Lecture 2: Linear Regression

Adityanarayanan Radhakrishnan

Edited by: Max Ruiz Luyten, George Stefanakis, Cathy Cai

January 21, 2022

1 Introduction

We will begin this course with a review of *linear regression*, i.e. finding a line of best fit to training data. Although studied for hundreds of years, linear regression remarkably continues to serve as a simple model that provides invaluable intuition into modern machine learning phenomena. For example, one can provably establish the generalization benefits of over-parameterization given by the double descent curve in linear models [1, 2, 3, 5]. Importantly, our analysis of linear regression will provide us tools for developing nonlinear regression methods, namely kernel regression.

2 Preliminaries

Suppose we are given a data distribution $(x, y) \sim \mathbb{P}_{(x,y)}$ where $x \in \mathbb{R}^d$ is a feature vector and $y \in \mathbb{R}$ is the corresponding label. Assume we are given n independent, identically distributed (i.i.d.) training pairs $\{x^{(i)}, y^{(i)}\}_{i=1}^n \subset \mathbb{R}^d \times \mathbb{R}$. Our goal is to learn a function $\hat{f}$ in a class of functions $\mathcal{F} = \{f : \mathbb{R}^d \to \mathbb{R}\}$ that is able to "predict well" on unseen samples, i.e. $\hat{f}(x) \approx y$ for $x \neq x^{(i)}$ for all $i \in [n]$. We give concrete examples of problems in this setting below.

Example 1 (Housing Price Prediction)**.** *Real estate companies, such as Zillow, provide services that present price estimates (Zestimates) for houses. Such software is easily framed in the context above where y is the price of a house and the entries of x contain information relevant to predicting the price of a house, e.g. number of bedrooms, number of bathrooms, location, etc.:*

$$x = \begin{bmatrix} \#\ bedrooms \\ \#\ bathrooms \\ square footage \\ \vdots \end{bmatrix} \mapsto y = house\ price$$

Our training data would be a set of feature vectors for houses that were sold, $\{x^{(i)}\}$, along with the price for which they were sold, $\{y^{(i)}\}$. Our goal would then be to use this historical data to build a model that can accurately estimate the price of a new house.

Example 2 (Image Classification)**.** *Another prominent example of such a prediction problem involves classifying what kind of object is in a given image (e.g. does the image contain a dog or cat?). In this case, the entries of x are pixel values in an image and y is simply a label encoding the type of object in the image (e.g. label 1 for dogs, label -1 for cats):*

$$x = \begin{bmatrix} pixel\ 1\ value \\ pixel\ 2\ value \\ \vdots \end{bmatrix} \mapsto y = \begin{cases} 1 & \text{if a dog is in the image} \\ -1 & \text{if a cat is in the image} \end{cases}$$

Our training data in this case would be a labelled set of images (e.g. ImageNet [7] or CIFAR10 [6]), and our goal would be to use these labelled images to build a model that could accurately predict whether an object is in an image.

After framing the general problem above, the following need to be formalized:

1. How do we measure model performance, i.e. how do we assess $\hat{f}(x) \approx y$?
2. What class of functions $\mathcal{F}$ do we want to use?
3. How do we select a function $\hat{f} \in \mathcal{F}$?

These application dependent questions are of generalization and optimization and lie at the heart of machine learning. In this lecture, we will select simple answers to these questions, leading to the linear regression framework.

3 Linear Regression

To address question 1 above, we will utilize the squared Euclidean distance from $\hat{f}(x)$ and y, which is typically referred to as mean squared error:

Definition 1 (MSE)**.** *Given $y, \tilde{y} \in \mathbb{R}^p$, the mean squared error (MSE) between $y, \tilde{y}$ is $\mathcal{L}(y, \tilde{y}) = \frac{1}{2}\|y - \tilde{y}\|_2^2$.*

To address question 2, we will at the moment restrict ourselves to the simple class of linear functions, i.e. $\mathcal{F} = \{f : \mathbb{R}^d \to \mathbb{R} \ ; \ f(x) = wx \ , \ w \in \mathbb{R}^{1\times d}\}$.

Lastly, in order to address question 3, we follow an empirical risk minimization (ERM) framework and aim to select $w \in \mathbb{R}^{1\times d}$ such that $wx^{(i)} \approx y^{(i)}$ for all $i \in [n]$. To provide some intuition around this choice, it is reasonable to expect that the training data and test data come from the same distribution, and so, we expect that there should be patterns in training data that can aid in building a model that generalizes to test data. For a more formal treatment of the ERM framework see [8]. We can enforce this formally by minimizing the MSE on the training data. In particular, in order to learn a function $\hat{f}(x) = \hat{w}x$ from data, we find $\hat{w}$ by minimizing:

$$\mathcal{L}(w) = \frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - wx^{(i)})^2$$

After learning $\hat{w}$ by minimizing the loss above, given a new sample x, our prediction is just given by $\hat{w}x$. We illustrate this procedure concretely on the real-world example of using linear regression for housing price estimates below.

Example 3 (Housing Price Prediction via Linear Regression)**.** *Suppose we are given feature vectors containing three features (# of bedrooms, # of bathrooms, square footage) for* 100 *sold houses, $\{x^{(i)}\}_{i=1}^{100} \subset \mathbb{R}^3$, along with the prices they were sold at, $\{y^{(i)}\}_{i=1}^{100} \subset \mathbb{R}$. We first minimize the following training loss in order to construct a predictor, $\hat{w} \in \mathbb{R}^{1\times 3}$:*

$$\mathcal{L}(w) = \frac{1}{2}\sum_{i=1}^{100}(y^{(i)} - wx^{(i)})^2 = \frac{1}{2}\sum_{i=1}^{100}(y^{(i)} - w_1x_1^{(i)} - w_2x_2^{(i)} - w_3x_3^{(i)})^2$$

Given the features for a new house, x, our prediction is then given by $wx = w_1x_1 + w_2x_2 + w_3x_3$.

Now that we have the linear regression framework set up, all that remains is to provide an algorithm to minimize the MSE, $\mathcal{L}(w)$. In this case, we will make the simple choice of selecting gradient descent as our optimization algorithm.

Definition 2 (Gradient Descent)**.** *Given a loss function $\mathcal{L}(w) : \mathbb{R}^d \to \mathbb{R}$, an initial value $w^{(0)} \in \mathbb{R}^d$, and a learning rate (a.k.a step size) $\eta \in \mathbb{R}$, **gradient descent** is used to minimize the loss $\mathcal{L}(w)$ by iteratively computing*

$$w^{(t+1)} = w^{(t)} - \eta\nabla_w\mathcal{L}(w^{(t)})$$

for $t \in \mathbb{Z}_+$.

An important aspect of the linear regression setting, is that as $t \to \infty$, we can explicitly identify both the largest possible learning rate for which the recurrence above converges and the corresponding limit point. This well-known result [4] is presented in the theorem below for the case when $w^{(0)} = \mathbf{0}$, and the general form appears in homework 2.

Theorem 1. *Let* $\{x^{(i)}, y^{(i)}\}_{i=1}^n \subset \mathbb{R}^d \times \mathbb{R}$ *denote training samples and labels. Let* $X = [x^{(1)}|x^{(2)}|\dots|x^{(n)}]$ *and* $y = [y^{(1)}, y^{(2)}, \dots y^{(n)}]$. *Let* σ_1 *be the top singular value of* X. *Given initialization* $w^{(0)} = \mathbf{0}$, *gradient descent used to minimize:*

$$\mathcal{L}(w) = \frac{1}{2}\sum_{i=1}^{n}(y^{(i)} - wx^{(i)})^2$$

converges to the global minimum $w^{(\infty)} = yX^\dagger$, *where* $X^\dagger$ *is the Moore-Penrose pseudoinverse of* X, *iff the learning rate* η *satisfies* $0 < \eta < \frac{2}{\sigma_1^2}$.

Proof. We first compute the terms in the gradient descent updates explicitly. Namely, we have:

$$w^{(t+1)} = w^{(t)} + \eta(y - wX)X^T$$

Now, let $S = XX^T$ and $S' = yX^T$. Then, $w^{(t+1)} = w^{(t)}(I - \eta S) + \eta S'$. Next, we directly solve the recurrence relation. Namely,

$$w^{(t)} = \eta S' \left[(I - \eta S)^{t-1} + (I - \eta S)^{t-2} + ... + (I - \eta S)^1 + I\right]$$

Let $X = U\Sigma V^T$ denote the singular value decomposition of X where $\{\sigma_1, \dots, \sigma_r\}$ are the non-zero singular values of Σ and r is the rank of X. Then, $S = U\Sigma^2 U^T$, and $S' = yV\Sigma U^T$. Thus, we can simplify the recurrence relation:

$$w^{(t)} = \eta S' U \left[(I - \eta\Sigma^2)^{t-1} + (I - \eta\Sigma^2)^{t-2} + ... + (I - \eta\Sigma^2)^1 + I\right] U^T$$

As the diagonal entries of $(I - \eta\Sigma^2)^{t-1} + (I - \eta\Sigma^2)^{t-2} + ... + (I - \eta\Sigma^2)^1 + I$ form geometric series, for $\eta < \frac{2}{\sigma_1^2}$, we have:

$$w^{(t)} = \eta S' U \Sigma^+ U^T$$

$$\Sigma^+ = \begin{bmatrix} \frac{1-(1-\eta\sigma_1^2)^t}{\eta\sigma_1^2} & 0 & \dots 0 & \\ 0 & \frac{1-(1-\eta\sigma_2^2)^t}{\eta\sigma_2^2} & \dots 0 & \mathbf{0}_{r\times d-r} \\ 0 & \dots & \frac{1-(1-\eta\sigma_r^2)^t}{\eta\sigma_r^2} & \\ & \mathbf{0}_{d-r\times r} & & t\mathbf{I}_{d-r\times d-r} \end{bmatrix}$$

Now substituting in $S' = yV\Sigma U^T$ gives us:

$$w^{(t)} = yV\Sigma^\dagger U^T$$

$$\Sigma^\dagger = \begin{bmatrix} \frac{1-(1-\eta\sigma_1^2)^t}{\sigma_1} & 0 & \dots 0 & \\ 0 & \frac{1-(1-\eta\sigma_2^2)^t}{\sigma_2} & \dots 0 & \mathbf{0}_{r\times d-r} \\ 0 & \dots & \frac{1-(1-\eta\sigma_r^2)^t}{\sigma_r} & \\ & \mathbf{0}_{d-r\times r} & & \mathbf{0}_{d-r\times d-r} \end{bmatrix}$$

Lastly, we can take the limit as $t \to \infty$ to conclude:

$$w^{(\infty)} = \lim_{t\to\infty} w^{(t)} = yV\Sigma^\dagger U^T \text{ where } \Sigma^\dagger = \begin{bmatrix} \frac{1}{\sigma_1} & 0 & \dots 0 & \\ 0 & \frac{1}{\sigma_2} & \dots 0 & \mathbf{0}_{r\times d-r} \\ 0 & \dots & \frac{1}{\sigma_r} & \\ & \mathbf{0}_{d-r\times r} & & \mathbf{0}_{d-r\times d-r} \end{bmatrix}$$

We lastly check that $X^\dagger = V\Sigma^\dagger U^T$ satisfies the following 4 properties of a Moore-Penrose inverse for a real matrix X:

$$\begin{aligned} &(1)\ \ XX^\dagger X = X \\ &(2)\ \ X^\dagger XX^\dagger = X^\dagger \\ &(3)\ \ (XX^\dagger)^T = XX^\dagger \\ &(4)\ \ (X^\dagger X)^T = X^\dagger X \end{aligned}$$

□

We note the following important point regarding the analysis in Theorem 1. When $w^{(0)} = \mathbf{0}$, note that $w^{(t)}$ for any $t \in \mathbb{Z}_+$ is given as a linear combination of training examples. In particular, from the proof of Theorem 1, we have for $\{\alpha_t^{(i)}\}_{i=1}^n \subset \mathbb{R}$ and all $t \in \mathbb{Z}_+$:

$$w^{(t)} = \sum_{i=1}^n \alpha_t^{(i)} {x^{(i)}}^T$$

This result implies that the output of the trained predictor is always a linear combination of training examples and thus lies in the span of the training data. This will be an extremely useful property for solving kernel regression, where it will appear as the Representer theorem.

Now that we have a closed form for the solution to linear regression, we can understand our solution in the context of sufficiently parameterized ($n = d$), under-parameterized ($n > d$), and over-parameterized ($n < d$) regimes.

Sufficiently Parameterized Regime ($n = d$)

In the case when $n = d$ (and assuming the rank of X is d), we know that there is exactly one solution to linear regression from linear algebra. In particular, we can achieve zero training loss by solving the following system of equations:

$$wX = y \implies w = yX^{-1}$$

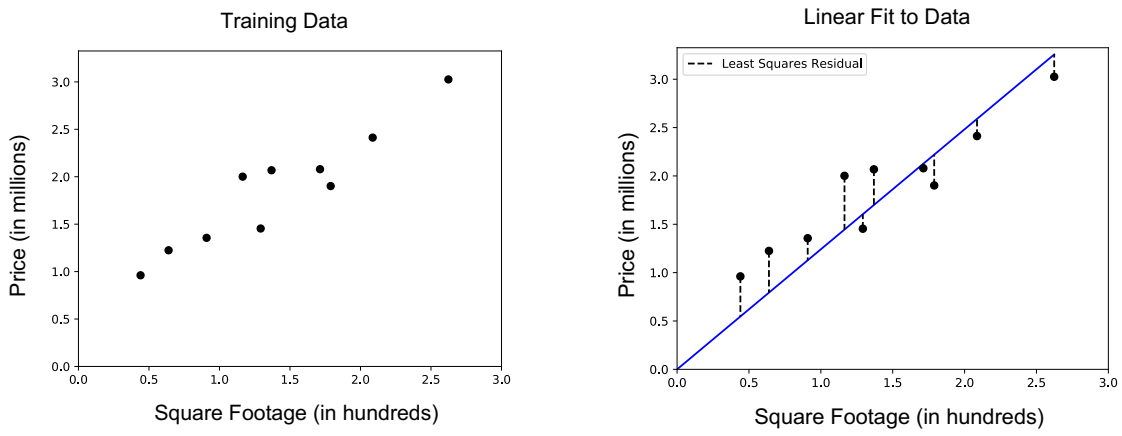
Naturally, when X is invertible, $X^\dagger = X^{-1}$ and so gradient descent gives the expected solution.

Under-parameterized Regime ($n > d$)

In the case when $n > d$ (and assuming the rank of X is d), we know that there is no interpolating solution (no solution with zero training loss) to linear regression from linear algebra. Instead, a solution that minimizes the MSE is found by setting the gradient of the MSE equal to zero:

$$\nabla_w \mathcal{L}(w) = 0 \implies (y - wX)X^T = 0 \implies w = yX^T(XX^T)^{-1}$$

Here, XX^T is invertible since $n > d$ and the rank of X is d. However, by substituting $X = U\Sigma V^T$ given by SVD, we see that $yX^T(XX^T)^{-1} = yX^\dagger$.



(a) (b)

Figure 1: A visualization of the solution learned when using linear regression to fit training data. We consider the example of predicting price of a house (in millions of dollars) given only the square footage (in hundreds of sq.ft.). (a) A visualization of the data that we want to fit. (b) A visualization in blue of the line of best fit given by minimizing the mean squared error to the training data. The residuals from the data to the line of best fit are shown as dashed black lines.

Over-parameterized Regime ($n < d$)

In the case when $n < d$, we know that there are infinitely many interpolating solutions to linear regression from linear algebra. We showed that the solution selected by gradient descent from zero initialization is the Moore-Penrose solution. This solution is the minimum ℓ_2 norm solution, as will be proved in homework 2.

4 An Empirical Demonstration

We now present a simple of example of performing linear regression to find a line of best fit to data. In order to easily visualize the solution given by linear regression, we will consider a simple setting in which we have only 1 feature, i.e. $x \in \mathbb{R}$.

Example 4 (Housing Price Prediction from Square Footage)**.** *Suppose we are given the square footage of* 10 *houses/apartments in New York City, along with their corresponding price, as shown in Figure 1a. In Figure 1b, we visualize the linear regression solution given by Theorem 1. We see that the line of best fit minimizes the MSE to the training data, and the corresponding residuals to the data points are shown as black dashed lines. Note that since we are given more samples (*10*) than the dimension (*1*) and since the samples are not collinear, we are in the under-parameterized regime and cannot achieve zero training MSE.*

5 Discussion

Thus far, we have presented the linear regression framework and analyzed the closed form of the solution to linear regression given by gradient descent. While effective as a simple baseline in many applications, most machine learning problems require learning a non-linear mapping from features to labels.

Fortunately, there is a simple method for extending the linear regression framework to perform nonlinear regression. Namely, instead of using the features $x \in \mathbb{R}^d$ directly, we can map the features x to a higher dimensional space by combining entries of x. For example, we can map $x \in \mathbb{R}^d$ to $\tilde{x} \in \mathbb{R}^{d+1}$ by concatenating the feature $x_1 x_2$ to x. Then, we can simply perform linear regression on the transformed features $\tilde{x} \in \mathbb{R}^{d+1}$ instead of on the original features x.

However, hand-crafting features is often an involved procedure, and it is unclear what combinations or

functions of features we should use. Instead of hand-crafting finitely many such features, we will turn to mapping the features x into an infinite dimensional space (in particular, a Hilbert space) and then efficiently performing linear regression in this space.

References

[1] P. L. Bartlett, P. M. Long, G. Lugosi, and A. Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

[2] M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[3] M. Belkin, D. Hsu, and J. Xu. Two models of double descent for weak features. *Society for Industrial and Applied Mathematics Journal on Mathematics of Data Science*, 2(4):1167–1180, 2020.

[4] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.

[5] T. Hastie, A. Montanari, S. Rosset, and R. J. Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *arXiv:1903.08560*, 2019.

[6] A. Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 2009.

[7] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F.-F. Li. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 2015.

[8] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, 1998.