Advanced Machine Learning

Muchang Bahng

Spring 2024

Contents

Machine learning in the 1980s have been focused on developing rigorous theory of learning algorithms, and the field has been dominated by statisticians. They strived to develop the theoretical foundation of algorithms that can be implemented and applied to real-world data. These days, machine learning is more of an engineering discipline than a science. With the advent of deep learning, the theory behind these black box algorithms has slowed down, but their applications have exploded. It is now a field of trying out a bunch of things and sticking to what works. These set of notes are for the former theory, while my deep learning notes are for the latter. It is covered in a separate set of notes since a lot of space is needed to talk about recent developments and architectures (e.g. RCNN, YOLO, LSTMs, Transformers, VAEs, GANs, etc.). We will focus more on establishing the theoretical foundations of most learning algorithms and analyze interpretable algorithms.

I've spent a good amount of time trying to create a map of machine learning, but after rewriting these notes multiple times. I've come to the conclusion that it is impossible to create a nice chronological map of machine learning. Like math, you keep on revisiting the same topics over and over again, but at a higher level, and it's not as simple to organize everything into parametric vs nonparametric[1], supervised vs unsupervised[2], or discriminative vs generative models.[3] Therefore, this is what I recommend.

1. If you are new to machine learning, go over my notes on Stanford CS229, which simply covers basic algorithms and their implementation.
2. Then go over my supervised and unsupervised machine learning notes to get a better grasp of these algorithms, with a bit of theory behind them.
3. Now you can learn the deeper theory of machine learning. This is what these notes are for.

You should know measure (probability) theory, a bit of functional analysis, and some statistics. I will reintroduce all the necessary definitions in a way that is as general as possible, as we move along.

1 Statistical Learning Theory

1.1 Concentration Inequalities

1.2 Minimax Theory

1.3 Empirical Risk Minimization

Unlike unsupervised learning, which comes in many different shapes and forms (anomaly detection, feature extraction, density estimation, dimensionality reduction, etc.), supervised learning comes in a much cleaner format.

1. We start off with a general probability space $(\Omega, \mathcal{F}, \mathbb{P})$. This is our model of the world and everything that we are interested in.
2. A measurable function $X : \Omega \to \mathcal{X}$ extracts a set of features, which we call the **covariates** and induces a probability measure on $\mathcal{X}$, say $\mathbb{P}_X$.
3. Another measurable function $Y : \Omega \to \mathcal{Y}$ extracts another set of features called the **labels** and induces another probability measure on $\mathcal{Y}$, the **label set**, say $\mathbb{P}_Y$.
4. At this point the function $X \times Y$ is all we are interested in, and we throw away Ω since we only care about the distribution over $\mathcal{X} \times \mathcal{Y}$.

[1] K nearest neighbors is a nonparameteric model given that the data is not fixed. When the data is fixed, then our function search space is finite.

[2] There are semi-supervised or weakly supervised models, and models like autoencoders use a supervised algorithm without any labels.

[3] Using Bayes rule, we can always reduce generative models into discriminative models.

5. We model the generation of data from Ω by sampling N samples from $\mathbb{P}_{X\times Y}$, which we assume to be iid (this assumption will be relaxed later). This gives us the **dataset**

$$\mathcal{D} = \{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})\}_{i=1}^N$$

6. We want to learn a prediction rule, or a **hypothesis function** $h : \mathcal{X} \to \mathcal{Y}$, which is searched within a set of functions $\mathcal{H}$, called a **hypothesis class**.

7. We want a measure of how good the hypothesis is. This can be measured with some custom loss function $L : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}_+$ which measures the "loss" of a function $h \in \mathcal{H}$ on the data point (x, y). This results in us being able to define the **error** of our hypothesis to be

$$E(h) \coloneqq \int_{\mathcal{X}\times\mathcal{Y}} L(x, y, h)\, d\mathbb{P}_{X\times Y}$$

The problem is that we don't know the distribution of $\mathbb{P}_{X\times Y}$, and so we must resort to our dataset, which yields the empirical error.

1.4 Decision Theory

How do we choose our loss functions?

2 Low Dimensional Linear Regression

Low dimensional linear regression is what statisticians worked in back in the early days, where data was generally low dimensional.[4] Generally, we had $d < n$, but these days, we are in the regime where $d > n$. For example, in genetic data, you could have a sample of $n = 100$ people but each of them have genetic sequences at $d = 10^6$. When the dimensions become high, the original methods of linear regression tend to break down, which is why I separate low and high dimensional linear regression.

In here, we start with **multiple linear regression**, which assumes that we have several covariates and one response. If we extend this to multiple responses (i.e. a response vector), this is called **multivariate linear regression**. The simple case for one response is called **simple linear regression**, and we will mention some nice formulas and intuition that come out from working with this.

2.1 Construction

Definition 2.1 (Multiple Linear Regression)

Given a dataset $\mathcal{D} = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^N$, where $\mathbf{x}^{(i)} \in \mathbb{R}^d$ and $y^{(i)} \in \mathbb{R}$, the multiple linear regression model is

$$y = \beta^T x + \epsilon \tag{1}$$

with the following assumptions:

1. *Weak exogeneity*: the covariates are observed without error.
2. *Linearity*: the mean of the variate is a linear combination of the parameters and the covariates.
3. *Gaussian errors*: the errors are Gaussian.[a]
4. *Homoscedasticity*: the errors (the observations of Y) have constant variance.
5. *Independence of errors*: The errors are uncorrelated.
6. *No multicollinearity*: more properly, the lack of perfect multicollinearity. Assume that the covariates aren't perfectly correlated.[b]

[4] Quoting Larry Wasserman, even 5 dimensions was considered high and 10 was considered massive.

[a] We can relax this assumption when we get into generalized linear models, and in most cases we assume some closed form of the error for computational convenience, like when computing the maximum likelihood.

[b] This is the assumption that breaks down in high dimensional linear regression.

In order to check multicollinearity, we compute the correlation matrix.

Definition 2.2 (Correlation Matrix)

The correlation matrix of random variables $X_1, \dots, X_d$ is

$$\mathbf{C}_{ij} = \text{Corr}(X_i, X_j) = \frac{\text{Cov}(X_i, X_j)}{\sigma_{X_i}\sigma_{X_j}}$$

given that $\sigma_{X_i}\sigma_{X_j} > 0$. Clearly, the diagonal entries are 1, but if there are entries that are very close to 1, then we have multicollinearity.

Assume that two variables are perfectly correlated. Then, there would be pairs of parameters that are indistinguishable if moved in a certain linear combination. This means that the variance of $\hat{\boldsymbol{\beta}}$ will be very ill conditioned, and you would get a huge standard error in some direction of the β_i's. We can fix this by making sure that the data is not redundant and manually removing them, standardizing the variables, making a change of basis to remove the correlation, or just leaving the model as it is.

If these assumptions don't hold,

1. *Weak exogeneity*: the sensitivity of the model can be tested to the assumption of weak exogeneity by doing bootstrap sampling for the covariates and seeing how the sampling affects the parameter estimates. Covariates measured with error used to be a difficult problem to solve, as they required errors-in-variables models, which have very complicated likelihoods. In addition, there is no universal fitting library to deal with these. But nowadays, with the availability of Markov Chain Monte Carlo (MCMC) estimation through probabilistic programming languages, it is a lot easier to deal with these using Bayesian hierarchical models (or multilevel models, or Bayesian graphical models—these have many names).
2. *Linearity*: the linear regression model only assumes linearity in the parameters, not the covariates. Therefore you could build a regression using non-linear transformations of the covariates, for instance,

$$Y = X_1\beta_1 + X_1^2\beta_2 + \log(X_1)\beta_3 \tag{2}$$

   If you need to further relax the assumption, you are better off using non-linear modelling.
3. *Constant variance*: the simplest fix is to do a variance-stabilising transformation on the data. Assuming a constant coefficient of variation rather than a constant mean could also work. Some estimation libraries (such as the `glm` package in R) allow specifying the variance as a function of the mean.
4. *Independence of errors*: this is dangerous because in the financial world things are usually highly correlated in times of crisis. The most important thing is to understand how risky this assumption is for your setting. If necessary, add a correlation structure to your model, or do a multivariate regression. Both of these require significant resources to estimate parameters, not only in terms of computational power but also in the amount of data required.
5. *No multicollinearity*: If the covariates are correlated, they can still be used in the regression, but numerical problems might occur depending on how the fitting algorithms invert the matrices involved. The t-tests that the regression produces can no longer be trusted. All the covariates must be included regardless of what their significance tests say. A big problem with multicollinearity, however, is overfitting. Depending on how bad the situation is, the parameter values might have huge uncertainties around them, and if you fit the model using new data their values might change significantly.[5] Multicollinearity is a favourite topic of discussion for quant interviewers, and they usually have strong opinions about how it should be handled. The model's intended use will determine how sensitive it is to ignoring the error distribution. In many cases, fitting a line using least-squares estimation is equivalent to assuming errors have a normal distribution. If the real distribution has heavier tails, like

[5] I suggest reading this Wikipedia article on multicollinearity, as it contains useful information: `https://en.wikipedia.org/wiki/Multicollinearity`

the t-distribution, how risky will it make decisions based on your outputs? One way to address this is to use a technique like robust-regression. Another way is to think about the dynamics behind the problem and which distribution would be best suited to model them—as opposed to just fitting a curve through a set of points.

2.2 Least Squares

Given the design matrix $\mathbf{X}$, we can present the linear model in vectorized form:

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},\ \boldsymbol{\epsilon} \sim N(\mathbf{0}, \sigma^2\mathbf{I})$$

The errors can be written as $\boldsymbol{\epsilon} = \mathbf{Y} - \mathbf{X}\boldsymbol{\beta}$, and you have the following total sum of squared errors:

$$S(\boldsymbol{\beta}) = \boldsymbol{\epsilon}^T\boldsymbol{\epsilon} = (\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})^T(\mathbf{Y} - \mathbf{X}\boldsymbol{\beta})$$

We want to find the value of $\boldsymbol{\beta}$ that minimizes the sum of squared errors. In order to do this, remember the following matrix derivative rules when differentiating with respect to vector $\mathbf{x}$.

1. $\mathbf{x}^T\mathbf{A} \mapsto \mathbf{A}$
2. $\mathbf{x}^T\mathbf{A}\mathbf{x} \mapsto 2\mathbf{A}\mathbf{x}$

Now this should be easy.

$$\begin{aligned} S(\boldsymbol{\beta}) &= \mathbf{Y}^T\mathbf{Y} - \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{Y} - \mathbf{Y}^T\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} \\ &= \mathbf{Y}^T\mathbf{Y} - 2\mathbf{Y}^T\mathbf{X}\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} \\ \frac{\partial}{\partial\boldsymbol{\beta}}S(\boldsymbol{\beta}) &= -2\mathbf{X}^T\mathbf{Y} + 2\mathbf{X}^{\mathbf{X}}\boldsymbol{\beta} \end{aligned}$$

and setting it to $\mathbf{0}$ gives

$$2\mathbf{X}^T\mathbf{X}\boldsymbol{\beta} - 2\mathbf{X}^T\mathbf{Y} = 0 \implies \mathbf{X}^T\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^T\mathbf{Y}$$

and the variance of $\boldsymbol{\beta}$, by using the fact that $\text{Var}[\mathbf{A}\mathbf{X}] = \mathbf{A}\,\text{Var}[X]\mathbf{A}^T$, is

$$\text{Var}(\hat{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\,\sigma^2\mathbf{I}\,\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1} = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}(\mathbf{X}'\mathbf{X})(\mathbf{X}'\mathbf{X})^{-1} = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$$

But we don't know the true σ^2, so we estimate it with $\hat{\sigma}^2$ by taking the variance of the residuals. Therefore, we have

$$\begin{aligned} \boldsymbol{\beta} &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y} \in \mathbb{R}^d \\ \text{Var}(\hat{\boldsymbol{\beta}}) &= \hat{\sigma}^2(\mathbf{X}^T\mathbf{X})^{-1} \in \mathbb{R}^{d\times d} \end{aligned}$$

Example 2.1 ()

What happens if you copy your data in OLS? In this case, our MLE estimate becomes

$$\begin{aligned} &\left(\begin{pmatrix}X\\X\end{pmatrix}^T\begin{pmatrix}X\\X\end{pmatrix}\right)^{-1}\begin{pmatrix}X\\X\end{pmatrix}^T\begin{pmatrix}Y\\Y\end{pmatrix} = \\ &\qquad = (X^TX + X^TX)^{-1}(X^TY + X^TY) = (2X^TX)^{-1}2X^TY = \hat{\beta} \end{aligned}$$

and our estimate is unaffected. However, the variance shrinks by a factor of 2 to

$$\frac{\sigma^2}{2}(\mathbf{X}^T\mathbf{X})^{-1}$$

A consequence of that is that confidence intervals will shrink with a factor of $1/\sqrt{2}$. The reason is that we have calculated as if we still had iid data, which is untrue. The pair of doubled values are obviously dependent and have a correlation of 1.

2.3 Likelihood Estimation

Given a dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^N$, our likelihood is

$$L(\theta; \mathcal{D}) = \prod_{i=1}^N p(y^{(i)} \mid x^{(i)}; \theta) = \prod_{i=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(- \frac{(y^{(i)} - \theta^T x^{(i)})^2}{2\sigma^2} \right)$$

We can take its negative log, remove additive constants, and scale accordingly to get

$$\begin{aligned} \ell(\theta) &= -\frac{N}{2} \ln \sigma^2 - \frac{N}{2} \ln(2\pi) + \frac{1}{2\sigma^2} \sum_{i=1}^N \left(y^{(i)} - \boldsymbol{\theta}^T \mathbf{x}^{(i)}\right)^2 \\ &= \frac{1}{2} \sum_{i=1}^N \left(y^{(i)} - \boldsymbol{\theta}^T \mathbf{x}^{(i)}\right)^2 \end{aligned}$$

which then corresponds to minimizing the sum of squares error function. Taking the gradient of this log likelihood w.r.t. θ gives

$$\nabla_\theta \ell(\theta) = \sum_{i=1}^N (y^{(i)} - \theta^T x^{(i)}) x^{(i)}$$

and running gradient descent over a minibatch $M \subset \mathcal{D}$ gives

$$\begin{aligned} \theta &= \theta - \eta \nabla_\theta \ell(\theta) \\ &= \theta - \eta \sum_{(x,y) \in M} (y - \theta^T x) x \end{aligned}$$

This is guaranteed to converge since $\ell(\theta)$, as the sum of convex functions, is also convex.

Note that since we can solve this in closed form, by setting the gradient to 0, we have

$$0 = \sum_{n=1}^N y^{(n)} \boldsymbol{\phi}(\mathbf{x}^{(n)})^T - \mathbf{w}^T \left(\sum_{n=1}^N \boldsymbol{\phi}(\mathbf{x}^{(n)}) \boldsymbol{\phi}(\mathbf{x}^{(n)})^T \right)$$

which is equivalent to solving the least squares equation

$$\mathbf{w}_{ML} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \mathbf{Y}$$

Note that if we write out the bias term out explicitly, we can see that it just accounts for the translation (difference) between the average of the outputs $\bar{y} = \frac{1}{N} \sum_{n=1}^N y_n$ and the average of the basis functions $\bar{\phi_j} = \frac{1}{N} \sum_{n=1}^N \phi_j(\mathbf{x}^{(n)})$.

$$w_0 = \bar{y} - \sum_{j=1}^{M-1} w_j \bar{\phi_j}$$

We can also maximize the log likelihood w.r.t. σ^2, which gives the MLE

$$\sigma^2_{ML} = \frac{1}{N} \sum_{n=1}^N \left(y^{(n)} - \mathbf{w}_{ML}^T \boldsymbol{\phi}(\mathbf{x}^{(n)})\right)^2$$

2.4 Weighted Least Squares

2.5 Simple Linear Regression

The simple linear regression is the special case of the linear regression with only one covariate.

$$y = \alpha + x\beta$$

which is just a straight line fit. Interviewers like this model for its aesthetically pleasing theoretical properties. A few of them are described here, beginning with parameter estimation. For n pairs of (x_i, y_i),

$$y_i = \alpha + \beta x_i + \epsilon_i$$

To minimize the sum of squared errors

$$\sum_i \epsilon_i^2 = \sum_i (y_i - \alpha - \beta x_i)^2$$

Taking the partial derivatives w.r.t. α and β and setting them equal to 0 gives

$$\sum_i (y_i - \hat{\alpha} - \hat{\beta} x_i) = 0$$
$$\sum_i (y_i - \hat{\alpha} - \hat{\beta} x_i) x_i = 0$$

From just the first equation, we can write

$$n\bar{y} = n\hat{\alpha} + n\hat{\beta}\bar{x} \implies y = \hat{\alpha} + \hat{\beta}\bar{x} \implies \hat{\alpha} = \bar{y} - \hat{\beta}\bar{x}$$

The second equation gives

$$\sum_i x_i y_i = \hat{\alpha} n \bar{x} + \hat{\beta} \sum_i x_i^2$$

and substituting what we derived gives

$$\begin{aligned} \sum_i x_i y_i &= (\bar{y} - \hat{\beta}\bar{x}) n\bar{x} + \hat{\beta} \sum_i x_i^2 \\ &= n\bar{x}\bar{y} + \hat{\beta}\left(\left(\sum_i x_i^2 \right) - n\bar{x}^2 \right) \end{aligned}$$

and so we have

$$\hat{\beta} = \frac{\left(\sum_i x_i y_i\right) - n\bar{x}\bar{y}}{\left(\sum x_i^2\right) - n\bar{x}^2} = \frac{\sum_i x_i y_i - \bar{x} y_i}{\sum x_i^2 - \bar{x} x_i} = \frac{\sum_i (x_i - \bar{x}) y_i}{\sum_i (x_i - \bar{x}) x_i}$$

Now we can use the identity

$$\sum_i (x_i - \bar{x})(y_i - \bar{y}) = \sum_i y_i (x_i - \bar{x}) = \sum_i x_i (y_i - \bar{y})$$

to substitute both the numerator and denominator of the equation to

$$\hat{\beta} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} = \frac{\text{cov}(x, y)}{\text{var}(x)} = \rho_{xy} \frac{s_y}{s_x}$$

where ρ_{xy} is the correlation between x and y, and the variance and covariance represent the sample variance and covariance (indicated in lower case letters). Therefore, the correlation coefficient ρ_{xy} is precisely equal to the slope of the best fit line when x and y have been standardized first, i.e. $s_x = s_y = 1$.

Example 2.2 ()

Say that we are fitting Y onto X in a simple regression setting with MLE β_1, and now we wish to fit X onto Y. How will the MLE slope change? We can see that

$$\beta_1 = \rho \frac{s_y}{s_x}, \quad \beta_2 = \rho \frac{s_x}{s_y}$$

and so

$$\beta_2 = \rho^2 \frac{1}{\rho} \frac{s_x}{s_y} = \rho^2 \frac{1}{\beta_1} = \beta_1 \frac{\text{var}(x)}{\text{var}(y)}$$

The reason for this is because regression lines don't necessarily correspond to one-to-one to a casual relationship. Rather, they relate more directly to a conditional probability or best prediction.

The **coefficient of determination** R^2 is a measure tells you how well your line fits the data. When you have your y_i's, their deviation around its mean is captured by the sample variance $s_y^2 = \sum_i (y_i - \bar{y})^2$. When we fit our line, we want the deviation of y_i around our predicted values $\hat{y}_i$, i.e. our sum of squared loss $\sum_i (y_i - \hat{y}_i)^2$, to be lower. Therefore, we can define

$$R^2 = 1 - \frac{\text{MSELoss}}{\text{var}(y)} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

In simple linear regression, we have

$$R^2 = \rho_{yx}^2$$

An R^2 of 0 means that the model does not improve prediction over the mean model and 1 indicates perfect prediction. However, a drawback of R^2 is that it can increase if we add predictors to the regression model, leading to a possible overfitting.

Theorem 2.1 ()

The residual sum of squares (RSS) is equal to the a proportion of the variance of the y_i's.

$$\text{RSS} = \sum (y_i - \hat{y}_i)^2 = (1 - \rho^2) \sum (y_i - \bar{y})^2$$

2.6 Significance Tests

2.6.1 T Test

Given some multilinear regression problem where we must estimate $\boldsymbol{\beta} \in \mathbb{R}^{D+1}$ (D coefficients and 1 bias), we must determine whether there is actually a linear relationship between the x and y variables in our dataset $\mathcal{D}$. Say that we have a sample of N points $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$. Then, for each ensemble of datasets $\mathcal{D}$ that we sample from the distribution $(X \times Y)^N$, we will have some estimator $\boldsymbol{\beta}$ for each of them. This will create a sampling distribution of $\boldsymbol{\beta}$'s where we can construct our significance test on.

So what should our sampling distribution of $\hat{\boldsymbol{\beta}}$ be? It is clearly normal since it is just a transformation of the normally distributed Y: $\hat{\boldsymbol{\beta}} \sim N(\boldsymbol{\beta}, \sigma^2 (X^T X)^{-1})$. Therefore, only considering one element β_i here,

$$\frac{\hat{\beta}_i - \beta_i}{\sigma \sqrt{(X^T X)_{ii}^{-1}}} \sim N(0, 1)$$

But the problem is that we don't know the true σ^2, and we are estimating it with $\hat{\sigma}^2$. If we knew the true σ^2 then this would be a normal, but because of this estimate, our normalizing factor is also random. It turns out that the residual sum of squares (RSS) for a multiple linear regression

$$\sum_i (y_i - x_i^T \beta)^2$$

follows a χ^2_{n-d} distribution. Additionally from the χ^2 distribution of RSS we have

$$\frac{(n-d)\hat{\sigma}^2}{\sigma^2} \sim \chi^2_{n-d}$$

where we define $\hat{\sigma}^2 = \frac{\text{RSS}}{n-d}$ which is an unbiased estimator for σ^2. Now there is a theorem that says that if you divide a $N(0,1)$ distribution by a χ^2_k/k distribution (with k degrees of freedom), then it gives you a t-distribution with the same degrees of freedom. Therefore, we divide

$$\frac{\frac{\hat{\beta}_i - \beta_i}{\sqrt{(X^TX)^{-1}_{ii}}}}{\hat{\sigma}} = \frac{\sigma \sim N(0,1)}{\sigma\chi^2_{n-d}/(n-d)} = \frac{\sim N(0,1)}{\chi^2_{n-d}/(n-d)} = t_{n-d}$$

where the standard error of the distribution is

$$\text{SE}(\hat{\beta}_i) = \sigma_{\hat{\beta}_i} = \sigma\sqrt{(X^TX)^{-1}_{ii}}$$

In ordinary linear regression, we have the null hypothesis $h_0 : \beta_i = 0$ and the alternative $h_a : \beta_i \neq 0$ for a two sided test or $h_a : \beta_i > 0$ for a one sided test. Given a certain significance level, we compute the critical values of the t-distribution at that level and compare it with the test statistic

$$t = \frac{\hat{\beta} - 0}{\text{SE}(\hat{\beta})}$$

Now given our β, how do we find the standard error of it? Well this is just the variance of our estimator $\boldsymbol{\beta}$, which is $\hat{\sigma}^2(\mathbf{X}^T\mathbf{X})^{-1}$, where $\hat{\sigma}^2$ is estimated by taking the variance of the residuals ϵ_i. When there is a single variable, the model reduces to

$$y = \beta_0 + \beta_1 x + \epsilon$$

and

$$\mathbf{X} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}, \qquad \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix}$$

and so

$$(\mathbf{X}'\mathbf{X})^{-1} = \frac{1}{n\sum x_i^2 - (\sum x_i)^2}\begin{pmatrix} \sum x_i^2 & -\sum x_i \\ -\sum x_i & n \end{pmatrix}$$

and substituting this in gives

$$\sqrt{\widehat{\text{Var}}(\hat{\beta}_1)} = \sqrt{[\hat{\sigma}^2(\mathbf{X}'\mathbf{X})^{-1}]_{22}} = \sqrt{\frac{\hat{\sigma}^2}{\sum x_i^2 - (\sum x_i)^2}} = \sqrt{\frac{\hat{\sigma}^2}{\sum (x_i - \bar{x}_i)^2}}$$

Example 2.3 ()

Given a dataset

```
Hours Studied for Exam 20 16 20 18 17 16 15 17 15 16 15 17 16 17 14
Grade on Exam 89 72 93 84 81 75 70 82 69 83 80 83 81 84 76
```

The hypotheses are $h_0 : \beta = 0$ and $h_a : \beta \neq 0$, and the degrees of freedom for the t-test is $df = N - (D+1) = 13$, where $N = 15$ is the number of datapoints and $D = 1$ is the number of coefficients (plus the 1 bias term). The critical values is ± 2.160, which can be found by taking the inverse CDF of the t-distribution evaluated at 0.975.

Now we calculate the t score. We have our estimate $\beta_1 = 3.216, \beta_0 = 26.742$, and so we calculate

$$\hat{\sigma}^2 = \frac{1}{15}\sum_{i=1}^{15}\big(y_i - (3.216x_i + 26.742)\big) = 13.426$$
$$\sum_i (x_i - \hat{x}_i)^2 = 41.6$$

and therefore, we can compute

$$t = \frac{\beta_1}{\sqrt{\hat{\sigma}^2 / \sum_i (x_i - \hat{x}_i)^2}} = \frac{3.216}{\sqrt{13.426/41.6}} = 5.661$$

and therefore, this is way further than our critical value of 2.16, meaning that we reject the null hypothesis.

Note that when multicolinearity is present, then $\sum_i (x_i - \hat{x}_i)^2$ will be very small causing the denominator to blow up, and therefore you cannot place too much emphasis on the interpretation of these statistics. While it is hard to see for the single linear regression case, we know that some eigenvalue of $(\mathbf{X}^T\mathbf{X})^{-1}$ will blow up, causing the diagonal entries $(\mathbf{X}^T\mathbf{X})^{-1}_{ii}$ to be very small. When we calculate the standard error by dividing by this small value, the error blows up.

Theorem 2.2 ()

We can compute this t-statistic w.r.t. just the sample size n and the correlation coefficient ρ as such.

$$t = \frac{\hat{\beta} - 0}{\text{SE}(\hat{\beta})}$$

and the denominator is simply

$$\text{SE}(\hat{\beta}) = \sqrt{\frac{\frac{1}{n-1}\sum(y_i - \hat{y})^2}{\sum(x_i - \bar{x})^2}} \implies t = \frac{\hat{\beta}\sqrt{\sum(x_i - \bar{x})^2}\sqrt{n-1}}{\sqrt{\sum(y_i - \hat{y})^2}} = \frac{\hat{\beta}\sqrt{\sum(x_i - \bar{x})^2}\sqrt{n-1}}{\sqrt{(1-\rho^2)}\sqrt{\sum(y_i - \bar{y})^2}}$$
$$= \frac{\rho}{\sqrt{1-\rho^2}}\sqrt{n-1}$$

where the residual sum of squares on the top can be substituted according to our theorem. Therefore

$$t = \frac{\rho}{\sqrt{1-\rho^2}}\sqrt{n-1} \tag{3}$$

2.6.2 F Test

Given that you have n data points that have been fit on a linear model, the F-statistic is based on the ratio of two variances.

3 High Dimensional Linear Regression

3.1 Stepwise Linear Regression

3.2 Ridge Regression

3.3 Lasso Regression

4 Nonparametric Regression

4.1 Kernel Regression

This is a local linear smoother.

Linear smoothers, Kernel regression, Gaussian smoothing, Kernel smoothers, but has the desgin bias and boundary bias problem. Good fix is local linear regression.

4.2 Local Polynomial Regression

Local linear regression, and polynomial regression. This is a local linear smoother.

4.3 Regularized: Spline Smoothing

This is not local, but it's a linear smoother.

4.4 Regularized: RKHS Regression

This is not local, but it's a linear smoother.

4.5 Additive Models

We've learned about linear smoothers to create nonparametric models in 1 dimension. We can then extend this to multiple input dimensions with additive models, which aren't as flexible since they can't capture dependencies, but we can create dependency functions.

4.6 Nonlinear Smoothers, Trend Filtering

Tough example of the Dobbler function (like topologists sine curve). It's a pretty good fit but it's not too good since it's using a linear smoother (homogeneous). So we might need to fit it with nonlinear smoothers.

4.7 High Dimensional Nonparametric Regression

4.8 Regression Trees

5 Cross Validation

5.1 Leave 1 Out Cross Validation

5.1.1 Generalized (Approximate) Cross Validation

5.1.2 Cp Statistic

5.2 K Fold Cross Validation

5.3 Data Leakage

6 Linear Classification

6.1 Empirical Risk Minimizer

You literally just try to build a hyperplane to minimize the number of misclassifications, but this is not really differentiable and is hard. It's just a stepwise function. Therefore, you use a **surrogate loss function** to approximate the 0-1 loss function. The logistic uses some function, and the SVM uses the smallest convex function to approximate the 0-1 loss function.

6.2 Gaussian/Linear Discriminant Analysis

This is the first example of a generative model. In GDA, we basically write the likelihood as

$$\prod_{i=1}^{n} p(x_i, y_i) = \prod_i p(x_i \mid y_i) p(y_i) \tag{4}$$

where each $p(x_i \mid y_i)$ is Gaussian and $p(y_i)$ is Bernoulli. This specifies $p(x_i, y_i)$ and therefore is called a generative model. In logistic regression, we have

$$\prod_{i=1} p(x_i, y_i) = \left(\prod_i p(y_i \mid x_i) \right) \left(\prod_i p(x_i) \right) \tag{5}$$

and the first term is the logistic function and the second term is unknown. We only use the first part to classify, and this is a discriminative model. You can be agnostic about the data generating process and you can work with less data since there are less things to fit. Some people ask why should you model more unless you have to, so people tend to try to model the minimum, which is why logistic regression is more popular.

6.3 Fisher Linear Discriminant

6.4 Perceptron

6.5 Logistic and Softmax Regression

6.5.1 Sparse Logistic Regression

6.6 Support Vector Machines

7 Nonparametric Classification

7.1 K Nearest Neighbors

Maybe similar like a kernel regression?