Understanding Double Descent In the Case of Simple Linear Regression

CS590 (Theory in ML) Final Project Report

Zion Sheng (NetID: zs144)
Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708
zion.sheng@duke.edu

Abstract

The bias and variance tradeoff used to dominate the model design in the classic machine learning era. The traditional theory believes overly complicated models with low training risk tend to have weak generalizability (a high test risk). However, recent studies have justified the value of overparameterized models since they can achieve both zero training risk and low test risk. The new pattern behind this phenomenon is called "Double Descent", where practitioners and theoreticians find that there is a second descent after the model enters the over-parameterized regime, while the underfitting and overfitting are in the under-parameterized regime. This paper tries to understand and investigate this phenomenon in a simple linear regression setting. Specifically, we fix the model size and vary the sample size to reproduce the double descent curve on some synthesized data based on Nakkiran [2019]. After we successfully reproduce the result, we then train the same linear regression model but with regularization to compare the performances. Finally, we explore the prospects and limitations of "Double Descent" on the application of the P300 speller, which is our research interest outside of theory learning in this course.

Keywords machine learning, double descent, bias-variance tradeoff, P300 BCI

1 Introduction

1.1 Traditional view of bias and variance tradeoff

Bias and variance tradeoff is one of the most important concepts in classic machine learning. This rule regulates model test risks with different model complexity. To formalize these concepts, suppose we are given n training samples $(\mathbf{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, where $\mathbf{x}_i$ is the features for one data point and y_i is the corresponding response. Our learning task is to train a model $f_n : \mathbb{R}^d \to \mathbb{R}$ such that we can accurately predict response y given the features $\mathbf{x}$ of any input data points. In practice, the strategy is that we want to maximize the test accuracy while maintaining the training accuracy at a high level. The test accuracy, or relatively the test risk, indicates how well the model can generalize on unseen data, which is not something we can directly control as the training accuracy. We define l as the function to quantify the loss between predictions and the ground truth. Specifically, during training, we use empirical risk minimization (ERM) or some other derivative algorithm to find a proper model f in some function class $\mathcal{H}$ such that it has the minimal empirical risk (also called training loss) $\frac{1}{n}\sum_{i=1}^{n} l(f(\mathbf{x}_i), y_i)$. Although the training set and testing set are generated on the same distribution $\mathcal{P}$, the concern is that the f derived by ERM may not simultaneously minimize the true risk (also called test risk) $\mathbb{E}[l(\mathbf{x}), y|(\mathbf{x}, y) \sim \mathcal{P}]$. The conventional bias and variance tradeoff believes there are two stages when we look at the change of training risk and test risk: underfitting and overfitting Hastie et al. [2009]. If the model is too simple (i.e., a small model class $\mathcal{H}$ capacity), all participants in $\mathcal{H}$ tend to underfit the training set, resulting in both a low training risk and a low test risk. On the other hand, if the model is overly complicated (i.e., a big model class $\mathcal{H}$ capacity), then models will overfit the training data and generate badly on new samples, so the test risk is still high this time but the training risk is optimized. Generally, the training risk decreases as the model becomes more complex, whereas the test risk reduces first and then bounces back. Why do we have a U-shaped curve for the change of test risk? This is because it can be decomposed into three parts: an increasing bias, a decreasing variance, and a constant irreducible error in Neville and Jensen [2008]. Therefore, between the two stages, there exists a balanced point where the model is neither too simple nor too complicated, leading to the ideal training risk and test risk. Another widespread rule of thumb is that: don't overfit the model to a zero training risk because it will definitely harm its performance during testing.

1.2 Modern viewpoint: Double Descent

However, modern practice has started to extend the previous view on the choice of models. Practitioners find that modern machine learning techniques, such as deep neural networks, can still perform relatively well on testing data even if they have already achieved low or even zero training risk. For these complicated neural networks, the number of weights (parameters) is usually much more than the number of samples so that the model can easily reach zero training risk. We name such phenomena of those heavily overparameterized models as "interpolation", which was first officially proposed by Belkin et al. [2019]. This sparked the beginning of a wave of research in this area, and more interesting findings were revealed. Nakkiran [2019] turns the attention to the performance of the fixed model size with different sample size and derive a similar conclusion. In Nakkiran et al. [2021], they further explore more different training settings and experiment with the most cutting-edge deep neural networks such as the transformer, which overall provides an in-depth and surrounding summarization of the double descent. Zhang et al. [2021] suggested that deep neural networks are competent even for classification tasks with data generated completely randomly. Although our focus is more on the application side, there are still many theoretical works on this topic worthy to be referred to. Works that analyze the double descent in simple settings (e.g., linear regression, random Fourier features) include: Bartlett et al. [2019], Muthukumar et al. [2019], Mitra [2019], Derezinski et al. [2019], Liang and Rakhlin [2020], Mei and Montanari [2022]. For theoretical analysis on double descent in neural networks, the related works we recommended include: d'Ascoli et al. [2020] and Baldassi et al. [2021].

Now, it's time to introduce the main character in the long line of papers we breifly covered above: double descent. We mentioned it several times in the previous paragraph, and grant it the title as the updated version of the traditional bias and variance tradeoff. But what it is? And how does it reconcile with the classic theory? We will elaborate on these questions right now.

Double descent refers to the behavior of the test risk curve when it enters the interpolating regime. Empirical evidence of many model architectures shows that, if the model becomes sufficiently over-parameterized that it starts to interpolate the training data, the U-shaped test risk curve will first surge to the peak at some point called interpolating threshold, and then immediately drop back to a comparatively low level as we keep increasing the model complexity. In other words, a second descent appears after the previous U-shaped curve. At this point, much theoretical analysis has proved that the double descent is not a coincidence, but rather it holds under many different assumptions and learning settings. The mathematical derivations to obtain the expression of loss and variance could last for several pages (such as Hastie et al. [2022]), so we decide to skip this here in our work as our focus is on empirical analysis, but the key reason for a second descent is that the bias and variance are different in the over-parameterize regime, and their sum (roughly the test risk) is decreasing as the model complexity increases.

1.3 Our work

Due to the limited amount of time and computation resources, it's impossible for us to explore the double descent in the deep nerual network setting. Therefore, we choose to use simple linear regression where double descent also holds. Note that there is a big difference in the problem formalization in the context of linear regression. In deep neural networks, we can simply make the model more complex by deepening and broadening the networks to investigate the change of performance while the amount of data is fixed. However, we can't do this in linear regression because the input dimension equals to the number of model parameters. In this case, adding more parameters will also increase the sample size, leading to an increment of available information to the model. This violates the requirement of variable controls. As an alternative method, we can instead fix the model size (d) while changing the number of samples (n). The result should be same as long as the ratio n/d is changed, so either tuning d or n shouldn't be a problem.

The organization of this paper can be listed as follows: In Section 2, we start by setting up the linear regression problem, including the assumptions, variables naming, and model definition. Then, we will derive the closed-form solution of the weight estimation, and use the results to further derive the analytical expression of bias and variance in different regimes. In Section 3, we will explain how we conducted the experiment. The visualization results and result analysis are included in Section 4. Finally in Section 5, we investigate the possible way to apply the double descent to the development of P300 spellers, which is essentially training a classifier to classify different brain waves. This section is ended by a brief conclusion of the whole report.

2 Theoretical Analysis

2.1 Problem Setup

Recall that the pattern we are trying to learn is $f : \mathbb{R}^d \to \mathbb{R}$. We define the features as x, and the response as y. The training set and testing set are all both generated from the same distribution $\mathcal{P}$

where the covariates of features $\mathbf{x}$ is isotropic, i.e., $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I}_d)$, where $\mathbf{I}_d$ is the identity matrix of dimension d. The true model is defined as:

$$y = \mathbf{x}^\intercal \beta + \eta, \text{ where } \eta \in \mathcal{N}(0, \sigma^2) \tag{1}$$

β is the true weights, and η is the noise term. To simplify the analysis, we require $||\beta||_2 = 1$. Suppose we have n samples $(\mathbf{x}_i, y_i)$, and we want to train a linear model $\hat{f} = \mathbf{x}^\intercal \hat{\beta}$. $\hat{\beta}$ is the learnable parameter here, and we want to find a $\hat{\beta}$ with the minimum square error on the test set. Since the features matrix is isotropic, test risk $R(\hat{\beta})$ can be expressed as:

$$R(\hat{\beta}) = \mathbb{E}[(\mathbf{X}\hat{\beta} - \mathbf{y})^2 | (\mathbf{X}, \mathbf{y}) \sim \mathcal{P}] = ||\beta - \hat{\beta}||^2 + \sigma^2 \tag{2}$$

, where $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]^\intercal$ and $\mathbf{y} = [y_1, y_2, ..., y_n]^\intercal$. We can't directly minimize the test risk. Instead, we need to train the model and hope that it can be nicely generalized on test data. To do this, we use vanilla gradient descent as the optimization algorithm to find the model with the lowest training risk. In other words, the learning objective can be written as $\min_{\hat{\beta}} ||\mathbf{X}\hat{\beta} - \mathbf{y}||^2$. The learning task here has the closed-form solution $\hat{\beta} = \mathbf{X}^\dagger \mathbf{y}$, where $\mathbf{X}^\dagger$ is the Moore-Penrose pseudoinverse. Interestingly, $\mathbf{X}^\dagger$ has different expressions on the different n/d ratio. For $n \leq d$, we call it the under-parameterized regime. Otherwise, it's over-parameterized regime. Recall that we can always derive the unique solution for the simple linear regression model when $n > d$. But when $n > d$ (interpolation), there could be multiple available solutions, so we will choose the one with the smallest $\mathcal{L}_2$ norm, i.e., $||\hat{\beta}||^2$. Therefore, the solution $\hat{\beta}$ can be expressed as:

$$\hat{\beta} = \mathbf{X}^\dagger \mathbf{y} = \begin{cases} \arg\min\limits_{\hat{\beta}} ||\hat{\beta}||^2 \text{ s.t. } \mathbf{X}\hat{\hat{\beta}} = \mathbf{y} & \text{when } n < d \\ \arg\min\limits_{\hat{\beta}} ||\mathbf{X}\hat{\beta} - \mathbf{y}||^2 & \text{when } n \geq d \end{cases} \tag{3}$$

At this point, we can already get some intuition to foresee a peak when $n = d$. Remember the samples are noisy due to the noise term $\eta \in \mathcal{N}(0, \sigma^2)$. In the over-parameterized regime, we have multiple interpolating models, and we can find the one with the smallest $\mathcal{L}_2$ norm. However, when $n = d$, we only have one interpolating model, and the $\mathcal{L}_2$ norm of this only solution must be high to fit the noise term. As a result, the varying term in the test risk $||\beta - \hat{\beta}||^2$ must be high too. After we give a more comprehensive explanation of the bias and variance term in the next subsection, we believe readers will have a clearer understanding, even if it may seem confusing at this point.

2.2 Analysis on the bias and variance

According to Nakkiran et al. [2021], we can approximately compute the bias, variance, and test risk as follows. For the over-parameterized regime, denote $\gamma = n/d < 1$ as the over-parameterization ratio. The bias B_n and variance V_n can be expressed as:

$$B_n = (1 - \gamma)^2 ||\beta||^2 \tag{4}$$

$$V_n \approx \gamma(1 - \gamma)||\beta||^2 + \sigma^2 \frac{\gamma}{1 - \gamma} \tag{5}$$

The expected test risk can be approximated by the sum of the bias and variance, which is:

$$\mathbb{E}[\bar{R}(\hat{\beta})] \approx (1 - \gamma)||\beta||^2 + \sigma^2 \frac{\gamma}{1 - \gamma} \tag{6}$$

For the under-parameterized regime, set $\gamma = n/d \leq << 1$ as the under-parameterization ratio. The bias B_n and variance V_n can be expressed as:

$$B_n = 0 \tag{7}$$

$$V_n \approx \sigma^2 \frac{1}{\gamma - 1} \tag{8}$$

In this case, the expected test risk is mainly determined by the variance term. Note that the detailed derivation is enclosed in Nakkiran et al. [2021] and Hastie et al. [2022], which are very long and beyond our mathematical ability to fully understand. We assume that the "outsourced" derivations are solidly correct (we will verify by the simulation in Section 3). Having the final expressions above is sufficient for us to generate the theoretical results in the experiment below.

3 Experiment

3.1 Experiment 1

We first want to verify whether the theoretical results are correct. To do this, we fix the dimension d of the linear regression model to 200. We also pre-set the true parameter β to an arbitrary number (the norm must be 1) and fix the standard deviation of the noise term as $\sigma = 0.5$. Then, we select 151 candidates of the sample size ranging from 0 to 600. We select more samples around the $n = 200$ since the change should be more dramatic compared to the other intervals. For each candidate, we run 40 simulations and compute the average performance (test loss and train loss). In each simulation, we will randomly generate a train set and a test set. The randomness here makes the train risk and test risk different each time, but the trend is unveiled after averaging the 40 simulation on each n.

After affirming that the simulation aligns with the theoretical results, we try a regularized linear model with a $\mathcal{L}_2$ regularization term. As a comparison, for each d, we also plot its average train risk and test risk in the same graph of the vanilla linear model.

3.2 Experiment 2

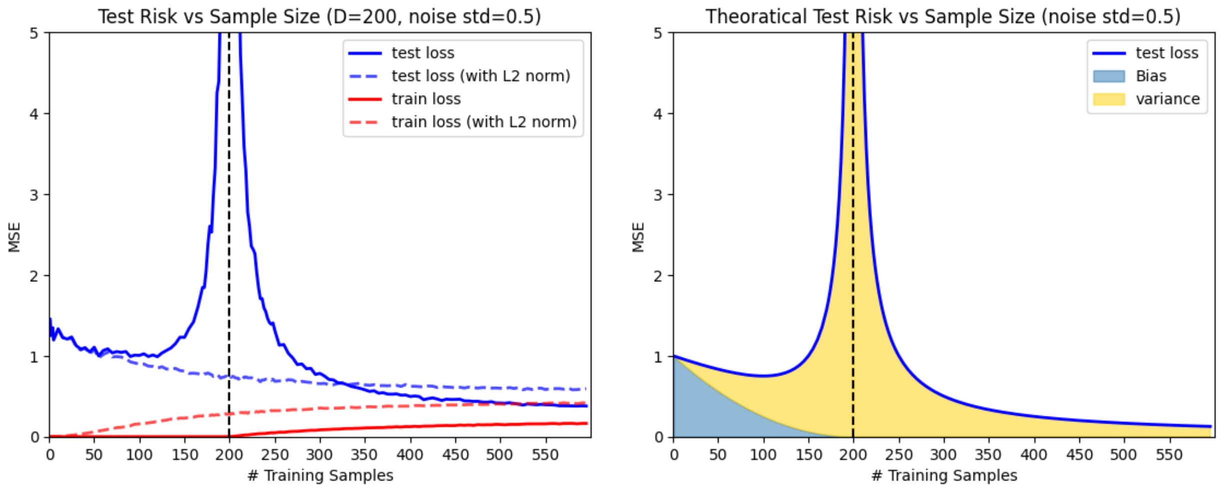
Next, we experiment with different model sizes and noise levels by varying dimension d and the standard deviation of the noise term σ. For d, we choose two different values: 200 and 500. For σ, we pick three different levels: $0.1, 0.5$, and 1.0. Note that we select 172 candidates ranging from 0 to 800 when $d = 500$. Unsurprisingly, we need to repeat the above process on the different combinations of d and n. This time we only visualize the test risk curves. By comparing these curves, we can observe how the double descent looks like in different configurations.

Both experiments are run on a Macbook Pro 2020 with Intel Core i5 CPU. The code was written in Python. We also tested the code on a Macbook Air 2021 with M1 CPU and ended with the same result, so the reproducibility should be guaranteed.

4 Results

Let's first look at the results of experiment 1 shown in Figure 1. The right subplot is the theoretical results we derived by the expression in Section 2.2. When $n > 200$ (the under-parameterized regime), the test risk increases with less training data (the ratio n/d is decreasing). The test risk diverges to ∞ as the ratio approaches 1^+. Then, we will fall into the over-parameterized regime ($n \leq 200$), starting from an infinite test risk. As we keep reducing the number of samples, the test risk soon drops to a local minimum (between $n = 100$ and $n = 150$), and finally, it slowly grows back again and reaches 1 when $n = 0$ (random guess). We also paint the areas representing the bias term and variance term under the test risk curve. This tells us that the non-monotonicity of test risk comes from the non-monotonicity of variance, which decreases, then increases, and then decreases again after the interpolation threshold ($n = d = 200$). Before we proceed to compare the theoretical results with the simulation, we have to emphasize that the second descent (in the context of varying the model size) appears at the left half of the curve here. Because we change the sample size while fixing the model size, the left half is the over-parameterized regime, while the right half is the under-parameterized regime. If we flip the plot horizontally, then it should look similar to those plots whose x-axis measures the model complexity (e.g. Belkin et al. [2019]).

The left subplot displays the results of simulations. The solid blue curve depicts the change of the averaged test risk. If we overlook the tiny fluctuations due to the noise, we can consider it perfectly matches with the theoratical test risk. The training risk curve is also consistent with our understanding as it keeps decreasing when the ratio n/d is decreasing (from right to left on x-axis). After it enters the over-parameterized regimem, those interpolating models are guaranteed to reach zero training risk. Another interesting phenomenon is that regularization can simply fix the risk explosion problem when n around $d = 200$.

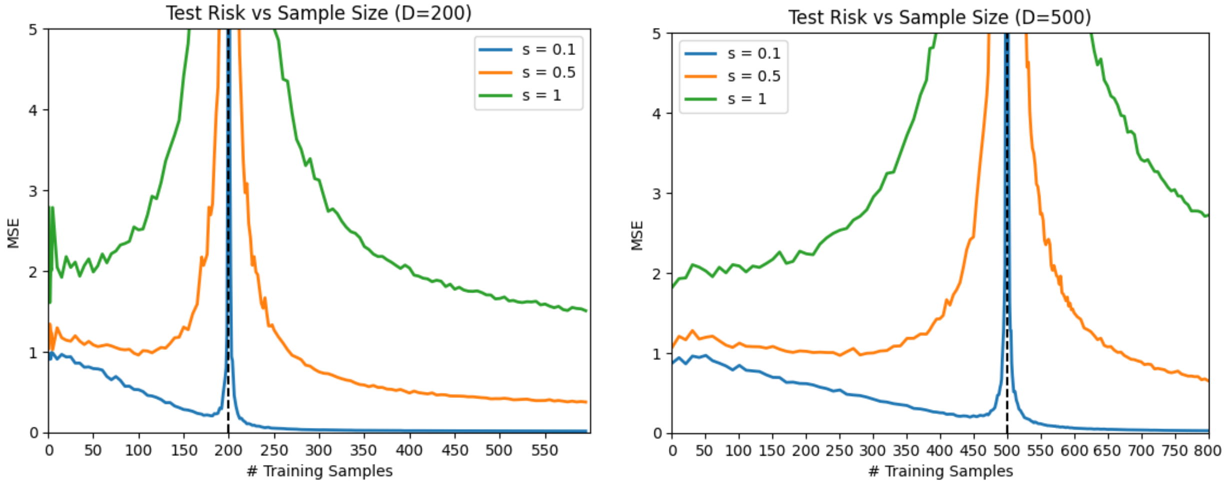


(a) averaged simulation results

(b) theoratical results

Figure 1: Test risk of simulation results (left) and theoratical results (right)

Now let's turn to analyze the results of experiment 2.



(a) d = 200

(b) d = 500

Figure 2: Test risk of simulation results with $d = 200$ (left) and $d = 500$ (right)

5 Discussion and Conclusion

5.1 Thoughts on applying double descent on P300 spellers development

5.2 Conclusion

References

Preetum Nakkiran. More data can hurt for linear regression: Sample-wise double descent. *arXiv preprint arXiv:1912.07242*, 2019.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

Jennifer Neville and David Jensen. A bias/variance decomposition for models using collective inference. *Machine Learning*, 73:87–106, 2008.

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik 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.

Preetum Nakkiran, Gal Kaplun, Yamini Bansal, Tristan Yang, Boaz Barak, and Ilya Sutskever. Deep double descent: Where bigger models and more data hurt. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124003, 2021.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115, 2021.

PeterL. Bartlett, PhilipM. Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *arXiv: Machine Learning,arXiv: Machine Learning*, Jun 2019.

Vidya Muthukumar, Kailas Vodrahalli, Vignesh Subramanian, and Anant Sahai. Harmless interpolation of noisy data in regression. *Cornell University - arXiv,Cornell University - arXiv*, Mar 2019.

ParthaP. Mitra. Understanding overfitting peaks in generalization error: Analytical risk curves for l 2 and l 1 penalized interpolation. *arXiv: Learning,arXiv: Learning*, Jun 2019.

Michal Derezinski, FeynmanT. Liang, and MichaelW. Mahoney. Exact expressions for double descent and implicit regularization via surrogate random design. *Cornell University - arXiv,Cornell University - arXiv*, Dec 2019.

Tengyuan Liang and Alexander Rakhlin. Just interpolate: Kernel "ridgeless" regression can generalize. *The Annals of Statistics*, May 2020. doi:10.1214/19-aos1849. URL `http://dx.doi.org/10.1214/19-aos1849`.

Song Mei and Andrea Montanari. The generalization error of random features regression: Precise asymptotics and the double descent curve. *Communications on Pure and Applied Mathematics*, page 667–766, Mar 2022. doi:10.1002/cpa.22008. URL `http://dx.doi.org/10.1002/cpa.22008`.

Stéphane d'Ascoli, Maria Refinetti, Giulio Biroli, and Florent Krzakala. Double trouble in double descent: Bias and variance (s) in the lazy regime. In *International Conference on Machine Learning*, pages 2280–2290. PMLR, 2020.

Carlo Baldassi, Clarissa Lauditi, Enrico M Malatesta, Gabriele Perugini, and Riccardo Zecchina. Unveiling the structure of wide flat minima in neural networks. *Physical Review Letters*, 127(27): 278301, 2021.

Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *Annals of statistics*, 50(2):949, 2022.