LMU LUDWIG-MAXIMILIANS-UNIVERSITÄT MÜNCHEN

FCIM / Predictive Modeling

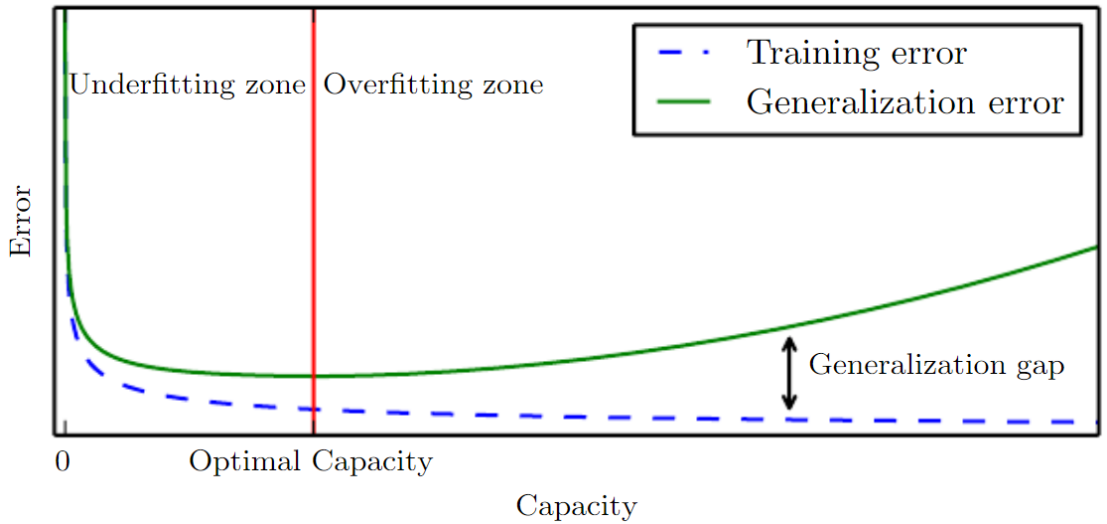
Chapter 3: Hypothesis Spaces and Capacity

Bernd Bischl, Julia Moosbauer

Department of Statistics – LMU Munich

Summer term 2019

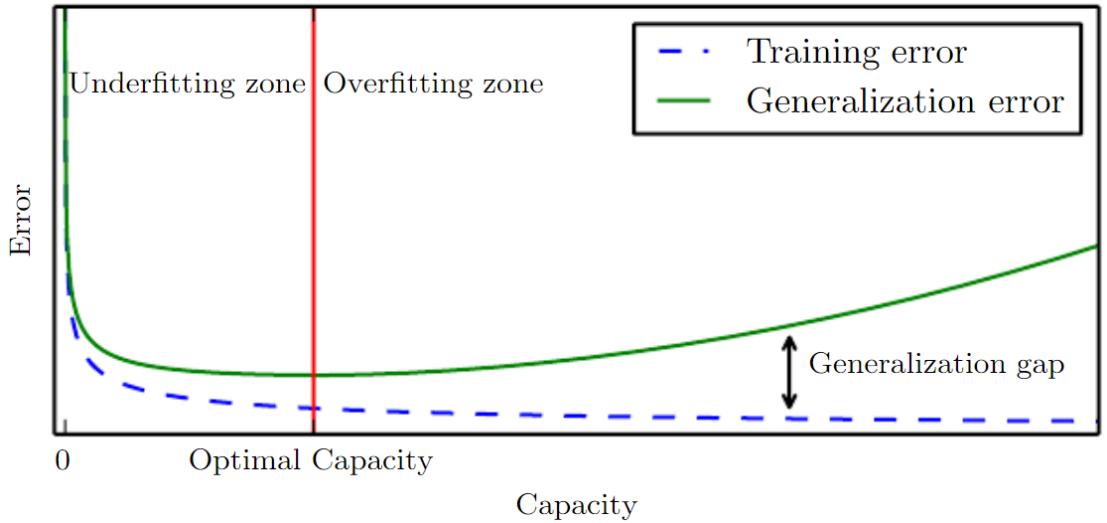
CAPACITY



credit: Ian Goodfellow

- The performance of a learner depends on its ability to:
  - Minimize the training error
  - Generalize well to new data
- Failure to obtain a sufficiently low training error is known as **underfitting**.
- On the other hand, if there is a large difference in training and test error, this is known as **overfitting**.

CAPACITY



credit: Ian Goodfellow

- The tendency of a model to over/underfit is a function of its **capacity**, determined by the type of hypotheses it can learn.
- Loosely speaking, a model with low capacity can only learn a few simple hypotheses, whereas a model with a large capacity can learn many, possibly complex hypotheses.
- As the figure shows, the test error is minimized when the model neither underfits nor overfits, that is, when it has the right capacity.

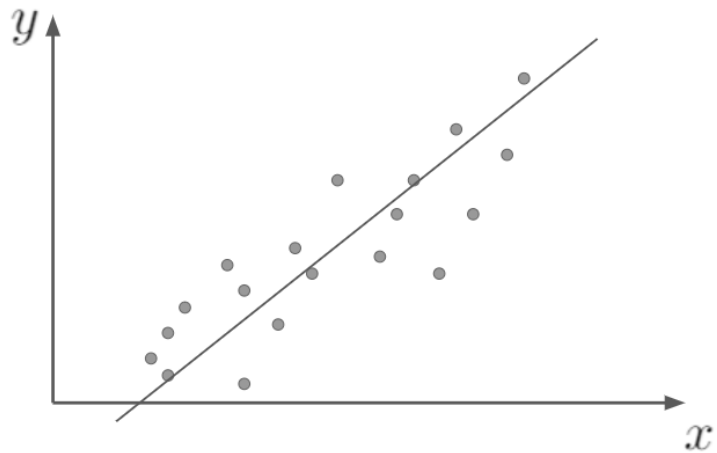
HYPOTHESIS SPACES

Recall, the **representation** of a learner is the space of allowed models (also **hypothesis space**). The hypothesis space $\mathcal{H}$ is a space of functions that have a certain functional form:

$$\mathcal{H} := \{ f : \mathcal{X} \to \mathbb{R}^g \mid f \text{ has a specific form (often parametrized by } \boldsymbol{\theta} \in \Theta \text{)} \}$$

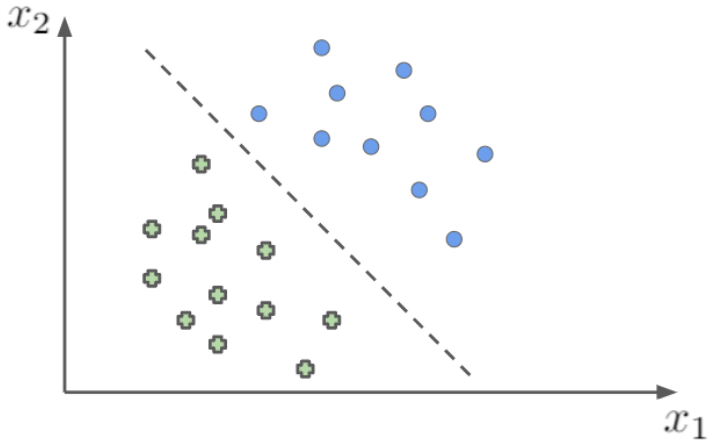
Examples of hypothesis spaces :

- **Linear regression** : $f(x) = x^T\theta + \theta_0$

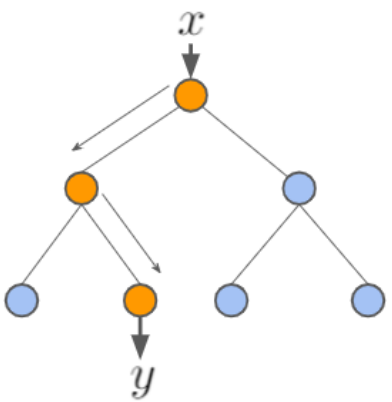


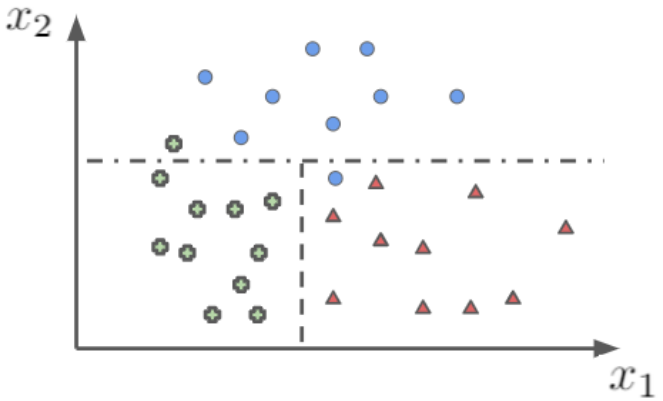
HYPOTHESIS SPACES

- **Separating hyperplanes** : $f(x) = \mathbb{I}[\mathbf{x}^T\boldsymbol{\theta} - \theta_0 > 0]$



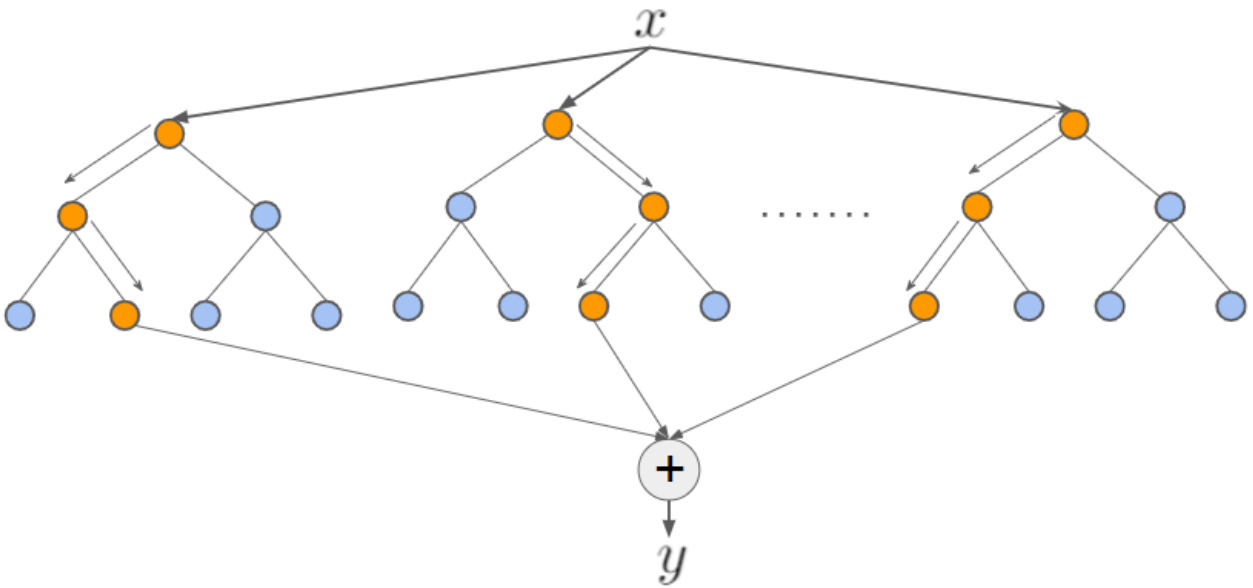
- **Decision trees** : $f(x) = \sum_i^m c_m \mathbb{I}(x \in R_m)$ (Recursively divides the feature space into axis-aligned rectangles.)





HYPOTHESIS SPACES

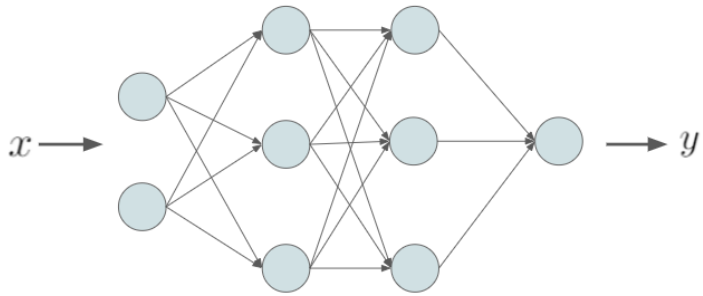
- **Ensemble methods** : $f(x) = \sum_1^m \beta^{[m]} b^{[m]}(x)$ (For a given input, the predictions of several models are aggregated in some manner.)



HYPOTHESIS SPACES

Neural networks :

$$f(x) = \tau \circ \phi \circ \sigma^{(h)} \circ \phi^{(h)} \circ \sigma^{(h-1)} \circ \phi^{(h-1)} \circ \ldots \circ \sigma^{(1)} \circ \phi^{(1)}(x)$$



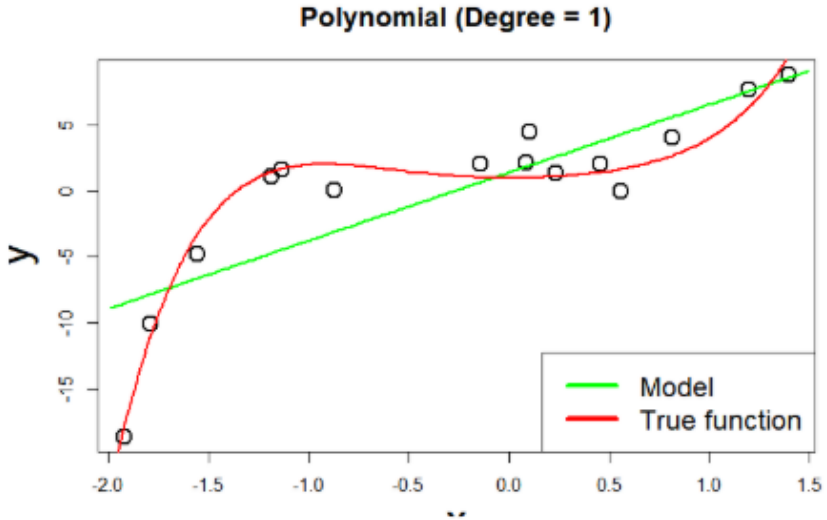
- A neural network consists of layers of simple computational units known as 'neurons' (there are h such layers in the formula above).
- Each neuron in a given layer performs a two-step computation : a weighted sum (ϕ) of its inputs from the previous layer followed by a non-linear transformation (σ) of the sum.
- The network as a whole represents a nested composition of such operations.

OVERFITTING

- The capacity (or “complexity”) of a model can be increased by increasing the size of the hypothesis space.
- This (usually) also increases the number of learnable parameters.
- Examples: Increasing the degree of the polynomial in linear regression, increasing the depth of a decision tree or a neural network, adding additional predictors, etc.
- As the size of the hypothesis space increases, the tendency of a model to overfit also increases.
- Such a model might fit even the random quirks in the training data, thereby failing to generalize.

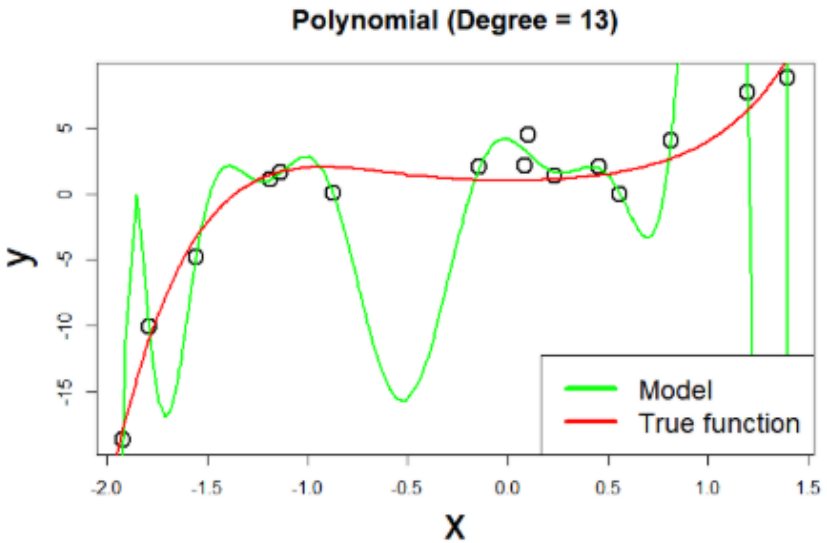
OVERFITTING: POLYNOMIAL REGRESSION

Degree = 1 (highest degree of a term in the polynomial)



Underfitting
(Low Capacity)

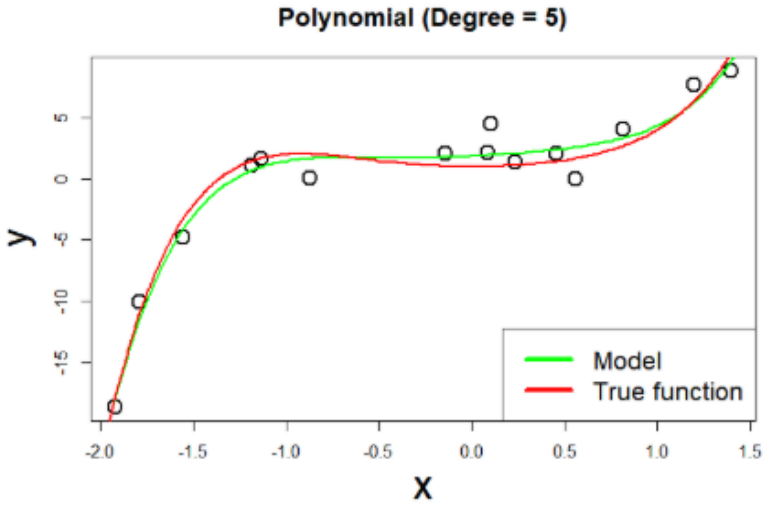
Degree = 13



Overfitting
(High Capacity)

OVERFITTING: POLYNOMIAL REGRESSION

Degree = 5

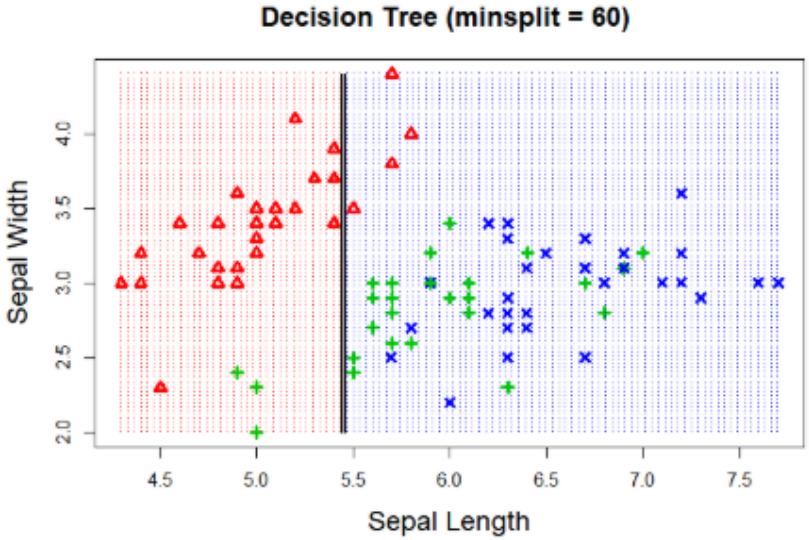


Good fit (Appropriate capacity)

	Degree = 1	Degree = 5	Degree = 13
Training error (RMSE)	3.87	1.23	0.48
Test error (RMSE)	4.11	**1.55**	148.5

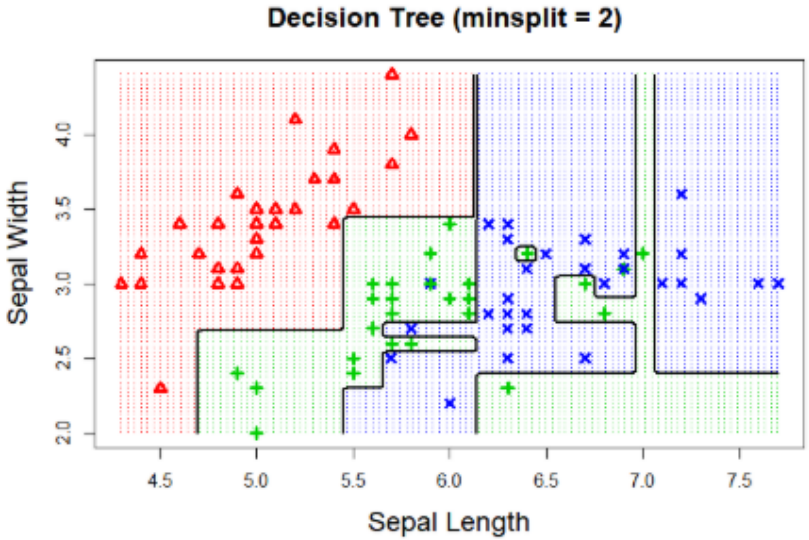
OVERFITTING : DECISION TREES

minsplit = 60 (minimum number of samples in a node being split)



Underfitting
(Low Capacity)

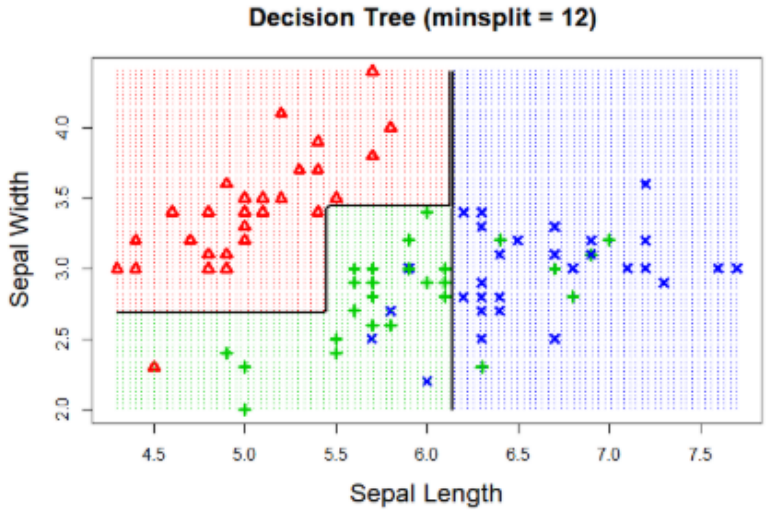
minsplit = 2



Overfitting
(High Capacity)

OVERFITTING : DECISION TREES

minsplit = 12

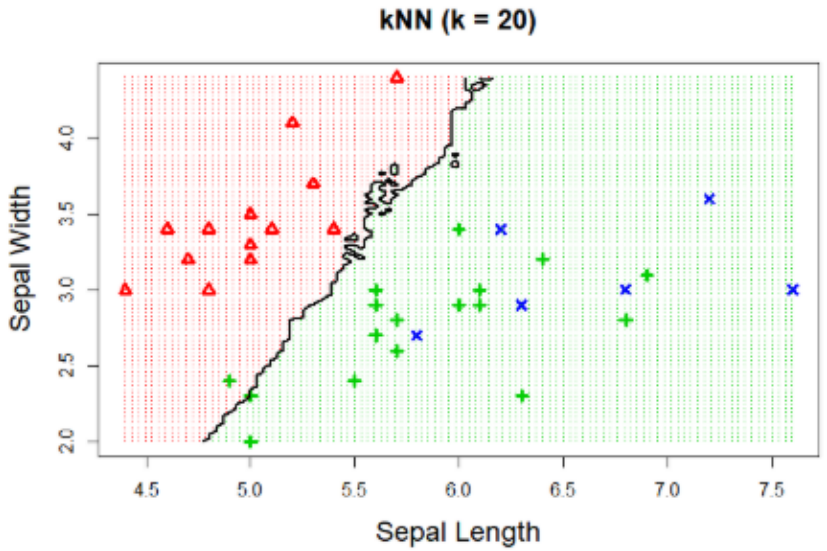


Good fit (Appropriate capacity)

	minsplit = 60	minsplit = 12	minsplit = 2
Training error (Misclassification)	0.36	0.12	0.02
Test error (Misclassification)	0.40	**0.32**	0.35

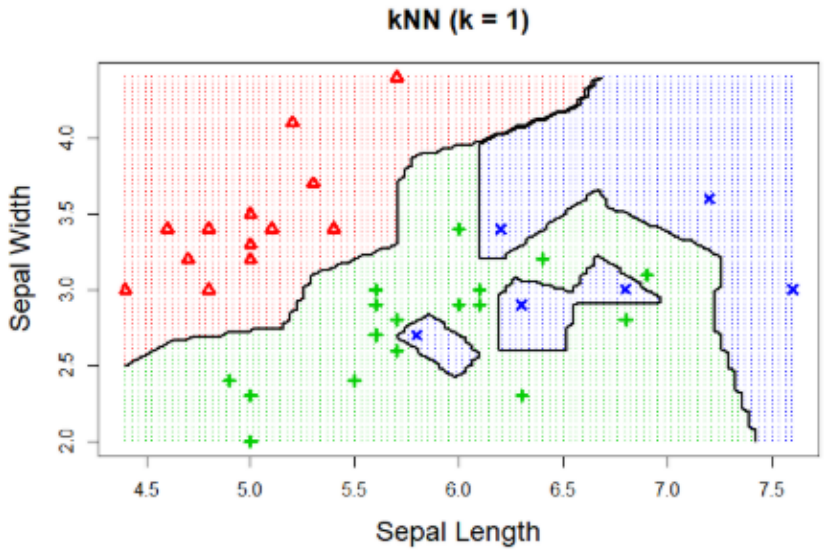
OVERFITTING : K-NEAREST NEIGHBOURS

k = 20
(number of neighbours)



Underfitting
(Low Capacity)

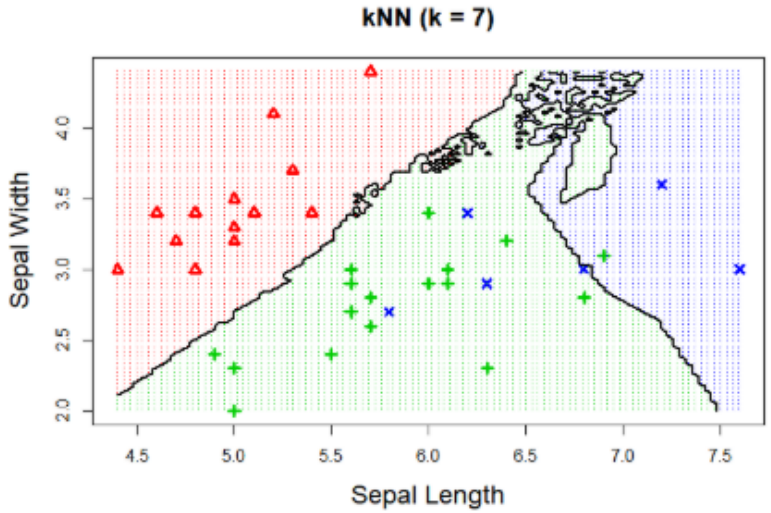
k = 1



Overfitting
(High Capacity)

OVERFITTING : K-NEAREST NEIGHBOURS

k = 7



Good fit (Appropriate capacity)

	k = 20	k = 7	k = 1
Training error (Misclassification)	0.22	0.13	0
Test error (Misclassification)	0.40	**0.25**	0.33

THE COMPLEXITY OF HYPOTHESIS SPACES

A general measure of the complexity of a function space is the **Vapnik-Chervonenkis (VC)** dimension.

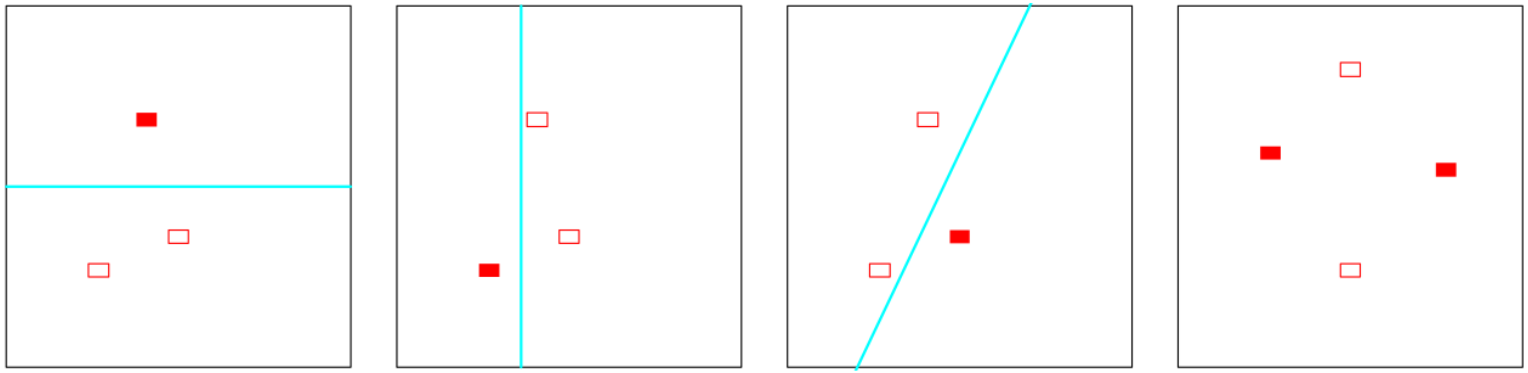
The **VC dimension** of a class of binary-valued functions $\mathcal{H} = \{f(x|\theta)\}$ is defined to be the largest number of points (in some configuration) that can be "shattered" by members of $\mathcal{H}$. We write $VC(\mathcal{H})$.

A set of points is said to be **shattered** by a class of functions if a member of this class can perfectly separate them no matter how we assign binary labels to the points.

Note: If the VC-dimension of a hypothesis class is d, it doesn't mean that **all** sets of size d can be shattered. Rather, it simply means that there is at least **one** such set which can be shattered and **no** set of size $d + 1$ which can be shattered.

THE COMPLEXITY OF HYPOTHESIS SPACES

Example:



For $\mathbf{x} \in \mathbb{R}^2$, the class of linear indicator functions
$\mathcal{H} = \{f(\mathbf{x}) \mid f(\mathbf{x}) = \mathbb{I}[\mathbf{x}^T\boldsymbol{\theta} - \theta_0 > 0]\}$

- can shatter 3 points: No matter how we assign labels to the configuration of three points shown above, we can find a linear line separating them perfectly;
- cannot shatter no configuration of 4 points;

Hence $VC(\mathcal{H}) = 3$.

THE COMPLEXITY OF HYPOTHESIS SPACES

Theorem : The VC dimension of the class of homogeneous halfspaces, $\mathcal{H} = \{f(\mathbf{x}) \mid f(\mathbf{x}) = \text{sign}(\mathbf{x}^T\theta)\}$, in $\mathbb{R}^p$ is p.

(**Proof**) p as a lower bound : Consider the set of standard basis vectors $\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_p$ in $\mathbb{R}^p$. For every possible labelling $y_1, y_2, \ldots y_p$, where $y_i \in \{-1, +1\}$, if we set $\theta = (y_1, y_2, \ldots y_p)$, then $\langle \theta, \mathbf{e}_i \rangle = \mathbf{e}_i^T\theta = y_i$, for all i. Therefore, the p points are shattered.

p as an upper bound :

- Let $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{p+1}$ be a set of $p + 1$ vectors in $\mathbb{R}^p$.
- Because any set of $p + 1$ vectors in $\mathbb{R}^p$ is linearly dependent, there must exist real numbers $a_1, a_2, \ldots, a_{p+1}$, not all of them zero, such that $\sum_{i=1}^{p+1} a_i\mathbf{x}_i = 0$.
- Let $I = \{i : a_i > 0\}$ and $J = \{j : a_j < 0\}$. Either I or J is nonempty.

THE COMPLEXITY OF HYPOTHESIS SPACES

If we assume both I and J are nonempty, then :

- $\sum_{i \in I} a_i \mathbf{x}_i = \sum_{j \in J} |a_j| \mathbf{x}_j$
- Let's assume $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{p+1}$ are shattered by this class.
- there must exist a vector θ such that $\langle \theta, \mathbf{x}_i \rangle > 0$ for all $i \in I$ while $\langle \theta, \mathbf{x}_j \rangle < 0$ for all $j \in J$.
- This implies

$$0 < \sum_{i \in I} a_i \langle \mathbf{x}_i, \theta \rangle = \left\langle \sum_{i \in I} a_i \mathbf{x}_i, \theta \right\rangle$$

$$= \left\langle \sum_{j \in J} |a_j| \mathbf{x}_j, \theta \right\rangle = \sum_{j \in J} |a_j| \langle \mathbf{x}_j, \theta \rangle < 0$$

which is a contradiction.

THE COMPLEXITY OF HYPOTHESIS SPACES

- This implies (copied from the previous slide)

$$0 < \sum_{i \in I} a_i \langle \mathbf{x}_i, \theta \rangle = \left\langle \sum_{i \in I} a_i \mathbf{x}_i, \theta \right\rangle$$
$$= \left\langle \sum_{j \in J} |a_j| \, \mathbf{x}_j, \theta \right\rangle = \sum_{j \in J} |a_j| \langle \mathbf{x}_j, \theta \rangle < 0$$

  which is a contradiction.

On the other hand, if we assume J (respectively, I) is empty, then the right-most (respectively, left-most) inequality should be replaced by an equality, which is still a contradiction.

THE COMPLEXITY OF HYPOTHESIS SPACES

Theorem : The VC dimension of the class of nonhomogeneous halfspaces, $\mathcal{H} = \{f(\mathbf{x}) \mid f(\mathbf{x}) = \text{sign}(\mathbf{x}^T\theta + \theta_0)\}$ in $\mathbb{R}^p$ is $p + 1$.

(**Proof**) $p + 1$ as a lower bound : Similar to the proof of the previous theorem, the set of basis vectors and the origin, that is, $0, \mathbf{e}_1, \ldots, \mathbf{e}_p$ can be shattered by nonhomogenous halfspaces.

$p + 1$ as an upper bound :

- Let's assume that $p + 2$ vectors $\mathbf{x}_1, \ldots \mathbf{x}_{p+2}$ are shattered by the class of nonhomogeneous halfspaces.
- If we denote $\theta' = (\theta_0, \theta_1, \theta_2, \ldots, \theta_p) \in \mathbb{R}^{p+1}$, where θ_0 is the bias/intercept, and $\mathbf{x}' = (1, x_1, x_2, \ldots x_p) \in \mathbb{R}^{p+1}$, then $f_{\theta,\theta_0}(\mathbf{x}) = \langle\theta, \mathbf{x}\rangle + \theta_0 = \langle\theta', \mathbf{x}'\rangle$. Therefore, any affine function in $\mathbb{R}^p$ can be rewritten as a homogeneous linear function in $\mathbb{R}^{p+1}$.

THE COMPLEXITY OF HYPOTHESIS SPACES

- By the previous theorem, the set of homogeneous halfspaces in $\mathbb{R}^{p+1}$ cannot shatter any set of $p + 2$ points. This leads us to a contradiction.

THE COMPLEXITY OF HYPOTHESIS SPACES

Example : For k-nearest neighbours with $k = 1$, the VC dimension is infinite.

Example : Let $\mathcal{H}$ be the class of axis-aligned rectangles

$$\mathcal{H} = \left\{ f_{(a_1,a_2,b_1,b_2)} : a_1 \leq a_2 \text{ and } b_1 \leq b_2 \right\}$$

where,

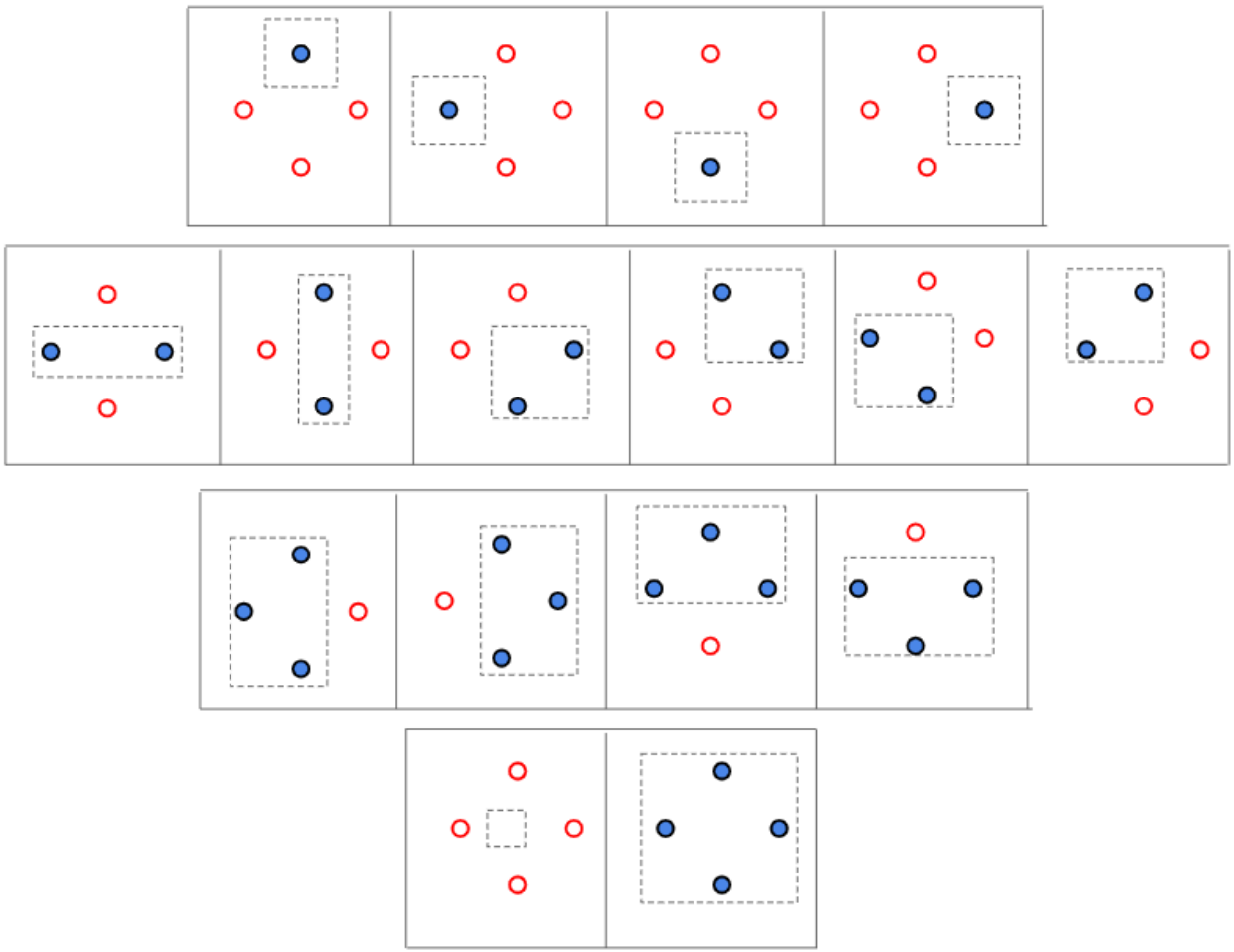
$$f_{(a_1,a_2,b_1,b_2)}(x_1, x_2) = \begin{cases} 1 & a_1 \leq x_1 \leq a_2 \text{ and } b_1 \leq x_2 \leq b_2 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

Claim : The VC-dimension of $\mathcal{H}$ is 4.

Proof : (next slide)

THE COMPLEXITY OF HYPOTHESIS SPACES

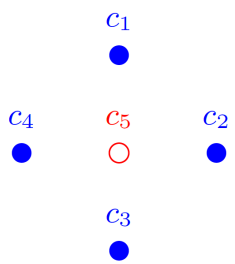
4 as a lower bound : There exists a set of 4 points that can be shattered.



THE COMPLEXITY OF HYPOTHESIS SPACES

4 as an upper bound : For any set C of 5 points in $\mathbb{R}^2$:

- Assign the leftmost point (lowest x_1) , rightmost point (highest x_1) , highest point (highest x_2) and lowest point (lowest x_2) to class 1. (In the figure, these are c_4, c_2, c_1 and c_3, respectively).
- The point not chosen, c_5, is assigned to class 0.
- Without loss of generality, such a labelling is impossible using an axis-aligned rectangle because the coordinates of c_5 are within the intervals defined by the other 4.



credit: Shai Shalev-Shwartz and Shai Ben-David

Therefore, the VC-dimension of axis-aligned rectangles is 4.

THE COMPLEXITY OF HYPOTHESIS SPACES

Recall that the training error is an optimistic estimate of the generalization (or test) error. For a classification model with VC dimension d, the VC dimension can predict a probabilistic upper bound on the test error :

$$\mathbb{P}(\text{err}_{\text{test}} \leq \text{err}_{\text{train}} + \underbrace{\sqrt{\frac{1}{|\mathcal{D}_{\text{train}}|}\left[d\left(\log\frac{2|\mathcal{D}_{\text{train}}|}{d}+1\right)-\log\frac{\delta}{4}\right]}}_{\epsilon}) = 1-\delta,$$

for $\delta \in [0,1]$ if the training data set is large enough ($d < |\mathcal{D}_{\text{train}}|$ required).

- The probability of the true error, err_{test}, being greater than $\text{err}_{\text{train}} + \epsilon$ is bounded by some constant δ.
- This probability is over all possible datasets of size $|\mathcal{D}_{\text{train}}|$ drawn from $\mathbb{P}_{xy}$ (which can be arbitrary).
- If d is finite, both δ and ϵ can be made arbitrarily small by increasing the sample size.

THE COMPLEXITY OF HYPOTHESIS SPACES

Recall that the training error is an optimistic estimate of the generalization (or test) error. For a classification model with VC dimension d, the VC dimension can predict a probabilistic upper bound on the test error.

$$\mathbb{P}(\text{err}_{\text{test}} \leq \text{err}_{\text{train}} + \underbrace{\sqrt{\frac{1}{|\mathcal{D}_{\text{train}}|}\left[d\left(\log\frac{2|\mathcal{D}_{\text{train}}|}{d}+1\right)-\log\frac{\delta}{4}\right]}}_{\epsilon}) = 1-\delta,$$

for $\delta \in [0,1]$ if the training data set is large enough ($d < |\mathcal{D}_{\text{train}}|$ required).

- As a corollary, if the training error is low, the test error is also (probably) low.
- In other words, an algorithm which minimizes training error reliably picks a "good" hypothesis from the hypothesis set.
- Such an algorithm is known as a **Probably Approximately Correct** (or **PAC**) algorithm.

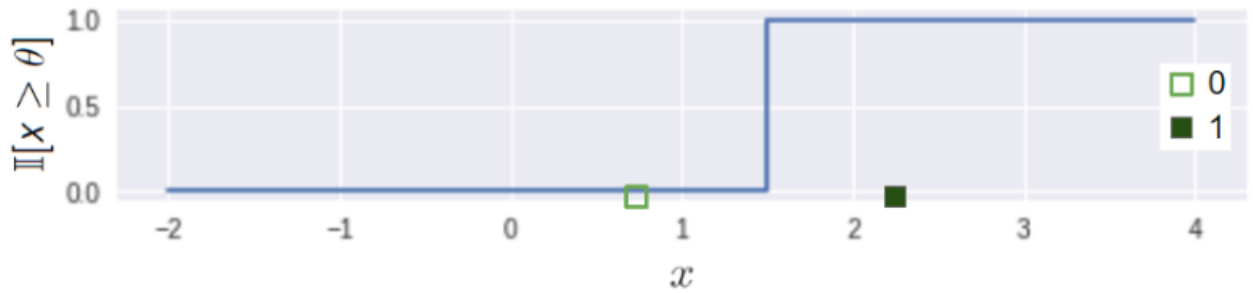
THE COMPLEXITY OF HYPOTHESIS SPACES

In general, the VC dimension of a hypothesis space increases as the number of learnable parameters increases.

However, it is not always possible to judge the capacity of a model based solely on the number of parameters.

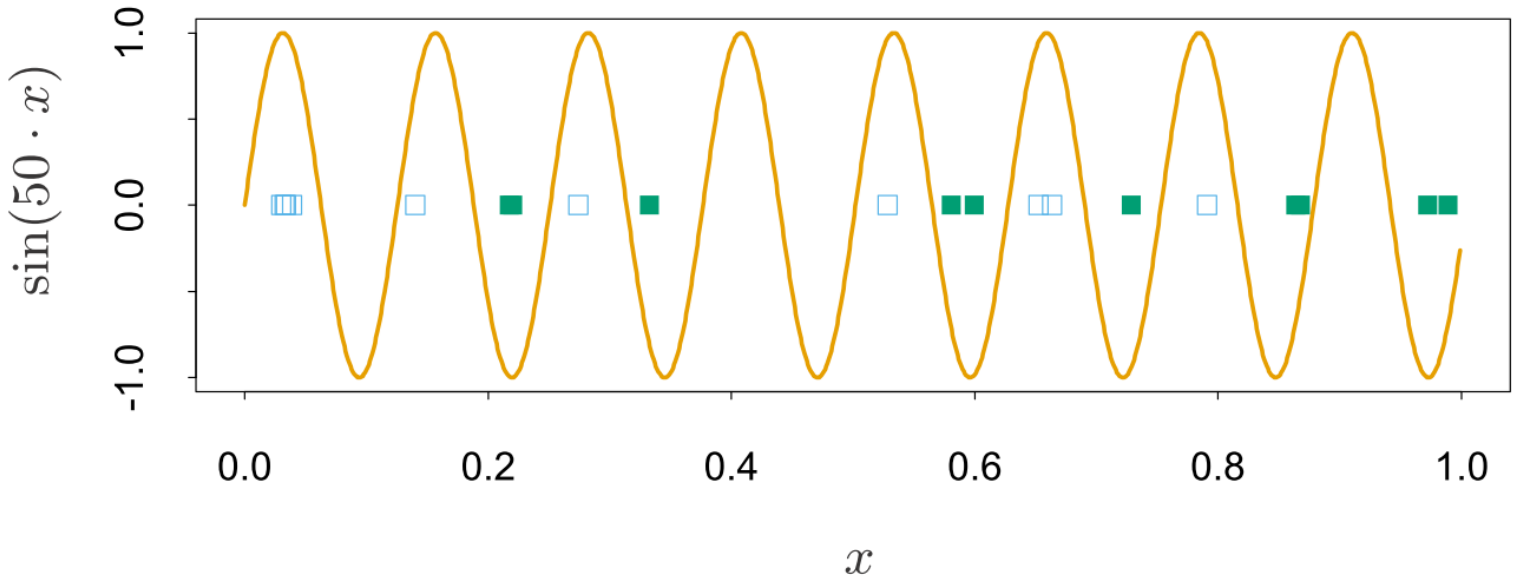
Example : A single-parametric threshold classifier ($f(x) = \mathbb{I}[x \geq \theta]$) has VC dimension 1:

- It can shatter a single point.
- It cannot shatter any set of 2 points (for every set of 2 numbers, if the smaller is labeled 1, the larger must also be labeled 1).



THE COMPLEXITY OF HYPOTHESIS SPACES

A single-parametric sine classifier, however, ($f(x) = \mathbb{I}[x \geq \sin(\theta x)]$, for $x \in \mathbb{R}$) has infinite VC dimension, since it can shatter any set of points if the frequency θ is chosen large enough.



Hastie, The Elements of Statistical Learning, 2009

THE COMPLEXITY OF HYPOTHESIS SPACES

- The bounds derived by the VC analysis are extremely loose and pessimistic.
- Because they have to hold for an arbitrary $\mathbb{P}_{xy}$, tightening the bounds to a desired level requires the training sets to be extremely large.
- In practice, complex models (such as neural networks) often perform much better than these bounds suggest.
- Other measures of model capacity such as **Rademacher complexity** may be easier to compute or provide tighter bounds.
- In addition to the hypothesis space, the effective capacity of a learning algorithm also depends in a complicated way on the optimizer.
- A better estimate of the generalization error can be obtained simply by evaluating the learned hypothesis on the test set.

BIAS-VARIANCE DECOMPOSITION

Let's take a closer look at the generalized prediction error

$$GE\left(\hat{f}_{\mathcal{D}}\right) = \mathbb{E}_{xy}\left(L\left(y, \hat{f}_{\mathcal{D}}(x)\right)|\mathcal{D}\right).$$

Assuming the normal regression model

$$y = f(x) + \epsilon$$

with $\epsilon \sim N(0, \sigma^2)$ and squared error loss,

$$GE\left(\hat{f}_{\mathcal{D}}\right) = \mathbb{E}_{xy}\left(L\left(y, \hat{f}_{\mathcal{D}}(x)\right)|\mathcal{D}\right) = \mathbb{E}_{xy}\left(\left(y - \hat{f}_{\mathcal{D}}(x)\right)^2\right)$$

Interestingly, this error can be broken down into three distinct components.

BIAS-VARIANCE DECOMPOSITION

Derivation :

$$
\begin{aligned}
GE\left(\hat{f}_{\mathcal{D}}\right) &= \mathbb{E}_{xy}\left(L\left(y, \hat{f}_{\mathcal{D}}(x)\right)|\mathcal{D}\right) = \mathbb{E}_{xy}\left(\left(y - \hat{f}_{\mathcal{D}}(x)\right)^2\right) \\
&= \mathbb{E}_y\left(y^2\right) + \mathbb{E}_x\left(\hat{f}_{\mathcal{D}}(x)^2\right) - \mathbb{E}_{xy}\left(2y\hat{f}_{\mathcal{D}}(x)\right) \\
&= \mathrm{Var}(y) + \mathbb{E}_{xy}\left(y\right)^2 + \mathrm{Var}\left(\hat{f}_{\mathcal{D}}(x)\right) + \mathbb{E}_x\left(\hat{f}_{\mathcal{D}}(x)\right)^2 - \\
&\qquad -2\mathbb{E}_x\left(f(x)\hat{f}_{\mathcal{D}}(x)\right) \\
&= \mathrm{Var}(y) + \mathrm{Var}\left(\hat{f}_{\mathcal{D}}(x)\right) + \mathbb{E}_x\left(f(x) - \hat{f}_{\mathcal{D}}(x)\right)^2 \\
&= \sigma^2 + \mathrm{Var}\left(\hat{f}_{\mathcal{D}}(x)\right) + \mathrm{Bias}\left(\hat{f}_{\mathcal{D}}(x)\right)^2.
\end{aligned}
$$

BIAS-VARIANCE DECOMPOSITION

For the k-NN regression fit

$$\hat{f}_{\mathcal{D}}(x) = \frac{1}{k} \sum_{i: x^{(i)} \in N_k(x)} y^{(i)},$$

the generalized prediction error (as a function of x) becomes

$$\begin{aligned} GE\left(\hat{f}_{\mathcal{D}}\right) &= \sigma^2 + \text{Var}\left(\hat{f}_{\mathcal{D}}(x)\right) + \text{Bias}\left(\hat{f}_{\mathcal{D}}(x)\right)^2 \\ &= \sigma^2 + \frac{\sigma^2}{k} + \mathbb{E}_x \left(f(x) - \frac{1}{k} \sum_{x^{(i)} \in N_k(x)} f\left(x^{(i)}\right) \right)^2 \end{aligned}$$

where we assumed for simplicity that training inputs $x^{(i)}$ are fixed and the randomness arises only from y.

BIAS-VARIANCE DECOMPOSITION

So for the squared error loss, the generalized prediction error can be decomposed into

- **Noise**: Intrinsic error, independent from the learner, cannot be avoided
- **Variance**: Learner's tendency to learn random things irrespective of the real signal (overfitting)
- **Bias**: Learner's tendency to *consistently* misclassify certain instances (underfitting)

BIAS-VARIANCE DECOMPOSITION

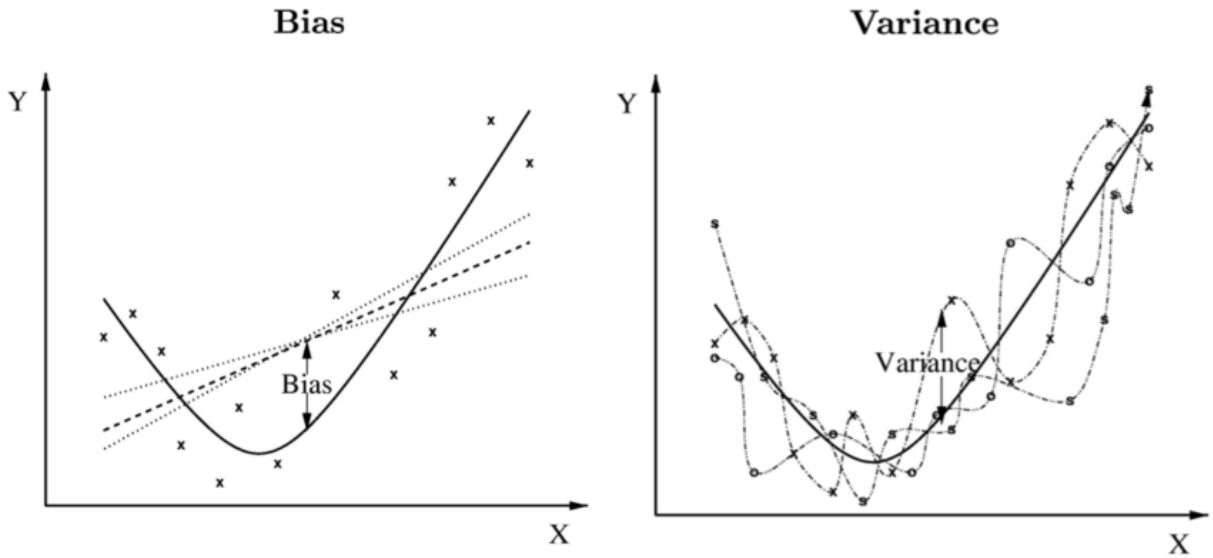
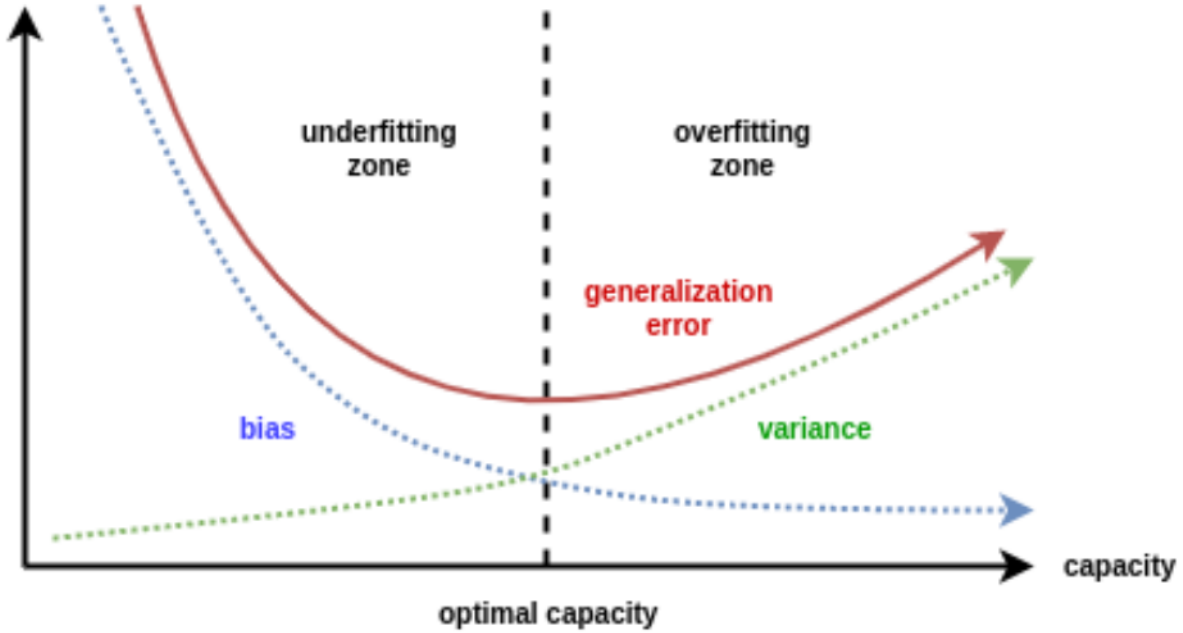


Figure: *Left* : A model with high bias is unable to fit the curved relationship present in the data. *Right* : A model with no bias and high variance can, in principle, learn the true pattern in the data. However, in practice, the learner outputs wildly different hypotheses for different training sets.

LEARNING CURVES

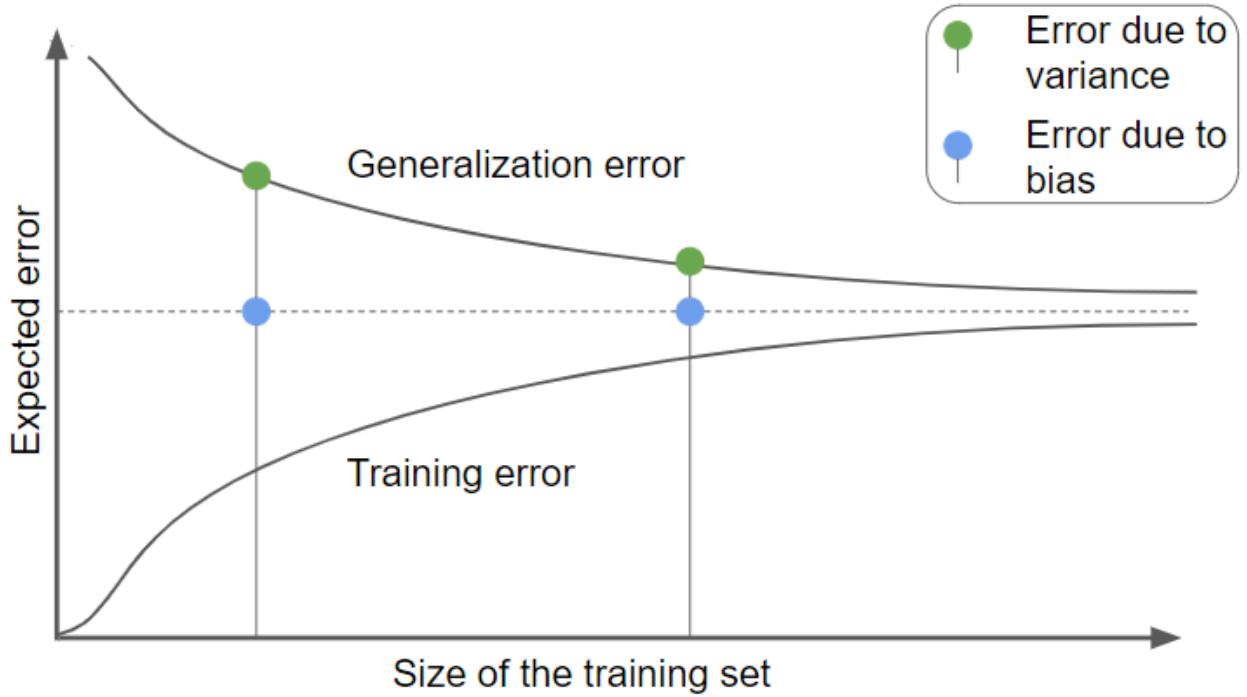


credit : digitalvidya

As the capacity of the model increases,

- The error due to variance increases.
- The error due to bias decreases.
- In the overfitting region, the generalization error increases because the increase in variance is much greater than the decrease in bias.

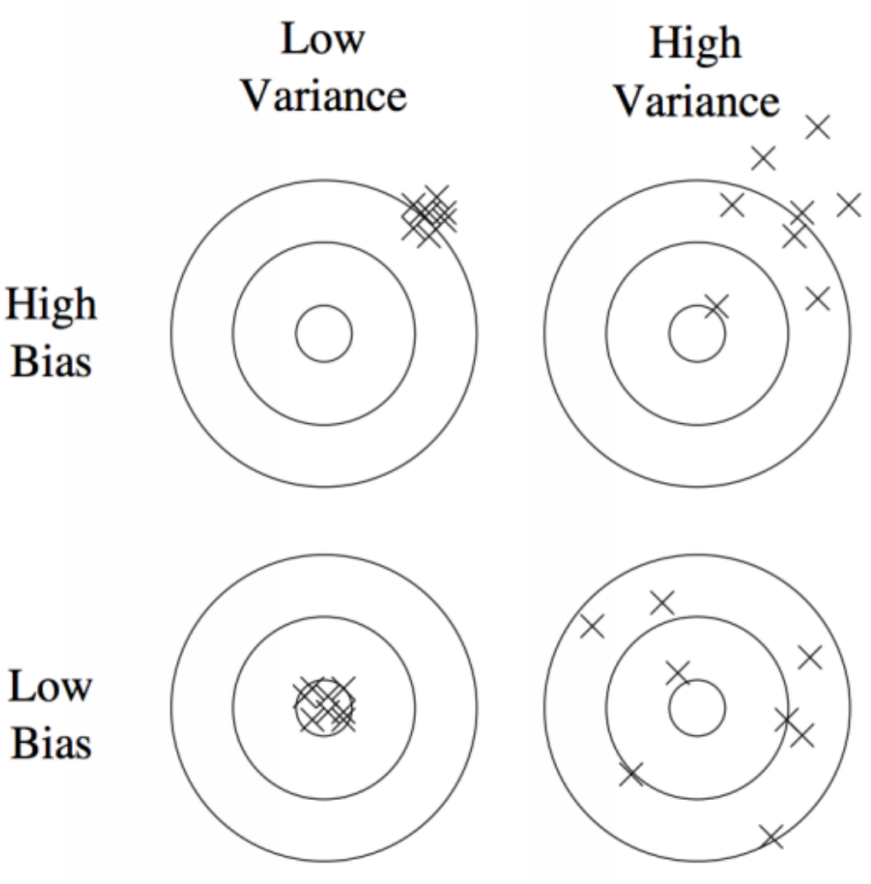
LEARNING CURVES



As the size of the training set grows,

- The error due to variance vanishes.
- Both the generalization error and training error converge to the bias of the algorithm (assuming noise is zero).
- As a result, the generalization gap also vanishes.

LEARNING CURVES



↓ Reduce underfitting by decreasing bias. Make model more flexible (or choose another).

← Reduce overfitting by decreasing variance. Make model less flexible (regularization), or add more data.

ML AS AN ILL-POSED PROBLEM

- Recall that a learning algorithm must perform well on previously unseen data.
- Let's assume we're trying to learn a boolean-valued function over 4 boolean features. There are $2^{16} = 65536$ possible functions.
- The training set (below) contains 7 examples.

Example	x_1	x_2	x_3	x_4	y
1	0	0	1	0	0
2	0	1	0	0	0
3	0	0	1	1	1
4	1	0	0	1	1
5	0	1	1	0	0
6	1	1	0	0	0
7	0	1	0	1	0

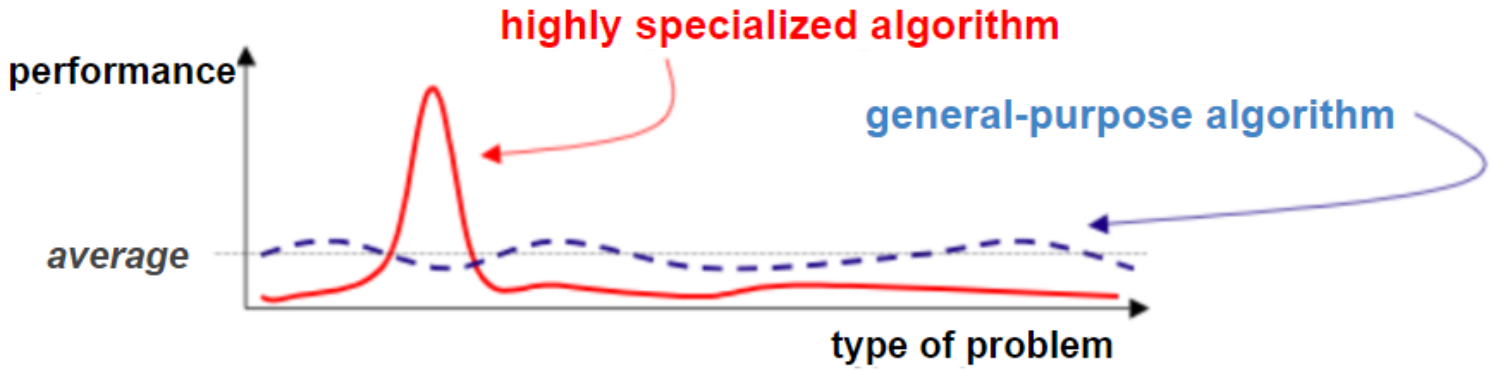
source : USC

ML AS AN ILL-POSED PROBLEM

- Even after observing 7 examples, there are still 2^9 possible functions left.
- Therefore, machine learning is an ill-posed problem.
- Given that the unseen datapoints can have any labels, can a machine learning algorithm really perform better than random guessing?

x_1	x_2	x_3	x_4	y
0	0	0	0	?
0	0	0	1	?
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	?
1	0	0	0	?
1	0	0	1	1
1	0	1	0	?
1	0	1	1	?
1	1	0	0	0
1	1	0	1	?
1	1	1	0	?
1	1	1	1	?

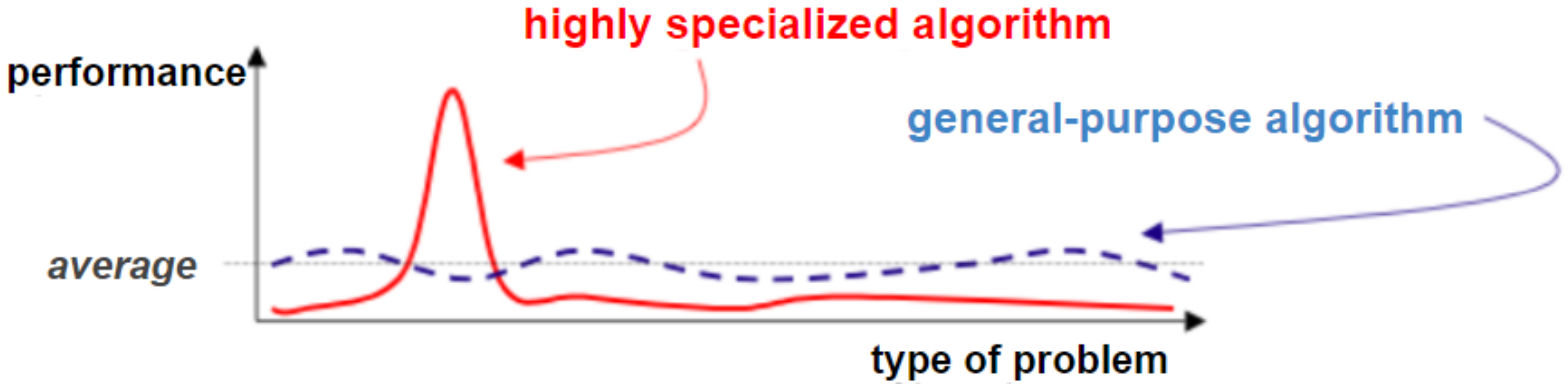
NO FREE LUNCH



credit : Leon Fedden

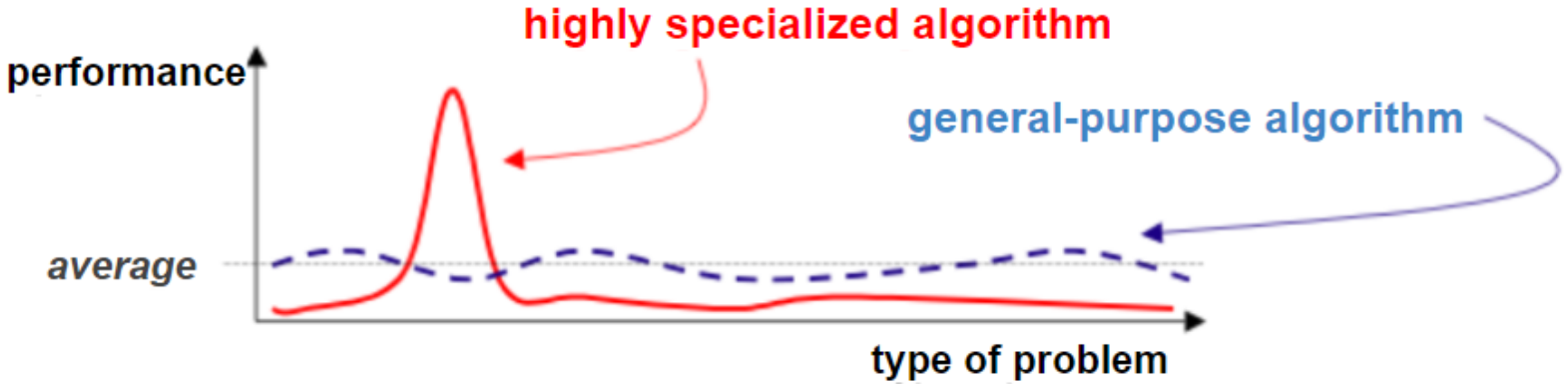
- For a specific problem, that is, for a specific distribution over target functions, it is possible for some algorithms to perform better than some others.
- When averaged over all possible problems, however, no algorithm is better than any other (including random guessing). This important result is known as the **No Free Lunch theorem**.
- The essence of the NFL theorem is that an algorithm that performs well on one set of problems has to necessarily perform badly on another set of problems. There is no "free lunch".

NO FREE LUNCH



- In other words, there is no algorithm that is universally better than all other algorithms.
- The main takeaway is that a learning algorithm must be tailored to a specific prior. Without making assumptions about the problem at hand, machine learning simply isn't possible!
- If these assumptions are very specific, it's possible for an algorithm to perform very well on a narrow set of problems. If these assumptions are very broad, we end up with an algorithm that is a "jack of all trades, master of none".

NO FREE LUNCH



- In the real world, however, we encounter only an infinitesimal subset of all possible problems.
- This is because data generating distributions are constrained by the laws of physics.
- Additionally, for a given task, one also has domain-specific knowledge about the nature of the target function.
- Therefore, by incorporating assumptions about the task (such as smoothness, linearity, etc), in practice, machine learning algorithms indeed perform quite well.