CSCC11 Week 2 Notes

Linear Regression:

1. Introduction:

- Is a linear approach to modelling the relationship btwn a dep var and an indep var.

2. 1D Linear Regression:

- We want to find $y = f(x) + \varepsilon$ where:
  a) $f(x) = \underbrace{w}_{\text{weight}} x + \underbrace{b}_{\text{bias}}$

  "w" and "b" are the parameters of f.

  b) ε is the error term (I.e. noise)

- We want to find/estimate "w" and "b" s.t. $f(x)$ fits the training data as well as possible.

  The training data is just a set of input/output pairs. I.e. $\{(x_1, y_1), \ldots, (x_n, y_n)\}$.

- One way to do this is to min the vertical dist between the actual value and the predicted value. We can do this using the Least Squares Method.

- Let $e_i = y_i - f(x_i)$
  $= y_i - (wx_i + b)$

  The loss function, $L(w,b)$, is equal to $\sum_{i=1}^{n} (e_i)^2$

  $= \sum_{i=1}^{n} (y_i - (wx_i + b))^2$

Note: We need to square the error because of possible negative values.

- Finding the line that minimizes the squared error is equivalent to solving for "w" and "b" that minimize $L(w,b)$. This can be done by setting the derivatives of L w.r.t these parameters to 0 and then solving.

$$\frac{\partial L}{\partial b} = -2\sum_{i=1}^{N}(y_i - (wx_i + b)) = 0$$

$$0 = \sum_{i=1}^{N}(y_i - wx_i - b)$$

$$= \sum_{i=1}^{N} y_i - \sum_{i=1}^{N} wx_i - \sum_{i=1}^{N} b$$

$$= \sum_{i=1}^{N} y_i - w\sum_{i=1}^{N} x_i - bN$$

$$bN = \sum_{i=1}^{N} y_i - w\sum_{i=1}^{N} x_i$$

$$b^* = \frac{\sum_{i=1}^{N} y_i}{N} - \frac{w\sum_{i=1}^{N} x_i}{N}$$

$$= \hat{y} - w\hat{x}$$

We'll define $\hat{x}$ and $\hat{y}$ as the avgs of the x's and y's respectively.

Now, we can rewrite $L(\omega, b)$ as:

$$\sum_{i=1}^{N} (y_i - (\omega x_i + (\hat{y} - \omega \hat{x})))^2$$

$$= \sum_{i=1}^{N} (y_i - (\omega x_i + \hat{y} - \omega \hat{x}))^2$$

$$= \sum_{i=1}^{N} (y_i - \hat{y} - (\omega x_i - \omega \hat{x}))^2$$

$$= \sum_{i=1}^{N} ((y_i - \hat{y}) - \omega (x_i - \hat{x}))^2$$

Using this new form of L, we can try to solve for ω.

$$\frac{\partial L}{\partial \omega} = -2 \sum_{i=1}^{N} ((y_i - \hat{y}) - \omega (x_i - \hat{x}))(x_i - \hat{x})$$

$$= 0$$

$$0 = \sum_{i=1}^{N} (y_i - \hat{y})(x_i - \hat{x}) - \omega (x_i - \hat{x})^2$$

$$= \sum_{i=1}^{N} (y_i - \hat{y})(x_i - \hat{x}) - \omega \sum_{i=1}^{N} (x_i - \hat{x})^2$$

$$w \sum_{i=1}^{N} (x_i - \hat{x})^2 = \sum_{i=1}^{N} (y_i - \hat{y})(x_i - \hat{x})$$

$$w^* = \frac{\sum_{i=1}^{N} (y_i - \hat{y})(x_i - \hat{x})}{\sum_{i=1}^{N} (x_i - \hat{x})^2}$$

w^* and b^* are the least-square estimates for the parameters of the linear regression.

3. Multi-Dimensional Inputs:

- Now, let $x \in R^D$ (x is now a $1 \times D$ column vector.)

  $y \in R$

  $$f(x) = w^T x + b = \sum_{j=1}^{D} w_j x_j + b \qquad x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{bmatrix}$$

  ← 1 data point; each with D features

- To make $f(x)$ the result of a dot product, we can add b as the last element in w and add a 1 to x.

  I.e. $w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_D \\ b \end{bmatrix}$, $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \\ 1 \end{bmatrix}$

  Then, $f(x) = w^T x$

- We can define our loss function $L(\omega)$, to be $\sum_{i=1}^{N} (y - \omega^T x_i)^2$.

$$L(\omega) = \sum_{i=1}^{N} (y - \omega^T x_i)^2$$

This is the 2-norm squared

$$= \|\vec{y} - \tilde{X}\omega\|_2^2 \text{ where}$$

$$\vec{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

$$\tilde{X} = \begin{bmatrix} \text{---} & \tilde{x}_1^T & \text{---} \\ \text{---} & \tilde{x}_2^T & \text{---} \\ & \vdots & \\ \text{---} & \tilde{x}_N^T & \text{---} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1d} \\ 1 & x_{21} & x_{22} & \cdots & x_{2d} \\ & & \vdots & & \\ & & \vdots & & \end{bmatrix}$$

- Now, we want to find ω^*.

$$\begin{aligned} L(\omega) &= \|\vec{y} - \tilde{X}\omega\|_2^2 \\ &= (\vec{y} - \tilde{X}\omega)^T (\vec{y} - \tilde{X}\omega) \\ &= (\vec{y}^T - \omega^T \tilde{X}^T)(\vec{y} - \tilde{X}\omega) \\ &= \vec{y}^T y - \vec{y}^T \tilde{X}\omega - \underline{\omega^T \tilde{X}^T y} + \omega^T \tilde{X}^T \tilde{X}\omega \\ &= \omega^T \tilde{X}^T \tilde{X}\omega - 2\vec{y}^T \tilde{X}\omega + \vec{y}^T \vec{y} \end{aligned}$$

Recall: $(a \pm b)^T = a^T \pm b^T$

$(ab)^T = b^T a^T$

$(abc)^T = c^T b^T a^T = c^T (ab)^T$

$$\frac{\partial L(w)}{\partial w} = 2(\tilde{X}^T\tilde{X})w - 2\tilde{X}^T\vec{y} + 0 = 0$$

$$0 = (\tilde{X}^T\tilde{X})w - \tilde{X}^T\vec{y}$$

$$(\tilde{X}^T\tilde{X})w = \tilde{X}^T\vec{y}$$

$$w = \underbrace{(\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T}_{\text{Pseudo Inverse}}\vec{y}$$

$$\therefore w^* = (\tilde{X}^T\tilde{X})^{-1}\tilde{X}^T\vec{y}$$

Note: w^* must be a min bc L is convex w.r.t w.

Note: Because finding the inverse is expensive, another approach we can do to find the min is to use the gradient descent.

$$w_{i+1} = w_i - \underset{\substack{\uparrow \\ \text{Learning rate/Step size}}}{\alpha}\left(\frac{\partial L(w)}{\partial w}\right) \leftarrow \text{Gradient descent formula}$$

Non-linear Regression:

1. Introduction:

- We can introduce non-linearity by adding/using a basis function.

2. Basis Function Regression:

- In basis function regression:

$$f(x) = \sum_{i=1}^{N} w_i b_i(x)$$

E.g.

Linear Model: $b_0(x) = 1, \quad b_1(x) = x$

$$f(x) = w_0 b_0(x) + w_1 b_1(x)$$
$$= w_1 x + w_0$$

Polynomial Model: $b_k(x) = x^k$

$$f(x) = \sum_{k=1}^{N} w_k x^k$$

Radial Basis Function (RBF): $b_k(x) = \exp\left(\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$,

μ_k and σ_k are hyperparameters meaning that they are expensive to choose.

$$f(x) = \sum_{k=1}^{N} w_k \exp\left(\frac{(x-\mu_k)^2}{2\sigma_k^2}\right)$$

Furthermore, μ_k is the center of the basis function and σ_k^2 is the width of the basis function.

- The polynomial model and RBF are 2 common choices of basis functions.

- The polynomial model is more susceptive to outliers but can extrapolate while RBF is not as susceptive to noise but can't extrapolate.

  RBF - local fit
  Polynomial - global fit

- In the above basis functions, there are hyperparameters to decide:
  - Polynomial: Degree of polynomial
  - RBF: # of RBFs, μ and σ

  Generally speaking, how you choose the hyperparameters is to do the training and validation and use the loss to decide the best parameters.

  For RBF, we can use the following guidelines
  a) To pick the center:
     1. Place the centers uniformly spaced containing the data. This is simple but can lead to empty regions with basis functions and will have an impractical number of data points in higher dimensional input spaces.

2. Place one center at each data point. This is used more often since it limits the num of centers needed although it can be expensive if the num of data points is too big.

3. Cluster the data and use one center for each cluster.

b) To pick the width:

1. Manually try diff values and pick the best.

2. Use the avg squared dist to neighbouring centers, scaled by a constant. This approach also allows you to use diff widths for diff basis functions and it allows the basis functions to be spaced non-uniformly.

- Directly minimizing squared-error can lead to overfitting. There are 2 important solns to this:
  1. Adding prior knowledge. ← We'll use this for now.
  2. Handling uncertainity.

- In many cases, there is some sort of prior knowledge we can use. A common assumption is that the underlying function is likely to be smooth. We can use regularization. This means we add an extra term, often to encourage smooth models.

- Least Squares/Ordinary Least Squares:

  $L(\omega) = \|\tilde{y} - B\omega\|_2^2$

- Regularized Least Squares:

  $L(\omega) = \underbrace{\|\tilde{y} - B\omega\|_2^2}_{\text{Data term}} + \underbrace{\lambda\|\omega\|_2^2}_{\text{Smoothness term}} \quad (\lambda \in \mathbb{R}^+)$

The smoothness term forces your parameters to be smaller, causing your function to be smoother.

This is also called L2 Regularization or Ridge Regression.

We can also use the L1 term as regularizer. This is called Lasso Regression.

$L(\omega) = \|\vec{y} - B\omega\|_2^2 - \lambda\|\omega\|_1$

Lagrange multiplier variant of a constrained optimizer.