Linear Regression

Muchang Bahng

Spring 2024

Contents

In introductory courses, we start with linear predictors since it is easy to understand. Low dimensional linear regression is what statisticians worked in back in the early days, where data was generally low dimensional.[1] 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. The line tends to be fuzzy between these two regimes, but we will not worry about strictly defining that now. Let's introduce the most general variant of linear regression.

Definition 0.1 (Linear Regression Model)

A **linear regression model** is a probabilistic model that predicts the conditional distribution of y given x as

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

Another common and compact way of writing it is to encode x as a $(d+1)$-dimensional vector where $x_0 = 1$, and write

$$y = \beta^T x + \epsilon, \beta = (b, w) \in \mathbb{R}^{d+1} \tag{2}$$

Get used to both methods. It has the following assumptions.

1. *Linearity in Parameters.* Note that this does not mean linearity in the *covariates.*[a]
2. *Weak exogeneity.* The covariates are observed without error.
3. ϵ is 0-mean.
4. *Homoscedasticity*: ϵ has constant variance.
5. The ϵ's are uncorrelated with each other.
6. *No multicolinearity*: There exists no covariates that are perfectly correlated.

[a] Therefore you could build a regression using non-linear transformations of the covariates, for instance, $y = w_1 x_1 + w_2 x_2^2 + w_3 \log(x_1)$.

Let's go through these assumptions in detail.

Linearity: This is pretty straightforward, but many beginners assume that we must only fit *lines* with respect to the covariates x_i. This is not true (consider polynomial regression), and all we are assuming is linearity with respect to the *parameters.*

In fact, we can consider combinations of fixed nonlinear functions of the input variables of the form

$$f(x, w) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x) = \beta^T \phi(x) \tag{3}$$

where each *basis function* $\phi_j : \mathbb{R}^D \longrightarrow \mathbb{R}$. If you need to further relax the assumption, you are better off using non-linear modeling.

Example 0.1 (Basis Functions)

We can choose many different types of basis functions. The following examples are for 1-dimensional x.

1. The *polynomial basis functions* form powers of x such that

$$\phi_j(x) = x^j \tag{4}$$

   One limitation of polynomial basis function is that they are global functions on the input variable, so that changes in one region of input space affect all other regions. This can be resolved by dividing up the input space up into regions and fit a different polynomial in each region, leading to *spline functions.*

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

2. The *Gaussian basis functions* (which can be, but not necessarily must be interpreted in the probabilistic way), have the form

$$\phi_j(x) = \exp\bigg(-\frac{(x-\mu_j)^2}{2s^2}\bigg) \tag{5}$$

   where the μ_j govern the locations of the basis functions in input space, and the parameter s governs their spatial scale.
3. The *sigmoidal basis functions* are of form

$$\phi_j(x) = \sigma\bigg(\frac{x-\mu_j}{s}\bigg), \text{ where } \sigma(a) = \frac{1}{1+e^{-a}} \tag{6}$$

   Rather than using the sigmoid function σ, we could also use the hyperbolic tangent $\tanh(a) = 2\sigma(a) - 1$.
4. The *Fourier basis functions* leads to an expansion in sinusoidal functions, which has specific frequency and infinite spatial extent. By contrast, basis functions that are localized to finite regions of input space necessarily comprise a spectrum of different spatial frequencies. In many signal processing applications, it is of interest to consider basis functions that are localized in both space and frequency, leading to a class of functions known as *wavelets*.

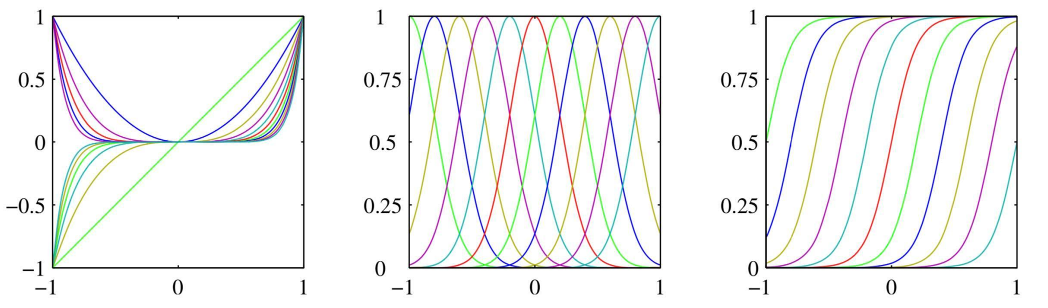

Figure 1: Different types of basis functions

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).

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.

Independence of errors: this is dangerous because in fields like finance, 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. Another field that looks for correlation between samples is time series analysis, which we will get to in another set of notes.

No 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, or making a change of basis to remove the correlation. If we just leave the model as is, 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.[2]

In order to check multicollinearity, we compute the correlation matrix, and see if there are any off-diagonal entries that are very close to 1, then there is multicollinearity.

Some more terminology: **multiple linear regression** 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] I suggest reading this Wikipedia article on multicollinearity, as it contains useful information: `https://en.wikipedia.org/wiki/Multicollinearity`. Multicollinearity is a favorite 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 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.

1 Low-Dimensional Ordinary Least Squares

When you learn linear regression for the first time, you are really learning a very specific part of linear regression called ordinary least squares, which is the linear model that comes with a very specific loss function. Recall the MSE loss.

Definition 1.1 (Ordinary Least Squares Regression)

The **OLS linear regression** model is a linear regression model that tells us to minimize the MSE loss.

Theorem 1.1 (Prediction Risk)

The **prediction risk**[a] of f is

$$R(f) = \mathbb{E}_{x,y}[(y - f(x))^2] = \mathbb{E}_{x,y}[(y - \beta^T x)^2] \tag{7}$$

and the empricial risk is

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^n (y^{(i)} - f(x^{(i)}))^2 = \frac{1}{n} \|Y - X\beta\|^2 \tag{8}$$

where $X \in \mathbb{R}^{n \times d}$ is the data matrix where the ith row is sample $x^{(i)}$, and $Y \in \mathbb{R}^d$ is the vector of sample predictors.

[a] This is a bit different from how Wasserman defines it in his lectures, but I think this is better.

This is a bit weird, since we are just *given* a loss function rather than having derived one from our model. There are two paths that we can take to derive this loss function.

1. The first is to have the extra convenient assumption that the errors ϵ are Gaussian. This is not too unrealistic since combinations of many random noise gives us a Gaussian by the CLT. As often done in machine learning, by computing the likelihood, we can take its negative logarithm to get our loss.
2. The second does *not* assume a distribution on ϵ and rather uses the Gauss-Markov theorem to directly say that the MSE loss minimizes variance among unbiased estimators. This is in a sense more fundamental.

Sometimes, the Gaussian error is given as an assumption, and sometimes it is not. We will go through all these points by first talking about the nice bias-variance decomposition of the MSE loss. Then, we will use the Gauss-Markov theorem justify the MSE loss and introduce the least-squares solution. Finally, we will look at the likelihood approach using Gaussian residuals.

1.1 Least Squares Solution

Using simple calculus, we can simply find the point β where the derivative of the loss is 0.

Theorem 1.2 (Least Squares Solution For Linear Regression)

The solution that minimizes the MSE is

$$\beta = (X^T X)^{-1} X^T Y \tag{9}$$

Proof.

We simply minimize

$$\begin{aligned} L(\beta) = \|Y - X\beta\|^2 = \langle Y - X\beta, Y - X\beta\rangle & \qquad (10) \\ = (Y - X\beta)^T(Y - X\beta) & \qquad (11) \\ = Y^TY - \beta^TX^TY - Y^TX\beta + \beta^TX^TX\beta & \qquad (12) \\ = Y^TY - 2Y^TX\beta + \beta^TX^TX\beta & \qquad (13) \end{aligned}$$

where we omit the constant term because it doesn't affect the argmin. Taking the derivative[a] and setting it equal to 0 gives

$$\begin{aligned} \frac{\partial L}{\partial \beta} = -2X^TY + 2X^TX\beta = 0 \implies X^TX\beta = X^TY & \qquad (14) \\ \implies \hat{\beta} = (X^TX)^{-1}X^TY & \qquad (15) \end{aligned}$$

[a] Remember the following matrix derivative rules when differentiating with respect to vector x. $\frac{\partial}{\partial x}x^TA = A$, $\frac{\partial}{\partial x}x^TAx = 2Ax$.

Example 1.1 (Simulation)

Let's do this for a simple dataset of 200 points across 10 covariates with some noise $\epsilon \sim N(0, 0.01)$.

```
>>> import numpy as np
>>> n, d = 200, 10
>>> beta_true = np.array([1, 2, 3, 1, -2, -3, 1, -2, 3, -1])
>>> X = np.random.randn(n, d)
>>> Y = X.dot(beta_true) + 0.1 * np.random.randn(n)
>>> beta_hat = np.linalg.inv(X.T @ X) @ X.T @ Y
>>> print(beta_hat)
[ 1.00981636  1.99629568  2.99167398  0.98981948 -1.99678893 -2.99819217
  0.99060477 -1.99468555  2.99490104 -0.99246088]
```

We can clearly see that we have good approximates on this.

Corollary 1.1 (Variance of MSE Minimizer)

The variance of the MSE minimizer with respect to Y is

$$\mathrm{Var}(\hat{\beta}) = \hat{\sigma}^2(X^TX)^{-1} \in \mathbb{R}^{d \times d} \qquad (16)$$

Proof.

We can calculate the variance of β by using the fact that $\mathrm{Var}[AX] = A\,\mathrm{Var}[X]A^T$, is

$$\mathrm{Var}(\hat{\beta}) = (X^TX)^{-1}X^T\ \sigma^2 I\ X(X^TX)^{-1} = \sigma^2(X^TX)^{-1}(X^TX)(X^TX)^{-1} = \sigma^2(X^TX)^{-1} \qquad (17)$$

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

$$\begin{aligned} \beta = (X^TX)^{-1}X^TY \in \mathbb{R}^d & \qquad (18) \\ \mathrm{Var}(\hat{\beta}) = \hat{\sigma}^2(X^TX)^{-1} \in \mathbb{R}^{d \times d} & \qquad (19) \end{aligned}$$

Example 1.2 (Simulation)

Let's empirically demonstrate the previous theorem by taking a dataset of fixed X, and then generating the Y 10000 times. For each Y we train $\hat{\beta}$, and take the sample covariance. For conciseness, we only print the top-left 3×3 block matrix.

```
>>> import numpy as np
>>> n, d = 200, 10
>>> beta_true = np.array([1, 2, 3, 1, -2, -3, 1, -2, 3, -1])
>>> X = np.random.randn(n, d)
>>> def run():
...   Y = X.dot(beta_true) + 0.1 * np.random.randn(n)
...   beta_hat = np.linalg.inv(X.T @ X) @ X.T @ Y
...   return beta_hat
...
>>>
>>> true_cov = 0.1 * np.linalg.inv(X.T.dot(X))
>>> print(true_cov[:3, :3]) # True covariance according to our theorem
[[ 5.51430990e-04  1.04398204e-05 -4.60641482e-06]
 [ 1.04398204e-05  5.82171568e-04 -1.70354446e-05]
 [-4.60641482e-06 -1.70354446e-05  4.38241743e-04]]
>>> betas = np.stack([run() for _ in range(10000)], axis=0)
>>> print(np.cov(betas.T)[:3, :3]) # estimated sample covariance
[[ 5.55882814e-05  5.47990882e-07 -4.53151038e-07]
 [ 5.47990882e-07  5.89809163e-05 -2.02919464e-06]
 [-4.53151038e-07 -2.02919464e-06  4.37200525e-05]]
>>>
```

Note that we have assumed that $X^T X$ was invertible in order for such a solution to be unique, i.e. X must be full rank. This process breaks down when it isn't invertible, e.g. if there are repetitions in the features (one feature is a linear combination of the others and hence not full column rank). We will talk more about this soon.

Definition 1.2 (Hat Matrix)

The **hat matrix** is defined

$$H = X(X^T X)^{-1} X^T \in \mathbb{R}^{n \times n} \tag{20}$$

It is the projection of the observed $y^{(i)}$'s to the least squared predictions.[a]

$$\hat{Y} = HY \tag{21}$$

[a] It projects to the column space of X.

We can also see that the residuals $\hat{\epsilon}_i = y_i - \hat{y}_i$ has the property that

$$\hat{\epsilon} = y - \hat{y} = (I_n - H)y \tag{22}$$

Note that this parallels the orthogonal projection of conditional expectation to the true function onto the subspace of X measurable functions. Except that we are not doing this in function space, but rather the sample space $\mathbb{R}^n$.

Example 1.3 (Copying Data)

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

$$\left(\begin{pmatrix} X \\ X \end{pmatrix}^T \begin{pmatrix} X \\ X \end{pmatrix} \right)^{-1} \begin{pmatrix} X \\ X \end{pmatrix}^T \begin{pmatrix} Y \\ Y \end{pmatrix} =$$

$$= (X^T X + X^T X)^{-1} (X^T Y + X^T Y) = (2X^T X)^{-1} 2X^T Y = \hat{\beta}$$

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

$$\frac{\sigma^2}{2} (X^T X)^{-1} \tag{23}$$

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.

1.2 Gauss Markov Theorem

Now recall from our frequentist inference notes that with the MSE loss, we have the bias-variance-noise decomposition as

$$\mathbb{E}_{\mathcal{D}} \mathbb{E}_{x,y} \big[(y - \hat{f}_{\mathcal{D}}(x))^2 \big] = \mathbb{E}_x \Big[\underbrace{\big(\mathbb{E}[y \mid x] - \mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] \big)^2}_{(\text{expected bias})^2} \Big] + \underbrace{\mathbb{E}_{\mathcal{D}} \Big[\mathbb{E}_x \Big[\big(\mathbb{E}_{\mathcal{D}}[\hat{f}_{\mathcal{D}}(x)] - \hat{f}_{\mathcal{D}}(x) \big)^2 \Big] \Big]}_{\text{expected variance}} \tag{24}$$

$$+ \underbrace{\mathbb{E}_{x,y} \big[(y - \mathbb{E}[y \mid x])^2 \big]}_{\text{noise}} \tag{25}$$

Theorem 1.3 (Gauss-Markov Theorem)

Given a dataset with

1. mean zero residuals $\mathbb{E}[\epsilon_i] = 0$, i.e. $\mathbb{E}[\mathbf{Y} \mid \mathbf{X}] = \mathbf{X}\boldsymbol{\beta}$.
2. homoscedacity $\text{Var}[\epsilon_i] = \sigma^2 < \infty$ for all i,
3. uncorrelated residuals $\text{Cov}(\epsilon_i, \epsilon_j) = 0$ for all $i \neq j$. This and the previous assumption can be combined into $\text{Cov}[\mathbf{Y} \mid \mathbf{X}] = \sigma^2 \mathbf{I}_n$.

We were concerned with estimating the parameters $\beta_1, \ldots, \beta_d$. Now let's generalize this and consider the problem of estimating, for some known constants $c_1, \ldots, c_{d+1}$, the point estimator

$$\theta = c_1 \beta_1 + c_2 \beta_2 + \ldots + c_d \beta_d + c_{d+1} \tag{26}$$

Then the estimator

$$\hat{\theta} = c_1 \hat{\beta}_1 + c_2 \hat{\beta}_2 + \ldots + c_d \hat{\beta}_d + c_{d+1} \tag{27}$$

where $\hat{\beta}_i$ is clearly an unbiased estimator of θ and it is a linear estimator of θ, i.e.

$$\hat{\theta} = \sum_{i=1}^n b_i y_i \tag{28}$$

for some known (given $\mathbf{X}$) constants b_i. Then, the Gauss-Markov theorem states that the estimator $\hat{\theta}$ has the smallest (best) variance among *all* linear unbiased estimators of θ, i.e. $\hat{\theta}$ is BLUE.

1.3 Likelihood Estimation

Now we add the *additional* assumption that the residuals are distributed according to a Gaussian. This gives us a parameteric closed-form of our likelihood, which we can then maximize with respect to β. The likelihood is simple to compute.

Lemma 1.1 (Likelihood)

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

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

Theorem 1.4 (Negative Log-Likelihood is MSE Loss)

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

$$\begin{aligned} \ell(\beta) &= -\frac{N}{2} \ln \sigma^2 - \frac{N}{2} \ln(2\pi) + \frac{1}{2\sigma^2} \sum_{i=1}^N \big(y^{(i)} - \beta^T x^{(i)}\big)^2 & (29) \\ &= \frac{1}{2} \sum_{i=1}^N \big(y^{(i)} - \beta^T x^{(i)}\big)^2 & (30) \end{aligned}$$

which then corresponds to minimizing the sum of squares error function.

So assuming Gaussian residuals also gives us back the MSE as a natural loss function. This is usually the way that beginners tend to use the loss function, but this is not as general as the Gauss-Markov theorem, which is *distribution-free*. As soon as we have the same loss, it should be clear that the MLE is the same as the least-squares optimizer.

Corollary 1.2 (MLE is the Same as Least-Squares Estimator)

The MLE is

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{31}$$

which is the same as the least squares estimator. We have an additional nusiance parameter σ^2, however, and the MLE is

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{i=1}^n \big(y^{(i)} - \hat{\beta}^T x^{(i)}\big)^2 \tag{32}$$

where $\hat{\beta}$ is already solved.

Proof.

This is not needed, but let's go through the computations anyways. Taking the gradient of this log likelihood w.r.t. β gives

$$\nabla_\beta \ell(\beta) = \sum_{i=1}^n (y^{(i)} - \beta^T x^{(i)}) x^{(i)} \tag{33}$$

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

$$0 = \sum_{i=1}^n y^{(i)} x^{(i)T} - \beta^T \bigg(\sum_{i=1}^n x^{(i)} x^{(i)T} \bigg) \tag{34}$$

which is equivalent to solving the least squares equation

$$\beta = (X^T X)^{-1} X^T Y \tag{35}$$

Now note that matrix inversion and multiplication are of cubic time complexity. This may be an actual problem when either n or d is large. Therefore, we can simply numerically optimize it with stochastic gradient descent. Over a minibatch $M \subset \mathcal{D}$, we have

$$\begin{align}
\beta &= \beta - \eta \nabla_\beta \ell(\beta) \tag{36} \\
&= \beta - \eta \sum_{(x,y) \in M} (\beta^T x - y) x \tag{37}
\end{align}$$

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

Example 1.4 (SGD From Scratch)

Let's see this in action.

```
>>> import numpy as np
>>> n, d = 200, 5
>>> beta_true = np.array([1, 2, 3, 1, -2])
>>> X = np.random.randn(n, d)
>>> Y = X.dot(beta_true) + 0.1 * np.random.randn(n)
>>> # Gradient descent
>>> lr = 1e-2
>>> beta_hat = np.random.randn(5)
>>> for _ in range(1000):
...   grad = (1/200) * np.sum((Y - X.dot(beta_hat)).reshape(-1, 1) * X, axis=0)
...   beta_hat += lr * grad
...
>>>
>>> print(beta_hat)
[ 1.01353957  1.99694533  2.99698439  1.00279151 -2.00176132]
```

Note that this is sensitive to numerical errors if we set the learning rate to be too high. On the other hand, it may not converge fast enough if we set the learning rate too low. The following are after 1000 steps of convergence.

```
lr: 10.0
[nan nan nan nan nan]
lr: 1.0
[ 0.99783623  2.00872921  3.00480298  1.00303178 -2.00464259]
lr: 0.1
[ 0.99783623  2.00872921  3.00480298  1.00303178 -2.00464259]
lr: 0.01
[ 0.99680738  2.00773822  3.00377349  1.00355067 -2.00464062]
lr: 0.001
[ 0.38075539  0.56665193  1.13758925  0.80863722 -1.51169666]
lr: 0.0001
[-0.12150123 -0.11091773  0.47147945  0.14935135 -0.28254488]
```

Code 1.1 (OLS Linear Regression in scikit-learn)

Here is a minimal working example of performing linear regression with scikit-learn. Note that the input data must be of shape (n, d).

```python
import numpy as np
from sklearn.linear_model import LinearRegression
X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
y = np.dot(X, np.array([1, 2])) + 3
model = LinearRegression()
model.fit(X, y)
print(X)
print(y)
print(model.score(X, y))
print(model.intercept_)
print(model.coef_)
print(model.predict(np.array([[3, 5]])))
```

```
[[1 1]
 [1 2]
 [2 2]
 [2 3]]
[ 6  8  9 11]
1.0
3.0000000000000018
[1. 2.]
[16.]
.
.
.
```

1.4 Coefficient of Determination

We now suggest a more interpretable metric for a linear regression model. Say that we did not have any of the X data and were told to try and predict the Y's according to the MSE loss. The best we can do is simply compute their mean, and our regression function would always be

$$f(x) = \bar{Y} \approx \mathbb{E}_y[y] \tag{38}$$

Rather than thinking of this as a sum of squared errors, we can think of this in term of variance. If we had our naive model above, then our sums of squares errors is equal to the sample variance.

$$\frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - f(x^{(i)}) \right)^2 = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - \bar{y} \right)^2 \approx \mathrm{Var}_y[y] \tag{39}$$

This is not a very good approximation, and this is shown by a large amount of variance that is not captured.

Definition 1.3 (Total Sum of Squares)

The **total sum of squares** is defined

$$TSS = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - \bar{y} \right)^2 \tag{40}$$

What we want to do is *reduce* this variance by having better estimates of the mean $\bar{y}$. Now if we are given the x, perhaps using the x's to model a better mean of y through f might help, and doing this will lead to a hopefully lower variance, i.e. the *residual sum of squares.*

Definition 1.4 (Residual Sum of Squares)

The **residual sum of squares** is defined

$$RSS = \frac{1}{n} \sum_{i=1}^{n} \left(y^{(i)} - f(x^{(i)}) \right)^2 \tag{41}$$

If we have a TSS of 2 and a RSS of 1, then we have decreased our variance by 50%. We can formalize this into the following.

Definition 1.5 (Coefficient of Determination)

The **coefficient of determination** is the proportion of the variation in the dependent variable that is predictable from the independent variables It is defined

$$R^2 = 1 - \frac{RSS}{TSS} \tag{42}$$

and R^2 value of 1 represents the best model.

However, a drawback of R^2 is that it always increases if we add predictors to the regression model, leading to a possible overfitting.

Example 1.5 (RSS and R^2 Values as we Account for More Covariates)

```python
import numpy as np
import matplotlib.pyplot as plt

n, d = 200, 10
beta_true = np.array([1, 2, 3, 1, -2, 1, 2, 3, 1, -2])
X = np.random.randn(n, d)
Y = X.dot(beta_true) + 0.1 * np.random.randn(n)

RSSs = []
R2s = []
TSS = np.var(Y)
RSSs.append(TSS)
R2s.append(0)

for k in range(1, 11):
    X_trunc = X[:,:k]
    beta_hat = np.linalg.inv(X_trunc.T @ X_trunc) @ X_trunc.T @ Y
    RSS = np.var(Y - X_trunc.dot(beta_hat))
    R2 = 1 - RSS/TSS
    RSSs.append(RSS)
    R2s.append(R2)

fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 5))

ax1.plot(range(11), RSSs, 'r-', marker='o')
ax1.set_title('Residual Sum of Squares (RSS)')
ax1.set_xlabel('Number of Features')
ax1.set_ylabel('RSS')
ax1.grid(True, alpha=0.3)

ax2.plot(range(11), R2s, 'b-', marker='o')
ax2.set_title('R-squared')
ax2.set_xlabel('Number of Features')
ax2.set_ylabel('R2')
ax2.grid(True, alpha=0.3)

plt.tight_layout()
plt.show()
```

This gives the following plot. You can indeed see that the RSS is monotonically decreasing and the R^2 is monotonically increasing as we add more covariates.

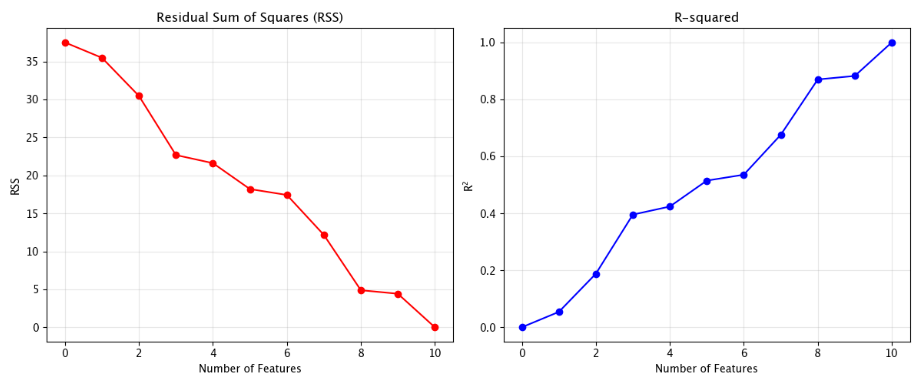

Figure 2: Note that the steps at which the RSS decrease the most are precisely the covariates where its true values are largest. The first covariate (true value of 1) doesn't decrease the RSS too much, but as we add the second (true value 2) and third (3), the RSS decreases much sharply.

There are cases when the $RSS > TSS$, leading to a negative R^2 value. This essentially means that the model is doing worse than just taking the mean.

1.5 Simple Linear Regression

The simple linear regression is the special case of the linear regression with only one covariate.[3] Interviewers like this model for its aesthetically pleasing theoretical properties. We will just list a bunch of them.

Definition 1.6 (Simple Linear Regression)

We will use the following notation.

$$y = \alpha + x\beta \tag{43}$$

Theorem 1.5 (Best Fit)

The least-squares best fit is

$$\hat{\beta} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} = \frac{\mathrm{cov}(x, y)}{\mathrm{var}(x)} = \rho_{xy} \frac{s_y}{s_x} \tag{44}$$

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. $s_x = s_y = 1$.

Proof.

For n pairs of (x_i, y_i),

$$y_i = \alpha + \beta x_i + \epsilon_i \tag{45}$$

To minimize the sum of squared errors

$$\sum_i \epsilon_i^2 = \sum_i (y_i - \alpha - \beta x_i)^2 \tag{46}$$

[3] I've included a separate section on this since this was especially important for quant interviews.

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

$$\sum_i (y_i - \hat{\alpha} - \hat{\beta} x_i) = 0 \tag{47}$$

$$\sum_i (y_i - \hat{\alpha} - \hat{\beta} x_i) x_i = 0 \tag{48}$$

From just the first equation, we can write

$$n\bar{y} = n\hat{\alpha} + n\hat{\beta}\bar{x} \implies y = \hat{\alpha} + \hat{\beta}\bar{x} \implies \hat{\alpha} = \bar{y} - \hat{\beta}\bar{x} \tag{49}$$

The second equation gives

$$\sum_i x_i y_i = \hat{\alpha} n \bar{x} + \hat{\beta} \sum_i x_i^2 \tag{50}$$

and substituting what we derived gives

$$\sum_i x_i y_i = (\bar{y} - \hat{\beta}\bar{x}) n \bar{x} + \hat{\beta} \sum_i x_i^2 \tag{51}$$

$$= n\bar{x}\bar{y} + \hat{\beta}\bigg(\Big(\sum_i x_i^2 \Big) - n\bar{x}^2 \bigg) \tag{52}$$

and so we have

$$\hat{\beta} = \frac{\big(\sum_i x_i y_i\big) - n\bar{x}\bar{y}}{\big(\sum x_i^2\big) - n\bar{x}^2} = \frac{\sum_i x_i y_i - \bar{x} y_i}{\sum x_i^2 - \bar{x} x_i} = \frac{\sum_i (x_i - \bar{x}) y_i}{\sum_i (x_i - \bar{x}) x_i} \tag{53}$$

Now we can use the identity

$$\sum_i (x_i - \bar{x})(y_i - \bar{y}) = \sum_i y_i (x_i - \bar{x}) = \sum_i x_i (y_i - \bar{y}) \tag{54}$$

to substitute both the numerator and denominator of the equation to

$$\hat{\beta} = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sum_i (x_i - \bar{x})^2} = \frac{\mathrm{cov}(x, y)}{\mathrm{var}(x)} = \rho_{xy} \frac{s_y}{s_x} \tag{55}$$

Example 1.6 (Switching Variables)

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

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

and so

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

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.

Theorem 1.6 (Coefficient of Determination)

In simple linear regression, we have

$$R^2 = \rho_{yx}^2 \tag{56}$$

That is, it is simply the square of the sample correlation between x and y. Now the notation R^2 makes sense.

Proof.

$$\text{RSS} = \sum (y_i - \hat{y}_i)^2 = (1 - \rho^2) \sum (y_i - \bar{y})^2 \tag{57}$$

1.6 Concentration Bounds

Let's get a deeper understanding on linear regression by examining the convergence of the empirical risk minimizer to the true risk minimizer. We can develop a naive bound using basic concentration of measure.

Theorem 1.7 (Exponential Bound)

Let $\mathcal{P}$ be the set of all distributions for $X \times Y$ supported on a compact set. There exists constants c_1, c_2 s.t. that the following is true. For any $\epsilon > 0$,

$$\sup_{\mathbb{P} \in \mathcal{P}} \mathbb{P}^n \big(r(\hat{\beta}_n) > r(\beta_*(\mathbb{P}) + 2\epsilon) \big) \leq c_1 e^{-nc_2\epsilon^2} \tag{58}$$

Hence

$$r(\hat{\beta}_n) - r(\beta_*) = O_{\mathbb{P}}\left(\sqrt{\frac{1}{n}} \right) \tag{59}$$

Proof.

Given any β, define $\tilde{\beta} = (-1, \beta)$ and $\Lambda = \mathbb{E}[ZZ^T]$ where $Z = (Y, X)$. Note that

$$r(\beta) = \mathbb{E}(Y - \beta^T X)^2 = \mathbb{E}[(Z^T\tilde{\beta})^2] = \tilde{\beta}^T \Lambda \tilde{\beta}. \tag{60}$$

Similarly,

$$\hat{r}_n(\beta) = \tilde{\beta}^T \hat{\Lambda}_n \tilde{\beta} \tag{61}$$

where

$$\hat{\Lambda}_n = \frac{1}{n} \sum_{i=1}^n Z_i Z_i^T. \tag{62}$$

So

$$|\hat{r}_n(\beta) - r(\beta)| = |\tilde{\beta}^T (\hat{\Lambda}_n - \Lambda) \tilde{\beta}| \leq \|\tilde{\beta}\|_1^2 \Delta_n \tag{63}$$

where

$$\Delta_n = \max_{j,k} |\hat{\Lambda}_n(j,k) - \Lambda(j,k)|. \tag{64}$$

By Hoeffding's inequality and the union bound (applied to each entry of the matrix $\hat{\Lambda}_n - \Lambda$),

$$\mathbb{P}\left(\sup_\beta |\hat{r}_n(\beta) - r(\beta)| > \epsilon \right) \leq c_1 e^{-nc_2\epsilon^2}. \tag{65}$$

On the event $\sup_\beta |\hat{r}_n(\beta) - r(\beta)| < \epsilon$, we have

$$r(\beta_*) \leq r(\hat{\beta}_n) \leq \hat{r}_n(\hat{\beta}_n) + \epsilon \leq \hat{r}_n(\beta_*) + \epsilon \leq r(\beta_*) + 2\epsilon. \tag{66}$$

The second inequality uses the fact that $|\hat{r}_n(\hat{\beta}_n) - r(\hat{\beta}_n)| < \epsilon$, the third uses the definition of $\hat{\beta}_n$ as the minimizer of $\hat{r}_n$, and the fourth uses $|\hat{r}_n(\beta_*) - r(\beta_*)| < \epsilon$.

Therefore,

$$\mathbb{P}^n(r(\hat{\beta}_n) > r(\beta_*(\mathbb{P})) + 2\epsilon) \leq \mathbb{P}\left(\sup_\beta |\hat{r}_n(\beta) - r(\beta)| \geq \epsilon\right) \leq c_1 e^{-nc_2\epsilon^2}. \tag{67}$$

Taking the supremum over $\mathbb{P} \in \mathcal{P}$ gives the first result.
For the second result, the exponential bound implies that for any $\delta > 0$,

$$\mathbb{P}(r(\hat{\beta}_n) - r(\beta_*) > t) \leq c_1 e^{-nc_2t^2/4} \tag{68}$$

for $t > 0$. Setting this equal to δ and solving for t gives $t = O(\sqrt{\log(1/\delta)/n})$. Since this holds for all $\delta > 0$, we have $r(\hat{\beta}_n) - r(\beta_*) = O_{\mathbb{P}}(\sqrt{1/n})$.

However, this is not a very tight bound, and we can do better. The next theorem reveals to us that in linear regression, the bounds are of order $\frac{d}{n}$, and so scales linearly with dimension d.

Theorem 1.8 (Gyorfi, Kohler, Krzyzak, Walk, 2002 [GKKW02])

Let $\sigma^2 = \sup_x \mathrm{Var}[Y \mid X = x] < \infty$. Assume that all random variables are bounded by $L < \infty$. Then

$$\mathbb{E}\int |\hat{\beta}^T x - m(x)|^2 \, d\mathbb{P}(x) \leq 8 \inf_\beta \int |\beta^T x - m(x)|^2 \, d\mathbb{P}(x) + \frac{Cd(\log(n) + 1)}{n} \tag{69}$$

Proof.

Straightforward but long. Omitted.

You can see that the bound contains a term of the form

$$\frac{d\log(n)}{n} \tag{70}$$

and under the low dimensional case, d is small and bound is good. However, as d becomes large, then we don't have as good of theoretical guarantees.

Theorem 1.9 (Central Limit Theorem of OLS)

We have

$$\sqrt{n}(\hat{\beta} - \beta) \xrightarrow{d} N(0, \Gamma) \tag{71}$$

where

$$\Gamma = \Sigma^{-1}\mathbb{E}\big[(Y - X^T\beta)^2 X X^T\big]\Sigma^{-1} \tag{72}$$

The covariance matrix Γ can be consistently estimated by

$$\hat{\Gamma} = \hat{\Sigma}^{-1}\hat{M}\hat{\Sigma}^{-1} \tag{73}$$

where

$$\hat{M}(j,k) = \frac{1}{n}\sum_{i=1}^n X_i(j) X_i(k) \hat{\epsilon}_i^2 \tag{74}$$

and $\hat{\epsilon}_i = Y_i - \hat{\beta}^T X_i$.

1.7 Multivariate OLS

Now we generalize this by considering multiple predictor variables $y_1, \ldots, y_m$.

Definition 1.7 (Multivariate Linear Regression)

A **multivariate linear regression model** is a probabilistic model that predicts the conditional distribution of $y \in \mathbb{R}^m$ given $x \in \mathbb{R}^d$ as

$$y = b + W^T x + \epsilon, \qquad \epsilon \sim N(0, \Sigma) \tag{75}$$

Another common and compact way of writing it is to encode x as a $(d+1)$-dimensional vector where $x_0 = 1$, and write

$$y = \beta^T x + \epsilon, \qquad \beta = (b, W) \in \mathbb{R}^{m \times (d+1)} \tag{76}$$

Note that now β is a matrix we have to fit and we have the variance of the noise Σ to fit as well, which may not be diagonal. It has the following assumptions.

1. *Linearity in Parameters.* Note that this does not mean linearity in the *covariates.*
2. *Weak exogeneity.* The covariates are observed without error.
3. ϵ is 0-mean.
4. The ϵ_i's may be correlated with each other across covariates.
5. *Homoscedasticity*: ϵ has constant variance.
6. The ϵ's are uncorrelated with each other across samples.
7. *No multicolinearity*: There exists no covariates that are perfectly correlated.

This is pretty much the linear regression model but now we must account for the general covariance of the error term, which may not be diagonal. We minimize this with the MSE.

Lemma 1.2 (Risk)

The prediction risk of f is

$$R(f) = \mathbb{E}_{x,y}\left[\|y - \beta^T x\|^2\right] \tag{77}$$

and the empirical risk is

$$\hat{R}(f) = \frac{1}{n}\sum_{i=1}^n \|y^{(i)} - \beta^T x^{(i)}\|^2 = \frac{1}{n}\|Y - X\beta\|_F^2 \tag{78}$$

where $X \in \mathbb{R}^{n \times d}$ is the data matrix where the ith row is sample $x^{(i)}$, $Y \in \mathbb{R}^d$ is the vector of sample predictors, and $\|\cdot\|_F$ is the Frobenius norm.

Theorem 1.10 (Least Squares Solution)

The OLS solution is

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{79}$$

$$\hat{\Sigma} = \frac{1}{n-p-1} Y^T (I - X(X^T X)^{-1} X^T) Y \tag{80}$$

Proof.

For the OLS estimator $\hat{\beta}$, we minimize the empirical risk:

$$\hat{\beta} = \arg\min_{\beta} \frac{1}{n}\|Y - X\beta\|_F^2 \tag{81}$$

Expanding the Frobenius norm:

$$\|Y - X\beta\|_F^2 = \operatorname{tr}((Y - X\beta)^T (Y - X\beta)) = \operatorname{tr}(Y^T Y - Y^T X\beta - \beta^T X^T Y + \beta^T X^T X\beta) \tag{82}$$

Taking the derivative with respect to β and setting to zero:

$$\frac{\partial}{\partial \beta} \|Y - X\beta\|_F^2 = -2X^T Y + 2X^T X\beta = 0 \tag{83}$$

Solving for β:

$$X^T X\beta = X^T Y \tag{84}$$

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{85}$$

For the error covariance estimate $\hat{\epsilon}$, we need the residual sum of squares. The residuals are:

$$e = Y - X\hat{\beta} = Y - X(X^T X)^{-1} X^T Y = (I - X(X^T X)^{-1} X^T)Y \tag{86}$$

The residual sum of squares is:

$$RSS = e^T e = Y^T (I - X(X^T X)^{-1} X^T)Y \tag{87}$$

For an unbiased estimate of the error variance, we divide by the degrees of freedom $(n - p - 1)$ where p is the number of parameters excluding the intercept:

$$\hat{\epsilon} = \frac{1}{n - p - 1} Y^T (I - X(X^T X)^{-1} X^T)Y \tag{88}$$

It turns out that if we do a maximum likelihood estimation, we get a slight bias in our estimated covariance. It becomes slightly smaller.

Theorem 1.11 (MLE Solution)

The MLE is

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{89}$$

$$\hat{\Sigma} = \frac{1}{n} Y^T (I - X(X^T X)^{-1} X^T)Y \tag{90}$$

Proof.

For the multivariate linear regression model $y = \beta^T x + \epsilon$ where $\epsilon \sim N(0, \Sigma)$, the likelihood for a single observation is:

$$p(y^{(i)} | x^{(i)}, \beta, \Sigma) = \frac{1}{(2\pi)^{m/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2} (y^{(i)} - \beta^T x^{(i)})^T \Sigma^{-1} (y^{(i)} - \beta^T x^{(i)}) \right) \tag{91}$$

The log-likelihood for all n observations is:

$$\ell(\beta, \Sigma) = -\frac{nm}{2} \log(2\pi) - \frac{n}{2} \log |\Sigma| - \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \beta^T x^{(i)})^T \Sigma^{-1} (y^{(i)} - \beta^T x^{(i)}) \tag{92}$$

In matrix form:

$$\ell(\beta, \Sigma) = -\frac{nm}{2} \log(2\pi) - \frac{n}{2} \log |\Sigma| - \frac{1}{2} \text{tr}(\Sigma^{-1} (Y - X\beta)^T (Y - X\beta)) \tag{93}$$

Taking the derivative with respect to β:

$$\frac{\partial \ell}{\partial \beta} = \text{tr}(\Sigma^{-1} X^T (Y - X\beta)) = 0 \tag{94}$$

which gives us $X^T(Y - X\beta) = 0 \implies \hat{\beta} = (X^TX)^{-1}X^TY$. For $\hat{\Sigma}$, substituting $\hat{\beta}$ back into the log-likelihood and maximizing with respect to Σ:

$$\frac{\partial \ell}{\partial \Sigma} = -\frac{n}{2}\Sigma^{-1} + \frac{1}{2}\Sigma^{-1}(Y - X\hat{\beta})^T(Y - X\hat{\beta})\Sigma^{-1} = 0 \tag{95}$$

Solving for Σ:

$$\hat{\Sigma} = \frac{1}{n}(Y - X\hat{\beta})^T(Y - X\hat{\beta}) \tag{96}$$

$$= \frac{1}{n}Y^T(I - X(X^TX)^{-1}X^T)Y \tag{97}$$

Therefore:

$$\hat{\epsilon} = \frac{1}{n}Y^T(I - X(X^TX)^{-1}X^T)Y \tag{98}$$

The key difference between OLS and MLE is that MLE uses $\frac{1}{n}$ (which gives a biased but consistent estimator) while OLS uses $\frac{1}{n-p-1}$ (which gives an unbiased estimator).

2 Significance Tests and Confidence Sets

This is not as emphasized in the machine learning literature, but it is useful to know from a statistical point of view.[4]

2.1 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,

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

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

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

where the standard error of the distribution is

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

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

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

[4] This is also asked in quant interviews.

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

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

Example 2.1 ()

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

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

$$\sum_i (x_i - \hat{x}_i)^2 = 41.6$$

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 2.1 ()

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

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

and the denominator is simply

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

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

$$t = \frac{\rho}{\sqrt{1-\rho^2}}\sqrt{n-1} \tag{99}$$

2.2 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 Ridge Regression

Ridge regression is used both in the high dimensional case or when our function space is too large/complex, which leads to overfitting. In the overfitting case, we have seen that either decreasing our function space or getting more training data helps. Another popular way is to add a *regularizing term* to the loss function in order to discourage the coefficients from reaching large values, effectively limiting the variance over $\mathcal{D}$. These are called *shrinkage models*, which "shrinks" the parameters to 0.

Definition 3.1 (Ridge Regression)

Ridge regression[a] refers to a linear model minimized with the *ridge loss*.

$$L(f, x, y) = (y - f(x))^2 + \lambda \|\beta\|^2 \tag{100}$$

where we penalize according to the L^2 norm of the coefficients.

[a] Also called weight decay in machine learning or Tikhonov regularization in signal processing.

Therefore, this regularization term effectively controls the variance that our estimator could have, which inevitably trades off with the bias. Therefore, λ acts as sort of a tuning knob between bias and variance. Think of the extreme cases when $\lambda \to \infty$. Then, all weights would be 0, and we would have extreme bias but no variance. On the other hand if $\lambda = 0$, then we are back to OLS.

Lemma 3.1 (Risk)

The prediction risk is of f is

$$R(f) = \mathbb{E}_{x,y} \left[(y - f(x))^2 + \|\beta\|^2 \right] = \mathbb{E}_{x,y} \left[(y - \beta^T x)^2 + \|\beta\|^2 \right] \tag{101}$$

and the empirical prediction risk is

$$\hat{R}(f) = \frac{1}{n} \left(\sum_{i=1}^n (y^{(i)} - f(x^{(i)}))^2 \right) + \lambda \|\beta\|^2 \tag{102}$$

Again, we should question why we should choose *this* form of the risk? Sure we should find some function that shrinks x to 0, but why the L^2 norm? One reason is that it is convenient and has a lot of nice properties as we will see later. Another is that later, in the Bayesian interpretation, this is equivalent to having a Gaussian prior on the parameter space. Other than these two reasons, I still have not yet found a good derivation, e.g. the analogue of the Gauss-Markov theorem or even some distributional assumptions that lead to this loss.

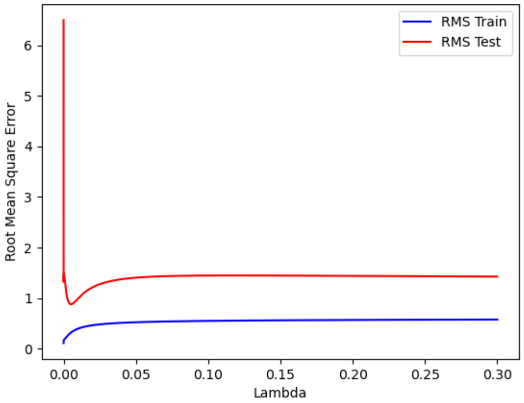

Figure 3: Even with a slight increase in the regularization term λ, the RMS error on the testing set heavily decreases.

3.1 Least Squares Solution

Now that we have this form, we might as well just solve it.

Theorem 3.1 (Least Squares Solution for Ridge Regression)

The minimizer of the ridge loss is

$$\hat{\beta} = (X^T X + \lambda I)^{-1} X^T Y \tag{103}$$

Proof.

TBD

Code 3.1 (MWS of Ridge Regression in scikit-learn)

```python
import numpy as np
from sklearn.linear_model import Ridge

X = np.random.randn(10, 5)
y = np.random.randn(10)
# regularization parameter
model = Ridge(alpha=1.0)
model.fit(X, y)
print(model.score(X, y))
print(model.intercept_)
print(model.coef_)
print(model.predict(np.array([[1, 2, 3, 4, 5]])))
```

```
0.8605535024325397
-0.28291076492665157
[-0.10400521 -0.7587073
    -0.05116735  1.16236649
    -0.0401323 ]
[2.39097184]
.
.
.
.
.
.
```

3.2 Bias Variance Tradeoff

Theorem 3.2 (Bias Variance Decomposition of Ridge Regression)

TBD

From a computational point of view, we can see that by adding the λI term, it *dampens* the matrix so that it does become invertible (or well conditioned), allowing us to find a solution. The higher the λ term, the higher the damping effect.

3.3 Concentration Bounds

The next theorem compares the performance of the best ridge regression estimator to the best linear predictor.

Theorem 3.3 (Hsu, Kakade, Zhang, 2014 [HKZ14])

Suppose that $||X_i|| \leq r$ and let $\beta^T x$ be the best linear approximation to $f(x)$. Then, with probability at least $1 - 4e^{-t}$, we have

$$r(\hat{\beta}) - r(\beta) \leq \left(1 + O\left(\frac{1 + r^2/\lambda}{n}\right)\right) \frac{\lambda ||\beta||^2}{2} + \frac{\sigma^2}{n} \frac{\operatorname{Tr}(\Sigma)}{2\lambda} \tag{104}$$

We can see that the λ term exists in the numerator on $\frac{\lambda ||\beta||^2}{2}$ and in the denominator on $\frac{\operatorname{Tr}(\Sigma)}{2\lambda}$. This is the bias variance tradeoff. The first term is the bias term, which is the penalty for not being able to fit the data as well. The second term is the variance term, which is the penalty for having a more complex model. So our optimal λ in the theoretical sense would be the one that minimizes the sum of these two terms. In practice, it's not this clean since we have unknown quantities in the formula, but just like how we did cross validation over the model complexity, we can also do cross validation over the λ. The decomposition above just gives you a theoretical feeling of how these things trade off.

3.4 Tuning the Regularization Coefficient

4 Stepwise Regression

Now we move to *sparse* linear regression.

Now suppose that $d > n$, then the first problem is that we can no longer use least squares since $X^T X$ is no longer invertible and the same problem happens with maximum likelihood. This is known as the **high dimensional** or **large** p**, small** n problem. The most straightforward way is simply to reduce the covariates to a dimension smaller than n. This can be done with three ways.

1. We perform PCA on the X and use the first k principal components where $k < n$.
2. We cluster the covariates based on their correlation. We can use one feature from each cluster or take the average of the covariates within each cluster.
3. We can screen the variables by choosing the k features that have the largest correlation with Y.

Once this is done, we are back in the low dimensional regime and can use least squares. Essentially, this is a way to find a good subset of the covariates, which can be formalized by the following. Let S be a subset of $[d]$ and let $X_S = (X_j : j \in S)$. If the size of S is not too large, we can regress Y on X_S instead of X.

4.1 Best Subset Regression

Definition 4.1 (Best Subset Regression)

Best subset regression is a linear regression model that wants to solve the best subset S that minimizes the constrained loss

$$L(\beta, x, y) = (y - \beta x)^2, \qquad \text{subject to } \|\beta\|_0 \leq K \tag{105}$$

where $\|\beta\|_0$ is the number of non-zero entries in β.[a]

[a] Note that $\|\cdot\|_0$ is not a norm.

There will be a bias variance tradeoff. As k increases, the bias decreases but the variance increases.

The minimization of the empirical error is over all subset of size k, and from this fact we can expect bad news.

Theorem 4.1 (Best Subset Regression is NP-Hard)

Solving the best subset loss is NP-hard.

Proof.

Even though best subset regression is infeasible, we can still approximate best subset regression in two different ways.

1. A greedy approximation leads to *forward stepwise regression.*
2. A convex relaxation of the problem leads to the *Lasso* regression.

It turns out that the theoretical guarantees and computational time for both are the same, but the Lasso is much more popular due to having cleaner form and performing better in practice.

4.2 Forward Stepwise Regression

Forward stepwise regression is a greedy algorithm that starts with an empty set of covariates and adds the covariate that most improves the fit. It avoids the NP-hardness of the best subset regression by adding covariates one by one.

Definition 4.2 (Greedy Forward Stepwise Regression)

Given your data $\mathcal{D}$, let's first standardize it to have mean 0 and variance 1.[a] You start off with a set $\mathcal{Q} = \{\}$ and choose the number of parameters K.

1. With each covariate $X = (X_1, \ldots, X_n)$, we compute the correlation between it and the Y, which reduces to the inner product (since we standardized).

$$\rho_j = \langle Y, X_{:,j} \rangle = \frac{1}{n} \sum_{i=1}^n Y_i X_{ji} \tag{106}$$

2. Then, we take the covariate index that has the highest empirical correlation with Y, add it to $\mathcal{Q}$ and regress Y only on this covariate.

$$q_1 = \operatorname*{argmax}_j \rho_j, \quad \mathcal{Q} = \{q_1\}, \quad \hat{\beta}_{q_1} = \operatorname*{argmin}_\beta \frac{1}{n} ||Y - X_{:,q_1}\beta||^2 \tag{107}$$

3. Then you repeat the process. You take the residual values $r = Y - X_{:,q_1}\hat{\beta}_{q_1} \in \mathbb{R}^n$ compute the correlation between r and the remaining covariates, and pick our the maximum covariate index q_2. Then, you *repeat the regression from start* with these two covariates

$$q_2 = \operatorname*{argmax}_j \langle r, X_{:;j} \rangle, \quad \mathcal{Q} = \{q_1, q_2\}, \quad \hat{\beta}_{q_1,q_2} = \operatorname*{argmin}_\beta \frac{1}{n} ||Y - X_{:,[q_1,q_2]}\beta||^2 \tag{108}$$

   Note that you're not going to get the same coefficient for $\hat{\beta}_{q_1}$ as before since you're doing two variable regression.

4. You get out the residual values $r = Y - X_{:,[q_1,q_2]}\hat{\beta}_{q_1,q_2} \in \mathbb{R}^n$ and keep repeating this process until you have K covariates in $\mathcal{Q}$.

[a] This may or may not be a good idea, since the variance of each covariate can tell you a lot about the importance of the covariate.

Again, there is a bias variance tradeoff in choosing the number of covariates K, but through cross-validation, we can find the optimal K. It is also easy to add constraints, e.g. if we wanted to place a restriction that two adjacent covariates can't be chosen, we can easily add this to the algorithm.

4.3 Bias Variance Tradeoff

4.4 Concentration Bounds

Theorem 4.2 (DeVore and Temlyakov, 1996)

For all $f \in \mathcal{L}_1$, the residual r_N after N steps of OGA satisfies

$$\|r_N\| \leq \frac{\|f\|_{\mathcal{L}_1}}{\sqrt{N+1}} \tag{4}$$

for all $N \geq 1$.

Proof.

Note that f_N is the best approximation to f from $\text{Span}(V_N)$. On the other hand, the best approximation from the set $\{a g_N : a \in \mathbb{R}\}$ is $\langle f, g_N \rangle g_N$. The error of the former must be smaller than the error of the latter. In other words, $\|f - f_N\|^2 \leq \|f - f_{N-1} - \langle r_{N-1}, g_N \rangle g_N\|^2$. Thus,

$$\begin{aligned}
\|r_N\|^2 &\leq \|r_{N-1} - \langle r_{N-1}, g_N\rangle g_N\|^2 && (109) \\
&= \|r_{N-1}\|^2 + |\langle r_{N-1}, g_N\rangle|^2 \|g_N\|^2 - 2|\langle r_{N-1}, g_N\rangle|^2 && (110) \\
&= \|r_{N-1}\|^2 - |\langle r_{N-1}, g_N\rangle|^2. && (5)
\end{aligned}$$

Now, $f = f_{N-1} + r_{N-1}$ and $\langle f_{N-1}, r_{N-1}\rangle = 0$. So,

$$\begin{aligned}
\|r_{N-1}\|^2 &= \langle r_{N-1}, r_{N-1}\rangle = \langle r_{N-1}, f - f_{N-1}\rangle = \langle r_{N-1}, f\rangle - \langle r_{N-1}, f_{N-1}\rangle && (111) \\
&= \langle r_{N-1}, f\rangle = \sum_j \beta_j \langle r_{N-1}, \psi_j\rangle \leq \sup_{\psi \in D} |\langle r_{N-1}, \psi\rangle| \sum_j |\beta_j| && (112) \\
&= \sup_{\psi \in D} |\langle r_{N-1}, \psi\rangle| \|f\|_{\mathcal{L}_1} = |\langle r_{N-1}, g_N\rangle| \|f\|_{\mathcal{L}_1}. && (113)
\end{aligned}$$

Continuing from equation (5), we have

$$\begin{aligned}
\|r_N\|^2 &\leq \|r_{N-1}\|^2 - |\langle r_{N-1}, g_N\rangle|^2 = \|r_{N-1}\|^2 \left(1 - \frac{\|r_{N-1}\|^2 |\langle r_{N-1}, g_N\rangle|^2}{\|r_{N-1}\|^4}\right) && (114) \\
&\leq \|r_{N-1}\|^2 \left(1 - \frac{\|r_{N-1}\|^2 |\langle r_{N-1}, g_N\rangle|^2}{|\langle r_{N-1}, g_N\rangle|^2 \|f\|_{\mathcal{L}_1}^2}\right) = \|r_{N-1}\|^2 \left(1 - \frac{\|r_{N-1}\|^2}{\|f\|_{\mathcal{L}_1}^2}\right). && (115)
\end{aligned}$$

If $a_0 \geq a_1 \geq a_2 \geq \cdots$ are nonnegative numbers such that $a_0 \leq M$ and $a_N \leq a_{N-1}(1 - a_{N-1}/M)$ then it follows from induction that $a_N \leq M/(N+1)$. The result follows by setting $a_N = \|r_N\|^2$ and $M = \|f\|_{\mathcal{L}_1}^2$. □

If f is not in $\mathcal{L}_1$, it is still possible to bound the error as follows.

Theorem 4.3 ()

For all $f \in \mathcal{H}$ and $h \in \mathcal{L}_1$,

$$\|r_N\|^2 \leq \|f - h\|^2 + \frac{4\|h\|_{\mathcal{L}_1}^2}{N}. \tag{6}$$

Proof.

Choose any $h \in \mathcal{L}_1$ and write $h = \sum_j \beta_j \psi_j$ where $\|h\|_{\mathcal{L}_1} = \sum_j |\beta_j|$. Write $f = f_{N-1} + f - f_{N-1} = f_{N-1} + r_{N-1}$ and note that r_{N-1} is orthogonal to f_{N-1}. Hence, $\|r_{N-1}\|^2 = \langle r_{N-1}, f\rangle$ and so

$$\begin{aligned}
\|r_{N-1}\|^2 &= \langle r_{N-1}, f\rangle = \langle r_{N-1}, h + f - h\rangle = \langle r_{N-1}, h\rangle + \langle r_{N-1}, f - h\rangle && (116) \\
&\leq \langle r_{N-1}, h\rangle + \|r_{N-1}\| \|f - h\| && (117) \\
&= \sum_j \beta_j \langle r_{N-1}, \psi_j\rangle + \|r_{N-1}\| \|f - h\| && (118) \\
&\leq \sum_j |\beta_j| |\langle r_{N-1}, \psi_j\rangle| + \|r_{N-1}\| \|f - h\| && (119) \\
&\leq \max_j |\langle r_{N-1}, \psi_j\rangle| \sum_j |\beta_j| + \|r_{N-1}\| \|f - h\| && (120) \\
&= |\langle r_{N-1}, g_N\rangle| \|h\|_{\mathcal{L}_1} + \|r_{N-1}\| \|f - h\| && (121) \\
&\leq |\langle r_{N-1}, g_N\rangle| \|h\|_{\mathcal{L}_1} + \frac{1}{2}(\|r_{N-1}\|^2 + \|f - h\|^2). && (122)
\end{aligned}$$

Hence,

$$|\langle r_{N-1}, g_N \rangle|^2 \geq \frac{(\|r_{N-1}\|^2 - \|f-h\|^2)^2}{4\|h\|^2_{\mathcal{L}_1}}. \tag{123}$$

Thus,

$$a_N \leq a_{N-1}\left(1 - \frac{a_{N-1}}{4\|h\|^2_{\mathcal{L}_1}}\right) \tag{124}$$

where $a_N = \|r_N\|^2 - \|f-h\|^2$. By induction, the last displayed inequality implies that $a_N \leq 4\|h\|^2_{\mathcal{L}_1}/k$ and the result follows. □

Corollary 4.1 ()

For each N,

$$\|r_N\|^2 \leq \sigma_N^2 + \frac{4\theta_N^2}{N} \tag{125}$$

where θ_N is the $\mathcal{L}_1$ norm of the best N-atom approximation.

By combining the previous results with concentration of measure arguments we get the following result, due to Barron, Cohen, Dahmen and DeVore (2008).

Theorem 4.4 ()

Let $\hat{h}_n = \arg\min_{h \in \mathcal{F}_n} \|f_0 - h\|^2$. Suppose that $\limsup_{n\to\infty} \|\hat{h}_n\|_{\mathcal{L}_1, n} < \infty$. Let $N \sim \sqrt{n}$. Then, for every $\gamma > 0$, there exist $C > 0$ such that

$$\|f - \hat{f}_N\|^2 \leq 4\sigma_N^2 + \frac{C \log n}{n^{1/2}} \tag{126}$$

except on a set of probability $n^{-\gamma}$.

Proof.

If we square root it to compare the norm, this reduces to a rate of about $\frac{1}{\sqrt[4]{n}}$. This is the optimal rate. What this is saying that forward stepwise gets to within about $\frac{1}{\sqrt{n}}$ of what you would get if you did the perfect best subset regression. Another interesting property is that this bound is dimensionless, which makes sense since we are approximating the best K-term linear predictor (not the true regressor), which is a weaker claim. On the other hand, if we use nonparameteric estimators to estimate the true regressor, then we will get the curse of dimensionality.

4.5 Stagewise Regression

Stagewise regression is a variant of forward stepwise regression where we add the covariate that most improves the fit, but we only take a small step in that direction. This is useful when we have a lot of covariates and we don't want to overfit.

5 Lasso Regression

The Lasso approximates the best subset regression by using a convex relaxation. In particular, the norm $||\beta||_0$ is not convex, but the L1 norm $||\beta||_1$ is. Therefore, we want relax our constraint equation as such:

$$\operatorname*{argmin}_{||\beta||_0 \leq L} R(\beta) \mapsto \operatorname*{argmin}_{||\beta||_1 \leq L} R(\beta) \tag{127}$$

This gives us a convex problem, which we can then solve. In fact, it turns out that optimizing the risk given the L1 restriction on the norm is equivalent to minimizing the risk plus a L1 penalty, as this is the Lagrangian form of the original equation (this is in convex optimization). Therefore, there exists a pair (L, λ) for which the two problems are equivalent

$$\operatorname*{argmin}_{||\beta||_1 \leq L} R(\beta) = \operatorname*{argmin}_{\beta} R(\beta) + \lambda ||\beta||_1 \tag{128}$$

Optimizing this gives us a sparse solution. That is, for large enough λ, many of the components of $\hat{\beta}$ are 0.

Definition 5.1 (LASSO Regression)

Lasso regression takes a linear regression model and minimizes the constrained loss

$$L(\beta, x, y) = (y - \beta^T x)^2 + \lambda \|\beta\|_1 \tag{129}$$

where we penalize according to the L1 norm of the coefficients.

Theorem 5.1 (Risk)

The risk is

$$R(\beta) = \mathbb{E}_{x,y} \big[(y - \beta^T x)^2 + \lambda \|\beta\|_1 \big] \tag{130}$$

The empirical risk is

$$\hat{R}(\beta) = \frac{1}{n} \sum_{i=1}^n (y^{(i)} - \beta^T x^{(i)})^2 + \lambda \|\beta\|_1 \tag{131}$$

Unfortunately, there is no closed form of this best estimator, so we must numerically optimize it.

5.1 Sparsity

We mentioned sparsity as a main motivation behind the Lasso, but let's formalize this treatment. To gain some intuition, take a look at the example below.

Example 5.1 (Comparison of L^0, L^1, L^2 Norms)

Let us take the two vectors

$$a = \bigg(\frac{1}{\sqrt{d}}, \ldots, \frac{1}{\sqrt{d}} \bigg), \qquad b = (1, 0, \ldots, 0) \tag{132}$$

Then the L0, L1, and L2 norms of a are $d, \sqrt{d}, 1$ and those of b are $1, 1, 1$. We want to choose a norm that capture the sparsity of b and distinguishes it from b., The L0 norm clearly does this, but the L2 norm does not. The L1 norm is a good compromise between the two.

Now let's establish this fact.

Theorem 5.2 (Sparse Estimators in Lasso)

Proof.

The classical intuition for this is the figure below, where the equipotential lines have "corners." In fact for any $0 < p < 1$, there are also corners, but the problem with using these p-norms is that they are not convex.

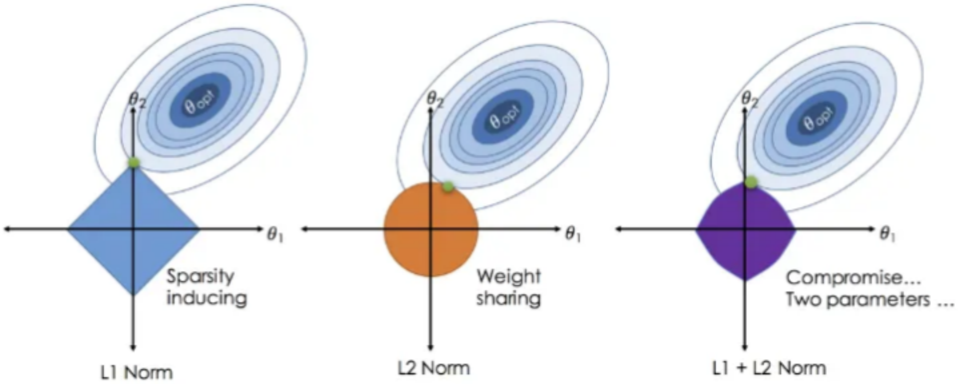

Figure 4: The ridge regularizer draws equipotential circles in our parameter space. The lasso draws a diamond, which tends to give a sparser solution since the loss is most likely to "touch" the corners of the contour plots of the regularizer. The elastic net is a linear combination of the ridge and lasso regularizers.

5.2 Bias Variance Tradeoff

5.3 Concentration Bounds

There are 3 measures we want to talk about: sparsistency, consistency, and risk consistency.

Theorem 5.3 (Sparsistency: Wainwright 2006)

Let $\beta = (\beta_1, \ldots, \beta_s, 0, \ldots, 0)$ and decompose the design matrix as $X = (X_S \;\; X_{S^c})$ where $S = \{1, \ldots, s\}$. Let $\beta_S = (\beta_1, \ldots, \beta_s)$. Suppose that:

1. The true model is linear.
2. The design matrix satisfies
$$\|X_S^c X_S (X_S^T X_S)^{-1}\|_\infty \leq 1 - \epsilon \quad \text{for some } 0 < \epsilon \leq 1. \tag{133}$$
3. $\phi_n(d_n) > 0$.
4. The ϵ_i are Normal.
5. λ_n satisfies
$$\frac{n\lambda_n^2}{\log(d_n - s_n)} \to \infty \tag{134}$$
and
$$\frac{1}{\min_{1 \leq j \leq s_n} |\beta_j|} \left(\sqrt{\frac{\log s_n}{n}} + \lambda_n \left\| \left(\frac{1}{n} X^T X \right)^{-1} \right\|_\infty \right) \to 0. \tag{135}$$

Then the lasso is sparsistent, meaning that $P(\text{support}(\hat{\beta}) = \text{support}(\beta)) \to 1$ where $\text{support}(\beta) = \{j : \beta(j) \neq 0\}$.

Theorem 5.4 (Consistency: Meinshausen and Yu 2006)

Assume that

1. The true regression function is linear.
2. The columns of X have norm n and the covariates are bounded.
3. $\mathbb{E}(\exp|\epsilon_i|) < \infty$ and $\mathbb{E}(\epsilon_i^2) = \sigma^2 < \infty$.
4. $\mathbb{E}(Y_i^2) \leq \sigma_i^2 < \infty$.
5. $0 < \phi_n(k_n) \leq \Phi_n(k_n) < \infty$ for $k_n = \min\{n, d_n\}$.
6. $\liminf_{n\to\infty} \phi_n(s_n \log n) > 0$ where $s_n = \|\beta_n\|_0$.

Then

$$\|\hat{\beta}_n - \beta_n\|^2 = O_P\left(\frac{\log n}{\phi_n^2(s_n \log n)}\right) + O\left(\frac{1}{\log n}\right) \tag{136}$$

If

$$s_n \log d_n \left(\frac{\log n}{n}\right) \to 0 \tag{137}$$

and

$$\lambda_n = \sqrt{\frac{\sigma_n^2 \Phi_n(\min\{n, d_n\}) n^2}{s_n \log n}} \tag{138}$$

then $\|\hat{\beta}_n - \beta_n\|^2 \xrightarrow{P} 0$.
Once again, the conditions of this theorem are very strong. They are not checkable and they are unlikely to ever be true in practice.

Theoretically and practically, these two results are important, but it has the unrealistic assumption that the true model is both linear, which is not checkable and unlikely to be every true in practice. Because of the linearity, we also have the strong assumption of multicollinearity, which can be a problem if it is high (though not perfect). For example, if for some reason—by chance or by some other factor—that covariate x_1 and x_{12} were correlated, then we won't be able to separate them.

The next concentration result on bounded covariates does not assume anything except for iid data (distribution free) and essentially proves why Lasso regression works in high dimensions.

Theorem 5.5 (Concentration of Lasso)

Given (X, Y), assume that $|Y| \leq B$ and $\max_j |X_j| \leq B$. Let

$$\beta^* = \operatorname*{argmin}_{||\beta||_1 \leq L} R(\beta) \tag{139}$$

be the best sparse linear predictor in the L1 sense, where $R(\beta) = \mathbb{E}[(Y - \beta^T X)^2]$. Let our lasso estimator be

$$\hat{\beta} = \operatorname*{argmin}_{||\beta||_1 \leq L} \hat{r}(\beta) = \operatorname*{argmin}_{||\beta||_1 \leq L} \frac{1}{n} \sum_{i=1}^n (Y_i - \beta^T X_i)^2 \tag{140}$$

which minimizes the empirical risk. Then, with probability at least $1 - \delta$,

$$r(\hat{\beta}) \leq R(\beta^*) + \sqrt{\frac{16(L+1)^4 B^2}{n} \log\left(\frac{\sqrt{2}d}{\sqrt{\delta}}\right)} \tag{141}$$

Proof.

Let $Z = (Y, X)$ and $Z_i = (Y_i, X_i)$. Define $\gamma \equiv \gamma(\beta) = (-1, \beta)$. Then

$$r(\beta) = \mathbb{E}(Y - \beta^T X)^2 = \gamma^T \Lambda \gamma \tag{142}$$

where $\Lambda = \mathbb{E}[ZZ^T]$. Note that $\|\gamma\|_1 = \|\beta\|_1 + 1$. Let $\mathcal{B} = \{\beta : \|\beta\|_1 \leq L\}$. The training error is

$$\hat{r}(\beta) = \frac{1}{n}\sum_{i=1}^n (Y_i - X_i^T\beta)^2 = \gamma^T\hat{\Lambda}\gamma \tag{143}$$

where $\hat{\Lambda} = \frac{1}{n}\sum_{i=1}^n Z_iZ_i^T$. For any $\beta \in \mathcal{B}$,

$$\begin{aligned} |\hat{r}(\beta) - r(\beta)| &= |\gamma^T(\hat{\Lambda} - \Lambda)\gamma| && (144)\\ &\leq \sum_{j,k} |\gamma(j)||\gamma(k)||\hat{\Lambda}(j,k) - \Lambda(j,k)| \leq \|\gamma\|_1^2\delta_n && (145)\\ &\leq (L+1)^2\Delta_n && (146) \end{aligned}$$

where

$$\Delta_n = \max_{j,k} |\hat{\Lambda}(j,k) - \Lambda(j,k)|. \tag{147}$$

So,

$$r(\hat{\beta}) \leq \hat{r}(\hat{\beta}) + (L+1)^2\Delta_n \leq \hat{r}(\beta_*) + (L+1)^2\Delta_n \leq r(\beta_*) + 2(L+1)^2\Delta_n. \tag{148}$$

Note that $|Z(j)Z(k)| \leq B^2 < \infty$. By Hoeffding's inequality,

$$\mathbb{P}(\Delta_n(j,k) \geq \epsilon) \leq 2e^{-n\epsilon^2/(2B^2)} \tag{149}$$

and so, by the union bound,

$$\mathbb{P}(\Delta_n \geq \epsilon) \leq 2d^2e^{-n\epsilon^2/(2B^2)} = \delta \tag{150}$$

if we choose $\epsilon = \sqrt{(4B^2/n)\log\left(\frac{\sqrt{2}d}{\sqrt{\delta}}\right)}$. Hence,

$$r(\hat{\beta}) \leq r(\beta_*) + \sqrt{\frac{16(L+1)^4B^2}{n}\log\left(\frac{\sqrt{2}d}{\sqrt{\delta}}\right)} \tag{151}$$

with probability at least $1 - \delta$. □

Note that $\epsilon \approx \sqrt{\log d/n}$, and since it scales nicely with d, this is what allows us to do sparse regression in high dimensions.

5.4 Optimization

Soft Thresholding and Proximal Gradient Descent

Code 5.1 (MWS of Lasso Regression in scikit-learn)

```python
from sklearn.linear_model import Lasso

X = np.random.randn(10, 5)
y = np.random.randn(10)
# regularization parameter
model = Lasso(alpha=1e-1)
model.fit(X, y)
print(model.score(X, y))
print(model.intercept_)
print(model.coef_)
print(model.predict(np.array([[1, 2, 3, 4, 5]])))
```

```
0.47590269719236045
-0.8861298412689853
[0.          0.10767647
    0.24172197 0.7427863  0.
         ]
[3.02553422]
.
.
.
.
.
```

5.5 Sparse Minimax Estimators

Now let's talk about the big weakness of sparse estimators. Intuitively, sparsity isn't really smooth (a variable is either 0 or not 0), and this is at a high-level a bad thing. In fact, in practice sparse estimators may not work well. They are terrible minimax estimators.

Say that $\hat{\beta}$ is weakly sparsistent if, for every β,

$$P_\beta(I(\hat{\beta}_j = 0) \leq I(\beta_j = 0) \text{ for all } j) \to 1 \tag{152}$$

as $n \to \infty$. In particular, if $\hat{\beta}_n$ is sparsistent, then it is weakly sparsistent. Suppose that d is fixed. Then the least squares estimator $\hat{\beta}_n$ is minimax and satisfies

$$\sup_\beta E_\beta(n\|\hat{\beta}_n - \beta\|^2) = O(1). \tag{153}$$

But sparsistent estimators have much larger risk:

Theorem 5.6 (Leeb and Potscher, 2007)

Given a regression model $y = \beta^T x + \epsilon$, suppose that the following conditions hold:

1. d is fixed.
2. The covariates are nonstochastic and $n^{-1} X^T X \to Q$ for some positive definite matrix Q.
3. The errors ϵ_i are independent with mean 0, finite variance σ^2 and have a density f satisfying

$$0 < \int \left(\frac{f'(x)}{f(x)} \right)^2 f(x) dx < \infty. \tag{154}$$

If $\hat{\beta}$ is weakly sparsistent then

$$\sup_\beta E_\beta(n\|\hat{\beta}_n - \beta\|^2) \to \infty. \tag{155}$$

More generally, if L is any nonnegative loss function then

$$\sup_\beta E_\beta(L(n^{1/2}(\hat{\beta}_n - \beta))) \to \sup_s L(s). \tag{156}$$

Proof.

Choose any $s \in \mathbb{R}^d$ and let $\beta_n = -s/\sqrt{n}$. Then,

$$\sup_\beta E_\beta(L(n^{1/2}(\hat{\beta} - \beta))) \geq E_{\beta_n}(L(n^{1/2}(\hat{\beta} - \beta))) \geq E_{\beta_n}(L(n^{1/2}(\hat{\beta} - \beta)))I(\hat{\beta} = 0) \tag{157}$$

$$= L(-\sqrt{n}\beta_n)P_{\beta_n}(\hat{\beta} = 0) = L(s)P_{\beta_n}(\hat{\beta} = 0). \tag{158}$$

Now, $P_0(\hat{\beta} = 0) \to 1$ by assumption. It can be shown that we also have $P_{\beta_n}(\hat{\beta} = 0) \to 1$.[2] Hence, with probability tending to 1,

$$\sup_\beta E_\beta(L(n^{1/2}(\hat{\beta} - \beta))) \geq L(s). \tag{159}$$

Since s was arbitrary the result follows. □

6 Robust Regression

7 Bayesian Linear Regression

7.1 Regularization with Priors

We will now demonstrate how having a normal $\alpha\mathbf{I}$ prior around the origin in a Bayesian setting is equivalent to having a ridge penalty of $\lambda = \sigma^2/\alpha^2$ in a frequentist setting. If we have a Gaussian prior of form

$$p(\mathbf{w} \mid \alpha^2) = N(\mathbf{w} \mid \mathbf{0}, \alpha^2 \mathbf{I}) = \bigg(\frac{1}{2\pi\alpha^2}\bigg)^{M/2} \exp\bigg(- \frac{1}{2\alpha^2} ||\mathbf{w}||_2^2 \bigg)$$

We can use Bayes rule to compute

$$\begin{aligned} p(\mathbf{w} \mid \mathbf{X}, \mathbf{Y}, \alpha^2, \sigma^2) & \propto p(\mathbf{Y} \mid \mathbf{w}, \mathbf{X}, \alpha^2, \sigma^2)\, p(\mathbf{w} \mid \mathbf{X}, \alpha^2, \sigma^2) \\ & = \bigg[\prod_{n=1}^N p(y^{(n)} \mid \mathbf{w}, \mathbf{x}^{(n)}, \alpha^2, \sigma^2) \bigg] p(\mathbf{w} \mid \mathbf{X}, \alpha^2, \sigma^2) \\ & = \bigg[\prod_{n=1}^N \frac{1}{\sqrt{2\pi\sigma^2}} \exp\bigg(- \frac{(y^{(n)} - h_{\mathbf{w}}(x^{(n)}))^2}{2\sigma^2} \bigg) \bigg] \cdot \bigg(\frac{1}{2\pi\alpha^2}\bigg)^{M/2} \exp\bigg(- \frac{1}{2\alpha^2} ||\mathbf{w}||_2^2 \bigg) \end{aligned}$$

and taking the negative logarithm gives us

$$\ell(\mathbf{w}) = \frac{1}{2\sigma^2} \sum_{n=1}^N \big(y^{(n)} - h_{\mathbf{w}}(\mathbf{x}^{(n)}) \big)^2 + \frac{N}{2} \ln \sigma^2 + \frac{N}{2} \ln(2\pi) - \frac{M}{2} \ln(2\pi\alpha^2) + \frac{1}{2\alpha^2} ||\mathbf{w}||_2^2$$

taking out the constant terms relative to $\mathbf{w}$ and multiplying by $2\sigma^2$ (which doesn't affect optima) gives us the ridge penalized error with a penalty term of $\lambda = \sigma^2/\alpha^2$.

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \big(y^{(n)} - h_{\mathbf{w}}(\mathbf{x}^{(n)}) \big)^2 + \frac{\sigma^2}{\alpha^2} ||\mathbf{w}||_2^2$$

But minimizing this still gives a point estimate of $\mathbf{w}$, which is not the full Bayesian treatment. In a Bayesian setting, we are given the training data $(\mathbf{X}, \mathbf{Y})$ along with a new test point $\mathbf{x}'$ and want to evaluate the predictive distribution $p(y \mid \mathbf{x}', \mathbf{X}, \mathbf{Y})$. We can do this by integrating over $\mathbf{w}$.

$$\begin{aligned} p(y \mid \mathbf{x}', \mathbf{X}, \mathbf{Y}) & = \int p(y \mid \mathbf{x}', \mathbf{w}, \mathbf{X}, \mathbf{Y})\, p(\mathbf{w} \mid \mathbf{x}', \mathbf{X}, \mathbf{Y})\, d\mathbf{w} \\ & = \int p(y \mid \mathbf{x}', \mathbf{w})\, p(\mathbf{w} \mid \mathbf{X}, \mathbf{Y})\, d\mathbf{w} \end{aligned}$$

where we have omitted the irrelevant variables, along with α^2 and σ^2 to simplify notation. By substituting the posterior $p(\mathbf{w} \mid \mathbf{X}, \mathbf{Y})$ with a normalized version of our calculation above and by noting that

$$p(y \mid \mathbf{x}', \mathbf{w}) = N(y \mid h_{\mathbf{w}}(\mathbf{x}'), \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\bigg(- \frac{\big(y - h_{\mathbf{w}}(\mathbf{x}')\big)^2}{2\sigma^2} \bigg)$$

Now this integral may or may not have a closed form, but if we consider the polynomial regression with the hypothesis function of form

$$h_{\mathbf{w}}(x) = w_0 + w_1 x + w_2 x^2 + \ldots + w_{M-1} x^{M-1}$$

then this integral turns out to have a closed form solution given by

$$p(y \mid \mathbf{x}', \mathbf{X}, \mathbf{Y}) = N\big(y \mid m(x'), s^2(x')\big)$$

where

$$\begin{aligned} m(x') &= \frac{1}{\sigma^2} \boldsymbol{\phi}(x')^T \mathbf{S} \bigg(\sum_{n=1}^N \boldsymbol{\phi}(x^{(n)}) y^{(n)} \bigg) \\ s^2(x') &= \sigma^2 + \boldsymbol{\phi}(x')^T \mathbf{S} \boldsymbol{\phi}(x') \\ \mathbf{S}^{-1} &= \alpha^{-2} \mathbf{I} + \frac{1}{\sigma^2} \sum_{n=1}^N \boldsymbol{\phi}(x^{(n)}) \boldsymbol{\phi}(x')^T \end{aligned}$$

and $\boldsymbol{\phi}(x)$ is the vector of functions $\phi_i(x) = x^i$ from $i = 0, \ldots, M-1$.

Bibliography

[GKKW02] L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A Distribution-Free Theory of Nonparametric Regression.* Springer Series in Statistics. Springer New York, 2002.

[HKZ14] Daniel Hsu, Sham M. Kakade, and Tong Zhang. Random design analysis of ridge regression, 2014.