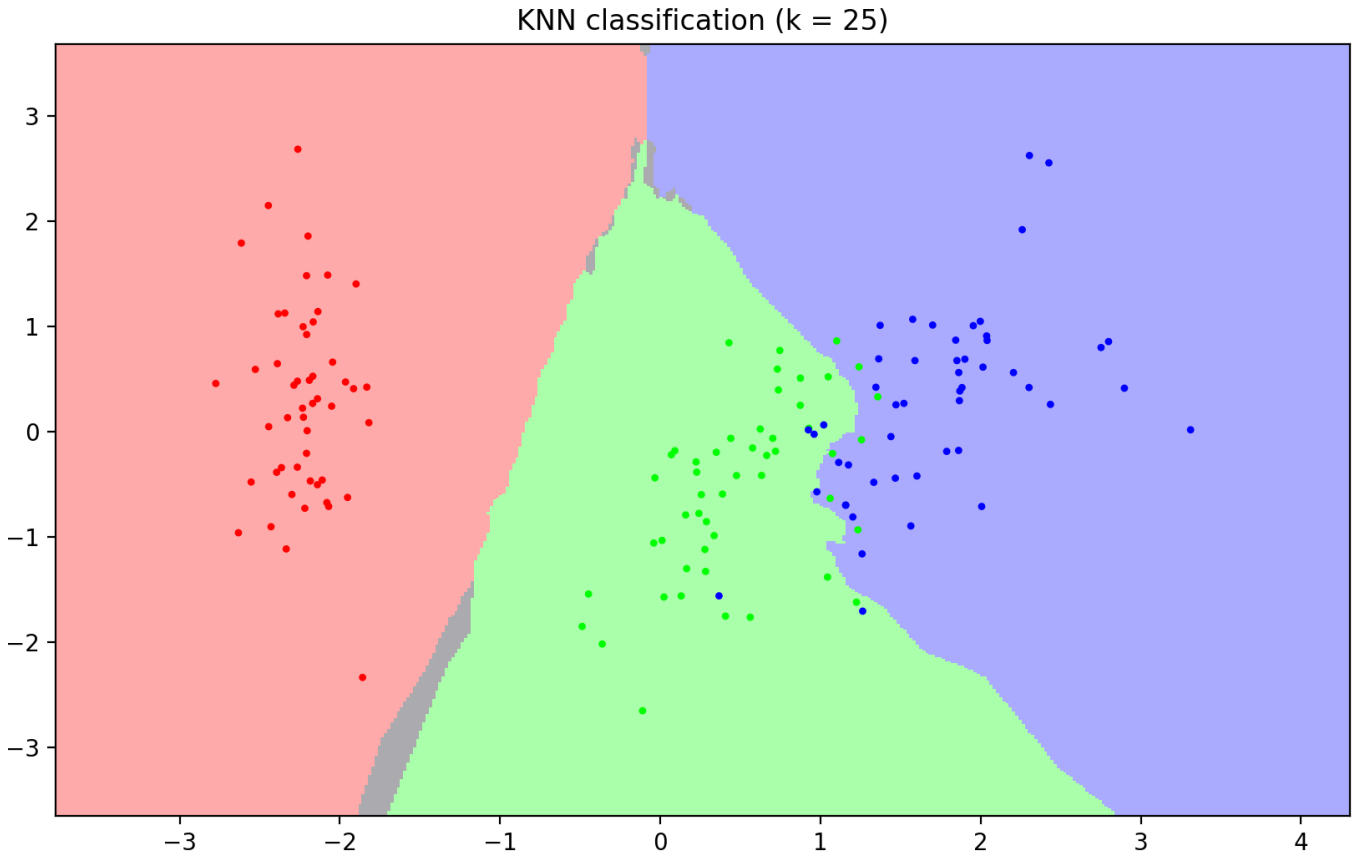
K is a free parameter (Cont.)

Here are the decision areas for K=15.



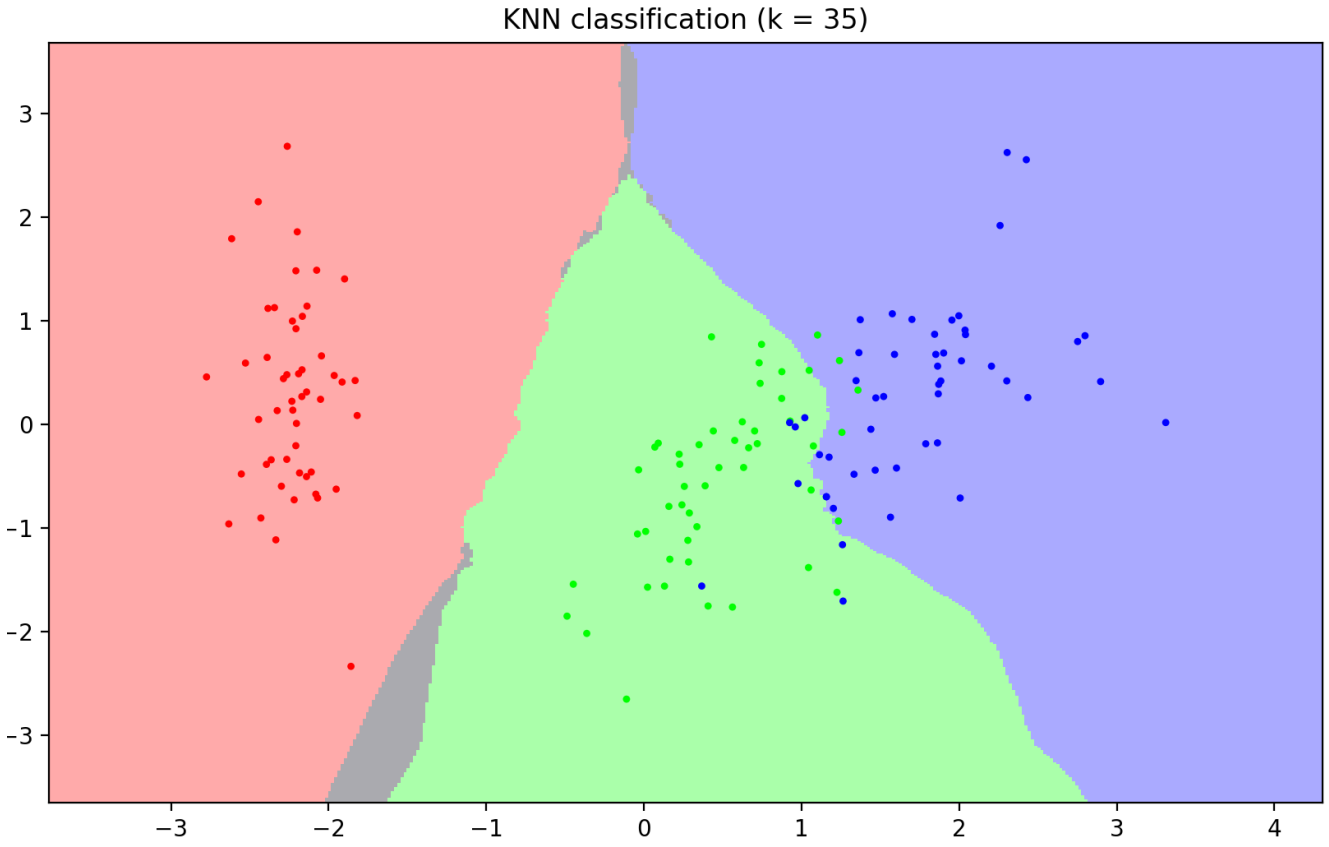
K is a free parameter (Cont.)

Here are the decision areas for K=25.



K is a free parameter (Cont.)

Here are the decision areas for K=35.



It can be seen that as K gets larger, the 'holes' starts to fill-in, but the grey areas grow bigger.

The K-NN algorithm with exhaustive search

- Note that the algorithm is exhaustively computing the distances between **z** (a test data) and all the other points in the training data. This might not be efficient.

- It is also interesting to note that K-NN is a ‘learning algorithm’ that actually does not involve learning/training. Once a set of labelled training data is available, you can start testing a new point.

Concluding remarks

- K-NN is one of the simplest methods for supervised learning. Given a classification task, one should test with K-NN first as a baseline, before attempting more complicated methods.

- However, computational efficiency is an issue. Among the 3 constants N (the number of training samples), M (the number of features) and K that effect the total running time of the algorithm, N and M usually have the largest effects.
  - In many practical problems, N can be massive (possibly in the range of millions), while M can also be large.