CS 217: Artificial Intelligence and Machine Learning **Jan-Apr 2024**

Lecture 4: Regression and Gradient Descent

Lecturer: Swaprava Nath *Scribe(s): SG7, SG8*

4.1 Linear Regression

We have defined our error function for linear regression $E(w, D)$ as $||Xw - y||^2$, this can be thought of as the square of Euclidean distance between our predicted output Xw, and the actual output y.

Theorem 4.1 *The vector function $f(w)$ is a convex function if $\nabla_w^2 f(w)$ (called the* Hessian *matrix of f) is* ***positive semi-definite****, i.e., the eigenvalues of this matrix are non-negative.*

We found the minimum of $E(w, D)$ by equating the gradient of E w.r.t w i.e. $\nabla_w(E) = 0$, and we get the following equation: $(X^TX)w - X^Ty = 0$. Solving for w, $w^* = (X^TX)^{-1}X^Ty$ (w^* is the optimal value of w). But the problem is that we do not know if X^TX is invertible or not. If it is not invertible, we use something known as the *pseudo-inverse* of X^TX.

4.1.1 Geometrical Interpretation

Our equation $(X^TX)w - X^Ty = 0$ can be factored as $X^T(Xw - y) = 0$, let us just focus on $Xw - y$ for now.

$$\begin{bmatrix} x_{11} & x_{12} & \dots & x_{1d} \\ x_{21} & x_{22} & \dots & x_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \dots & x_{nd} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \tag{4.1}$$

The column picture of $Xw = y$ can be represented as a linear combination of the columns of X:

$$w_1 \begin{bmatrix} x_{11} \\ x_{21} \\ \vdots \\ x_{n1} \end{bmatrix} + w_2 \begin{bmatrix} x_{12} \\ x_{22} \\ \vdots \\ x_{n2} \end{bmatrix} + \dots + w_d \begin{bmatrix} x_{1d} \\ x_{2d} \\ \vdots \\ x_{nd} \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

In other words, the above equation is trying to find a solution vector w whose components take a linear combination of the columns of X. This equation has a solution only if the vector y is in the column space of X, which might not be always the case. In case y does not belong to the column space and our objective is to minimize $||Xw - y||^2$, i.e., the Euclidean distance between y and Xw, the geometric solution is to pick that point Xw which lies in the column space of X which is perpendicular to this column space from y. This is the geometric interpretation and also shown in Fig. 4.1.

Case 1: y is in the column space of X

In this case, we can actually just find a w such that $Xw = y$, which makes $||Xw - y||^2 = 0$. And this solution of w will obviously satisfy $X^T(Xw - y) = 0$.

Case 2: y is not in the column space of X

Let's take a simple example where columns of X are $\binom{2}{1}$ and $\binom{4}{2}$, and y is $\binom{2.5}{3}$.

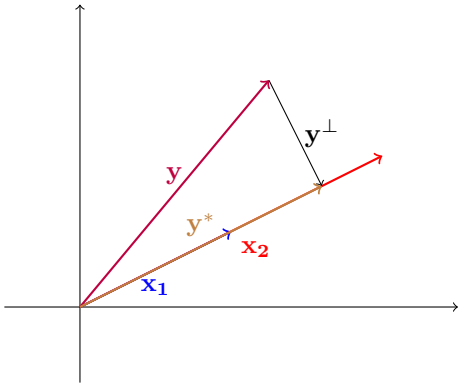

Figure 4.1: Example of case where $Xw = y$ has no solution

Here y isn't in the span of columns of X, so we can't have a solution for $Xw = y$. But in order to minimize magnitude of $Xw - y$ let us project y onto the column space of X, and call this projection y^*. We can now choose w such that $Xw = y^*$, and $Xw - y = y^* - y = y^\perp$. But $y^\perp$ should be a vector which is orthogonal to the column space of X i.e. its dot product with any vector in the column space should be 0. Now observe the expression $X^T(Xw - y) = X^T y^\perp$

$$\mathbf{X^T y^\perp} = \begin{bmatrix} \leftarrow \mathbf{x_1} \rightarrow \\ \leftarrow \mathbf{x_2} \rightarrow \\ \vdots \\ \leftarrow \mathbf{x_n} \rightarrow \end{bmatrix} \mathbf{y^\perp} = \begin{bmatrix} \mathbf{x_1} \cdot \mathbf{y^\perp} \\ \mathbf{x_2} \cdot \mathbf{y^\perp} \\ \vdots \\ \mathbf{x_n} \cdot \mathbf{y^\perp} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

So, the w which satisfies $Xw = y^*$ is a solution of $X^T(Xw - y) = 0$. Thus we always have an optimal value for w.

4.2 Regression model with basis functions

Consider a simple linear regression model:

$$y_i = w_0 + w_1 x_i + \epsilon_i, \tag{4.2}$$

where y_i is the dependent variable, x_i is the independent variable, w_0 is the intercept, w_1 is the slope, and ϵ_i is the error term. When we have data points as described in Fig4.2 which fit into a curve i.e., when we have higher powers of x_i's the expression for $\hat{y_i}$ we deal with this using basis functions.

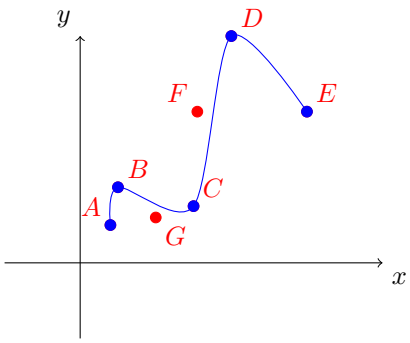

Figure 4.2: Non Linear data

To incorporate basis functions, we can extend the model as follows:

$$y_i = w_0 + w_1\phi_1(x_i) + w_2\phi_2(x_i) + \ldots + w_k\phi_k(x_i) + \epsilon_i, \tag{4.3}$$

where $\phi_1(x_i), \phi_2(x_i), \ldots, \phi_k(x_i)$ are basis functions applied to the input variable x_i.

This is now in the form of an m-dimensional regression model which can be done similarly to a d-dimensional linear regression model as follows,

$$\hat{y_i} = \sum_{j=0}^{m} w_j \cdot \phi_j(x_i) \qquad (m >> d) \tag{4.4}$$

For higher dimensional x_is we usually consider the basis functions to be norms of respective powers.

$$\Phi = \begin{bmatrix} \phi_0(x_1) & \phi_1(x_1) & \ldots & \phi_m(x_1) \\ \phi_0(x_2) & \phi_1(x_2) & \ldots & \phi_m(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(x_n) & \phi_1(x_n) & \ldots & \phi_m(x_n) \end{bmatrix}_{n\times(m+1)}, w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_m \end{bmatrix}_{(m+1)\times 1} \text{and } y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}_{n\times 1}$$

The error function E here will be, $\|\Phi \cdot w - y\|^2$ and we are to minimize this function with respect to w.

According to our previous results, the optimum solution would be

$$w^* = (\Phi^T\Phi)^{-1}\Phi^T y.$$

This is non-linear regression but the analysis is similar to linear regression.

4.2.1 Common Basis Functions

Some standard basis functions include polynomial basis functions, Gaussian basis functions, and piecewise linear basis functions. Each basis function introduces a different form of nonlinearity into the model.

Polynomial Basis Functions

Polynomial basis functions involve raising the input variable to different powers:

$$\phi_j(x) = x^j. \tag{4.5}$$

Gaussian Basis Functions

Gaussian basis functions are based on Gaussian distributions:

$$\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{2\sigma^2}\right), \tag{4.6}$$

where μ_j is the mean and σ is the standard deviation.

Piecewise Linear Basis Functions

Piecewise linear basis functions create linear segments within specified intervals:

$$\phi_j(x) = \begin{cases} 0 & \text{if } x < a_j \\ (x - a_j) & \text{if } a_j \leq x < b_j \\ 0 & \text{if } x \geq b_j \end{cases}. \tag{4.7}$$

4.3 Probabilistic model of Linear Regression

We need to know that almost no dataset exists without noise. To get a better approximation of the Dataset without the noise, even when the noise is present, we use the **Probabilistic Model of Linear Regression**. A probabilistic model of linear regression on the dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ can be given by:

$$y_i = w^T x_i + \epsilon_i.$$

Here, w is the parameter of the noisy linear model with noise ϵ_i where ϵ_i follows a standard normal distribution $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ with ϵ_i being independent and identically distributed (*i.i.d.*) and hence $cov(\epsilon_i, \epsilon_j) = 0, \forall i \neq j$.

Our goal here is to estimate the parameter of linear regression. Hence, we need to estimate w. We use a method called **Maximum Likelihood Estimation (MLE)**.

4.4 Maximum Likelihood Estimation (MLE)

A set of i.i.d. observations $\{y_1, y_2, \ldots, y_n\}$ are generated by a probabilistic model parametrized by θ and is represented as:

$$y_i \sim \mathcal{P}(y|\theta) \text{ or } \mathcal{P}(y;\theta)$$

where $\mathcal{P}(y|\theta)$ is called the **Likelihood function**, or simply the **Likelihood**, and our goal is to maximize this.

But, instead, something called the **Log Likelihood**, which is *log* of the *Likelihood*, is maximized, which will therefore result in the maximization of $\mathcal{P}(y|\theta)$, the *Likelihood*, as the *log* function is increasing.

The motivation behind maximizing the *Log Likelihood* instead of the *Likelihood* is that:

1. It is mathematically nicer and easy to deal with the *logarithmic* mathematics as the Guassians have cumbersome exponents.
2. The *log* function increases monotonically.

3. The most important one, *log* functions, transforms a product to a sum, which has a numerical advantage. For instance, the standard *Likelihood* $\prod_{i=1}^{n} \mathcal{P}(y_i|x_i, w)$ for large numbers evaluates to a very small number (negligibly smaller) and a finite precision computer may not recognize that and treats it as 0. But with the *log likelihood*, mathematically it becomes $\sum_{i=1}^{n} \log \mathcal{P}(y_i|x_i, w))$ where many terms add up to give a significantly reasonable number.

The *Log Likelihood* function is given by:

$$\mathcal{L}(\theta) = \sum_{i=1}^{n} \log \mathcal{P}(y_i|\theta)$$

and the Maximum Likelihood Estimator is given by:

$$\theta_{MLE} = \arg\max_{\theta} \mathcal{L}(\theta) = \arg\max_{\theta} \sum_{j=1}^{n} \log \mathcal{P}(y_i|\theta)$$

4.4.1 Example - Coin Toss

Tossing a coin n times, each is a binary RV with Bernoulli random distribution with parameter θ.

$$\mathbb{P}(y_j|\theta) = \theta^{y_j}(1-\theta)^{1-y_j}$$

$$\begin{aligned} L(\theta) = \log \mathbb{P}(y|\theta) = \log\left(\prod_{j=1}^{n} \mathbb{P}(y_j|\theta)\right) = \sum_{j=1}^{n} log(\mathbb{P}(y_j|\theta) \\ L(\theta) = \sum_{j=1}^{n} y_j \log\theta + (1-y_j)\log(1-\theta) \end{aligned} \tag{4.8}$$

Differentiating w.r.t θ and setting it to 0 we get

$$\begin{aligned} L'(\theta) = \sum_{j=1}^{n} y_i \times \frac{1}{\theta} + \sum_{j=1}^{n}(1-y_j)\frac{-1}{1-\theta} = 0 \\ (1-\theta)\sum_{j=1}^{n} y_j = \theta \sum_{j=1}^{n}(1-y_j) \\ \sum_{j=1}^{n} y_j - \theta\sum_{j=1}^{n} y_j = n\theta - \theta\sum_{j=1}^{n} y_j \\ \theta = \frac{\sum_{j=1}^{n} y_j}{n} \end{aligned}$$

4.4.2 MLE for Regression

$$
\begin{aligned}
y_i &= w^T x_i + \epsilon_i \sim \mathcal{N}(0, \sigma^2) \\
y_i &\sim \mathcal{N}(w^T x_i, \sigma^2) \\
\mathbb{P}(y_1 | x_i, w) &= \frac{1}{\sqrt{2\pi\sigma^2}} \exp \frac{-(y_i - w^T x_i)^2}{2\sigma^2} \\
L(w) &= \text{constant} - \sum_{i=1}^{n} \frac{(y_i - w^T x_i)^2}{2\sigma^2}
\end{aligned} \tag{4.9}
$$

The w for which the above log-likelihood is maximized is given by

$$
\begin{aligned}
w^* &= \arg\max_w L(w) = \arg\min_w \sum_{i=1}^{n} (y_i - w^T x_i)^2 \\
w^* &= \arg\min_w ||Xw - y||^2
\end{aligned}
$$

As we saw in the previous lecture, if $X^T X$ is non-singular, this gives

$$w^* = (X^T X)^{-1} X^T Y$$

4.5 Gradient Descent

Gradient descent is an optimization algorithm that efficiently searches for the optimal parameter values.

Note: Gradient Descent will only be applicable when the error function E is Differentiable.

Initialize the parameter matrix w with a random matrix w_0 or a zero matrix.

$$w \leftarrow w_0$$

Choose a learning rate (η), which determines the step size in the parameter space. Avoid taking very large or very small values for η, for taking a very large value of η will result in divergence rather than convergence or may also result in oscillations about the minima. Taking a small value of η will result in an increased number of steps, thus making the convergence slower.

Calculate the gradient of the error function $\nabla_w E$ using the recently assumed value of w.

Update the parameter matrix w, inputting the gradient computed using the previous value of the w. This step adjusts the parameters in the direction that reduces the Error function E.

$$w \leftarrow w - \eta \cdot \nabla_w E$$

Update the value of the $\nabla_w E$ using the new w and again update the value of the w using the new $\nabla_w E$ and the previous w.
Repeat this process until the $\nabla_w E$ becomes insignificant or in other words the w converges.