Cross Validation and Regularized Least Squares

Rohit Budhiraja

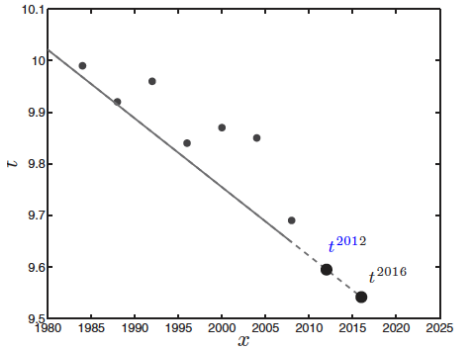
Machine Learning for Wireless Communications (EE798L)

Jan 15, 2024

Recap of last lecture and today's agenda

- Recap of last class
  - Generalized the derivation by writing the model in vector/matrix notation
  - Obtained non-linear response from a linear model and fitted model of various orders
  - Discussed the problem of higher order model not being able to predict (generalize) well
- Today's agenda
  - Discuss generalisation and over-fitting
  - Idea of cross validation and regularized least squares to solve above problems

First order model for Olympic data (recap)

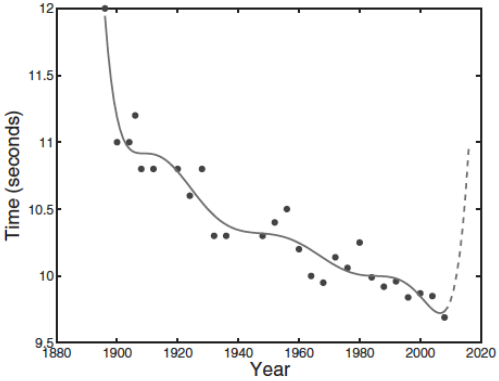


- Fitted first-oder model and predicted winning time for 2012 and 2016

$$
\begin{aligned}
f(x; w_0 = 36.416, w_1 = -0.0133) &= 36.416 - 0.0133x \\
t^{2012} = f(2012; w_0, w_1) &= 36.416 - 0.0133 \times 2012 = 9.595s \\
t^{2016} = f(2016; w_0, w_1) &= 36.416 - 0.0133 \times 2016 = 9.541s
\end{aligned}
$$

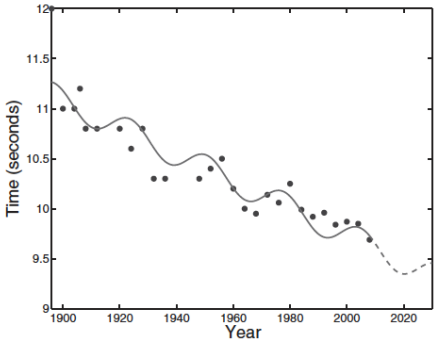
- Predictions look sensible

Eighth order model for Olympic data (recap)



- Eighth-order polynomial gets closer to observed data than first-order polynomial
- Reflected in a lower value of the loss function: $\mathcal{L}^8 = 0.459, \mathcal{L}^1 = 1.358$
  - Increasing the polynomial order will always result in a model that gets closer to the training data
- Secondly, predictions (shown by the dashed line) do not look sensible

Third order model for Olympic data (recap)



- Model fits the observed data better than the first order polynomial with $\mathcal{L} = 1.1037$
  - But not as well as the eighth order polynomial
- Predictions, however, look sensible

Idea of generalisation and over-fitting

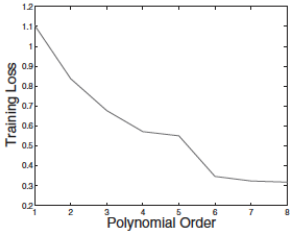
- We posed the question of which was better, the first- or eighth order polynomial
- Aim to build these models is to predict
  - Sensible to think of the best model as the one which makes the most accurate predictions
  - Such a model should generalize beyond the examples we have for training
- Ideally, we would like to choose a model that performs best (i.e. minimises the loss) on unseen data
  - but, by the very nature of the problem, this data is unavailable
- Eight-order polynomial indicated that we should be suspicious of using the loss on training data to choose a model that will be used to make predictions
  - Much lower loss on training data than a first order polynomial
  - At the same time, predictions for future Olympics are very poor
  - Model pays too much attention to training data (it over-fits) – does not generalise well to new data
- As we make models more complex, they will be able to get closer and closer to training data
  - beyond a certain point, quality of predictions can deteriorate rapidly
- Determining optimal model complexity such that it generalizes well without over-fitting is challenging
  - Trade-off is often referred to as bias-variance trade-off and we will briefly discuss this later

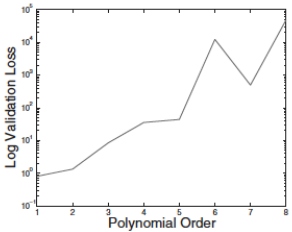
Techniques to overcome generalisation/over-fitting problem

- Use validation data
- Regularization

Overcome generalisation/over-fitting using validation data

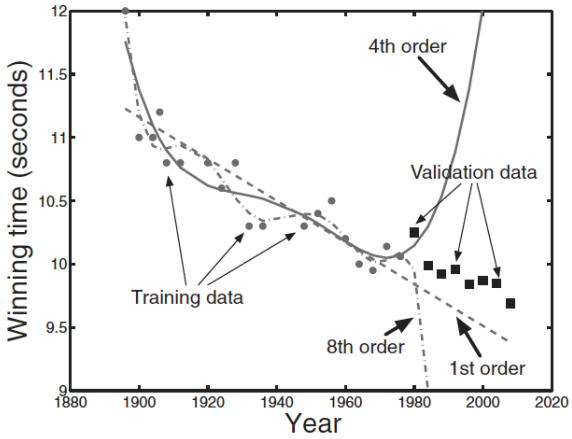
- Common way to overcome generalisation/over-fitting problem is to use a second dataset
  - Referred as validation data set – validates the predictive performance of our model
  - Validation data could be provided separately or created by removing data from original training set
- To choose between model orders, we train each model on reduced training set, and then compute their loss on the validation set
- Plots of training and (log) validation losses





- Validation loss: $\mathcal{L} = \frac{1}{N_v} \sum_{n=1}^{N_v} (t_n - \mathbf{w}^T \mathbf{x}_n)^2$ with N_v being amount of validation data
- With polynomial order, training loss decreases monotonically but validation loss increases
  - First-order polynomial has best generalisation ability and will produce the most reliable predictions

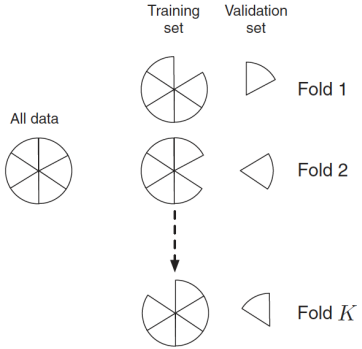
Confirming the hypothesis with validation data



- This hypothesis is easily tested

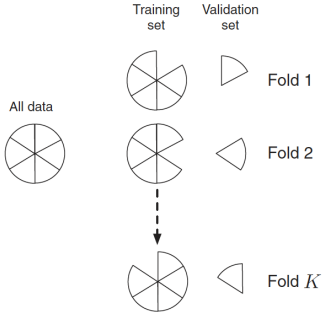
Cross validation (1)

- Loss calculated from validation data will be sensitive to the choice of data in our validation set
  - Particularly problematic if our dataset (and hence our validation set) is small
- Cross-validation is a technique that allows us to make more efficient use of the data we have



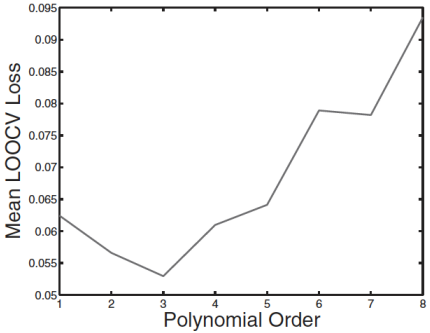
- K-fold cross-validation splits data into K equally (or as close to equal as possible) sized blocks
- Each block takes its turn as a validation set for a training set comprised of the other $K-1$ blocks
- Averaging over the resulting K loss values gives us our final loss value

Cross validation (2)



- An extreme case of K-fold cross-validation is where $K = N$, number of observations in dataset
- Each data observation is held out in turn and used to test a model trained on other $N - 1$ objects
- This particular form of cross-validation is called Leave-One-Out Cross-Validation (LOOCV)
- Squared validation loss for LOOCV while leaving out first sample $= (t_1 - \widehat{\mathbf{w}}_{-1}^T \mathbf{x}_1)^2$
- Average squared validation loss for LOOCV $\mathcal{L}^{CV} = \frac{1}{N} \sum_{n=1}^{N} (t_n - \widehat{\mathbf{w}}_{-n}^T \mathbf{x}_n)^2$
  - $\widehat{\mathbf{w}}_{-n}$ is parameters estimate without the nth training example

Cross validation (3)



- Above figure shows mean LOOCV error for the Olympic men's 100m data
- Plot suggests that a third-order polynomial would be best
- Disagreement with the value obtained from using the last few data points as a validation set
  - Disagreement like this is not uncommon – model selection is a very difficult problem
- Two methods do agree on one thing – model certainly shouldn't be sixth-order or above

Regularized least squares

- Discussed how validation loss could ensure good predictive (generalisation) performance
  - This process prevents the model from i) becoming too complex; and ii) overfitting data
- There is another way to stop model from becoming too complex, known as regularisation
- Consider a trivial model $f(x; \mathbf{w}) = \mathbf{w}^T \mathbf{x}$ where $\mathbf{w} = [0, 0 \ldots, 0]^T$
  - Model always predicts a value 0
- Any change made to elements of $\mathbf{w}$ increases their absolute value and makes model more complex
- Specifically, consider the fifth order polynomial model

$$f(x; \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 + w_5 x^5$$

- If we start with all of the elements of $\mathbf{w}$ being zero, the function always predicts a value of zero
- Now imagine we set w_0 to some non-zero value – model now predicts a constant (w_0)
- Leaving w_0 at its new value, we can set w_1 to some value – model has become more complex
- Each additional parameter is given a non-zero value, model becomes more complex still
- In general, higher the sum of absolute values in $\mathbf{w}$, more complex the model
- As absolute values slightly complicates the maths, we could define model complexity $\sum_i w_i^2 = \mathbf{w}^T \mathbf{w}$
- We don't want our model to become too complex, it makes sense to try and keep this value low

Regularized least squares – optimal value of w

- Rather than just minimising the average squared loss $\mathcal{L}$, we could minimise a regularised loss $\mathcal{L}'$

$$\mathcal{L}' = \mathcal{L} + \lambda \mathbf{w}^T\mathbf{w} = \frac{1}{N}(\mathbf{X}\mathbf{w} - \mathbf{t})^T(\mathbf{X}\mathbf{w} - \mathbf{t}) + \lambda \mathbf{w}^T\mathbf{w}$$

- Parameter λ controls trade-off between penalizing not fitting the data well ($\mathcal{L}$), and penalizing overly complex models ($\mathbf{w}^T\mathbf{w}$)
- Adding the regularisation term to our original squared loss

$$\mathcal{L}' = \frac{1}{N}\mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} - \frac{2}{N}\mathbf{w}^T\mathbf{X}^T\mathbf{t} + \frac{1}{N}\mathbf{t}^T\mathbf{t} + \lambda \mathbf{w}^T\mathbf{w}$$

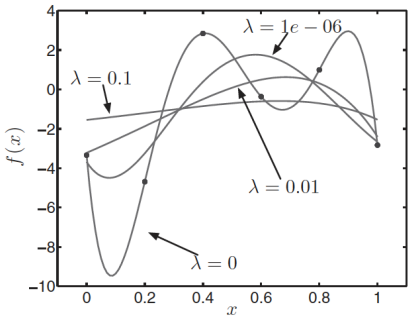
- Taking partial derivatives with respect to $\mathbf{w}$ and setting it to zero

$$\begin{aligned}
\frac{\partial \mathcal{L}'}{\partial \mathbf{w}} &= \frac{2}{N}\mathbf{X}^T\mathbf{X}\mathbf{w} - \frac{2}{N}\mathbf{X}^T\mathbf{t} + 2\lambda\mathbf{w} \\
\frac{2}{N}\mathbf{X}^T\mathbf{X}\mathbf{w} - \frac{2}{N}\mathbf{X}^T\mathbf{t} + 2\lambda\mathbf{w} &= 0 \\
(\mathbf{X}^T\mathbf{X} + N\lambda\mathbf{I})\mathbf{w} &= \mathbf{X}^T\mathbf{t} \\
\Rightarrow \widehat{\mathbf{w}} &= (\mathbf{X}^T\mathbf{X} + N\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{t}
\end{aligned}$$

- Clearly, if $\lambda = 0$, we retrieve the original solution

Regularized least squares – effect of λ

- Visualize effect of λ increasing with a synthetic example - six training points and 5th order model



- For $\lambda = 0$, 5th order model is exactly fitting (over-fitting) six data points
  - In general N data points can be perfectly fitted by an $(N-1)$th order polynomial
- If we increase λ, regularisation starts taking effect
- $\lambda = 1 \times 10^{-6}$ follows the general shape of exact fifth-order polynomial, but without as much variability and subsequently is further from the data points
- $\lambda = 0.01$ and $\lambda = 0.1$ continue this trend – function become less complex and does not over-fits