Simple Numeric Prediction

Machine Learning

Introduction I

Often we find tasks where the most natural representation is that of *prediction of numeric values*

For the class of *numeric* representations, machine learning is viewed as:

"searching" a space of *functions* ...

represented as mathematical models (linear equations, neural nets, ...).

Introduction II

Some methods:

- linear regression: the process of computing an expression that predicts a numeric quantity
- perceptron: a biologically-inspired linear prediction method
    - an "artificial neuron"
- logistic regression: learning a probability model using a non-linear transformation applied to the data
- multi-layer neural networks: learning non-linear predictors via hidden nodes between input and output (cascaded logistic regression)
- regression trees: tree where each leaf predicts a numeric quantity. The internal nodes are usually tests that decide how the tree is traversed (if *Mainmemory* > 512 then go to the right subtree, otherwise go to the left subtree)

Introduction III

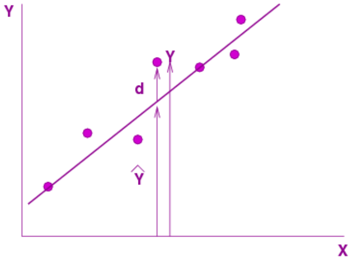
  - prediction in a leaf is the average value of (training) instances that reach the leaf
  - internal nodes test discrete **or** continuous attributes
- model trees: regression tree with linear or non-linear models at the leaf nodes

- We will look at the simplest model for numerica prediction: a *regression equation*
- The outcome will be a linear sum of feature values with appropriate weights.
- We will use this task to illustrate a number of aspects of Machine Learning

The Univariate Regression Problem

- Given a set of data points x_i, y_i, what is the relationship between them? (We can generalise this to the “multivariate” case later)
- One kind of question is to ask: are these linearly related in some manner? That is, can we draw a straight line that describes reasonably well the relationship between X and Y
- Remember, the correlation coefficient can tell us if there is a case for such a relationship
- In real life, even if such a relationship held, it will be unreasonable to expect all pairs x_i, y_i to lie precisely on a straight line. Instead, we can probably draw some reasonably well-fitting line. But which one?

Linear Relationship Between 2 Variables I



- GOAL: fit a line whose equation is of the form $\hat{Y} = a + bX$
- HOW: minimise $\sum_i d_i^2 = \sum_i (Y_i - \hat{Y}_i)^2$ (the "least squares estimator")

Linear Relationship Between 2 Variables II

- Solving for b gives:

$$b \;=\; \frac{\mathrm{cov}(x,y)}{\mathrm{var}(x)}$$

  where $\mathrm{cov}(x,y)$ is the covariance of x and y, given by $\sum_i (x_i - \overline{x})(y_i - \overline{y})$ (see slides on Mathematical Basics)

- This can be simplified to:

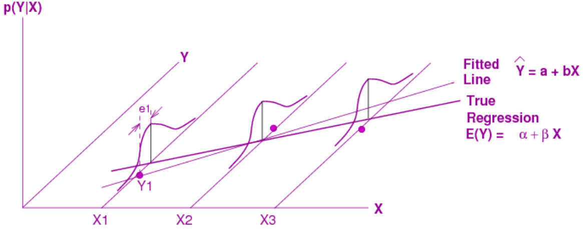
$$b = \sum(xy) / \sum x^2$$

  where $x = (X_i - \overline{X})$ and $y = (Y_i - \overline{Y})$

- $a = \overline{Y} - b\overline{X}$

The Probabilistic Setting for Regression I

- The least-square estimator fits a line using sample data
- To draw inferences about the population requires us to have a (probabilistic) model about what this line means
- What is being assumed is actually this:



- That is: Obtain Y values for many instances of X_1. This will result in a distribution of Y values $P(Y|X_1)$; and so on for $P(Y|X_2), P(Y|X_3), etc.$. The regression model makes the following assumptions:

The Probabilistic Setting for Regression II

- The x's are values of a r.v. X_i with some arbitrary distribution
- For $X_i = x_i$, the Y's are distributied according to $P(Y|x)$. It is assumed that $P(Y|X)$ distributions are the same, and have the same spread
- For each $P(Y|X_i)$ distribution, the true mean value μ_i lies on a straight line (this is the "true regression line"). That is $\mu_i = \alpha_i + \beta_i X_i$
- The Y_i are independent

- Uhe Y_i are identically distributed independent random variables with mean $\mu_i = \alpha + \beta X_i$ and variance σ^2
- Or: $Y_i = \alpha + \beta X_i + \epsilon_i$ where the ϵ_i are independent errors with mean 0 and variance σ^2

The Probabilistic Setting for Regression III

- In the tutorials, we will see 2 ways of building linear models: using Linear Algebra and using Optimisation. Here we look at the probabilistic approach, using some specific choices:
  1. The x's are values of a r.v. X with some arbitrary distribtion;
  2. If $X = x$, then $Y = \alpha + \beta x + \epsilon$, for values of the parameters α and β, and some random variable; and
  3. $\epsilon \sim N(0, \sigma^2)$. This is the same as saying $Y|X \sim N(\alpha + \beta x, \sigma^2)$; and
  4. ϵ is independent of the values of X
- Now, given a dataset $d = (x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$, of indeependent instances, what is the model with the highest posterior probability?
- The hypothesis here are the values of weights (α, β). That is, what we want are the pdfs of continuous valued r.v.'s

The Probabilistic Setting for Regression IV

- From Bayes:

$$f((\alpha,\beta)|d) \propto f(d|\alpha,\beta)f(\alpha,\beta)$$

Let us assume a uniform prior over all hypotheses. This will mean that maximising the posterior is equivalent to maximising the likelihood. Now, the likelihhod is:

$$L = f(d|\alpha,\beta) \;=\; \prod_{i=1}^{n} f((x_i,y_i)|\alpha,\beta) \;\propto f(y_i|x_i,\alpha,\beta)$$

With the specific assumptions:

$$f(y_i|x_i,\alpha,\beta) \propto \prod_{i=1}^{n} ke^{-\frac{(y_i-(\alpha+\beta x_i))^2}{2\sigma^2}}$$

We do not know the true values of α, β and σ

The Probabilistic Setting for Regression V

- If we want the estimates for α, β, σ that maximise the likelihood, or equivalently the log likelihood, then we want to maximise:

$$\log L = C - \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - (\alpha + \beta x_i))^2$$

- It is easy to see that maximising $\log L$ is equivalent to minimising squared error. Finding the partials w.r.t α, β, and setting to 0 gives the same estimates a, b for α, β as mimimising the MSE:

$$b = \frac{\sum (x_i - \overline{x})(y_i - \overline{y})}{(\sum (x_i - \overline{x})^2}$$

and:

$$a = \overline{y} - b\overline{x}$$

The Probabilistic Setting for Regression VI

- That is, under the assumption of independent Gaussian noise, and equiprobable priors, the least-square estimates of a and b are the same as the maximimum-likelihood estimates
- In your tutorials, you will also see that these estimates are unbiased. That is $\mathrm{E}[a] = \alpha$ and $\mathrm{E}[b] = \beta$; and that $MSE \;=\; Var + (Bias)^2$. So, under these assumptions, the estimates are the most effocient estimates of α, β
- The results hold under slightly more general condition, called the *Gauss-Markov* assumptions:
  1. The expected (average) values of residuals is 0 ($E(e_i) = 0$)
  2. The spread of residuals is constant for all X_i ($Var(e_i) = \sigma^2$)
  3. There is no relationship amongst the residuals ($cov(e_i, e_j) = 0$)
  4. There is no relationship between the residuals and the X_i ($cov(X_i, e_i) = 0$)

The Probabilistic Setting for Regression VII

- If these assumptions hold, then the Gauss-Markov theorem shows that $E(a) = \alpha$, $E(b) = \beta$, and that the variance in these estimates will have the lowest variance (*i.e.* the estimates are the most efficient)
- There is a special case of the assumptions that arises when the residuals are assumed to be distributed according to the Normal distribution, with mean 0
  - In this case, minimising least-squares is equivalent to maximising the probability of the Y_i, given the X_i (that is, least-squares is equivalent to *maximum likelihood estimation*)

Multivariate Regression

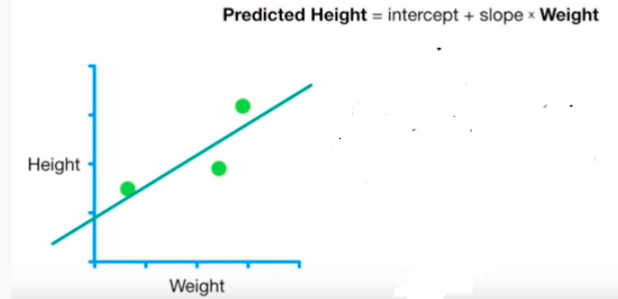
- Often, we are interesting in modelling the relationship of Y to several other variables
- In observational studies, the value of Y may be affected by the values of several variables. For example, carcinogenecity may be gender-specific. A regression model that ignores gender may find that carcinogenecity to be related to some surrogate variable (height, for example)
- Including more variables can give a narrower confidence interval on the prediction being made

The General Linear Model

- The Y_i are identically distributed independent variables with mean $\mu = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$ and variance σ^2
- Or: $Y_i = \beta_0 + \beta_1 X_1 + \cdots + \beta_n X_n + e_i$ where the e_i are independent errors with mean 0 and variance σ^2
- As before, this linear model is estimated from a sample by the equation $\hat{Y} = b_0 + b_1 X_1 + b_2 X_2 + \cdots + b_n X_n$
- With many variables, the regression equation and expressions for the b_i are expressed better using a matrix representation for sets of equations.

Parameter Estimation as Optimisation I

- Let us look at estimating values of one parameter for a simple linear model:[1]



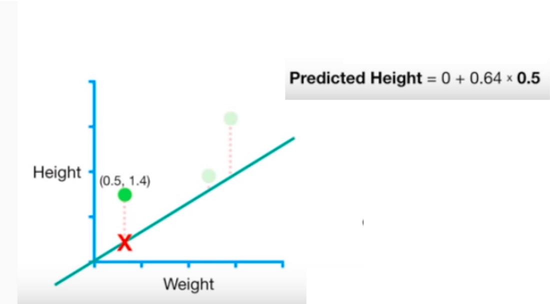
- For the moment, we will simply minimise the sum of squared residuals (SSR)
- The parameters of this model are the intercept and the slope. So, ideally, we want to find optimal values for these two quantities.

Parameter Estimation as Optimisation II

- Using the usual method of partial differentiation w.r.t the slope and and the intercept gives optimal values of $\mathit{Intercept} = 0.95$ and $\mathit{Slope} = 0.64$
- BUT: analytical optimisation is difficult to automate, and may not even be possible in some cases
  - Instead, we will look at a general-purpose greedy search (gradient descent)
- Let us first look at how gradient descent works with one parameter. We will assume we have already found the value of the slope to be 0.64
- For any predicted line, let us take the sum of squared residuals as the *loss function*. This is sometimes called the *squared loss* function

Parameter Estimation as Optimisation III

- We can now evaluate the sum of the squares of residuals for any predicted line



Predicted Height = 0 + 0.64 × **0.5**

Parameter Estimation as Optimisation IV

- We can write the sum of squares of residuals as the following:

  $$SSR \;=\; [1.4-(I+0.64\times0.5)]^2 + [1.9-(I+0.64\times2.3)]^2 + [3.2-(I+0.64\times2.9)]^2$$

  This can be visualised as function between I (X-axis) and SSR (Y-axis). As I gets the optimal value, the slope of this function will be close to 0
- The derivative of the loss function w.r.t. I is then:

  $$\frac{d(SSR)}{dI} \;=\; -2[1.4-(I+0.64\times0.5)] - 2[1.9-(I+0.64\times2.3)] - 2[3.2-(I+0.64\times2.9)]$$

- Let is start with a random value of *Intercept*, say $I=0$.
- The value of this derivative at $I=0$ (our first guess) is -5.7. This is the slope of the I vs SSR curve at $I=0$

Parameter Estimation as Optimisation V

- Gradient takes steps along I using the the slope:

$$I_{k+1} \;=\; I_k - \eta \times \frac{d(SSR)}{dI_k}$$

Here, if $\eta = 0.1$, then the step is $-5.7 \times 0.1 = -0.57$ and:

$$I_2 \;=\; 0 - (-0.57) \;=\; 0.57$$

- Repeat the process of calculating the derivative with $I = 0.57$, which gives a slope of -2.3 at $I = 0.57$. The new step size is therefore -0.23 and:

$$I_3 \;=\; 0.57 + 0.23 \;+\; 0.8$$

- Repeating will result in intercepts $I = 0.89, 0.92, 0.94, 0.95$

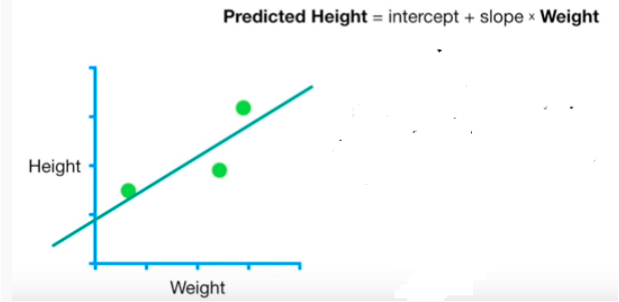
Parameter Estimation as Optimisation VI

- So, gradient descent gets to the optimal value (0.95). But how does it know to stop? Stops when the step size is very close to 0. Since $Step = \eta Slope$, this must mean that *Slope* is very close to 0.
- In this case, gradient descent will stop with $I = 0.95$

[1] The following example from Statquest's online presentation on Gradient Descent

Example: Multi-parameter Gradient Descent I

- Let us look at finding values of both parameters for the simple linear model:[2]



Predicted Height = intercept + slope × **Weight**

- We will now look at estimating both *Slope* and *Intercept* using gradient descent. As before, we start with the loss function:

Example: Multi-parameter Gradient Descent II

- We can write the sum of squares of residuals as the following:

  $$SSR = [1.4-(I + S\times 0.5)]^2 + [1.9-(I + S\times 2.3)]^2 + [3.2-(I + S\times 2.9)]^2$$

  SSR is now a function of both *I* and *S*, and we want to estimate the optimal value of both *I* and *S*.

- We will need to calculate slopes w.r.t *I* and *S* separately. These are the following:

  $$\frac{\partial(SSR)}{\partial I} = -2[1.4-(I + S\times 0.5)] - 2[1.9-(I + S\times 2.3)] - 2[3.2-(I + S\times 2.9)]$$

  $$\frac{\partial(SSR)}{\partial S} = -2\times 0.5[1.4-(I + S\times 0.5)] - 2\times 2.3[1.9-(I + S\times 2.3)] - 2\times 2.9[3.2-(I + S$$

- Like before, we will start with random choices, say: $I = 0$, $S = 1$. This gives us 2 slopes:

  $$\frac{\partial(SSR)}{\partial I} = -1.6 \qquad \frac{\partial(SSR)}{\partial S} = -0.8$$

Example: Multi-parameter Gradient Descent III

- With a learning rate of $\eta = 0.01$, we get step-sizes of 0.01×-1.6 for I and 0.01×-0.8 for S. So, the new values of I and S are

$$I_2 \ + \ I_1 - \eta \times \frac{\partial(SSR)}{\partial I_1} \ = \ 0 + 0.016 \ = \ 0.016$$

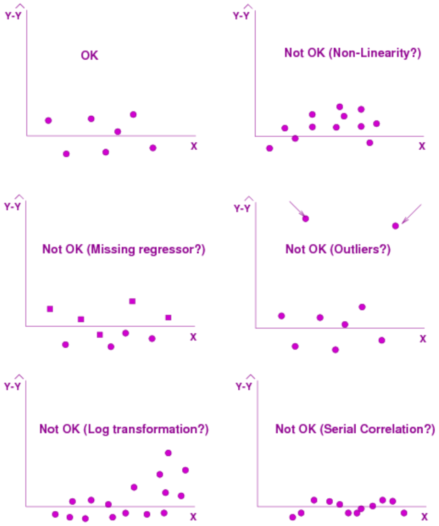
and

$$S_2 \ + \ S_1 - \eta \times \frac{\partial(SSR)}{\partial S_1} \ = \ 1 + 0.008 - \ = \ 1.008$$

- Repeat with the new values of I and S, until step sizes are very small. Here, gradient descent terminates with $I = 0.95$ and $S = 0.64$

[2] The following example from Statquest's online presentation on Gradient Descent

Is the Model Appropriate? I



Is the Model Appropriate? II

- The residuals from the regression line can be calculated numerically, along with their mean, variance and standard deviation. It can be shown that the residual standard deviation is related to the standard deviation of the Y values in the following manner:

$$rsd \;=\; s_y\sqrt{1-r^2}$$

- This helps us understand how much the regression line helped reduce the scatter of the y values (s_y gives a measure of the scatter of y values about the mean $\overline{y}$; and rsd gives a measure of the scatter of y values about the regression line)
- This also gives you another way of understanding the correlation coefficient. With $r = 0.9$, the scatter about the regression line is still almost 45% of the original scatter about the mean

Is the Model Appropriate? III

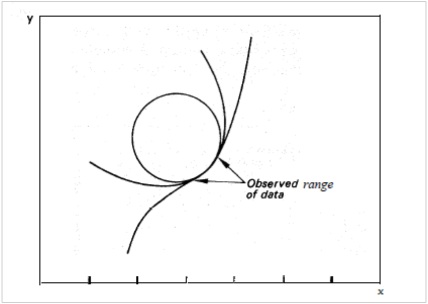
- If there is no systematic pattern to the residuals—that is, there are approximately half of them that are positive and half that are negative, then the line is a good fit
- It should also be the case that there should be no pattern to the residual scatter all along the line. If the average size of the residuals varies along the line (this condition is called *heteroscedasticity* then the relationship is probably more complex than a straight line
- Residuals from a well-fitting line should show an approximate symmetric, bell-shaped frequency distribution with a mean of 0

Changing the Model Structure

- Sometimes, the linear model may be inappropriate
- Some non-linear relationships can be captured in a linear model by a transformation ("trick"). For example, the curved model $\hat{Y} = b_0 + b_1 X_1 + b_2 X_1^2$ can be transformed by $X_2 = X_1^2$ into a linear model. This works for polynomial relationships.
- Some other non-linear relationships may require more complicated transformations. For example, the relationship is $Y = b_0 X_1^{b_1} X_2^{b_2}$ can be transformed into the linear relationship $\text{log}(Y) = \log(b_0) + b_1 \log X_1 + b_2 \log X_2$
- Other relationships cannot be transformed quite so easily, and will require full non-linear estimation (attend the ML course to find out more about these)

Non-Linear Relationships (contd.)

- The main difficulty with non-linear relationships is the choice of function
  - We can use a form of gradient descent to get an estimate of the parameters involved
- After a point, almost any sufficiently complex mathematical function will do the job in a sufficiently small range



- Some kind of prior knowledge or theory is the only real way to help here. Otherwise, it becomes a process of trial-and-error, in which case, beware of conclusions that can be drawn

Model Selection

- Suppose there are a lot of variables X_i, some of which may be representing products, powers, *etc*.
- Taking all the X_i will lead to an overly complex model. There are 3 ways to reduce complexity:
    1. Subset-selection, by search over subset lattice. Each subset results in a new model, and the problem is one of model-selection
    2. Regularization by optimization. There is a single model, and unimportant variables have near-zero coefficients.
    3. Dimensionality-reduction, by projecting points into a lower dimensional space (this is different to subset-selection, and we will look at it later)

Structure Estimation by Combinatorial Search I

- The subsets of the set of possible variables form a lattice with $S_1 \cap S_2$ as the g.l.b. or meet and $S_1 \cup S_2$ as the l.u.b. or join
- Each subset refers to a model structure, and a pair of subsets are connected if they differ by just 1 element
- A lattice is a graph, and we know how to search a graph
  - A^*, greedy, randomised *etc*.
  - "Cost" of node in the graph: MSE of the model. The parameters (coefficients) of the model can be found by gradient descent, if needed
- Historically, model-selection for regression has been done using "forward-selection", "backward-elimination", or "stepwise" methods

Structure Estimation by Combinatorial Search II

- These are greedy search techniques that either: (a) start at the top of the subset lattice, and add variables; (b) start at the bottom of the subset lattice and remove variables; or (c) start at some interior point and proceed by adding or removing single variables (examining nodes connected to the node above or below)
- Greedy selection done on the basis of calculating the *coefficient of determination* (often denoted by R^2) which denotes the proportion of total variation in the dependent variable Y that is explained by the model
- Given a model formed with a subset of variables X, it is possible to compute the observed change in r^2 due to the addition or deletion of some variable x
- This is used to select greedily the next best move in the graph-search

- NOTE: Each model's parameters still have to be estimated

Model Selection as Optimisation I

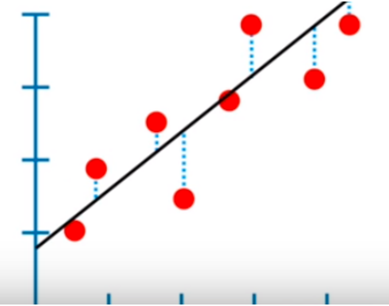
- It is possible to re-cast the model selection problem as an optimisation problem that trades-off the cost of increasing the complexity of the model against the performance of the model
- We will often describe a model as having *structure* and *parameters*
  - The structure of a linear model specifies the terms in the model and the parameters are the coefficients
- For the present, let start with a single variable and assume a fixed structure $Y = a + bX$.
- Finding the least-squares solution is in effect an optimisation problem of finding the values of a and b that minimizes $\sum_i d_i^2 = \sum_i (Y_i - \hat{Y}_i)^2$, where $\hat{Y}_i = a + bX_i$
- We can also try to include structure of the model as part of the optimisation problem so that we trade-off increases in model complexity with decrease in mean-square-error

Model Selection as Optimisation II

- We will look at ways of doing this first, and then look at numerical ways of performing the optimisation

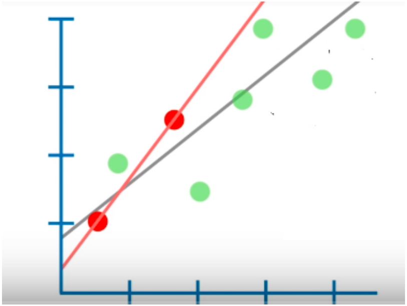
Including Structure into the Loss Function I

- So far, the loss function we have sought to minimise (optimise) is the sum of squared residuals (*SSR*) to fit a line for some data points. We have implicitly relied on prior knowledge to avoid fitting an overly complicated function



Including Structure into the Loss Function II

- But this will not always save us. Suppose we had just saw 2 of the data points:



- Minimising *SSR* necessarily results in the line through the 2 points. This clearly has a high error when used to estimate the Y-value on the rest of the data. This is an example of "over-fitting".

Including Structure into the Loss Function III

- In MORAL: over-fitting can happen not just by having a very complex structure, but even with very few data points. How can we deal with this?
- Suppose we are looking for lines of the form $Y = I + SX$. Here I is theintercept and S is the slope. Suppose instead of minimising SSR, we minimised instead:

  $$L = SSR + \lambda S^2$$

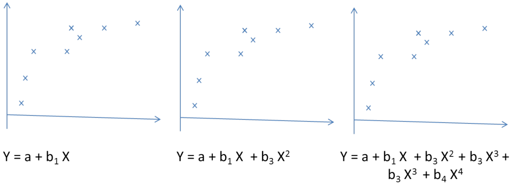
  where λ is some positive number. Then, if we consider a line with a large value of S (positive or negative), it will also have a large value of L.
- So, the line that minimises L will have a lower S than one that minimises SSR. In other words, it will be "flatter", and changes in X will result in smaller changes in Y

Including Structure into the Loss Function IV

- For the line through the 2 points, $SSR = 0$, and the slope is 1.3. If $\lambda = 1$, then $L = 1.69$. Increasing λ will result in increasingly flatter lines (less role for changes in X)
- In the limiting case, X is completely irrelevant (the line that minimises L has S close to 0). So, the model is simplified to $Y = I$

Regularisation I

- In general, the problem dealt with by regularisation is to make smodel-structure simpler. We can increase the number of terms in the linear model, even with a single variable



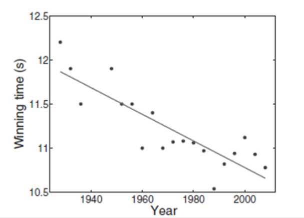
$Y = a + b_1 X$

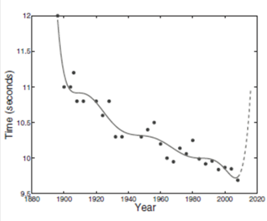
$Y = a + b_1 X + b_3 X^2$

$Y = a + b_1 X + b_3 X^2 + b_3 X^3 + b_3 X^3 + b_4 X^4$

Regularisation II

- In general, more complex structure will allow us to fit the observed data better (for example, using n^{th}-order polynomial to fit n data points)





In general, if the number of parameters in the model $\geq$ number of data points, then we can end up over-fitting.

- That is, we can end up with a model with low error on the observed data, but a high expected error on all data drawn from the same source

Regularisation III

- BUT: we don't have all the data from the same source. So, what can we do?
  1. Estimate the expected error using the data we have
  2. Estimate the expected error using a new data sample
  3. Identify the model that optimises a *cost function* that trades-off decrease in observed error against increase in model complexity

$$\textit{Total Cost} = \textit{Training Cost} \quad + \quad \textit{Complexity Cost}$$

  The "Complexity Cost" is sometimes called the *regularisation* term

Regularisation IV

- In general, suppose Y is modelled as some function f of $X_1, X_2, \ldots, X_d$, with parameters $w_0, w_1, \ldots, w_k$ (which we simply call w)

  $$Y \;=\; f_{\mathrm{w}}(\mathrm{X})$$

  For example, for a linear model f is a linear function with $d+1$ parameters. Given data $(\mathrm{x}_1, y_1), \ldots, (\mathrm{x}_\mathrm{n}, y_n)$. Here, each x_i is a d-dimensional vector. One possible training cost is the MSE:

  $$Training\ \ Cost \;=\; MSE \;=\; \frac{1}{n}\sum_i (f_{\mathrm{w}}(\mathrm{x}_\mathrm{i}) - y_i)^2$$

Regularisation V

- But, what should the regularisation term be? One answer is that this depends on how many of the coefficients are non-zero (more of these means a more complex model). So, one possibility for Complexity Cost is just the sum of the absolute value of the coefficients, or even $\sum_{i=0}^{d} w_i^2$
- So:

$$\mathit{Total\ Cost} \ = \ \mathit{MSE} \ + \ \lambda \sum_i w_i^2$$

  where λ is some weighting factor'
- As before, parameter estimation by optimisation will attempt to values for the w_i s.t. *Total Cost* is a minimum. As usual, there are two ways of doing this: analytically, or numerically
  - Analytically determining the values of w_i that minimise *Total Cost* requires us to find the partial derivatives of *Total Cost* w.r.t. the w_i and setting them to 0

Regularisation VI

- Numerically determining the values of w_i that minimise *Total Cost* requires us to find the gradient, and change the w_i in the opposite direction

$$w_i^{(k+1)} = w_i^{(k)} - \eta \nabla_{w_i}$$

- It is in fact common to use an update formula each value of w_i also gets "shrunk" on each iteration by multiplying the old value by an amount $\alpha < 1$:

$$w_i^{(k+1)} = \alpha w_i^{(k)} - \eta \nabla_{w_i}$$

Regularisation VII

Question. Derive the analytical solution for optimal weight values.

Question. How do we decide on values for λ, η, and α?

Regularisation as a Prior

- The addition of a term penalising complexity is a form of introducing a prior. In the usual Bayesian formulation, we want to minimise:

  $$-\log\ P(H|D) \ = \ -\log P(D|H) - \log P(H)$$

  Comparing back, the first term on the r.h.s. is the training cost, and the second (the regularisation penalty) is the complexity cost.
- For linear models, when we use a regularisation penalty of $\lambda \sum w_i^2$, we are implicitly using a prior:

  $$P(H) \ = \ 2^{-\lambda \sum w_i^2}$$

  where the w_i are parameters of H

Gradient Descent with any loss function I

1. Take the derivative ("gradient") of the loss function w.r.t. each parameter
2. Pick random values for the parameters
3. Calculate the values of the derivative (gradient) for the (current) values of the parameters
4. Calculate (current) values for parameters using old values and the step size (= learning rate × gradient)
5. Go to Step 3

Gradient Descent with any loss function II

Question. What happens the the cost function being minimised has a regulatisation term?

Question. What happens when there are 1000's of parameters and millions of data points?

Question. What happens when the cost function being minimised is not convex?

Stochastic Gradient Descent (SGD) I

- With gradient descent, there is 1 gradient equation for every parameter; and every equation has to be evaluated for each data point
- If there are 1,000's of parameters (can happen) and 1,000,000's of data points (can happen) then each step requires calculating at least 1 billion terms. If there 1000 iterations, then there may be 1 trillion terms to be calculated
- There are 3 was to reduce the computational effort: (a) reduce the number of parameters; or (b) reduce the number of data points; or (c) both. *Stochastic gradient descent* does (b) by simply using a single randomly chosen example on each iteration.
- SO, if there are 1,000,000 data points, then on each step the computation is 1,000,000 times lesser

Stochastic Gradient Descent (SGD) II

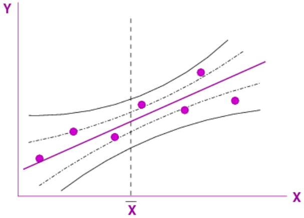
- ALSO, if data become available 1-at-a-time, SGD continues to update parameter values

Prediction I

- Let us look again at the linear model $Y = a + bX$. Two sorts of questions can be asked: (1) How will the mean of Y $(= \mu)$ values vary for repeated observations of X? and (2) What can we say about the value of Y $(= y_k)$ for a (given) single value of X $(= x_k)$?
- In both cases, the answers will be scattered around $a + bX$:
  1. $\mu = (a + bX) \pm t_{\alpha/2} \times s_1$
  2. $y_k = (a + bX) \pm t_{\alpha/2} \times s_2$
- For small samples, $s_2 > s_1$. Also, as we move away from $\overline{X}$, both $s_{1,2}$ increase. So, the prediction becomes increasingly less reliable

Prediction II

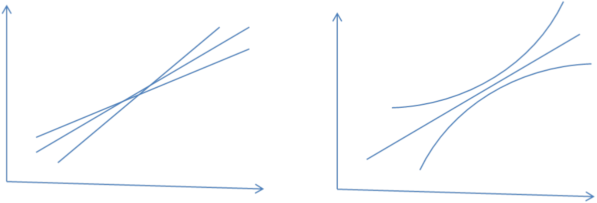
- It is therefore possible to quantify what happens if the regression line is used for prediction:



- The intuition is this:
  - Recall the regression line goes through the mean $(\overline{X}, \overline{Y})$
  - If the X_i are slightly different, then the mean is not going to change much. So, the regression line stays somewhat "fixed" at $(\overline{X}, \overline{Y})$ but with a different slope

Prediction III

- With each different sample of the X_i we will get a slightly different regression line
- The variation in Y values is greater further we move from $(\overline{X}, \overline{Y})$



- MORAL: Be careful, when predicting far away from the centre value
- ANOTHER MORAL: The model only works under the approximately the same conditions that held when collecting the data

Summary

- Linear models give us a glimpse into many aspects of Machine Learning

  Conceptual. Learning as search, learning as optimisation, assumptions underlOying a technique

  Optimisation. Gradient descent, stochastic gradient descent, regulatisation

  Application. Errors in prediction

  Each of these aspects will have counterparts in other kinds of machine learning
- NEXT: extend linear models to Linear Threshold Machines
  - Linear Threshold Machine: Perceptron
  - Multi-layer Perceptrons: Neural Networks
  - Linear Perceptrons with Regularisation: Support Vector Machines
  - Probabilistic Threshold Machine: logistic regression