Machine Learning - Michaelmas Term 2017
Lecture 5: Basis Expansion, Learning Curves, Overfitting

Lecturers: Christoph Haase & Varun Kanade

1 Basis Function Expansion

So far we have studied the linear model that can only capture a linear relationship between the output y and the inputs $\mathbf{x}$. Basis function expansion is an easy way to capture non-linear relationships.

1.1 Polynomial Basis Expansion

Let us first look at a simple example when the input is one dimensional. Suppose, we want the output to be a quadratic function of the input, then we can use the features 1, x and x^2, as inputs and then simply fit a linear model. Essentially, we have mapped the input from a 1-d space to a 3-d space, but the model in the 3-d space is just a linear model!

We will denote such a feature expansion of input x by $\phi(x)$. In this case, $\phi(x) \in \mathbb{R}^3$, $\phi(x) = [1, x, x^2]^\mathsf{T}$. For $\mathbf{w} \in \mathbb{R}^3$ a linear model of the form $\mathbf{w} \cdot \phi(x)$ captures a quadratic relationship between the output y and the input x. Figure 1(a) and 1(b) show a linear model and a quadratic model fit to the same dataset; it can be seen that the quadratic model is a better fit. In general, it is possible to fit more complex models using a feature expansion that includes higher degree terms. The degree d feature expansion is given by $\phi(x) = [1, x, \ldots, x^d]^\mathsf{T}$. When using higher degree polynomials there is a danger that we may overfit the data, *i.e.,* use a model that works very well for the training data but will most likely perform badly on unseen data. We will discuss overfitting in much greater detail below. Figure 1(c) shows degree 7 and 14 polynomials fit to the same data, both of which are unnaturally "wiggly" and are probably overfitting. In fact, if we have N data points and use degree $d = N - 1$, then there exists a polynomial of degree d that fits all the data perfectly! Having large quantities of data can alleviate the problem of overfitting significantly as shown in Figure 1(d). In this case, all of the degree 2, 7 and 14 polynomials fit the data well; however, the linear model still performs poorly because it is incapable of capturing the true relationship between input and output. This last problem is referred to as *underfitting.*

Polynomial basis expansion can be carried out in higher dimensions. For example, in order to fit a quadratic function in 2 dimensions, for vector $\mathbf{x} = [x_1, x_2]^\mathsf{T} \in \mathbb{R}^2$, we use the feature expansion $\phi(\mathbf{x}) = [1, x_1, x_2, x_1^2, x_2^2, x_1 x_2]^\mathsf{T}$. Now if we fit a linear model using $\phi(\mathbf{x})$ as the input, the output y can capture an arbitrary quadratic function of the input. Figure 2 shows a linear model and a quadratic model (linear model with degree 2 basis expansion) in two dimensions. We can calculate the dimension in which the feature vector $\phi(\mathbf{x})$ lies if we want to fit degree d polynomials and the original input $\mathbf{x}$ is in D dimensions; this number is very large, roughly D^d. The large number of parameters required to fit such models will present both computational (requiring more computational resources) and statistical (requiring more data) challenges. We will discuss to what extent these challenges may be addressed.

1.2 Basis Expansion Using Kernels

Let us now consider another way to capture non-linear relationships. We will perform basis expansion using kernels. For the time being, let us think of a kernel as a function that takes as input two points $\mathbf{x}$ and $\mathbf{x}'$ in the input space (*i.e.,* $\mathbb{R}^D$) and outputs a non-negative real

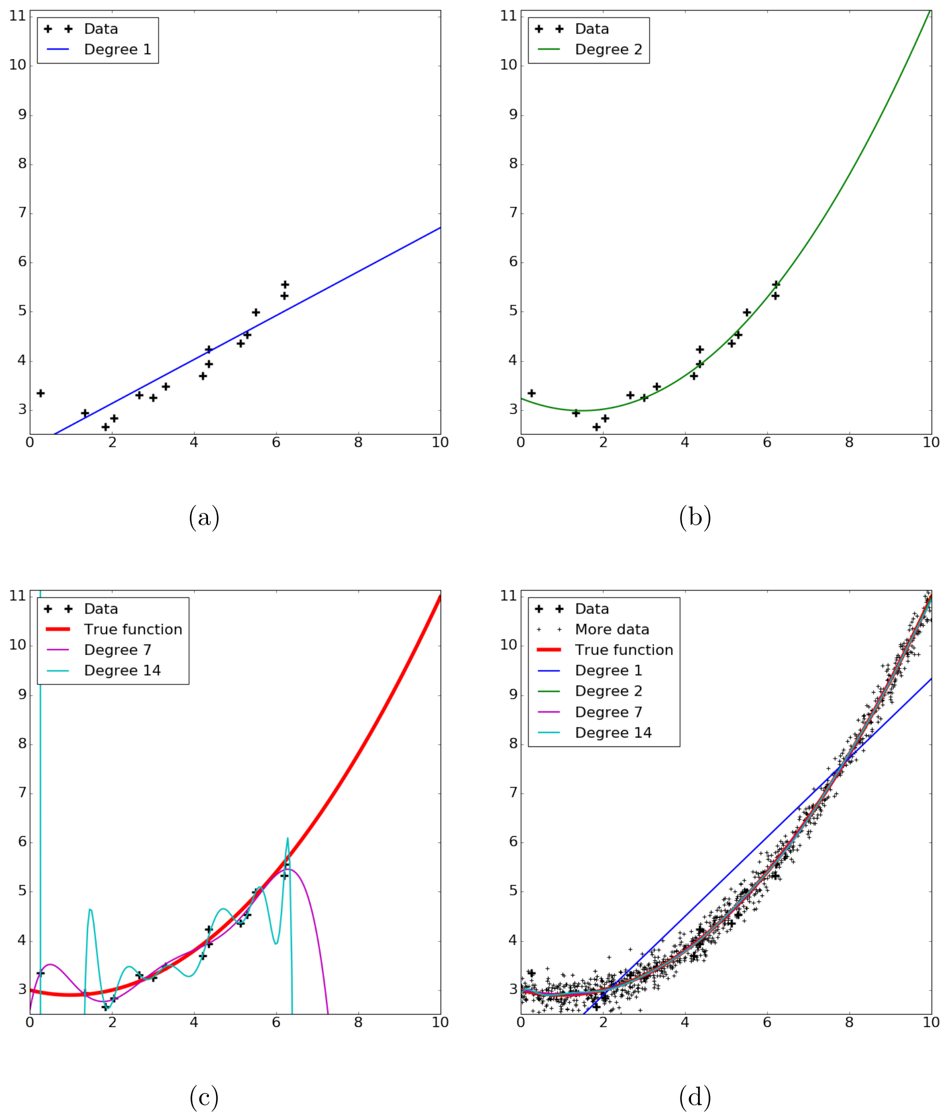


Figure 1: (a) Linear model fit to data. (b) Quadratic model fit to data. (c) The true model used to generate the data was quadratic. Higher degree models can be fit which may overfit the data. (d) When large quantities of data is available, overfitting can be avoided.

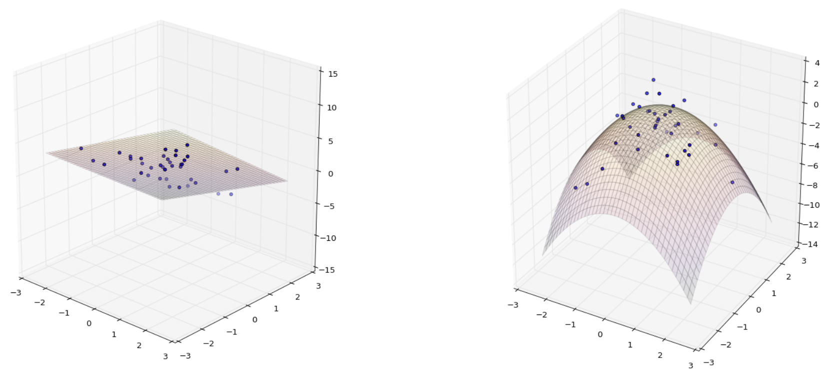
number.[1] In principle, any kernel may be used, however, let's focus on a particularly common one called the radial basis function (RBF) kernel.

Definition 1 (RBF Kernel)**.** *A radial basis function (RBF) kernel with width parameter* γ *is defined as* $\kappa(\mathbf{x}', \mathbf{x}) = \exp\left(-\gamma \left\|\mathbf{x} - \mathbf{x}'\right\|^2\right)$.

The name radial basis function reflects the fact that the value of $\kappa(\mathbf{x}', \mathbf{x})$ depends only on the distance between the two points $\mathbf{x}'$ and $\mathbf{x}$; for now, let us suppose that we are using Euclidean distances. We pick some centres, $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_M$ in the input space; we will address the issue of how to pick centres shortly. Then, for every point $\mathbf{x}$ we use the following kernel basis expansion:

$$\phi(\mathbf{x}) = [1, \kappa(\boldsymbol{\mu}_1, \mathbf{x}), \kappa(\boldsymbol{\mu}_2, \mathbf{x}), \ldots, \kappa(\boldsymbol{\mu}_M, \mathbf{x})]^{\mathsf{T}}$$

[1]For a more formal definition refer to Murphy (2012, Chap 14), or for some more informal discussion to Goodfellow et al. (2016, Chap 5.7.2).



(a) (b)

Figure 2: (a) Linear model in 2 dimensions. (b) Quadratic model in 2 dimensions obtained by first performing degree 2 polynomial basis expansion and then fitting a linear model.

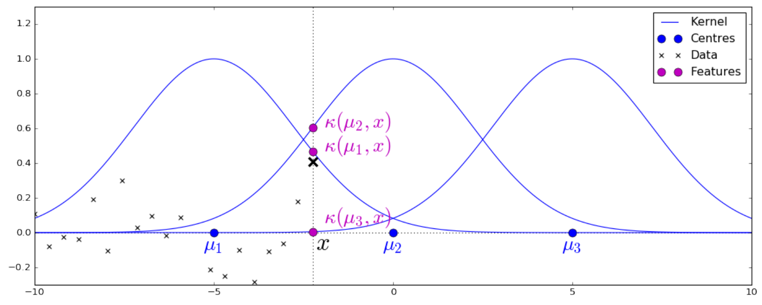


Figure 3: Basis expansion using kernels.

If we fit a linear model using the inputs $\phi(\mathbf{x})$, we get an output of the form:

$$\widehat{y}(\mathbf{x}) = w_0 + \sum_{i=1}^{M} w_i \kappa(\boldsymbol{\mu}_i, \mathbf{x})$$

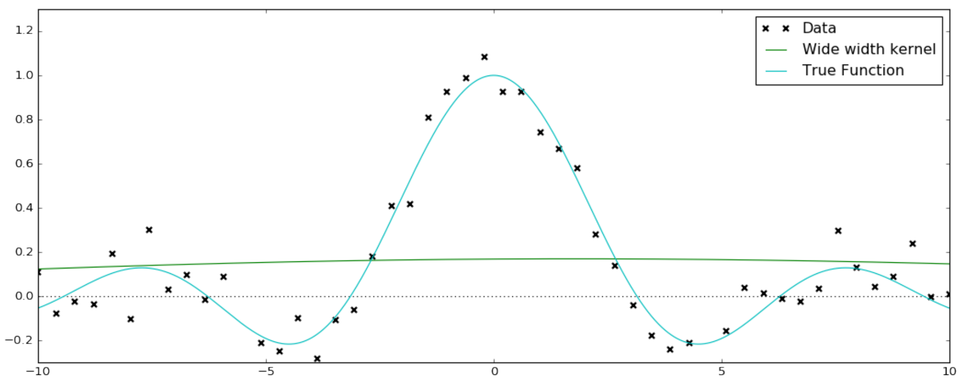
Thus, the model output is a linear combination of the kernel functions with M different centres.

Choice of Centres

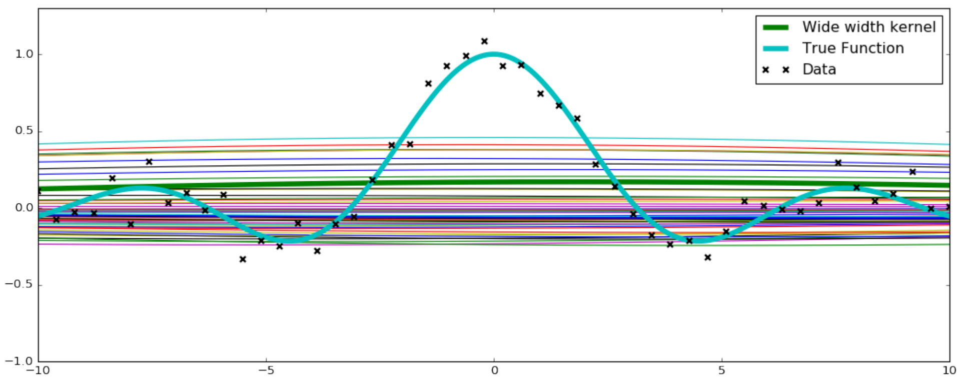
Figure 3 shows basis expansion using RBF kernels in one dimension using 3 centres. At the marked point x, the features will be given by $[1, \kappa(\mu_1, x), \kappa(\mu_2, x), \kappa(\mu_3, x)]^\mathsf{T}$. We see in the picture that $\kappa(\mu_3, x)$ is very close to 0 as the centre μ_3 is far from the point x. In fact in this picture most datapoints are very far from μ_3. The data mainly lies between $[-10, 2]$, but $\mu_3 = 5$. Thus, $\kappa(\mu_3, x) \approx 0$ for almost all the points in the dataset, making this a relatively redundant feature. One way to avoid choosing such centres is to choose the datapoints themselves to be centres. This is the most common approach employed in practice. There is also good theoretical justification for this choice, which is unfortunately beyond the scope of this course. The interested student is referred to books on kernel methods, *e.g.,* (Schölkopf and Smola, 2002; Shawe-Taylor and Cristianini, 2004).

Choice of Width Parameter γ

As with the choice of degree in polynomial basis expansion, the width parameter γ plays an important role in kernel basis expansion. When γ is very small, the resulting kernel is "wide";



(a)



(b)

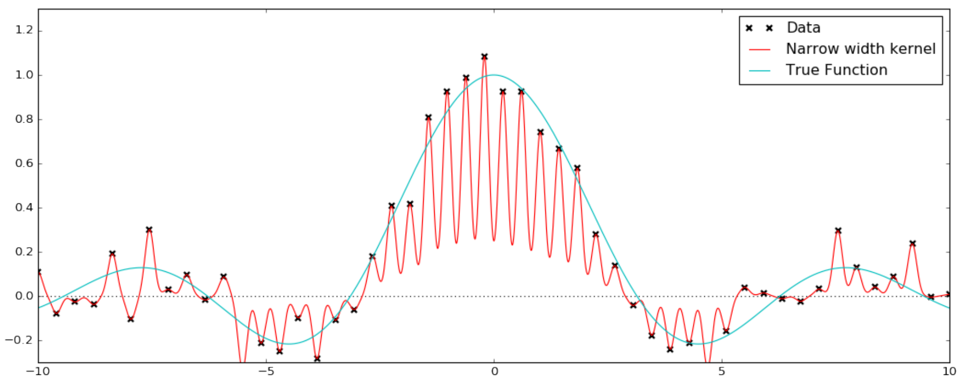
Figure 4: (a) True data generating function and a linear model with kernel basis expansion for a wide kernel. (b) Composition of the fit model as a linear combination of kernel functions, with datapoints as centres

suppose $\boldsymbol{\mu}$ is the centre, then even points $\mathbf{x}$ that are far from $\boldsymbol{\mu}$ will have $\kappa(\boldsymbol{\mu}, \mathbf{x}) \approx 1$. This often results in underfitting, as most points in the dataset will have the same feature value (see Fig. 4). On the other hand, if γ is very large, the kernel is “narrow”; except for points very close to the centre $\boldsymbol{\mu}$, the value $\kappa(\boldsymbol{\mu}, \mathbf{x}) \approx 0$. The resulting function fits the datapoints very well, but is unlikely to generalise well to unseen data (see Fig. 5). A suitable choice of the width parameter will avoid both overfitting and underfitting (Fig. 6).

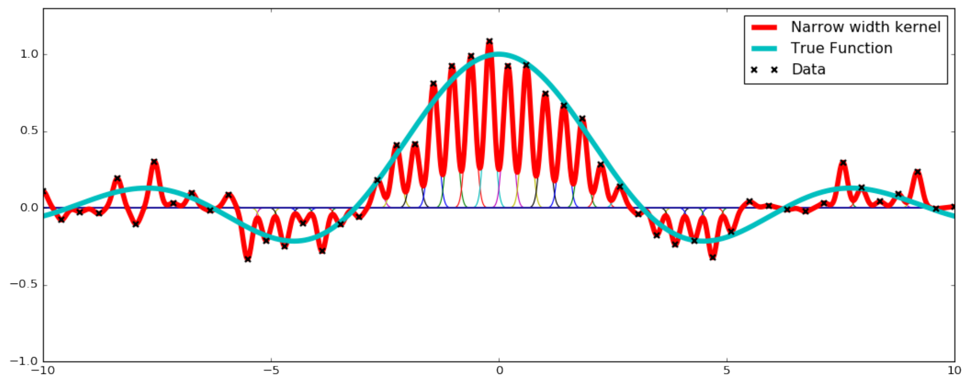
In high dimensions, we might suffer from the curse of dimensionality (see Problem 1 on Sheet 1). If the width of the kernel is chosen to be too large (very small γ), then it may result in underfitting. However, in high dimensions it is often not easy to find the “right” middle between a kernel that is ‘too wide’ and one that is ‘too narrow’

Remark 2. *High-dimensional geometry can sometimes appear counter-intuitive and takes some time and effort to get used to. For example, think about this simple question. What is the ratio of the area of a circle with diameter* 1 *to that of a unit square? In two dimensions the answer is simple, it's* $\pi/4$. *However, in higher dimensions the volume of the ball with unit diameter is exponentially (in the dimension) smaller than the volume of the unit box.*

Imagine the implications of this: if we sample points uniformly from the unit box in say 100 *dimensions, almost none of them lie in the sphere with unit diameter with centre at the centre of the box!*

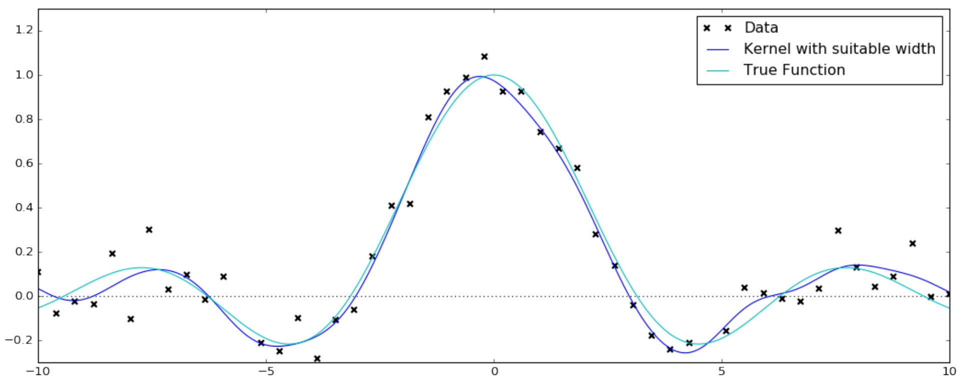


(a)

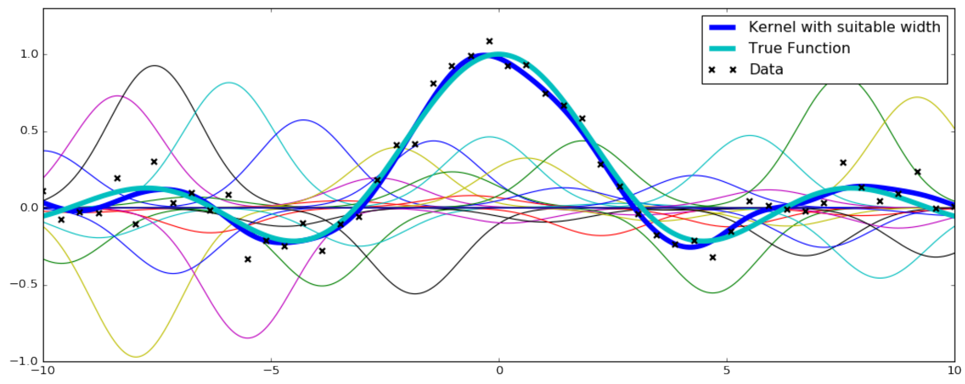


(b)

Figure 5: (a) True data generating function and a linear model with kernel basis expansion for a narrow kernel. (b) Composition of the fit model as a linear combination of kernel functions, with datapoints as centres

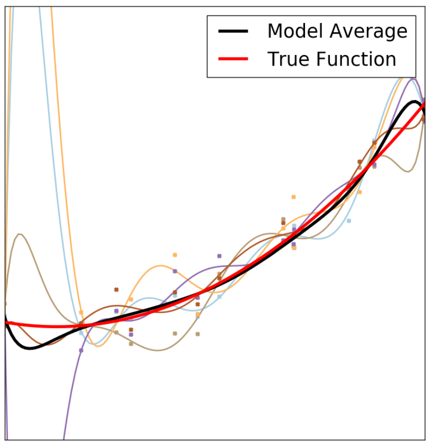


(a)

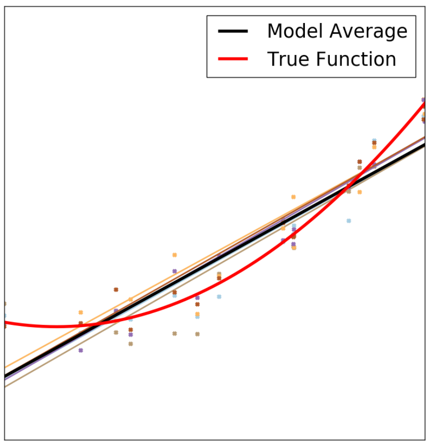


(b)

Figure 6: (a) True data generating function and a linear model with kernel basis expansion for an intermediate width kernel. (b) Composition of the fit model as a linear combination of kernel functions, with datapoints as centres



(a)



(b)

Figure 7: (a) Different high-degree polynomials fit to data generated from the same true quadratic function with (different) Gaussian noise. There is high-variance in the models, but the average (thick black line) is close to the true function (thick red line). (b) The same setting as Fig. (a) but this time a linear model is fit. There is very little variance, but high bias.

Discussion on Basis Expansion

Both polynomials and radial basis functions are universal; what this means is, as we increase the degree in the case of polynomials or the number of centres in the case of RBF kernels, the resulting model can approximate any *continuous* function. While this means that these models are very powerful, it also suggests that they may be too powerful! Ultimately, we want to fit models that can be trained using reasonable quantities of data and computational resources.

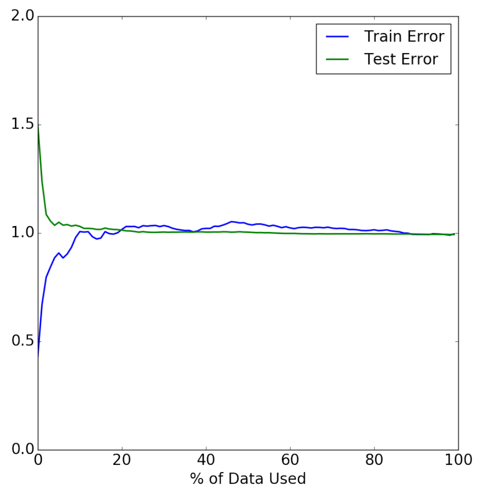
2 Overfitting and the Bias Variance Tradeoff

We have already alluded to the possibility of overfitting the data when we increase the complexity of the model by increasing the number of parameters as a result of basis expansion. On the other hand, when the model we attempt to fit is inadequate to explain the data, it results in *underfitting*. In statistical learning theory, these terms are referred to as having "high variance" and "high bias" respectively. Choosing the correct model complexity to avoid both overfitting and underfitting is referred to as the *bias-variance* tradeoff.

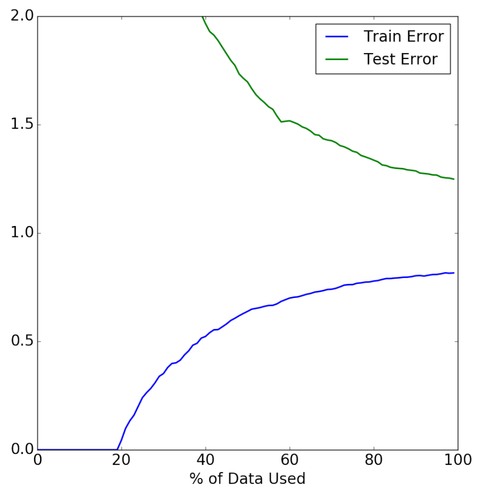
2.1 Bias-Variance Tradeoff

We will not go into the precise definitions of bias and variance in the context of statistical learning here. (On Problem Sheet 2 you are asked to work out the bias and variance of the least squares estimator.) High variance refers to the fact that when we fit the model to different realisations of the dataset we get very different models. As an example, if we fit a model using 5 different sets of 10 patients each, we might get completely different models (in terms of predictions on unseen patients) in each case. High bias refers to the fact that the model makes predictions that will inherently be different (either higher or lower) from the observations on unseen points.

Figure 7(a) shows five different high-degree polynomials fit to points generated from the same quadratic function with different noise added in each case. The model has high variance,



(a)



(b)

Figure 8: (a) Plot of training error and test error as a fraction of training data used. This shows that the model is either underfitting or fitting correctly. (b) Same as Fig. (a). In this case, the model is clearly overfitting.

but no bias. The average of the 5 models, shown by the thick black line, is actually pretty close to the true model generating the data (thick red line). Figure 7(b) shows linear models fit to the same five datasets. In this case, there is not much variance, but there is a high bias. In most of the input space the models will make predictions that are either higher or lower than the true function.

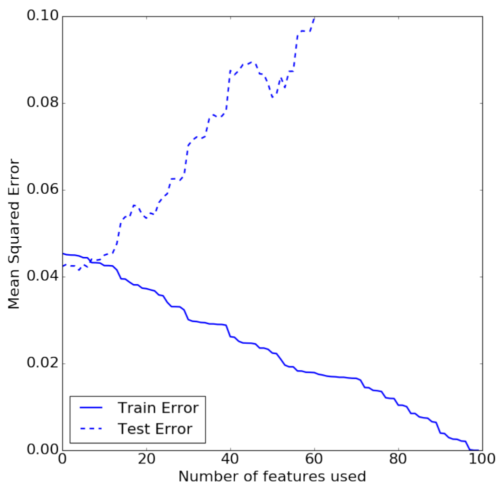
Learning Curves

An important task in practice is understanding whether the trained model has high bias (underfitting), high variance (overfitting) or neither. When the data is not one-dimensional and synthetically generated as we have seen in the pictures, how can we tell whether we are underfitting or overfitting?

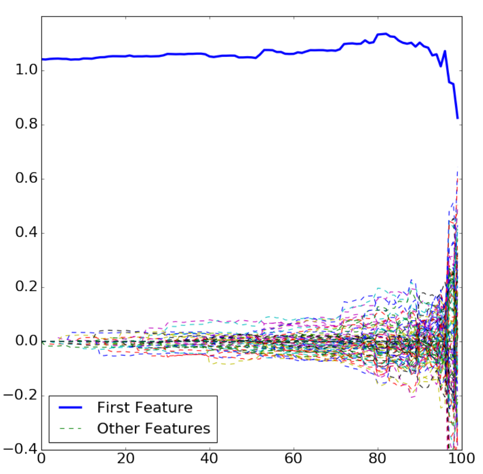
One way to do this is to plot learning curves (see Fig. 8). If sufficient data is available, we first keep aside part of the dataset called the "test set" which we will not use to fit the model at all. The remaining data is called the "training set". We can train the model on increasing fractions of the training data. For each model we have fit we can compute the training error (error on the data used for training) as well as the test error (error on the test set).

If the model has high bias (or suitable bias), the training and test error curves approach each other and then stay level. On the other hand, if the model has high-variance, the training error will start increasing as we increase the amount of training data, and the test error will start decreasing. Eventually, in the limit of infinite data, we do expect them to approach each other. Looking at the learning curves is a good way to understand whether our model is overfitting or underfitting, and to develop possible solutions if necessary.

If the model is underfitting, then, of course, we can try to design more complex models, *e.g.,* by basis function expansion. If the model is overfitting, then we can either simplify the model by directly reducing the number of parameters, or apply methods that control overfitting that we shall discuss later. Another, but often more expensive, solution is to obtain more data; if we get more data, the curves for the training and test error get closer to each other.



(a)



(b)

Figure 9: (a) Plot of training and test error vs number of features used in the linear model to fit the toy problem described in Section 2.2 (b) Plot showing the weights to each feature in the models as a function of the number of features used.

2.2 How does overfitting occur?

Let us try to understand why models overfit the data. Of course, the obvious answer is because the number of parameters in the model can be very large, especially after performing basis expansion. Here's a quote from Enrico Fermi as reported by Freeman Dyson, [2]

> I remember my friend Johnny von Neumann used to say, with four parameters I can fit an elephant, and with five I can make him wiggle his trunk.

Compare this to the number of parameters in our model, if our original inputs are in $\mathbb{R}^{100}$ and we use degree 10 basis expansion. The number of parameters in the model is $\approx 10^{20}$! It is unlikely that we will ever get datasets of this size and even if we did, we wouldn't have the computational resources to fit them.

Overfitting occurs because we have a finite-size dataset as well as noise in the observations. Let's perform the following thought experiment. Take two unbiased coins and flip them each independently. Repeat the experiment one hundred times and count the number of times the two coin tosses produced the same outcome. Now, we expect this number to be roughly 50. However, we know that actually outcome is more likely to be some number in the range 40 to 60 other than 50! Let's say this number was 43, then we can conclude that by knowing the result of the first coin, we can predict the output of the second coin with about 57% accuracy, which is higher than 50%! Of course, this is patently absurd as the second coin is independent of the first one. This is essentially what is going on when we fit a complex model to a finite-size dataset.

Consider a toy dataset generated as follows: $N = 100$ inputs are generated in $D = 100$ dimensions, each entry of the $N \times D$ matrix is drawn from the normal distribution with mean 0 and variance 1. The output $y = x_1 + \epsilon$ only depends on the first component of $\mathbf{x}$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ for $\sigma = 0.2$. We know that the input features $x_2, \ldots, x_{100}$ have nothing to do with the output y. We fit a series of linear models $\mathbf{w}_1, \ldots, \mathbf{w}_{100}$, where the model $\mathbf{w}_i$ only uses the first i features.

[2] `http://www.nature.com/nature/journal/v427/n6972/full/427297a.html`

Figure 9(a) shows the training error and test error as a function of the number of features used. We can see that as we add more features the training error goes down but the test error actually increases! Figure 9(b) shows the actual weights on the i^{th} feature. Obviously, for model $\mathbf{w}_k$ the weight on all features $j > k$ is 0 because they are unused. Initially almost all the weight is on the first feature, but, as we allow the model to use more features, we see that other features also start getting larger weights. Essentially, by the time we reach $\mathbf{w}_{100}$, the model has fit the noise in the output variables perfectly using the irrelevant features!

References

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

Kevin P. Murphy. *Machine Learning : A Probabilistic Perspective.* MIT Press, 2012.

Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

John Shawe-Taylor and Nello Cristianini. *Kernel methods for pattern analysis.* Cambridge university press, 2004.