Advanced Topics in Statistics

~ Introduction to Machine Learning ~

1

Introduction.

Suppose we have some observations Y_i and $\boldsymbol{X}_i = (X_{i1}, \ldots, X_{ip})$ for $i = 1, \ldots, n$, and we believe that there is a relationship between Y and at least one of the X's.

We are looking to model this relationship as

$$Y_i = f(\boldsymbol{X}_i) + \varepsilon_i,$$

where f is an unknown function that represents the systematic information that $\boldsymbol{X}$ provides about Y, and ε is a random error with mean zero.

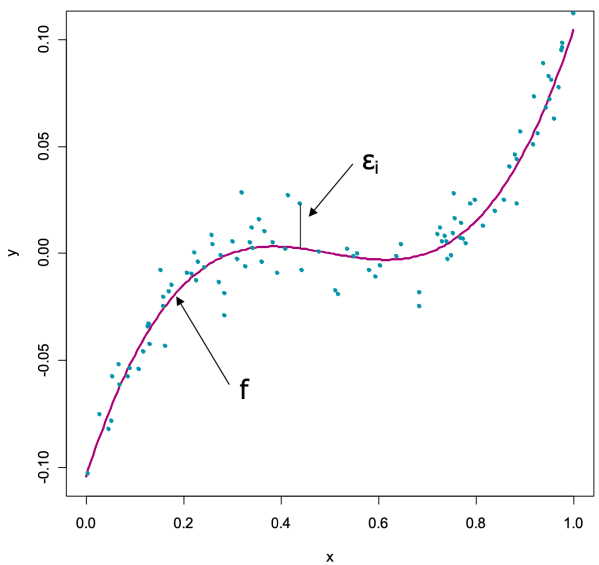
The set of observations

$$\{(\boldsymbol{X}_1, Y_1), (\boldsymbol{X}_2, Y_2), \ldots, (\boldsymbol{X}_n, Y_n)\}$$

are called training data.

We must then use the training data and a statistical/machine learning method to estimate f

There are two main reasons that we might want to estimate f; these are prediction and inference.



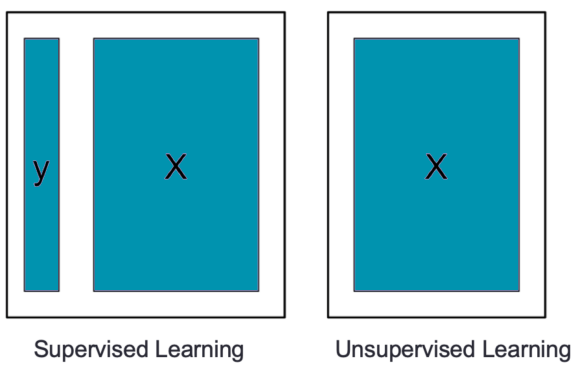
Supervised vs. Unsupervised Learning.

The majority of the most widely applied machine learning problems can be divided into two categories, supervised and unsupervised learning.

- Supervised Learning is where both the predictors, $\boldsymbol{X}_i$ and the response, Y_i, are observed.
- Unsupervised Learning is the situation where only the $\boldsymbol{X}_i$'s are observed. We need to use the $\boldsymbol{X}_i$'s to guess what Y would have been and build a model from there.

  An example is market segmentation where we try to divide potential customers into groups based on their characteristics. Customers in one group can then be treated similarly in marketing campaigns.

  A common approach to unsupervised learning is clustering.



Here we will focus on supervised learning algorithms.

Regression and Classification.

Supervised learning algorithms are trained using labelled examples that is, where both the input and the desired output are known.

It learns patterns, then uses these to predict labels for new, unlabelled data.

Supervised learning problems can be further divided into regression and classification problems.

Regression covers situations where Y is continuous/numerical. e.g.

- Predicting the value of the FTSE 100 in six months.
- Predicting the value of a given house based on various inputs.

The resulting mapping function maps the input values to continuous output. The predicted values are *ordered*.

Classification covers situations where Y is categorical/discrete e.g.

- Will the FTSE 100 be up (U) or down (D) in six months?
- Is this email a SPAM or not?

The resulting mapping function maps the input values to predefined classes (the output is discrete). The predicted values are *unordered*.

Measuring quality of fit - Regression.

Several different learning methods exist, and it depends on the particular dataset that which is the one that works best in that situation.

But how do we know which method produces the best results?

To evaluate performance we need to measure how well the method's predictions match the observed data.

For regression problems a common measure of accuracy is the mean squared error (MSE), given by

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where $\hat{y}_i$ is the prediction our method gives for the observation y_i in our training data.

Our methods have generally been designed to make MSE small on the training data we are looking at e.g. with linear regression we choose the line such that above MSE for the training data is minimised.

However what we really care about is how well the method works on new data. We call this new data 'Test Data'.

Training vs Test MSE.

Suppose we train a method that predicts the risk of diabetes based on some clinical measurements of the patient.

In the training data we have the clinical measurements as well as the information whether the given patient has diabetes or not.

When evaluating the method, we don't really care whether it accurately predicts diabetes for patients in the training data (we already know whether they have diabetes or not).

What we want the method to accurately predict is the risk of diabetes for future patients for whom we only have their clinical measurements.

The problem is that there is no guarantee that the method with the smallest training MSE will have the smallest MSE for the test (i.e. new) data.

In some cases we do have a test dataset. In this case we can train our method on the training data, evaluate the method on the test data, and select the method with the smallest test MSE.

But what can we do when there are no test observations available?

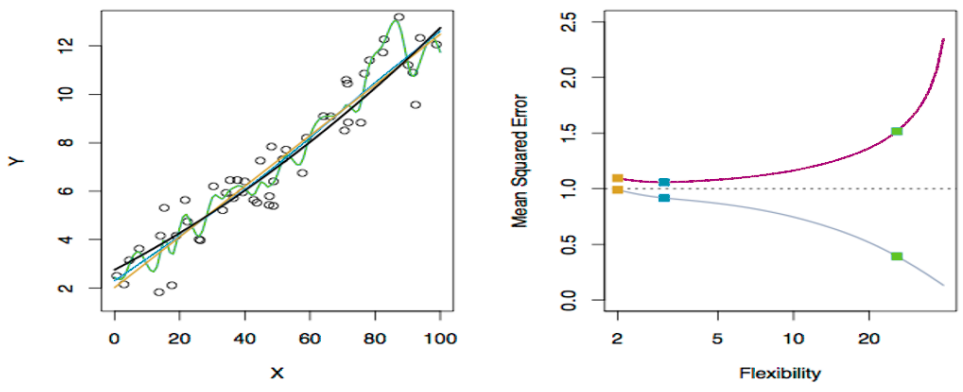
Model flexibility vs MSE.

In general the more flexible a method is the lower its training MSE will be i.e. it will 'fit' or explain the training data very well.

In the figure below we have fitted three different models with increasing level of flexibility (one linear regression, and two regression models using smoothing splines with different levels of smoothness).

More flexible methods (such as splines) can generate a wider range of possible shapes to estimate f as compared to less flexible and more restrictive methods (such as linear regression).

However, the test MSE may in fact be higher for a more flexible method than for a simple approach like linear regression, as seen on the right hand side of the figure.

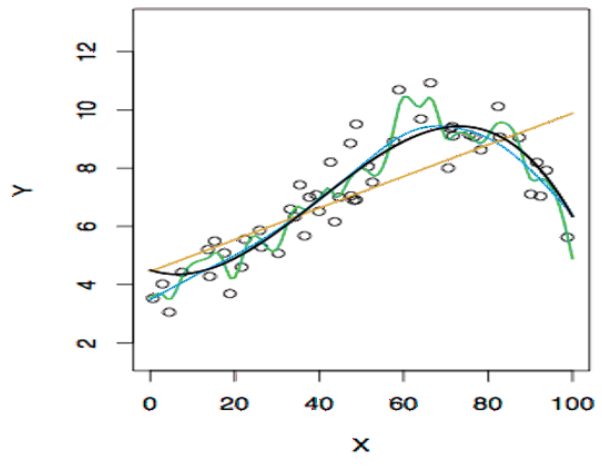


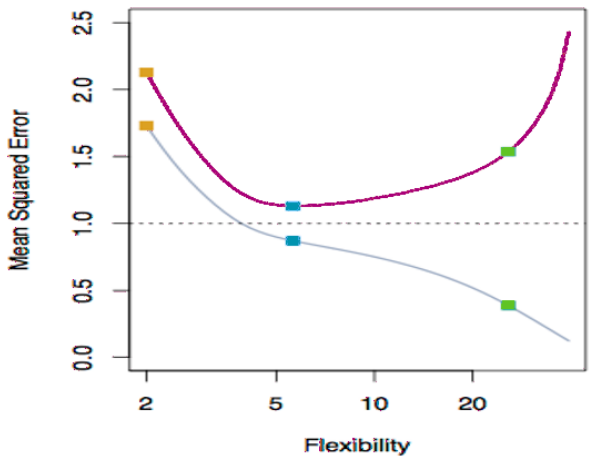
As the flexibility increases the training MSE (grey curve) decreases, however the test MSE (red curve), after an initial decrease, will start increasing.

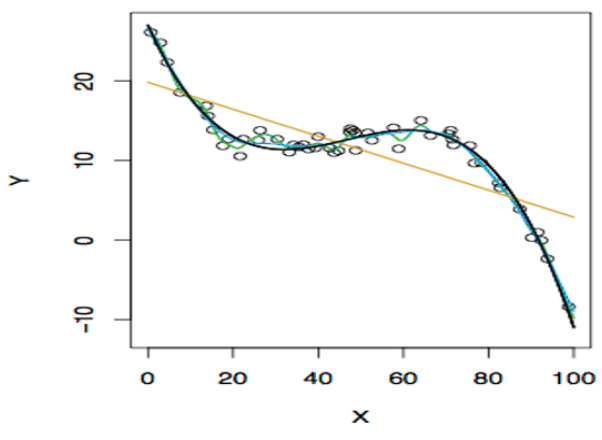
(The squares denote the three different methods we used to estimate f).

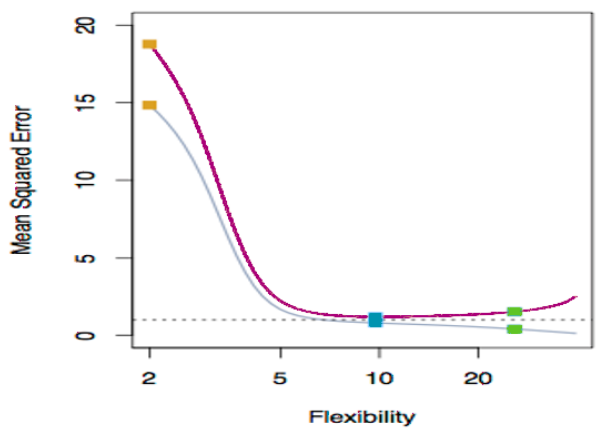
Training vs Test MSE Examples.

Here we can see two more examples of how the training and test MSE changes with the model flexibility. The decreasing training MSE is accompanied by a U-shaped test MSE.









Overfitting and Interpretability.

In all the previous examples the green curve is the most flexible, and it matches the observed data really well.

However it fits the true curve (shown in black) poorly, since it's too wiggly.

Too flexible methods tend to overfit the data. This means that the model is picking up 'patterns' caused by random chance in addition to the true patterns of the unknown underlying function f.

In summary, increasing the flexibility can help finding a better fitting model, up to a certain degree. Too flexible methods however will aim to model the noise as well, resulting in poor fit when it comes to a new dataset.

There are a number of approaches that can be used for finding the sweet-spot, e.g. cross-validation (see later).

Another trade-off we have to consider is flexibility vs interpretability. Usually the less flexible the method, the easier to interpret the model.

When the aim of the modelling is inference, interpretability becomes more important. We want to be able to understand how the individual predictors are associated with the response.

Bias vs Variance Trade-off.

The previous graphs of test versus training MSE's illustrates a very important trade-off that governs the choice of statistical/machine learning methods.

There are always two competing forces that govern the choice of learning method, bias and variance.

- Bias refers to the error that is introduced by representing a real-life problem (that is usually extremely complicated) by a model.

  For example, linear regression assumes that there is a linear relationship between Y and $\boldsymbol{X}$. It is unlikely that, in real life, the relationship is exactly linear so some bias will be present.

  The more flexible/complex a method is the less bias it will generally have.

- Variance refers to the amount the estimated $f, \hat{f}$ would change if we estimated it using a different data set.

  Different training sets will naturally result in different estimates for f, however we don't want these estimates to vary too much between training sets.

In general, more flexible statistical methods have lower bias, but higher variance.

Expected test MSE.

It can be shown that for any given, $\boldsymbol{X} = \boldsymbol{x}_0$, the expected test MSE can be decomposed into three quantities, the variance of $\hat{f}(\boldsymbol{x}_0)$, the squared bias of $\hat{f}(\boldsymbol{x}_0)$ and the variance of the error terms ε.

That is,

$$\text{Expected Test MSE} = E\left((Y - f(\boldsymbol{x}_0))^2\right) = \text{Bias}^2 + \text{Var} + \sigma^2,$$

where σ^2 is $Var(\varepsilon)$, which we call the irreducible error.

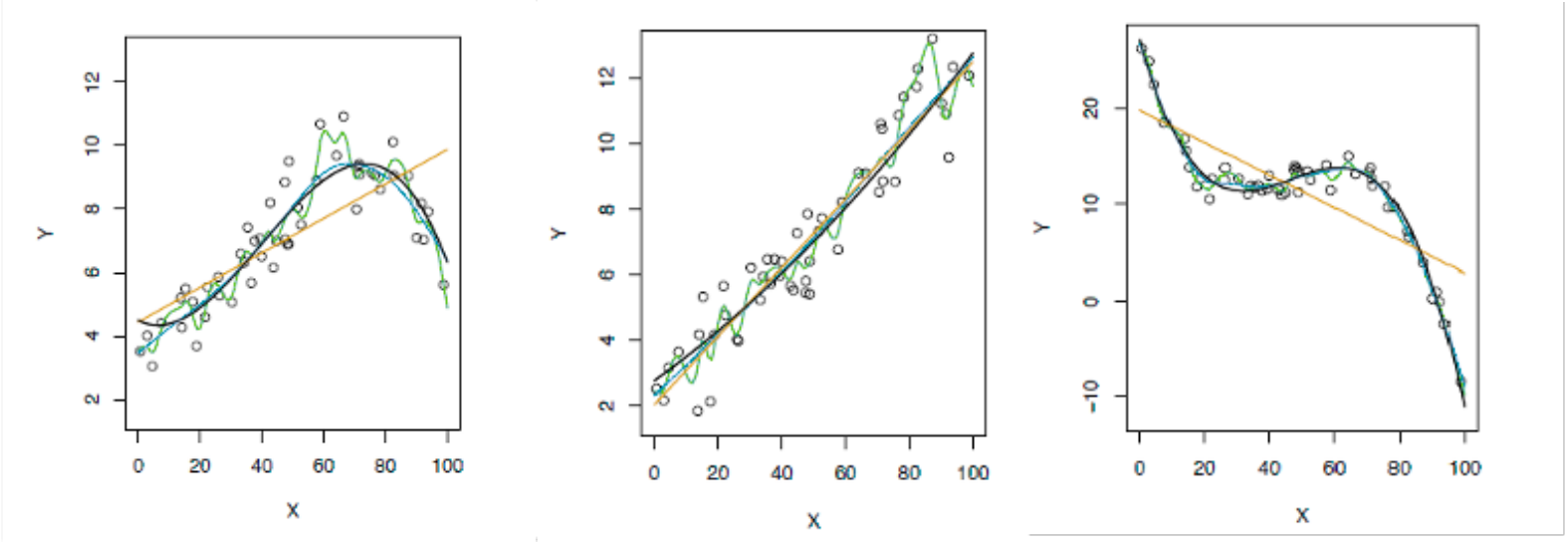
The expected test MSE is the average test MSE we would obtain if we repeatedly estimated f using a large number of training sets, and tested each at $\boldsymbol{x}_0$.

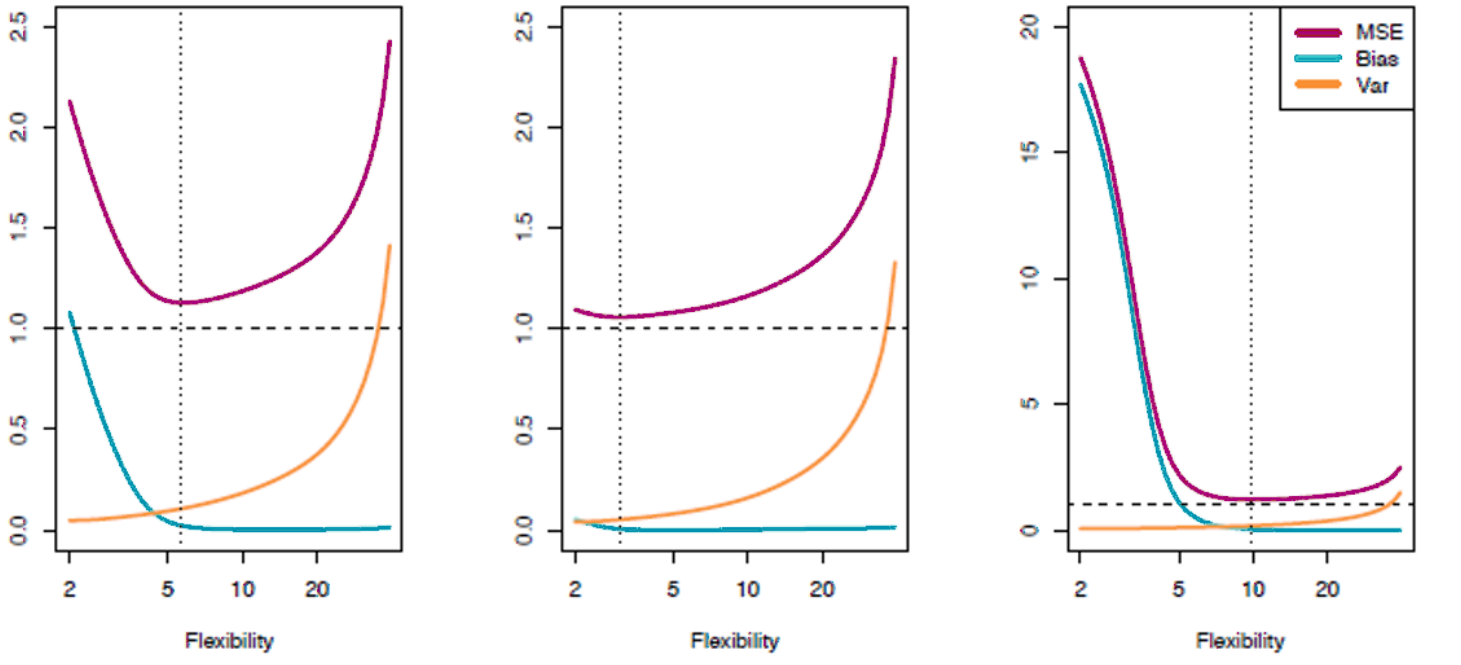
What we can conclude from the above formula is that as a method gets more complex the bias will decrease and the variance will increase but the expected test MSE may go up or down!

The challenge is finding a method that has both low variance and low bias.

But note, that even in an ideal world, we wouldn't be able to obtain an expected test MSE below $Var(\varepsilon) = \sigma^2$.

Test MSE, bias and variance - examples.





THE CLASSIFICATION SETTING.

For a regression problem, we used the MSE to assess the accuracy of the statistical learning method.

For a classification problem we can use the error rate i.e.

$$\text{Error rate} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{(y_i \neq \hat{y}_i)},$$

where $\hat{y}_i$ is the predicted class label for the ith observation, and $\mathbf{1}_{(y_i \neq \hat{y}_i)}$ is the indicator function that gives 1 if the condition $(y_i \neq \hat{y}_i)$ is correct, otherwise is gives 0.

Thus the error rate represents the fraction of incorrect classifications, or misclassifications.

The above error rate is referred to as the training error rate because it is computed based on the data we use to train our classifier.

But similarly to the regression setting, we are more interested in the so called test error rate, that gives the proportion of misclassifications when the classifier is applied to the test observations.

Ideally we would want to find the classifier that has the smallest test error.

BAYES CLASSIFIER.

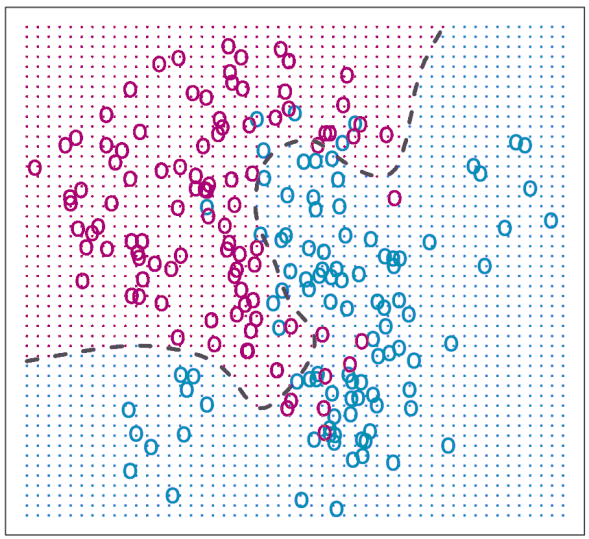
The test error rate is minimised by a very simple classifier, namely the one that assigns each observation to the most likely class given its predictor values.

That is, it assigns a test observation with predictor vector $\boldsymbol{x}_0$ to the class j for which

$$P(Y = j|\boldsymbol{X} = \boldsymbol{x}_0)$$

is the largest.

In a two-class problem, the Bayes classifier predicts class one if $P(Y = 1|\boldsymbol{X} = \boldsymbol{x}_0) > 0.5$, and class two otherwise.



The **pink** and **blue** observations correspond to training observations that belong to two different classes.

The pink shaded area gives the set of points for which $P(Y = \text{pink}|\boldsymbol{X})$ is greater than 0.5.

The grey dashed line represents the points for which this probability is exactly 0.5. This is called the Bayes decision boundary.

A new observation that falls on the pink side will be assigned the pink class (and blue o/w).

Bayes error rate

It can be shown that the Bayes classifier produces the lowest possible test error rate, called the Bayes error rate.

The Bayes error rate at $\boldsymbol{X} = \boldsymbol{x}_0$ is given by $1 - \max_j P(Y = j|\boldsymbol{X} = \boldsymbol{x}_0)$, while the overall Bayes error rate is

$$1 - E\left(\max_j P(Y = j|\boldsymbol{X})\right),$$

where the expectation averages the probability over all possible values of $\boldsymbol{X}$.

- The Bayes error rate is the error rate that could be achieved if somehow we could find exactly what the 'true' probability distribution of the data looked like.
- On test data, no classifier can get lower error rates than the Bayes error rate.
- Of course in real life problems the Bayes error rate can't be calculated exactly.

  The problem is that for real data we do not know the conditional distribution of Y given $\boldsymbol{X}$, and so computing the Bayes classifier is impossible. (The previous example showed simulated data, where we knew the 'truth').
- The Bayes error rate is analogous to the irreducible error of the regression setting.

K-NEAREST NEIGHBOUR.

The Bayes classifier serves as an unattainable gold standard against which we compare other methods.

Most classifiers attempt to estimate the Bayes classifier, by estimating the conditional distribution of Y given $\boldsymbol{X}$, then assign an observation to the class with the highest estimated probability.

One example is the K-nearest neighbour (KNN) classifier.

- KNN is a flexible approach to estimate the Bayes Classifier.
- For any given $\boldsymbol{X} = \boldsymbol{x}_0$ we find the K closest neighbours to $\boldsymbol{x}_0$ in the training data, and examine their corresponding Y.
- If $\mathcal{N}_0$ denotes the K nearest point to x_0, then the KNN classifier estimates the condititonal probability of class j as

$$P(Y = j|\boldsymbol{X} = \boldsymbol{x}_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} \mathbf{1}_{(y_i = j)},$$

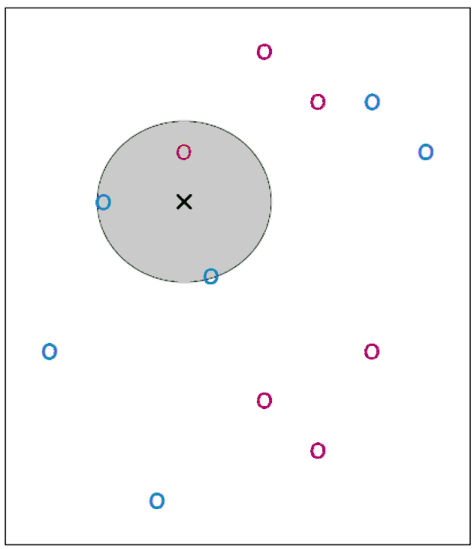
  and assigns the test observation $\boldsymbol{x}_0$ to the class with the largest probability.
- The smaller the K is the more flexible the method will be.

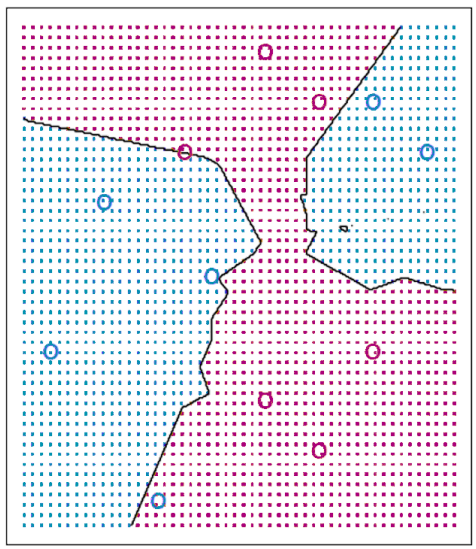
K-NEAREST NEIGHBOUR - EXAMPLE.

Essentially, the KNN classifier classifies a new object by the majority vote of its neighbours, with the object being assigned to the class most common among its k nearest neighbours.

If the two class example, if the majority of the Y's are pink, we predict **pink**, otherwise guess **blue**.

In the figure below, the cross denotes a new observation. We have identified its three nearest neighbours, the majority class among these objects is 'blue', therefore we predict that the object denoted by the cross belongs to class 'blue'.





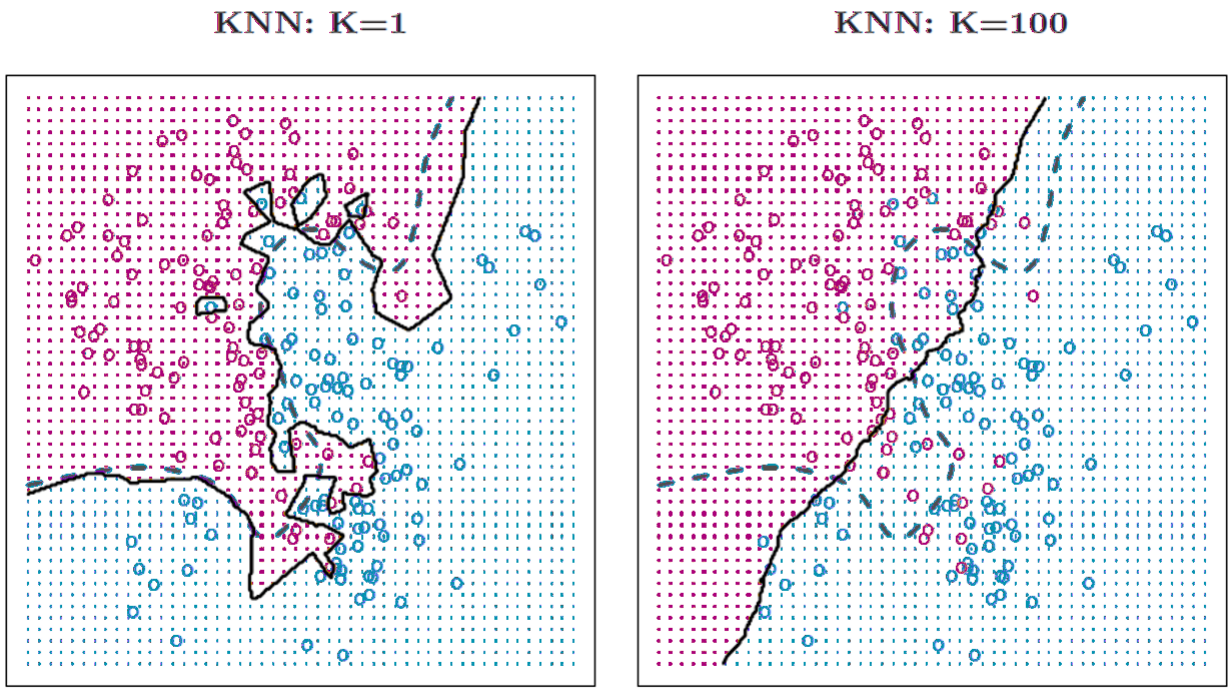
If we make predictions for all the points of the state space, we get the region where the method predicts class 'pink', and the region where the method predicts class 'blue'.

The solid black line is the KNN decision boundary.

Choosing K.

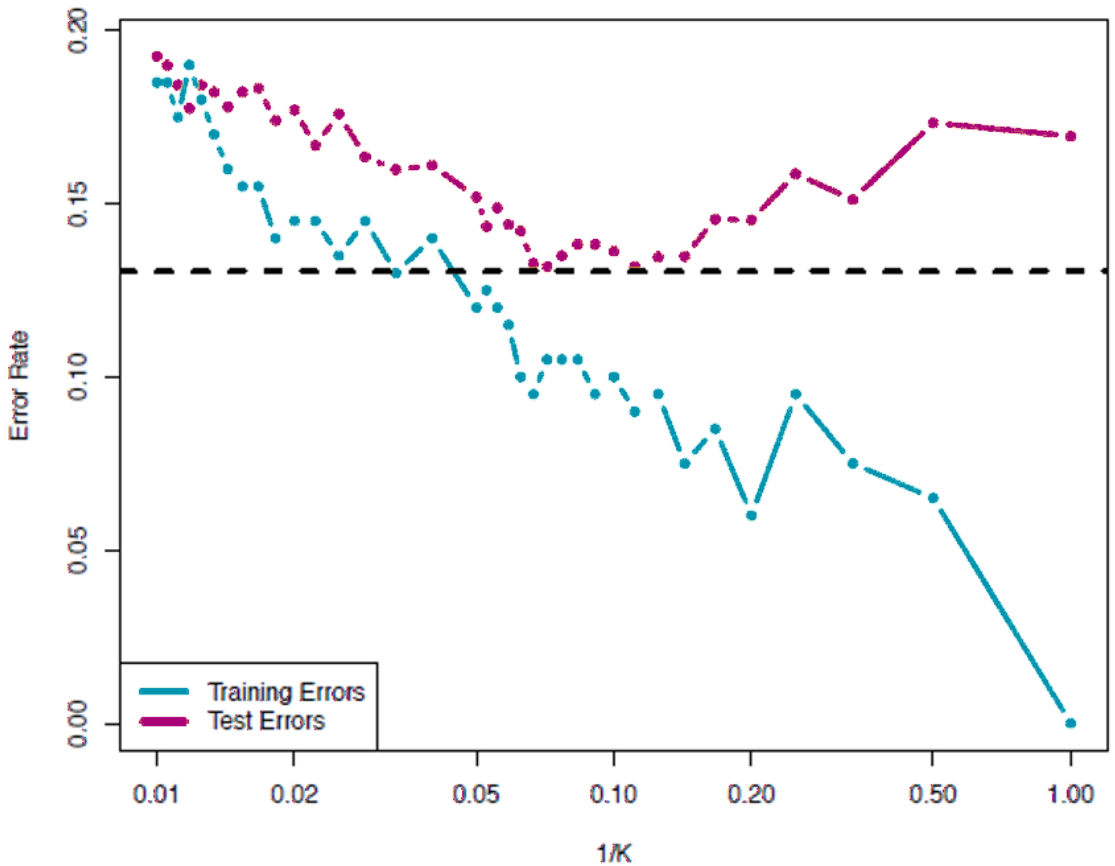
The choice of K has a drastic effect on the KNN classifier we obtain.

- In the example below, when $K = 1$, the resulting decision boundary is overly flexible. This corresponds to a classifier that has low bias but very high variance.
- When $K = 100$, the decision boundary is close to linear. This corresponds to a classifier that has low variance but high bias. *(The sweet-spot here is around $K = 10$).*



TRAINING VS. TEST ERROR RATES ON THE SIMULATED DATA

The figure shows the KNN test and training error rates as a function of $1/K$.



Notice that the training error rate keep going down as $1/K$ increases (thus K decreases) or equivalently, as the flexibility increases.

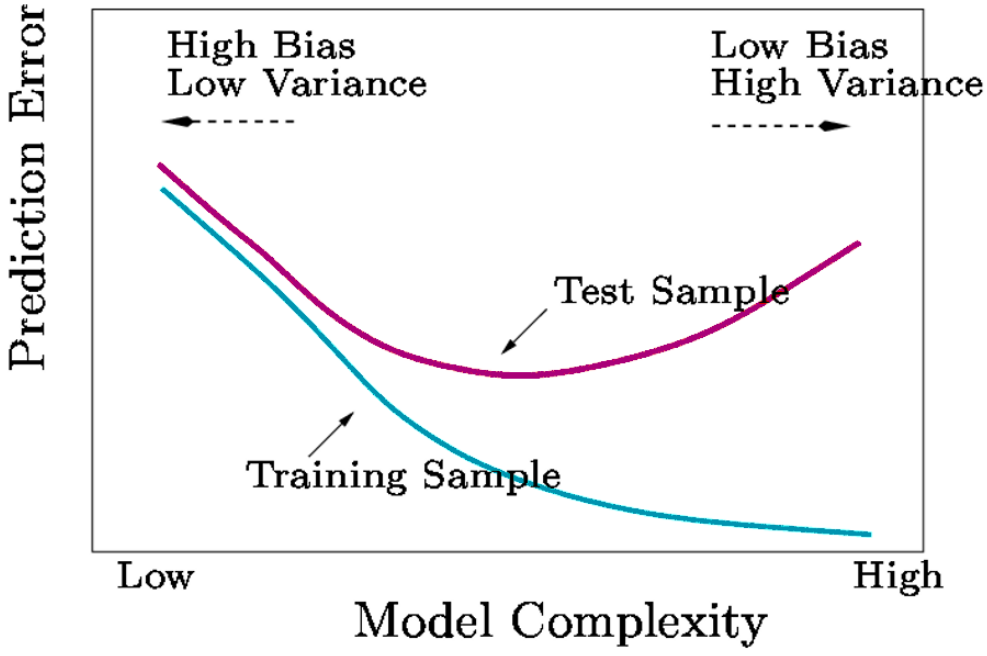
However, the test error rate at first decreases but then starts to increase again, displaying the U-shape characteristic we have seen in the regression setting.

The dashed black line indicates the Bayes error rate.

SUMMARY.

In general training errors will always decline.

However, test errors will decline at first (as reductions in bias dominate) but will then start to increase again (as increases in variance dominate).



In both regression and classification settings, choosing the correct level of flexibility is critical to the success of the statistical learning method.

The bias-variance trade-off, and the resulting U-shape in the test error, can make this difficult.