Introduction, Notation, and Overview

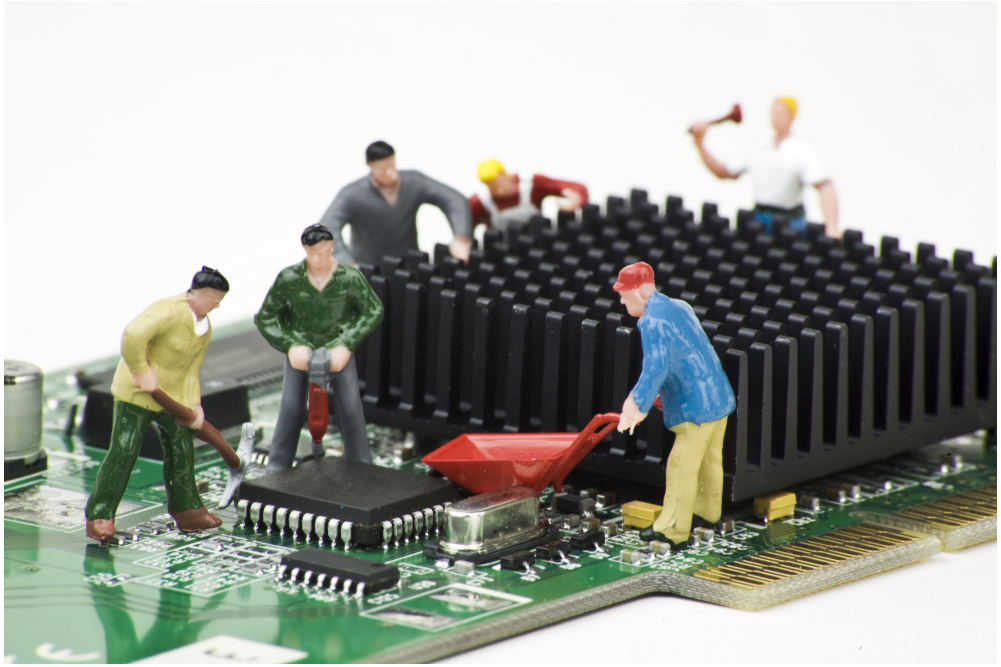
-Introduction to Data Science-

ISL: Chapter 2

Lecturer: Darren Homrighausen, PhD

Preamble:

- Define data science
- Go over terminology and introduce notation
- Outline a general framework for assessing the quality of an estimator/algorithm
- Cover the singular value decomposition (SVD)



DATA SCIENCE is about using data to ...

- ... discover structure
- ... glean non-obvious insights into a problem
- ... make predictions about unknown quantities

Class Overview

Practically speaking, this means we seek to:

- find relationships in data that give good predictive performance
- reduce the size of the group of variables for scientific, statistical, or computational purposes

and, perhaps most importantly..

Know the techniques, how they work, when they apply, and how to implement them

Class Outline

Over the next semester we will address a variety of topics

(The specific topics will be a function of how well I feel the materials are being handled and student requests)

For sure, we will cover

1. Techniques for model selection, regularization, and dimension reduction
2. Supervised, unsupervised, and semi-supervised methods
3. Kernelization and dimension expansion
4. Algorithms for large data analysis

   (In particular, the fundamental notion of convex vs. non-convex optimization)

This course will emphasize methods and applications over theory

(Major caveat: in my experience, one person's theory is another's application..)

References:

Main references:

- *An Introduction to Statistical Learning with Applications in R* (James, Witten, Hastie, Tibshirani)
- *The Elements of Statistical Learning* (Hastie, Tibshirani, Friedman)

(We will refer to these as ISL and ESL, respectively)

Secondary references:

- *Computer Age Statistical Inference* (Efron, Hastie)
- *Statistics for High-Dimensional Data: Methods, Theory and Applications* (Bühlmann, Van de Geer

  (Mostly the discussion in their Chapter 2 about variations on the lasso)
- Topic specific notes or lectures

Wait, statistical learning, what's that?

A harrowing aspect of data science: it's an academic chimera

Data science is in the union of computer science, statistics, mathematics, databases, library science, ...

Data science

Due to this multi-origin origin, data science goes by many names:

- business analytics
- statistical (machine) learning
- statistics
- data mining
- others?

and is often expressed as a formula such as

$$\text{Data science} = \text{Applied Statistics} + \text{Tech field}$$

or what it isn't

$$\text{Data science} \neq \text{Statistical Machine Learning} + \text{Programming}$$

(Caveat: I don't necessarily believe either of these statements)

My research is in statistical (machine) learning hence I'll be talking mainly from that perspective

Statistical learning terminology

Introduction

Statistical (Machine) Learning (SML) is statistics with a focus on prediction, scalability, and high dimensional problems

Regression: predict $Y \in \mathbb{R}$ from covariates or features X

Classification: predict $Y \in \{1, 2, \ldots, G\}$ from X

(Here, the labels or classes of Y are arbitrary)

Finding structure:

- Finding groups or clusters in the data
- Dimension reduction
- Graphical models (conditional independence structure)

Some main themes: Assumptions

What assumptions are needed to motivate the method or guarantee some property?

Example: Suppose I observe some data $Y_1, \ldots, Y_n \in \mathbb{R}$

I want to make a prediction about a new observation Y_{n+1}

I could use $\overline{Y} = \frac{1}{n} \sum_{i=1}^{n} Y_i$

I'm (implicitly?) assuming that the ...

- ...expectations are all (nearly) the same:

$$\mathbb{E}Y_1 \approx \mathbb{E}Y_2 \approx \cdots \approx \mathbb{E}Y_n$$

- ...observations have covariance (nearly) equal to zero
- ...observations all have (nearly) the same variance
- ...probability of Y_i being far from $\mathbb{E}Y_i$ is small

Some main themes: Convexity

Convex problems can be solved efficiently. If necessary, we try to approximate nonconvex problems with convex ones

- Convex set: A set B is convex if for any $\beta, \beta' \in B$ and any $\tau \in [0, 1]$

$$\tau\beta + (1 - \tau)\beta' \in B$$

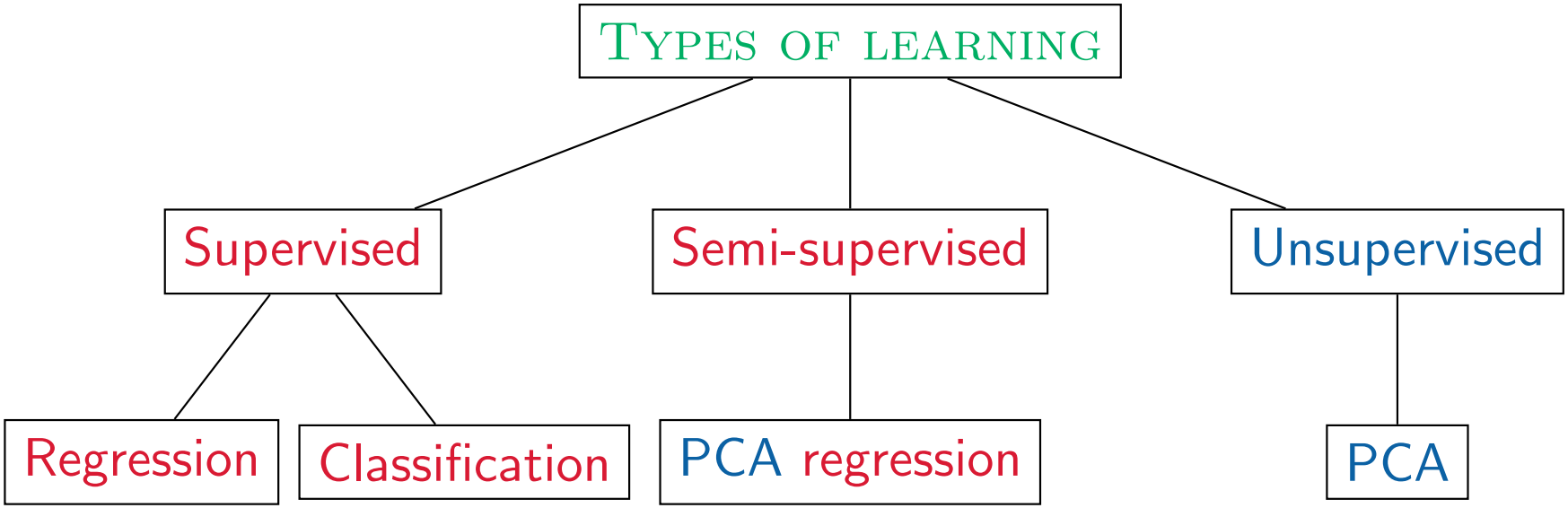
- Convex function: A function ℓ is convex if the "area" above the function is a convex set

  (This area is formally known as the epigraph of ℓ)

  This is the same as: for any $\beta, \beta' \in \mathbb{R}^p$ and any $\tau \in [0, 1]$

$$f(\tau\beta + (1 - \tau)\beta') \leq \tau f(\beta) + (1 - \tau) f(\beta')$$

  (Convex functions cannot have multiple local minima)



Some comments:

- Comparing predictions to Y gives a natural notion of prediction accuracy
- Much more heuristic, unclear what a good solution would be. We'll return to this later in the semester.

Supervised Methods

The set-up

We observe n pairs of data $(X_1^\top, Y_1)^\top, \ldots, (X_n^\top, Y_n)^\top$

Let[1] $Z_i^\top = (X_i^\top, Y_i) \in \mathbb{R}^p \times \mathbb{R}$

We'll refer to the training data as $\mathcal{D} = \{Z_1, \ldots, Z_n\}$

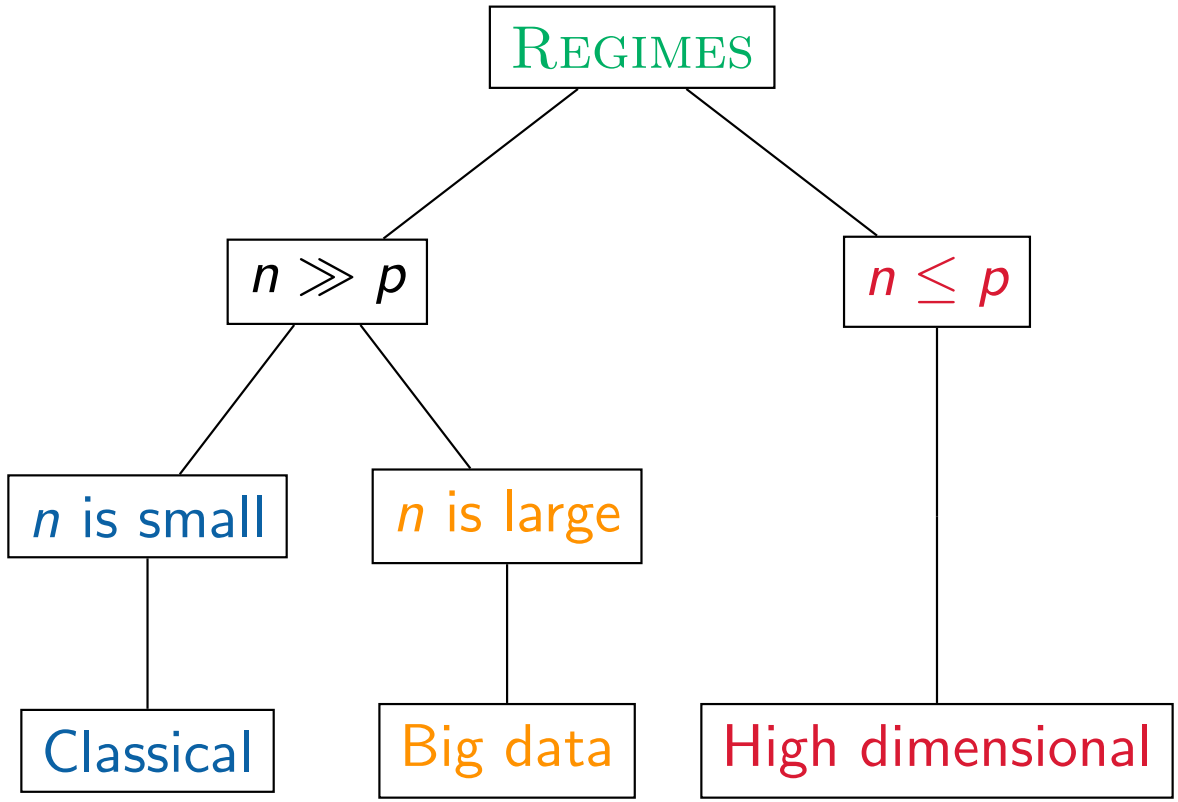
- Y_i the supervisor or response
  (NOT DEPENDENT VARIABLE)
- $X_i \in \mathbb{R}^p$ is the feature or covariate (vector)
  (or explanatory variables or predictors. NOT INDEPENDENT VARIABLES)

Example: Y_i is whether a threat is detected in an image and the X_{ij} is the value at the j^{th} pixel of an image (p might be $1024^2 = 1048576$)

[1] These transposes get tiredsome. We'll get a bit sloppy and drop them selectively in what follows.

The set-up

There are roughly three regimes of interest, assuming $\mathbb{X} \in \mathbb{R}^{n \times p}$



(We will return to these in more detail in the next set of lectures)

The set-up

We use the training data $\mathcal{D}$ to train an algorithm, producing a function $\hat{f} : \mathbb{R}^p \to \mathbb{R}$

Goal: Given a new $X \in \mathbb{R}^p$, we want to form predictions

$$\hat{f}(X) = \hat{Y}$$

Such that $\hat{Y}$ is a good prediction of Y, the unobserved supervisor

Example: Classically, this is often done with maximum likelihood

- likelihood ℓ

  (Ex: $\ell(\pi, Y) = \pi^Y(1-\pi)^{1-Y}$ is the Bernoulli likelihood for one observation)
- which is a function of a parameter θ and training data $\mathcal{D}$

$$\hat{\theta} = \arg\max_{\theta} \prod_{i=1}^{n} \ell(\theta, Z_i)$$

$$\Rightarrow \hat{Y} = \hat{f}(X) = \arg\max_{Y} \prod_{i=1}^{n} \ell(\hat{\theta}, Z_* = (X, Y))$$

Risk, Bayes, bias, variance, and approximation

Loss functions and risk

If we want a $\hat{f}(X)$ which is a good prediction, what does good mean?

Define a loss function which

- Inputs both
  - $\underbrace{\hat{f}(X)}_{\text{Our prediction}}$
  - $\underbrace{Y}_{\text{Unknown, true value}}$
- Outputs a number $\ell(\hat{f}(X), Y)$ between 0 and ∞...

...such that smaller $\ell(\hat{f}(X), Y)$ indicate better performance

(There is an intimate connection between loss and likelihoods, hence same notation)

Risky (and lossy) business

Any distance function could serve for the loss function ℓ

As both $\hat{f}(X)$ and Y are random, the loss function is random

Hence, we define the (prediction) risk to be the expectation of the loss

$$R(f) = \mathbb{E}\ell(f(X), Y)$$

(Hence, the risk is not random)

Definition: A good procedure f is one that has a small risk $R(f)$

Risky (and lossy) business

More Details: If we want the procedure with small risk it begs the question

$\rightarrow$ What procedure has the smallest risk?

The (unknown) function f_* with the smallest risk is known as the Bayes rule with respect to the loss function ℓ

$$f_* = \operatorname*{argmin}_f R(f) \qquad \text{and} \qquad \min_f R(f) = R(f_*)$$

(Warning: I will use argmin and min a lot in this class)

An example: Squared-error loss

An example: Squared-error loss

If the function $\ell(f(X), Y) = (f(X) - Y)^2$, then

$$f_*(X) = \mathbb{E}[Y|X]$$

This is known as the regression function; that is, the conditional expectation of Y given X.

(Emphasis: This is the Bayes rule with respect to the squared error loss function)

Example: In simple linear regression, the Bayes rule is modeled as

$$f_*(X) = \beta_0 + \beta_1 X$$

Giving rise to the model

$$Y = \beta_0 + \beta_1 X + \epsilon$$

where ϵ is some mean zero fluctuation

An example: Squared-error loss

Recap:

Given the training data $\mathcal{D}$, we want to predict some independent test data $Z = (X, Y)$

This means forming a $\hat{f}$, which is a function of both X and the training data $\mathcal{D}$, which provides predictions $\hat{Y} = \hat{f}(X)$.

The quality of this prediction is measured via the prediction risk

$$R(\hat{f}) = \mathbb{E}(Y - \hat{f}(X))^2$$

We know that the regression function, $f_*(X) = \mathbb{E}[Y|X]$, is the best possible prediction

However, as previously mentioned, it is *unknown*

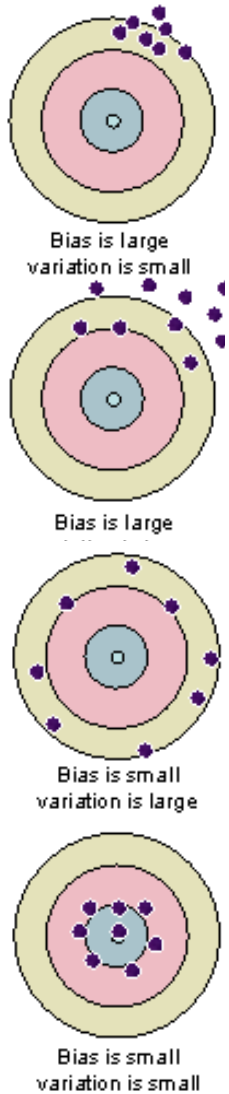
An example: Squared-error loss

Note that squared prediction risk at any X can be written as

$$\mathbb{E}(\hat{f}(X) - Y)^2 = \text{bias}^2(X) + \text{var}(X) + \sigma^2$$

where

$$\begin{aligned} \text{bias}(X) &= \mathbb{E}\hat{f}(X) - f_*(X) \\ \text{var}(X) &= \mathbb{V}\hat{f}(X) = \mathbb{E}(\hat{f}(X) - \mathbb{E}\hat{f}(X))^2 \\ \sigma^2 &= \mathbb{E}(Y - f_*(X))^2 = \mathbb{V}Y \end{aligned}$$



Bias-variance tradeoff

This can be heuristically thought of as

$$\text{Prediction risk} = \text{Bias}^2 + \text{Variance} + \text{Irreducible error}$$

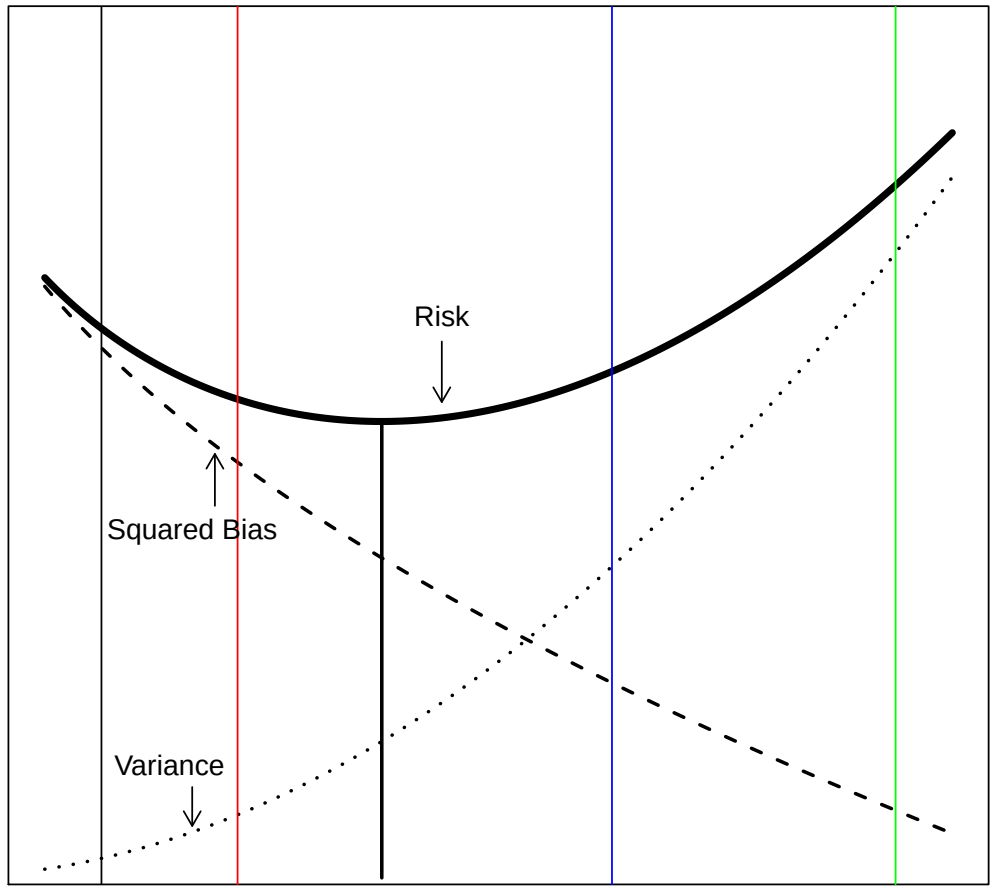
There is a natural conservation between these quantities

Low bias $\rightarrow$ complex model $\rightarrow$ many parameters $\rightarrow$ high variance

The opposite also holds
(Think: $\hat{f} \equiv 0$.)

We'd like to 'balance' these quantities to get the best possible predictions

Bias-variance tradeoff



Model Complexity ↗

The main idea and the main problem

Main idea and problem

In a certain sense, we are done: minimize $R(f)$ over the types of f we are willing to consider

(i.e.: over all $f(X) = X^\top \beta$)

Problem: we never know the distribution of (X, Y)!

Not only is the Bayes rule unknown, but the risk itself is as well!

$$R(f) = \underbrace{\mathbb{E}}_{\text{unknown!}} \Big[\ell(f(X), Y) \Big]$$

Every (supervised) procedure we discuss provides a model/algorithm for estimating some aspect of the distribution of (X, Y) using $\mathcal{D}$

Training error and risk estimation

Since we want to minimize $R(f)$, which is an expectation, perhaps we can approximate it with an average

For any loss function $\ell(f(X), Y)$, we can form the training error

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^{n} \ell(f(X_i), Y_i)$$

In many applied statistical applications, this plug-in estimator of the risk is used

(Think: how many techniques rely on an unconstrained minimization of squared error, or maximum likelihood, or estimating equations, or ...)

This sometimes has disastrous results

Example

Let's look at the regression version: mean squared error (MSE)

$$\hat{R}(f) = \frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2$$

Let's suppose $\mathcal{D}$ is drawn from

```
n = 30
X = (0:n)/n*2*pi
Y = sin(X) + rnorm(n,0,.25)
```

Example

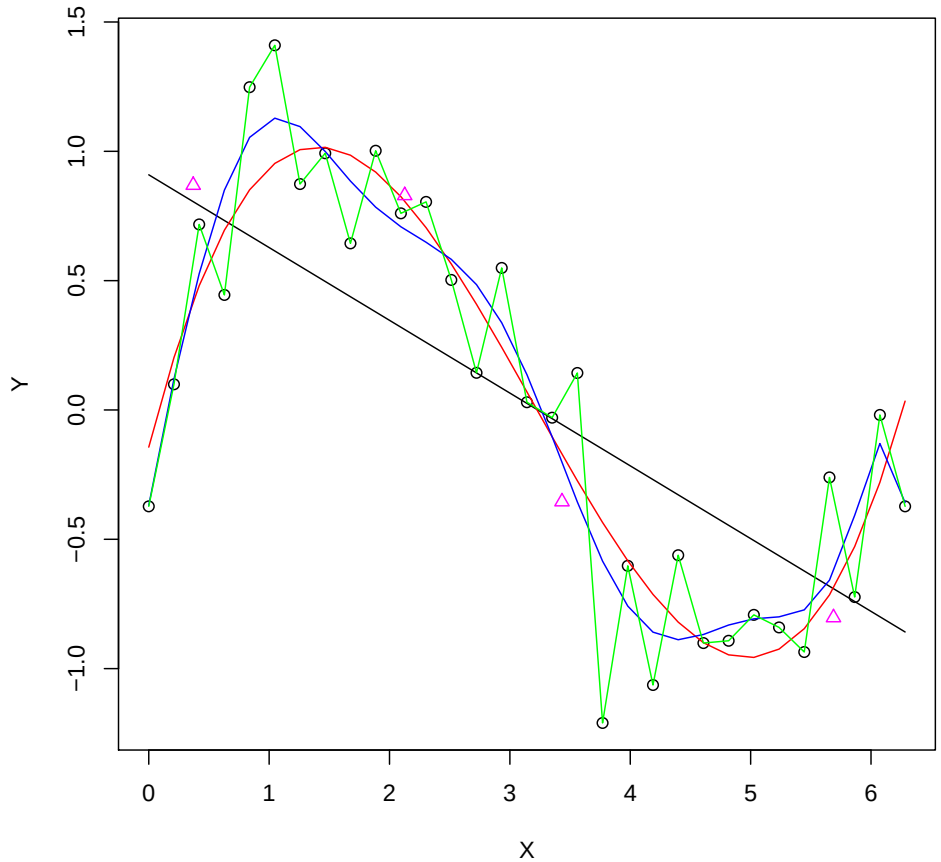
Now, let's fit some polynomials to this data.

We consider the following models:

- Model 1: $f(X_i) = \beta_0 + \beta_1 X_i$
- Model 2: $f(X_i) = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \beta_3 X_i^3$
- Model 3: $f(X_i) = \sum_{j=0}^{10} \beta_j X_i^j$
- Model 4: $f(X_i) = \sum_{j=0}^{n-1} \beta_j X_i^j$

Let's look at what happens...

Example



The $\hat{R}$'s are:

$\hat{R}(\text{Model 1}) = 10.98$

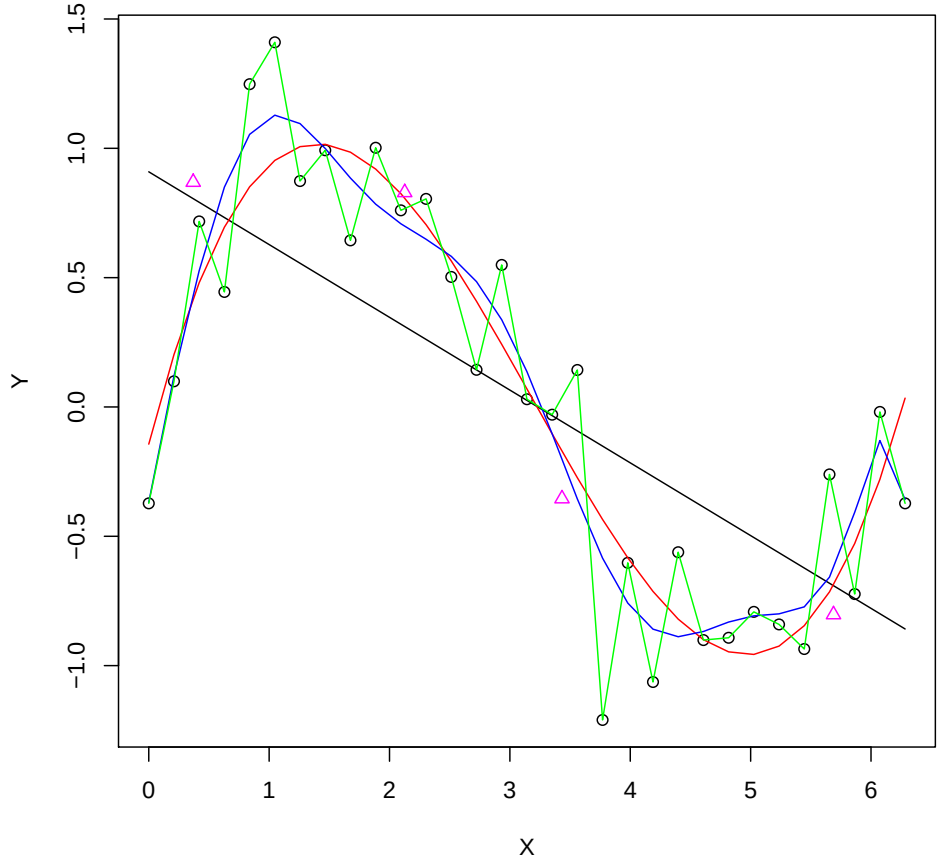
$\hat{R}(\text{Model 2}) = 2.86$

$\hat{R}(\text{Model 3}) = 2.28$

$\hat{R}(\text{Model 4}) = 0$

What about predicting new observations (Δ)?

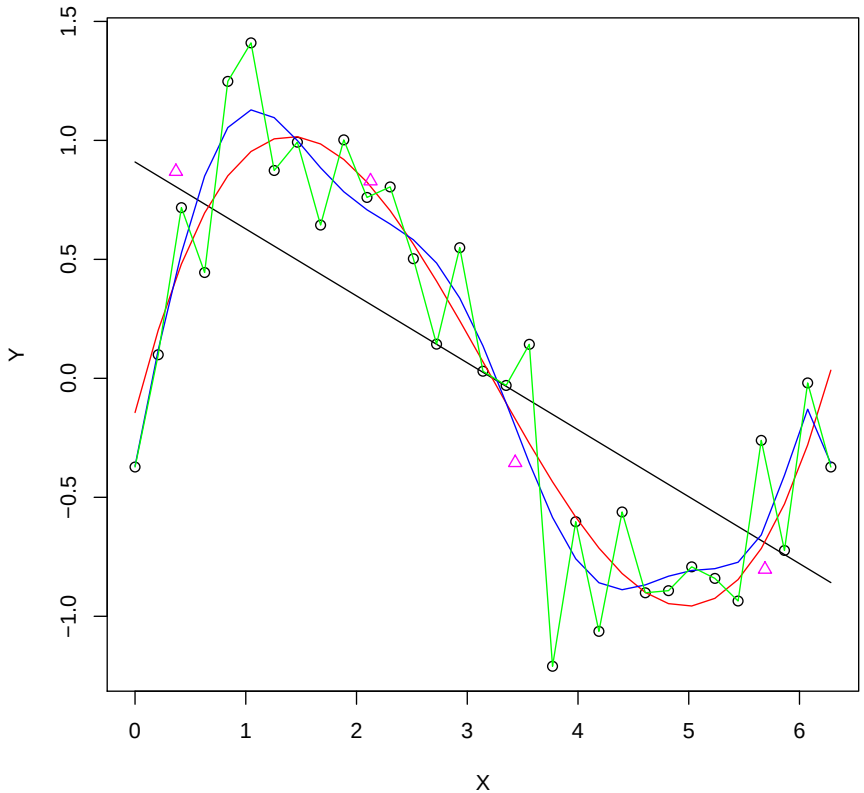
Example

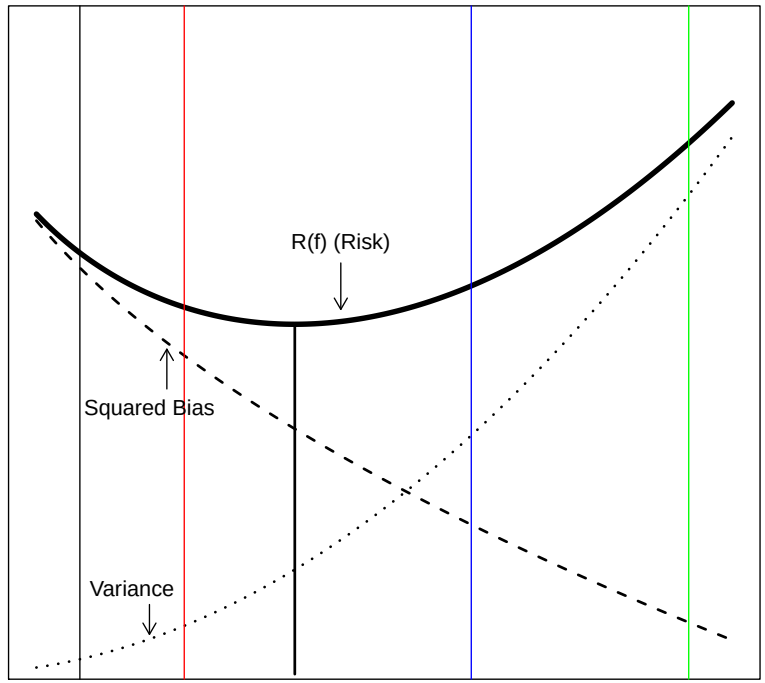


- Black model has low variance, high bias
- Green model has low bias, but high variance
- Red model and Blue model have intermediate bias and variance.

We want to balance these two quantities.

Bias vs. Variance





Model Complexity ↗

Background

Background

- We will write vectors as

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$$

  We write this as $z \in \mathbb{R}^n$, which is "z is a member of ar-en."
- We commonly will need to "turn" the vector, which we write as

$$z^\top = \begin{bmatrix} z_1 & z_2 & \dots & z_n \end{bmatrix}$$

Necessary Background: Notation

We will concatenate the covariates or features into the design or feature matrix $\mathbb{X}$, where

$$\mathbb{X} = \begin{bmatrix} x_1 & x_2 & \cdots & x_p \end{bmatrix} = \begin{bmatrix} X_1^\top \\ X_2^\top \\ \vdots \\ X_n^\top \end{bmatrix} \in \mathbb{R}^{n \times p}$$

In words: The features (columns) will be lower case letters and the observations (rows) will be upper case letters

Necessary Background: Lengths

We will need to measure the size of both vectors and matrices.

The most common is the one we use every day Euclidean distance
(Think: the Pythagorean theorem)

$$||x||_2 = \sqrt{\sum_{k=1}^{p} x_k^2}$$

We call this a norm and refer to this as the "ell two norm"

Additionally, we will need the Manhattan distance

$$||x||_1 = \sum_{k=1}^{p} |x_k|$$

We call this the "ell one norm"

Singular Value Decomposition (SVD)

SVD

A huge amount of statistics depends on (numerical) linear algebra concepts

Many, many topics in (numerical) linear algebra are implicitly motivated by the singular value decomposition (SVD)

The SVD is a generalization of the eigenvector decomposition

Instead of

$$\mathbb{X} = UDU^\top \longleftarrow \text{eigenvector decomposition}$$

we get

$$\mathbb{X} = UDV^\top \longleftarrow \text{singular value decomposition}$$

This change makes the (unique) SVD always exist

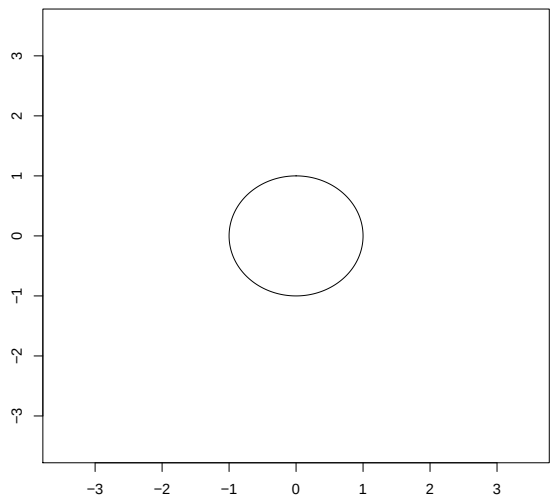
SVD

It turns out we can think of matrix multiplication in terms of circles and ellipsoids

Take a matrix $\mathbb{X}$ and let's look at the set of vectors

$$B = \{\beta : ||\beta||_2 \leq 1\}$$

This is a circle!

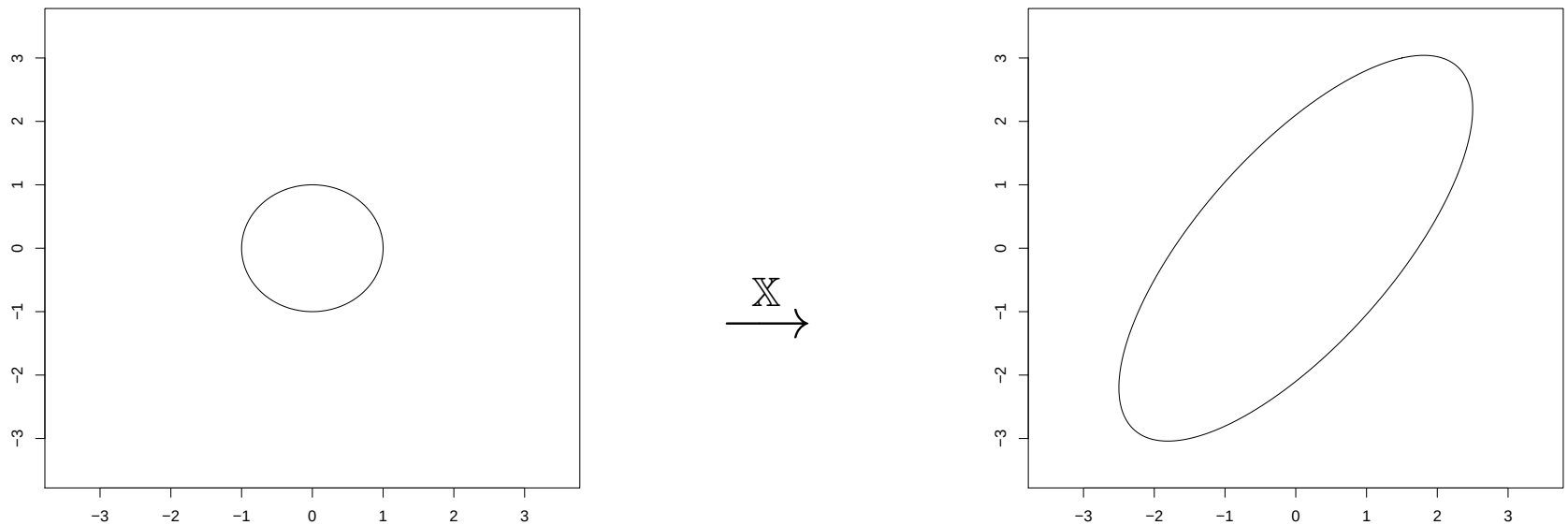


SVD

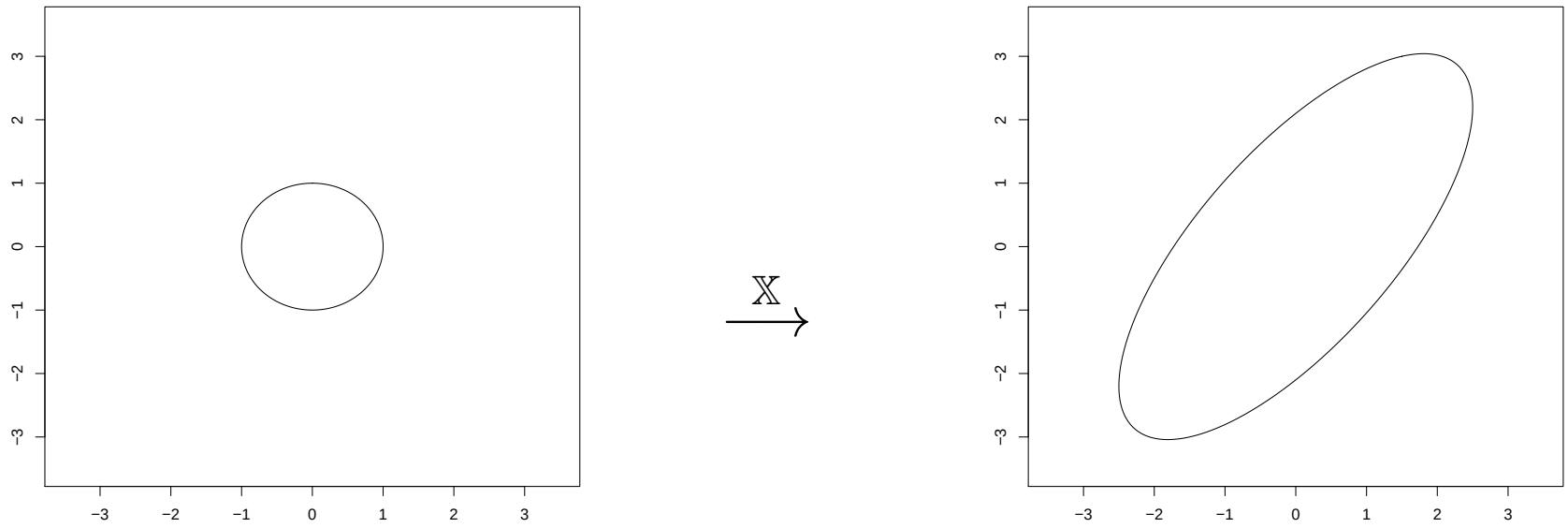
What happens when we multiply vectors in this circle by $\mathbb{X}$?

Let

$$\mathbb{X} = \begin{bmatrix} 2.0 & 0.5 \\ 1.5 & 3.0 \end{bmatrix} \text{ and } \mathbb{X}\beta = \begin{bmatrix} 2\beta_1 + 0.5\beta_2 \\ 1.5\beta_1 + 3\beta_2 \end{bmatrix}$$



SVD

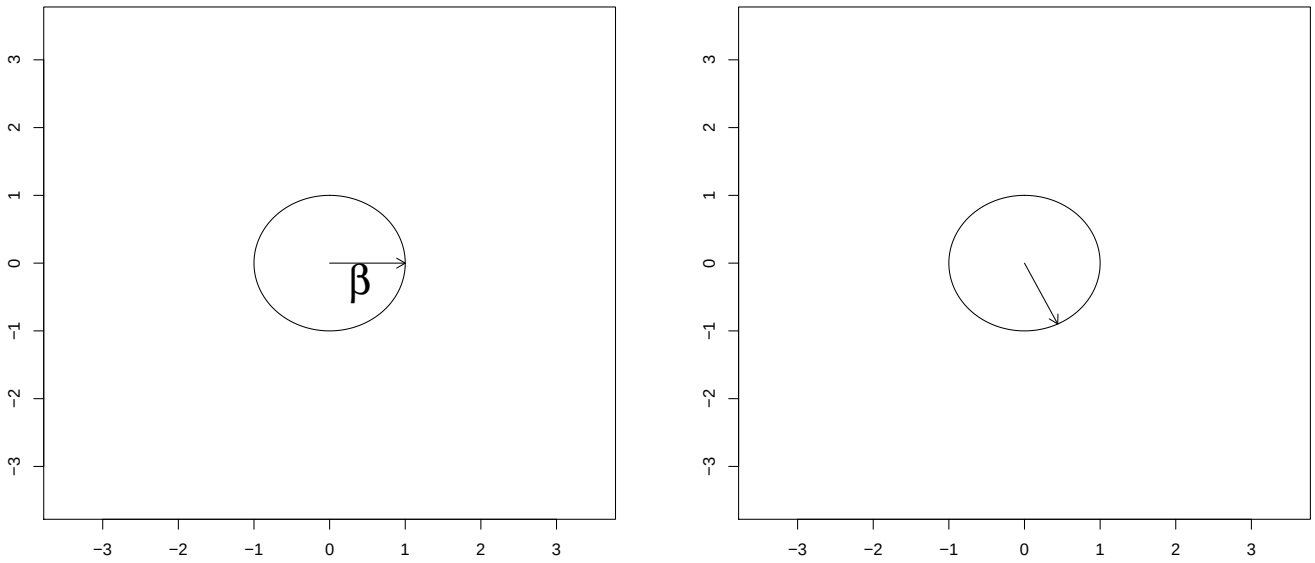


What happened?

1. The coodinate axis gets rotated
2. The new axis gets elongated (making an ellipse)
3. This ellipse gets rotated

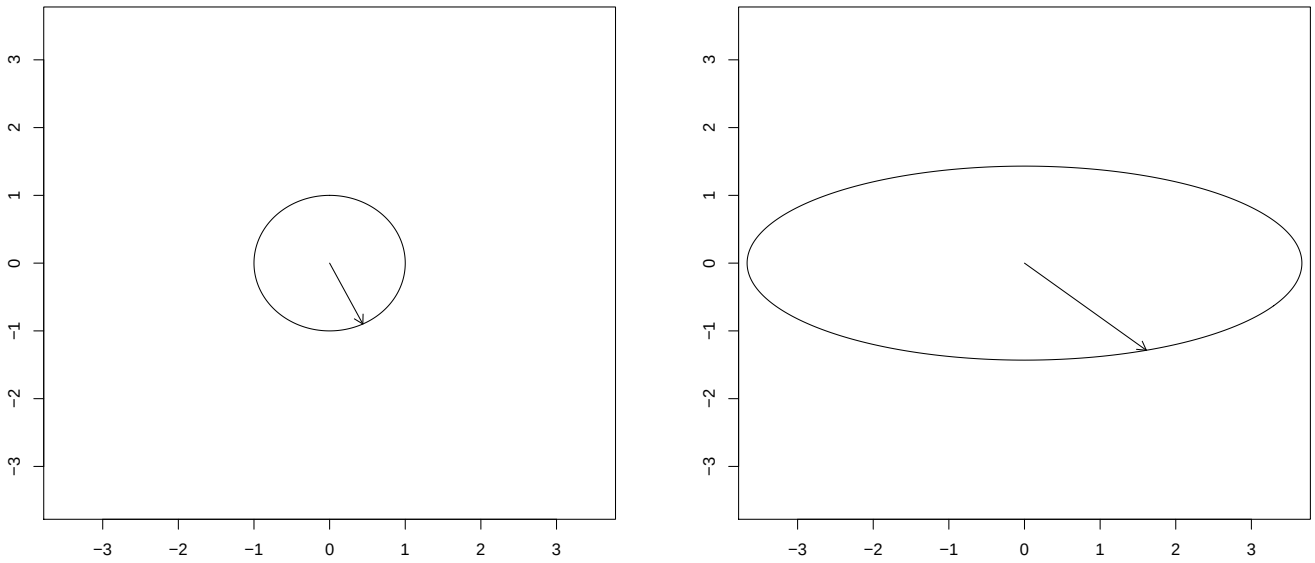
Let's break this down into parts...

SVD



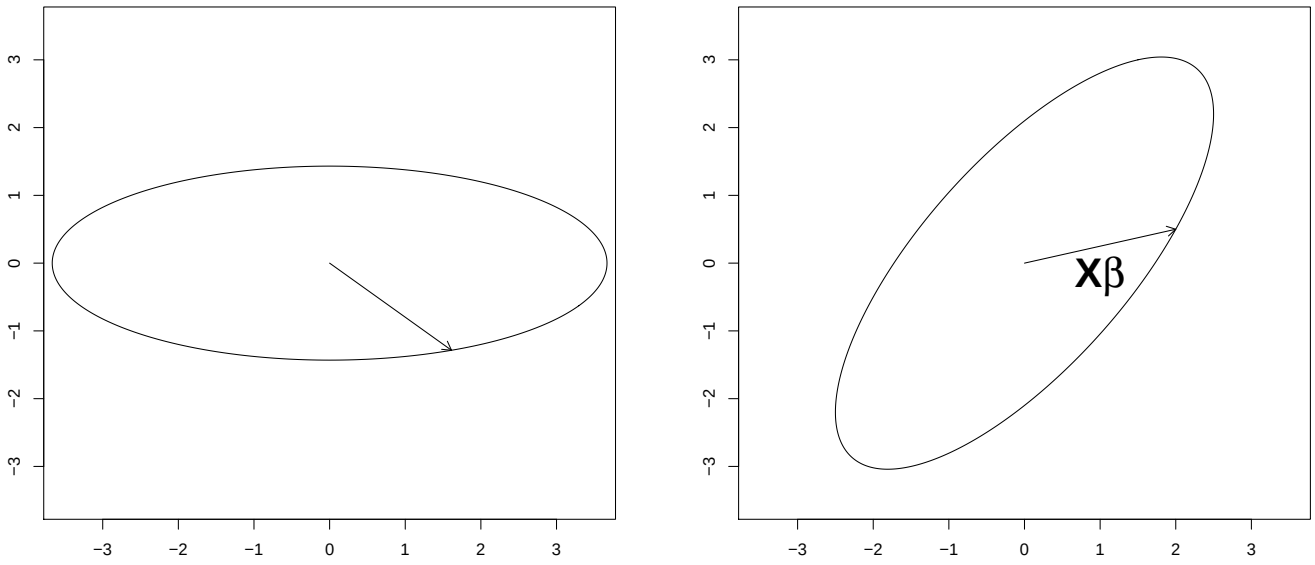
1. The coordinate axis gets rotated

SVD



1. The coordinate axis gets rotated
2. The new axis gets elongated (making an ellipse)

SVD



1. The coordinate axis gets rotated
2. The new axis gets elongated (making an ellipse)
3. This ellipse gets rotated

Necessary Background: Rotation

Rotations: These can be thought of as just reparameterizing the coordinate axis. This means that they don't change the geometry.

As the original axis was orthogonal (that is; perpendicular), the new axis must be as well.

Necessary Background: Rotation

Let v_1, v_2 be two normalized, orthogonal vectors. This means that:

$$v_1^\top v_2 = 0 \quad \text{and} \quad v_1^\top v_1 = v_2^\top v_2 = 1$$

In matrix notation, if we create V as a matrix with normalized, orthogonal vectors as columns, then:

$$V^\top V = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ & & \vdots & & \\ 0 & 0 & 0 & \dots & 1 \end{bmatrix} = I$$

Here, I is the identity matrix.

Necessary Background: Elongation

Elongation: These can be thought of as stretching vectors along the current coordinate axis. This means that they **do** change the geometry by distorting distances.

Elongations are the result of multiplication by a diagonal matrix (note: we just saw a very special case of such a matrix: the identity matrix I)

All diagonal matrices have the form:

$$D = \begin{bmatrix} d_1 & 0 & 0 & \dots & 0 \\ 0 & d_2 & 0 & \dots & 0 \\ & & \vdots & & \\ 0 & 0 & 0 & \dots & d_p \end{bmatrix}$$

SVD

Using this intuition, for any matrix $\mathbb{X}$ it is possible to write its SVD:

$$\mathbb{X} = UDV^\top$$

where

- U and V are orthogonal (think: rotations)
- D is diagonal (think: elongation)
- The diagonal elements of D are ordered as

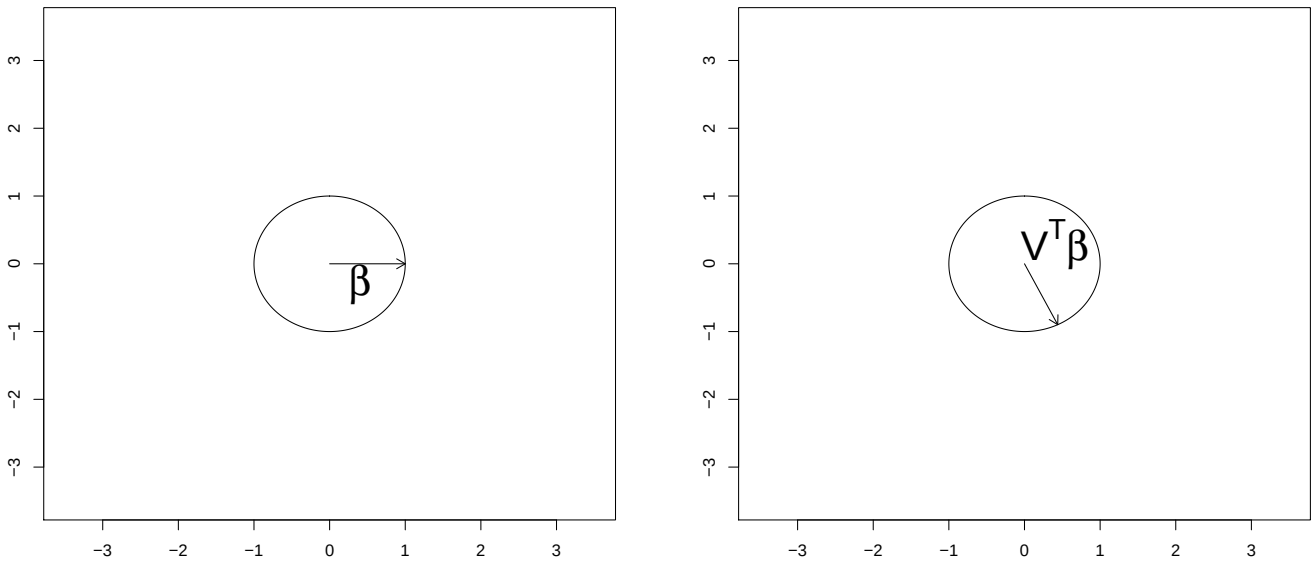
$$d_1 \geq d_2 \geq \ldots \geq d_p \geq 0$$

Many properties of matrices can be 'read off' from the SVD.

Rank: The rank of a matrix answers the question: how many dimensions does the ellipse live in? In other words, it is the number of columns of the matrix $\mathbb{X}$, not counting the columns that are 'redundant'

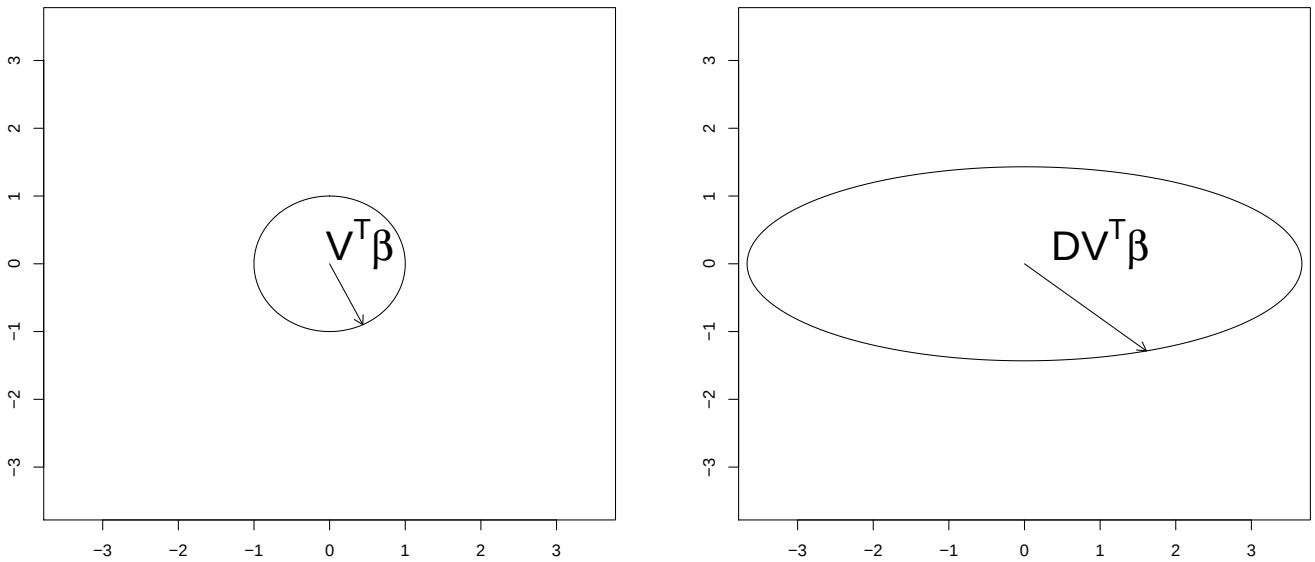
It turns out the rank is exactly the quantity q such that $d_q > 0$ and $d_{q+1} = 0$

SVD



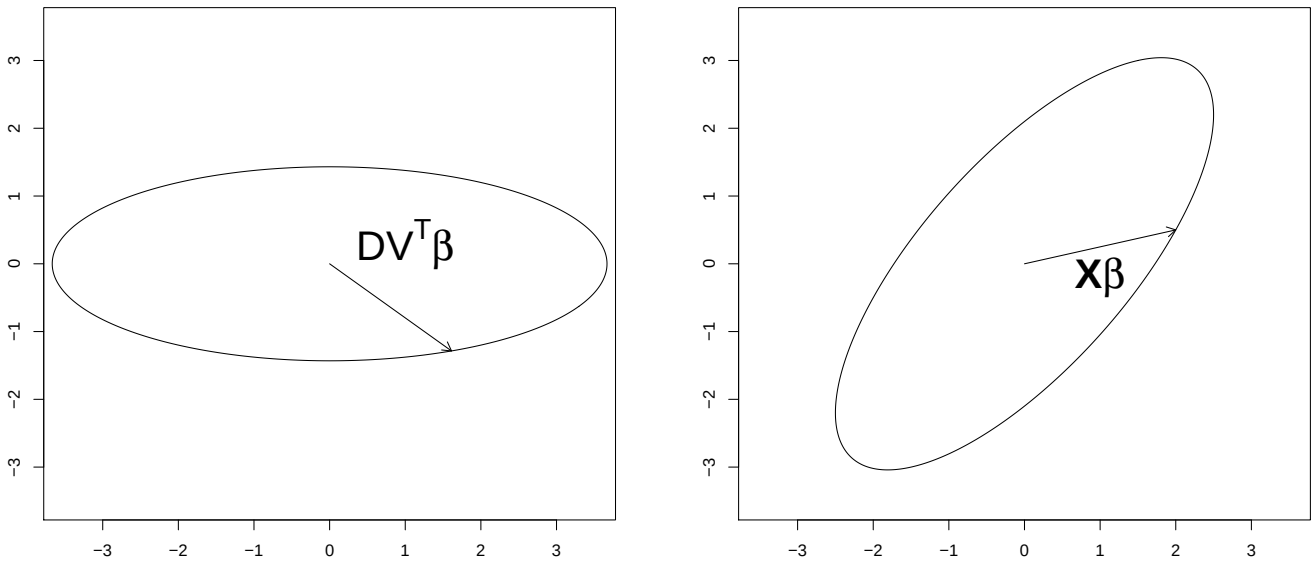
1. The coordinate axis gets rotated (Multiplication by $V^\top$)

SVD



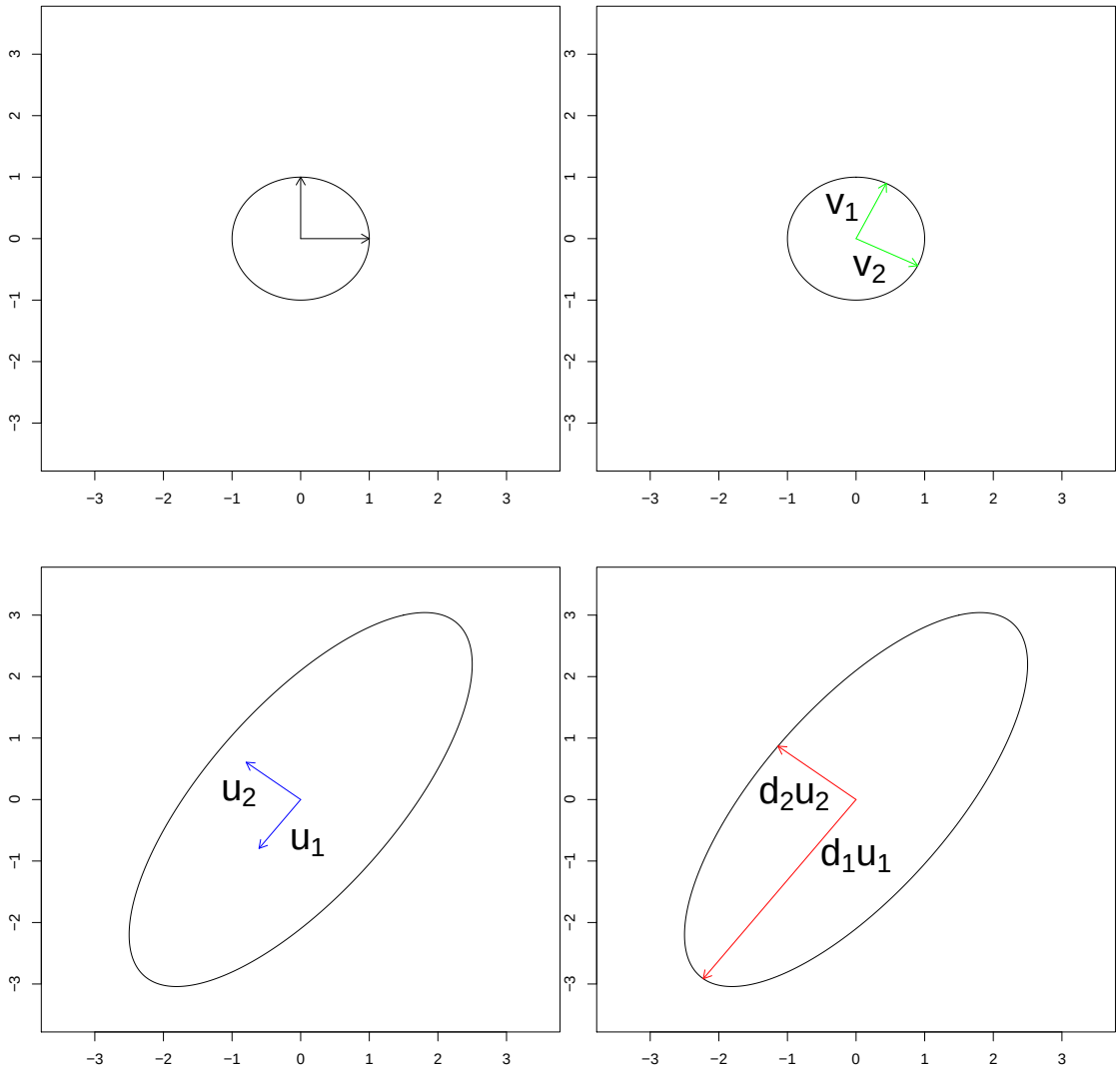
1. The coordinate axis gets rotated (Multiplication by $V^\top$)
1. The new axis gets elongated (Multiplication by D)

SVD



1. The coordinate axis gets rotated (Multiplication by $V^\top$)
1. The new axis gets elongated (Multiplication by D)
2. This ellipse gets rotated (Multiplication by U)

SVD [one last time]



Summary:

Of all the possible axes of the original circle, the one given by v_1, v_2 has the unique property:

$$\mathbb{X} v_j = d_j u_j$$

for all j.

Lastly:

$$\mathbb{X} = \sum_j d_j u_j v_j^\top$$

Postamble:

- Define data science

  (Amalgamation of many fields)
- Go over terminology and introduce notation

  (What is statistical learning? What types of learning are there?)
- Outline a general framework for assessing the quality of an estimator/algorithm

  (Loss and risk functions. A good procedure is one that has small risk.)
- The risk is unknown in practice $\rightarrow$ estimating it is paramount

  (The usual estimator of the risk, the training error, isn't reliable)
- Cover the singular value decomposition (SVD)

  (The SVD provides a unifying lens through which to view seemingly unrelated procedures)