CS 181 Spring 2021 Section 1 Notes: Linear Regression, MLE

1 Least Squares (Linear) Regression

1.1 Takeaways

1.1.1 Linear Regression

The simplest model for regression involves a linear combination of the input variables:

$$h(\mathbf{x}; \mathbf{w}) = w_1 x_1 + w_2 x_2 + \ldots + w_D x_D = \sum_{d=1}^{D} w_d x_d = \mathbf{w}^\top \mathbf{x} \tag{1}$$

where $x_j \in \mathbb{R}$ for $j \in \{1, \ldots, D\}$ are the features, $\mathbf{w} \in \mathbb{R}^D$ is the weight parameter, with $w_1 \in \mathbb{R}$ being the bias parameter. (Recall the trick of letting $x_1 = 1$ to merge bias.)

1.1.2 Least Squares Loss Function

The least squares loss function assuming a basic linear model is given as follows:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \left(y_n - \mathbf{w}^\top \mathbf{x}_n \right)^2 \tag{2}$$

For regularized regression, such as LASSO (L1) or Ridge (L2) regression, we use a different loss function with an added penalty term. The L1 penalty is $\lambda|w|$, and the L2 penalty is $\lambda\|w\|^2$.

1.1.3 Optimizing Weights to Minimize Loss Function

If we minimize the function with respect to the weights, we get the following solution:

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y} = \arg\min_{\mathbf{w}} \mathcal{L}(\mathbf{w}) \tag{3}$$

where $\mathbf{X} \in \mathbb{R}^{N \times D}$. Each row represents one data point and each column represents values of one feature across all the data points. In practice, gradient descent is often used to compute w^*. [1]

[1] Note: $(\mathbf{X}^T \mathbf{X})^{-1}$ is invertible iff X is full column rank (i.e. rank D, which implies $N \geq D$). **What if $(\mathbf{X}^T \mathbf{X})^{-1}$ is not invertible?** Then, there is not a unique solution for w^*. If $d > N$, computing the pseudoinverse of $\mathbf{X}^T \mathbf{X}$ will find one solution. Alternatively, in general applying ridge regression can fix the invertibility issue.

1.2 Concept Question

How is a model (such as linear regression) related to a loss function (such as least squares)?

- The model (of the data) and the loss functions are both important pieces to the ML pipeline, but they are distinct. The model describes how you believe the data is related and/or generated. Very commonly, you will be optimizing over a family of models. The loss function measures how well a specific model (i.e. with specific parameters) fits the data, and it is used in the previously mentioned optimization.
- Least squares and linear regression are often used together, especially since there are theoretical justifications (i.e. MLE connection) to why least squares is a good loss function for linear regression. However, you do **not** have to use them together. Another loss function that could be used with linear regression is an absolute difference (L1) loss.

1.3 Exercise: Practice Minimizing Least Squares

Let $\mathbf{X} \in \mathbb{R}^{N \times D-1}$ be our design matrix, $\mathbf{y}$ our vector of N target values, $\mathbf{w}$ our vector of $D-1$ parameters, and w_0 our bias parameter. The least squares error function of $\mathbf{w}$ and w_0 can be written as follows

$$\mathcal{L}(\mathbf{w}, w_0) = \frac{1}{2} \sum_{n=1}^{N} \left(y_n - w_0 - \sum_{d=1}^{D-1} w_d X_{nd} \right)^2 .$$

Find the value of w_0 that minimizes $\mathcal{L}$. Can you write it in both vector notation and summation notation? Does the result make sense intuitively?

Solution: We minimize by finding gradient w.r.t w_0, setting to 0, and solving.

$$\begin{aligned}
\frac{\partial L}{\partial w_0} &= -\sum_{n=1}^{N} (y_n - w_0 - \sum_{d=1}^{D-1} w_d X_{nd}) = 0 \\
N w_0^* &= \sum_{n=1}^{N} \left(y_n - \sum_{d=1}^{D-1} w_d X_{nd} \right) \\
w_0^* &= \frac{1}{N} \left[\left(\sum_{n=1}^{N} y_n \right) - \sum_{n=1}^{N} \sum_{d=1}^{D-1} w_d X_{nd} \right] \\
&= \frac{1}{N} \left(\mathbf{y}^\top \mathbf{1} - \sum_{n=1}^{N} \mathbf{w}^\top \mathbf{x}_n \right)
\end{aligned}$$

The result makes sense intuitively as it is the average deviation of the outputs from the predictions.

2 Maximum Likelihood Estimation

2.1 Takeaways

- Given a model and observed data, the **maximum likelihood estimate** (of the parameters) is the estimate that maximizes the probability of seeing the observed data under the model.
- It is obtained by maximizing the **likelihood function**, which is the same as the joint pdf of the data, but viewed as a function of the parameters rather than the data.
- Since log is monotonic function, we will often maximize the **log likelihood** rather than the likelihood as it is easier (turns products from independent data into sums) and results in the same solution.

2.2 Exercise: MLE for Gaussian Data

We are given a data set $(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ where each observation is drawn independently from a multivariate Gaussian distribution:

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{|(2\pi)\boldsymbol{\Sigma}|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right) \tag{4}$$

where $\boldsymbol{\mu}$ is a m-dimensional mean vector, $\boldsymbol{\Sigma}$ is a m by m covariance matrix, and $|\boldsymbol{\Sigma}|$ denotes the determinant of $\boldsymbol{\Sigma}$. Find the maximum likelihood value of the mean, $\boldsymbol{\mu}_{MLE}$.

Solution: We find the MLE by maximizing the log likelihood:

$$\begin{aligned} l(\boldsymbol{\mu}, \boldsymbol{\Sigma}; \mathbf{x}) &= \log\left(\prod_{i=1}^{n} \mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma})\right) = \sum_{i=1}^{n} \log(\mathcal{N}(\mathbf{x}_i|\boldsymbol{\mu}, \boldsymbol{\Sigma})) \\ &= -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log(|\boldsymbol{\Sigma}|) - \frac{1}{2}\sum_{i=1}^{n}(\mathbf{x}_i-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i-\boldsymbol{\mu}) \end{aligned}$$

Taking the derivative (matrix cookbook ch 2.4 eqn 78) with respect to $\boldsymbol{\mu}$ and setting it equal to 0, we get

$$0 = \frac{\partial l}{\partial \boldsymbol{\mu}} = \sum_{i=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_i - \boldsymbol{\mu})$$

and solving gives us that

$$\boldsymbol{\mu}_{MLE} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i.$$

3 Linear Basis Function Regression

3.1 Takeaways

We allow $h(\mathbf{x}; \mathbf{w})$ to be a non-linear function of the input vector $\mathbf{x} \in \mathbb{R}^D$, while remaining linear in $\mathbf{w} \in \mathbb{R}^M$ by using a basis function $\phi : \mathbb{R}^D \to \mathbb{R}^M$. The resulting basis regression model is below:

$$h(\mathbf{x}; \mathbf{w}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}) = \mathbf{w}^\top \phi(\mathbf{x}) \tag{5}$$

To merge the bias term, we can define $\phi_1(\mathbf{x}) = 1$. Some examples of basis functions include polynomial $\phi_m(x) = x^m$, Fourier $\phi_m(x) = \cos(m\pi x)$, and Gaussian $\phi_m(x) = \exp\{-\frac{(x-\mu_m)^2}{2s^2}\}$.

3.2 Concept Questions

- What are some advantages and disadvantages to using linear basis function regression to basic linear regression?
- How do we choose the bases?

Basis functions allow us to capture nonlinear relations that may exist in the data, which linear functions can not. There is, however, a greater risk of overfitting with the more flexible linear basis function regression - more on this in the upcoming weeks in lecture with bias-variance tradeoff.

We can choose the bases with expert domain knowledge, or they could even be learned themselves... (foreshadowing neural nets).

We don't talk about feature engineering and incorporating expert domain knowledge too much in this class; however, these are vital in real world situations for good performance. The practical assignment and pset problems may touch on this.

3.3 Exercise: HW1 Q4

If extra time, can have students work on this.