Job Hunting Grind

Yihao Hu
ACMS, University of Notre Dame

Contents

Chapter 1

Abstract

Yihao Hu
May, 2022

Part I

Probability & Brain Teasers

Chapter 2

Problems

2.1 Binormial Distribution

The probability of getting k successes in n independent Bernoulli picks is given by the probability dense function:

$$P(k, n, p) = \binom{n}{k} p^k (1-p)^{n-k}$$

- mean: np
- variance: npq

2.2 Exponential Distribution

Problem: What is exponential distribution? What are the mean and variance of it?

Solution:
The exponential distribution is the probability distribution of the time between events in a Poisson point process, or the time we must wait until a certain event occurs.

$$f(x) = \begin{cases} \lambda e^{-\lambda x}, & x >= 0 \\ 0, & x < 0 \end{cases} \tag{2.1}$$

Expectation:

$$\begin{aligned} E(x) &= \int_0^\infty x\lambda e^{-\lambda x} dx && (2.2) \\ &= -xe^{-\lambda x} \mid_0^\infty - \frac{1}{\lambda} e^{-\lambda x} \mid_0^\infty && (2.3) \\ &= \frac{1}{\lambda} && (2.4) \end{aligned}$$

Variance:

$$\begin{aligned} E(x^2) &= \int_0^\infty x^2 \lambda e^{-\lambda x} dx && (2.5) \\ &= -x^2 e^{-\lambda x} \mid_0^\infty + 2\int_0^\infty x e^{-\lambda x} dx && (2.6) \\ &= \frac{2}{\lambda^2} && (2.7) \\ Var(x) &= E(x^2) - E(x)^2 = \frac{1}{\lambda^2} && (2.8) \end{aligned}$$

Example:
The mean waiting time for a bus is 40 mins, what is the probability that the your waiting time for next bus arrival is 50 mins if you just missed the previous bus?

Solution:

$$\lambda = \frac{1}{40} \Rightarrow P(x \leq 50) = 1 - e^{-50/40} = 71\%$$

2.3 Probability Y is greater than X.

Problem: If X and Y are independent exponential randon variables with means 6 amd 8, respectively, what's the probability that Y is greater than X?

Solution:
Here $\lambda_1 = \frac{1}{6}, \lambda_2 = \frac{1}{8}$

$$P(Y \geq X) = \int_0^\infty \int_0^y \frac{1}{48} e^{-\frac{x}{6}-\frac{y}{8}} dx\ dy = \frac{1}{8}\int_0^\infty (e^{-\frac{7y}{24}} - e^{-\frac{y}{8}})dy = \frac{4}{7} \tag{2.9}$$

2.4 Poisson Distribution

Problem: What is Poission distribution? What are the expect value and variance of Poisson Distribution?

Solution:
By given the number of events k and the mean rate λ of a time interval, Poisson Distribution is a discrete distribution to describe the probability of $x = k$ in a specific time interval.:

$$P(x = k) = \frac{e^{-\lambda}\lambda^k}{k!}, for\ \forall k > 0 \tag{2.10}$$

Expectation:

$$E(x) = \sum_{k=1}^{\infty} k\frac{e^{-\lambda}\lambda^k}{k!} = e^{-\lambda}\lambda\sum_1^\infty \frac{\lambda^{k-1}}{(k-1)!} = e^{-\lambda}\lambda e^{\lambda}(\,Taylor\ Expansion\,) = \lambda$$

Variance

$$\begin{aligned} E(x^2) &= \sum_{k=1}^{\infty} k^2\frac{e^{-\lambda}\lambda^k}{k!} = e^{-\lambda}\sum_1^\infty\{\frac{(k-1)\lambda^k}{(k-1)!} + \frac{\lambda^k}{(k-1)!}\} \\ &= e^{-\lambda}\sum_2^\infty \frac{\lambda^{k-2}}{(k-2)!}\lambda^2 + e^{-\lambda}\sum_1^\infty\frac{\lambda^{k-1}}{(k-1)!}\lambda \\ &= \lambda^2 + \lambda \end{aligned}$$

$$Var(x) = E(x^2) - E(x)^2 = \lambda$$

2.5 Chi square distribution

- if $X \sim \mathcal{N}(0, 1)$, then $X^2 \sim \chi^2$
- $\chi^2_m + \chi^2_n = \chi^2_{m+n}$, where m, n is the degree of freedom.
- $E(\chi^2_n = X^2) = n \times E(\chi^2 = X^2) = n \times (var(X) + E(X)^2) = n$
- $Var(\chi^2) = 2$
- $\chi^2_n \sim N(n, 2n)$

2.6 Geometric distribution

- pdf: $f(k) = (1-p)^{k-1}p$
- mean: $\frac{1}{p}$
- variance: $\frac{1-p}{p^2}$
- median: $\frac{-1}{\log_2(1-p)}$

2.7 Normal Distribution

Continuous probability distribution for a real-valued random variable.

- $x \sim \mathcal{N}(0,1)$
- $f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$
- mean: μ
- variance: σ^2

2.8 Uncorrelated vs. Independent

- Random variables X, Y are uncorrelated if

$$\begin{aligned}\rho(X,Y) &= \frac{Cov(X,Y)}{\sigma_x\sigma_y} = 0 \\ Cov(X,Y) &= \mathbf{E}[(X-\mathbf{E}[X])(Y-\mathbf{E}[Y])] = \mathbf{E}[XY] - \mathbf{E}[X]\mathbf{E}[Y]\end{aligned}$$

- **Independent:** $p(X,Y) = p(X)p(Y), p(X|Y) = p(X)$
- If **Independent** then **Uncorrelated**:

$$\begin{aligned}\mathbf{E}[XY] &= \iint xyp_{X,Y}(x,y)dxdy \\ &= \iint xyp_X(x)p_Y(y)dxdy \\ &= \int xp_X(x)\left(\int yp_Y(y)dy\right)dx \\ &= \left(\int xp_X(x)dx\right)\left(\int yp_Y(y)dy\right) \\ &= \mathbf{E}[X]\mathbf{E}[Y]\end{aligned}$$

- **Uncorrelated** doesn't imply **Independent**: Let $Y = |X|,\ X \sim U(-1,1), E[X] = 0$, then

$$E(XY) = E(X|X|) = \int_{-1}^{0} -X^2\frac{1}{2}dX + \int_0^1 X^2\frac{1}{2}dX = 0$$

Obviously they are uncorrelated but dependent.

- Only general case **Uncorrelated** does imply **Independent**: joint distribution of X, Y is Gaussian.

$$f_{\mathrm{X}}(x_1,\ldots,x_k) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)}{\sqrt{(2\pi)^k|\boldsymbol{\Sigma}|}}$$

2.9 What is the law of large numbers?[1]

There is a strong law and a week law of large numbers.
Strong Law: The strong law states that average of a large number of i.i.d. integrable random variables convergences almost surely to their common mean.

$$P(\lim_{n\to\infty} \frac{S_n}{n} = \mu) = 1.$$

Weak Law: for $\forall \epsilon > 0$

$$\lim_{n\to\infty} P(|\frac{S_n}{n} - \mu| > \epsilon) = 0$$

Weak law states the convergence above is only in probability.
Note: Convergence almost surely implies convergence in probability. In which sense the strong law is "stronger".

2.10 What is Central Limit Theorem?[1]

The CLT states that the limiting distribution of the centered and scaled sum of an i.i.d. sequence of random variables is a normal distribution if the common distribution of the random variables has finite variance.
Let $X_1, X_2, \cdots, X_n$ be a sequence of i.i.d. random variables with finite expected value $\mu = E(X_i)$ and finite variance $\sigma^2 = Var(X_i)$, Let $S = \sum_i X_i$, then:

$$\lim_{n\to\infty} \frac{S_n - n\mu}{\sigma\sqrt{n}} = Z \sim \mathscr{N}(0,1) \Leftrightarrow \lim_{n\to\infty} \sqrt{n}(\frac{S_n}{n} - \mu) \sim \mathscr{N}(0, \frac{\sigma^2}{n}).$$

Putting together 1.8 and 1.9 we have:

$$\frac{S_n}{n} \approx \mu + \frac{\sigma}{\sqrt{n}} Z$$

2.11 Delta Method

I was asked in an Applied Scientist phone screen interview for this question, I just said I forgot it.

From CLT we have

$$\lim_{n\to\infty} \sqrt{n}(S_n - n\mu) = Z \sim \mathscr{N}(0,\sigma^2) \Leftrightarrow (X_n - \mu) \xrightarrow{n\to\infty} \mathscr{N}(0, \frac{\sigma^2}{n})$$

The Delta Method states that for any smooth asymptotically function g,

$$\frac{\sqrt{n}\,(g\,(X_n) - g(\mu))}{|g'(\mu)|\,\sigma} \approx \mathscr{N}(0,1)$$

Proof.

$$\begin{aligned}
&g(X_n) = g(X_n - \mu + \mu) = g(\mu) + g'(\mu)(X_n - \mu) + \frac{g'(\mu)}{2}(X_n - \mu)^2 + \mathscr{O}((X_n - \mu)^3) \\
&\Rightarrow \frac{g(X_n) - g(\mu)}{g'(\mu)} = (X_n - \mu) + \mathscr{O}((X_n - \mu)^2) \\
&\Rightarrow \frac{g(X_n) - g(\mu)}{g'(\mu)} \approx \mathscr{N}(0, \frac{\sigma^2}{n}) \Leftrightarrow \sqrt{n}\big(g(X_n) - g(\mu)\big) \approx \mathscr{N}(0, g'(\mu)^2\sigma^2) \\
&\Rightarrow g(X_n) \approx \mathscr{N}\big(g(\mu), g'(\mu)^2(\frac{\sigma^2}{n})\big)
\end{aligned}$$

□

2.12 CLT in coin toss game

Problem: Toss a coin 100 times, estimate the probability sum of heads are greater or equal to 60.

Solution:
Use CLT, let s be the sum of 100 rolls.

$$P(s > 60) = 1 - P(s \le 59)$$
$$\mu = 50, \sigma^2 = 25$$
$$P(s > 60) \approx 1 - P(\frac{s-50}{5} < \frac{59-50}{5}) = 1 - \Phi(1.8) = 0.0359$$

2.13 CLT in coin toss game 2

Problem: Toss a dice 100 times, the sum is X, toss a coin 600 times, the sum is Y, estimate $P(X > Y)$

Solution:
Use CLT, let $Z = X - Y$, $mean(Z) = \frac{21}{6}100 - 300 = 50, Var(Z) = Var(X) + Var(Y) = (100Var(X_i) + 600Var(Y_i)) = 100\frac{35}{12} + 150 = 442$

$$P(X > Y) = P(Z > 0) = 1 - \Phi(\frac{0-50}{21})$$

2.14 Distance to the center of a disc.

Problem: A point is chosen uniformly from the disk, what is the expected value of the distance between the point and the center of the disk? [1]

Solution:
The probability density function:

$$P(x) = \frac{1}{\pi(Area\ of\ the\ disk)}, for\ \forall x \in D \tag{2.11}$$

$$\begin{aligned} E(x) &= \int_0^1 \int_0^1 \frac{1}{\pi}\sqrt{x^2+y^2}\ dxdy \\ &= \frac{1}{\pi}\int_0^{2\pi}\int_0^1 r^2\ drd\theta \\ &= \frac{2}{3} \end{aligned}$$

2.15 PDF of Max i.i.d variables

Problem: Let $X_i \sim U(a,b)$ be iid variables, what is the pdf if $Y = \max(X_1, \cdots X_n)$

Solution:

$$P(Y \le x) = P(X_1 \le x, \cdots X_n \le x) = \prod P(X_i \le x) = F(x)^n = (\frac{x-a}{b-a})^n$$
$$\Rightarrow p(y) = n\frac{(x-a)^{n-1}}{(b-a)^n}$$

2.16 Concave pdf

Problem: If the pdf is a strictly decreasing function, prove the median is less than the mean.

Solution:
if pdf is strictly decreasing then cdf is a convave function with (Jensen's Inequality):

$$F(E(x)) \geq E(F(x)) = \int_0^\infty F(x)f(x)\ dx = \int_0^1 F(x)\ dF(x) = \frac{1}{2} = F(median)$$

2.17 Conditional probability $P(X > 0|Y < 0)$

Problem: Consider two random variables X and Y with mean 0 and variance 1, with joint normal distribution, if $Cov(X,Y) = \frac{1}{\sqrt{2}}$ what is the conditional probability $P(X > 0|Y < 0)$?[1]

Solution:
$X \sim \mathscr{N}(0,1)$, $Y \sim \mathscr{N}(0,1)$, $Cov(X,Y) = E\{(X - E(X))(Y - E(Y))\} = \frac{1}{\sqrt{2}}$.

$$P(X > 0|Y < 0) = \frac{P(X > 0, Y < 0)}{P(Y < 0) = \frac{1}{2}}$$

Let $W = \sqrt{2}X - Y$, $E(W) = 0$.

$$\begin{aligned} Var(W) &= Var(\sqrt{2}X - Y) = Var(\sqrt{2}X) + Var(Y) - 2Cov(\sqrt{2}X, Y) \\ &= 2 + 1 - 2\sqrt{2}Cov(X,Y) = 1 \end{aligned}$$

Then $W \sim \mathscr{N}(0,1)$.

$$Cov(W,Y) = Cov(\sqrt{2}X - Y, Y) = Cov(\sqrt{2}X, Y) - Cov(Y,Y) = 0$$

W, Y are independent.

$$P(X > 0, Y < 0) = P(\frac{1}{\sqrt{2}}(W + Y) > 0, Y < 0)$$

Image the following linear programming problem:

$$P(X > 0, Y < 0) = \begin{cases} W + Y > 0 \\ Y < 0 \end{cases} = \frac{1}{8}$$

Thus:

$$P(X > 0|Y < 0) = \frac{1}{4}$$

2.18 Probability $P(Y > 3X)$

Problem: Let $X, Y \sim \mathscr{N}(0,1)$, what is $P(Y - 3X > 0)$?

Solution:

$$\begin{aligned} &Z = Y - 3X \\ &\Rightarrow Z \sim \mathscr{N}(0,10) \\ &\quad \Rightarrow P(Y - 3X > 0) = P(Z > 0) = \frac{1}{2} \end{aligned}$$

Problem: Let $X, Y \sim \mathscr{N}(0,1)$, what is $P(Y - 3X > 0|X > 0)$?

Solution:
Think about in a Cartesian coordinate, the probability is just the angle between the line $y = 0$ and $y = 3x$ over $x > 0$, thus $P(Y - 3X > 0|X > 0) = \frac{arctan\frac{1}{3}}{\pi}$

2.19 Joint Distribution

- For two independent random variables $X.Y$ the joint distribution:

$$P(X,Y) = P(X)P(Y)$$
$$F_{X,Y}(X \leq x, Y \leq y) = F_X(x)F_Y(y)$$

- If they are dependent:

$$P(X,Y) = P(X)P(Y|X) = P(Y)P(X|Y)$$

2.20 Joint lognormal random distribution

Problem: If X and Y are lognormal random variables, is XY a lognormal random variable?[1]

Solution:
$log(X) \sim \mathscr{N}(0,1), log(Y) \sim \mathscr{N}(0,1)$. $X = e^{Z_1}, Z_1 \sim \mathscr{N}(0,1), Y = e^{Z_2}, Z_2 \sim \mathscr{N}(0,1)$.

$$log(XY) = log(e^{Z_1+Z_2}) = Z_1 + Z_2.$$

If $log(X)$ and $log(Y)$ have joint distribution then XY is lognormal distribution, else not. See question.

2.21 $E(\Phi(x))$. [1]

Problem: Let $x \sim \mathscr{N}(0,1)$ and Φ be the cdf of standard normal distribution. Find $E(\Phi(x))$

Solution:

$$E(Y) = E(\Phi(X)) = E(P(Z \leq X)|X) = E(E(1_{Z\leq X}|X)) = E(1_{Z\leq X}) = P(Z \leq X),\ Z \sim \mathscr{N}(0,1)$$

Let W be $Z - X$, $E(W) = -\mu, Var(W) = 1 + \sigma^2$

$$P(Z \leq X) = P(W < 0) \equiv P(\frac{W+\mu}{\sqrt{1+\sigma^2}} \leq \frac{\mu}{\sqrt{1+\sigma^2}}) = \Phi(\frac{\mu}{\sqrt{1+\sigma^2}})$$

2.22 Generate uncorrelated variables

Problem: Generate two uncorrelated variables from two i.i.d. random variable with correlation 0.2.

Solution:
Let $X \sim \mathscr{N}(0,1)$, $Y \sim \mathscr{N}(0,1)$, $Cov(X,Y) = 0.2$. Let $W = aX + bY$.

$$Cov(W,Y) = Cov(aX+bY,Y) = a \cdot Cov(X,Y) + b \cdot Var(Y) = 0.2a + b.$$

Let $a = 1, b = -0.2$, $W = X - 0.2Y, Cov(W,Y) = 0$, then W, Y are two uncorrelated random variables.

2.23 Generate variables $\sim \mathscr{N}(0,1)$

Problem: Generate two $\mathscr{N}(0,1)$ random variables with correlation ρ if you have a random number generator for standard normal distribution.

Solution:

$$\begin{aligned}x_1 &= z_1 \sim \mathcal{N}(0,1)\\ x_2 &= az_1 + bz_2\\ cov(x_1, x_2) &= cov(z_1, az_1 + bz_2) = a = \rho\\ Var(x_2) &= a^2 + b^2 = 1 \Rightarrow b = \sqrt{1-\rho^2}\end{aligned}$$

2.24 Corr(A,C)

Problem: If corr(A,B) = X and corr(B,C) = Y, what is corr(A,C)?

Solution:
For independent variables X and Y,

$$Var(X+Y) = Var(X) + Var(Y)$$

For a right triangle with side length a, b, c:

$$a^2 = b^2 + c^2$$

For a dependent variable and a regular triangle we have:

$$\begin{aligned}&Var(X+Y) = Var(X) + Var(Y) + 2cov(X,Y)\\ &a^2 = b^2 + c^2 - 2ab \cdot \cos(a,b)\\ &corr(X,Y) = \frac{cov(X,Y)}{\sigma(X)\sigma(Y)}\\ &\Rightarrow corr(X,Y) \equiv \cos(\theta)\end{aligned}$$

$$\begin{aligned}&corr(A,B) = \cos(\theta_1), corr(B,C) = \cos(\theta_2)\\ &\Rightarrow corr(A,C) \in [\cos(\theta_1 - \theta_2), \cos(\theta_1 + \theta_2)]\\ &= (XY - \sqrt{1-X^2}\sqrt{1-Y^2}) \sim (XY + \sqrt{1-Y^2}\sqrt{1-X^2})\end{aligned}$$

We could also solve by the determinant of the 3×3 correlation matrix is $det = 1 + 2(\rho(ab))\rho(ac))\rho(bc)) - (\rho(ab))^2 + \rho(ac))^2 + \rho(bc))^2) \geq 0$.

2.25 Dice Game

Problem: A casino offers a simple dice game. Rolling a dice one, if the number you get is greater than the house, you get 1 dollar, else you get 0, what is the expectation of the game ? Now you could add 2 to the dice value by paying another 0.25 dollar, what is the expectation value?[2]

Solution:
Similar with 1.11, consider the symmetry, the probability you win is $\frac{1}{2}(1-\frac{1}{6}) = \frac{5}{12}, E(x) = 1 \cdot \frac{5}{12} = \frac{5}{12}$. Now if you pay 0.25 to increase the value, The probability you win is $\frac{1}{6}(\frac{2}{6} + \frac{3}{6} + \frac{4}{6} + \frac{5}{6} + \frac{6}{6} + \frac{6}{6}) = \frac{13}{18}$, the probability you draw the game is $\frac{2}{3} \cdot \frac{1}{6} = \frac{1}{9}$.

$$E(x) = \frac{13}{18} \cdot 0.75 - \frac{1}{6} \cdot 0.25 = \frac{35}{72}$$

2.26 Dice Game 2

Problem: Suppose that you are rolling a dice. For each roll, you are paid the face value. If a roll gives 4, 5 or 6, you can roll the dice again. If you get 1,2 or 3, the game stops. What is the expected payoff of this game?[2]

Solution:
let n be the rolling time

$$p(n = N) = (1 - p)^{N-1}p = \frac{1}{2^N}$$

Expected Payoff is

$$E(S_n) = E(N) * E(i)$$

This is a geometry distribution thus the mean is 2 and $E(i) = 3.5$, $E(S_n) = 7$.

2.27 Dice Game 3

Problem: You have the option to throw a die up to three times. You will earn the face value of the die. You have the option to stop after each throw and walk away with the money earned. The earnings are not additive. What is the expected payoff of this game?

Solution:
Let's start backward, suppose you only have one chance to toss, then the expectation is $E(n = 1) = 3.5$. Now you have a second chance, similarity

$$E(n = 2) = \frac{1}{6}(4 + 5 + 6) + \frac{1}{2}3.5 = 4.25$$

That's saying, if I throw in $4, 5, 6$ and I know that this value is beyond my expectation, I would be happy to directly get the face value, otherwise I choose to do a second toss and the expectation for next toss is 3.5. Thus if you throw a number which is less than 5, you choose to do the third toss.

$$E(n = 3) = \frac{1}{6}(5 + 6) + \frac{2}{3}4.25 = 4.66$$

2.28 Dice Game: Dice rolling with prime number

Problem: You have a unbiased dice with numbers from 1 to 10, if you roll a prime number you win the number, else you lose half of the value, what is your expectation?

Solution:

$$E(x) = \frac{1}{10}(2 + 3 + 5 + 7) - \frac{1}{20}(55 - 17) = -0.2$$

2.29 Dice Game: Dice rolling sum divided by 6

Problem: Roll a unbiased dice 10 times, what is the probability the sum could be divided by 6?

Solution:
Let $P(k, n)$ be the probability n sum divides 6 with a remainder k.

$$P(0, n) = \sum_{k=0}^{5} P(k, n-1)P(n^{th} = 6 - k) = \frac{1}{6}\sum_{k=0}^{5} P(k, n-1) = \frac{1}{6}$$

Thus for any number and no matter how many times you throw, the answer is always $\frac{1}{6}$.

2.30 Dice Game: expectation of rolling a dice until you get 6?

Problem: Keep rolling a dice until you get 6, what is the expectation of the rolls?

Solution:
Simple geometric expectation: $E = \frac{1}{\frac{1}{6}} = 6$.

Problem: Now still rolling until you see 6, given the sequences you have are all even, for example (2,4,6), (2,6), (2,2,6), now what is the expectation.

Solution:
Note this problem is tricky! The sample space has been changed as we only consider all the sequence with even numbers and end with a first seen 6. A promising math proof is to calculate the pdf, for a n solution, the previous $n-1$ consist of only 2 or 4, thus:

$$
\begin{aligned}
P(k_n = 6, k_i(1 \le i < n) \in \{2,4\}) &= 2^{n-1}(\frac{1}{6})^n = \frac{1}{2}(\frac{1}{3})^n \\
P(A = \text{even sequence end with 6}) &= \frac{1}{2}\sum_{i=1}^{\infty}(\frac{1}{3})^i = \frac{1}{4} \\
\Rightarrow E = \sum_{i=1}^{\infty} iP(A_i|A) &= \sum_{i=1}^{\infty} i\frac{P(A_i)}{P(A)} = 2\sum_{i=1}^{\infty}\frac{i}{3^i} = 1.5
\end{aligned}
$$

Note for $0 < x < 1$,

$$
\begin{aligned}
\sum_{n=0}^{\infty} x^n &= \frac{1}{1-x} \\
\Rightarrow \sum_{n=0}^{\infty} nx^{n-1} &= \frac{1}{(1-x)^2} \\
\Rightarrow \sum_{i=1}^{\infty}\frac{i}{3^i} &= \frac{1}{3}\sum_{i=1}^{\infty}\frac{i}{3^{i-1}} = \frac{1}{3}\frac{1}{\frac{4}{9}} = \frac{3}{4}.
\end{aligned}
$$

A simple approach, you will stop if you first roll is in $\{1,3,5,6\}$, otherwise you keep rolling then:

$$
E = \frac{4}{6} + \frac{1}{3}(1+E) = 1.5
$$

2.31 Dice Game: Dice rolling sum expectation

Problem: Keep roll a unbiased until you get 1, what is the expectation of the sum?

Solution:

$$
\begin{aligned}
E(1) &= \frac{1}{6}\times 1 + \frac{1}{6}(2 + E(1) + 3 + E(1) + 4 + E(1) + 5 + E(1) + 6 + E(1)) \\
\Rightarrow E(1) &= 21
\end{aligned}
$$

Actually you could find that $E(n) = 21$.

2.32 Dice Game: Dice in increasing order

Problem: We throw 3 dice one by one. What is the probability that we obtain 3 points in strictly increasing order?

Solution:
Just think picking 3 dices from $1,2,3,4,5,6$.

$$
p = \frac{\binom{6}{3}}{6^3} = \frac{20}{216} = \frac{4}{54}
$$

2.33 Dice Game: Rolling dice until getting 'HHT' or 'HTH'

Problem: Rolling a dice until getting 'HHT' or 'HTH' in a sequence, what is the probability of ending at 'HHT' state?

Solution:
This question equivalents to rolling a HHT befroe HTH. Let $P(E)$ be the probability of end up with HHT. Obviously:

$$\begin{aligned}
P(E) &= P(E|T) = P(E|TT) = P(E|TTT\cdots) \\
P(E) &= P(E|T)P(T) + P(E|H)P(H) = P(E)\frac{1}{2} + P(E|H)\frac{1}{2} \Rightarrow P(E) = P(E|H) \\
P(E) &= P(E|H) = \big(P(E|HH) = 1\big)\frac{1}{2} + P(E|HT)\frac{1}{2} = \frac{1}{2} + P(E|HT)\frac{1}{2} \\
P(E|HT) &= \frac{1}{2}\big(P(E|HTH) = 0\big) + \frac{1}{2}P(E|HTT) = \frac{1}{2}P(E|HTT) = \frac{1}{2}P(E|T) = \frac{1}{2}P(E) \\
&\Rightarrow P(E) = \frac{1}{2} + \frac{1}{4}P(E) = \frac{2}{3}
\end{aligned}$$

2.34 Dice Game: Dice rolling with 12 or 2 7's first

Problem: Comparing the probabilities of rolling a 12 or two consecutive 7's first with a pair of dice

Solution:
Let A be the event that you just rolled a 12.
Let B be the event that you just rolled a 7.
Let C be the event that you just rolled a sum other than 12 or 7.
Let $P(W)$ be the event that A wins.

$$\begin{aligned}
&P(A) = \frac{1}{36}, P(B) = \frac{6}{36}, P(C) = \frac{29}{36} \\
&P(W) = P(W|A)P(A) + P(W|B)P(B) + P(W|S)P(S) \\
&P(W) = \frac{1}{36} + P(W|B)\frac{1}{6} + P(W)\frac{29}{36} \\
&P(W|B) = \frac{1}{36}P(W|B,A) + \frac{1}{6}P(W|B,B) + P(W|B,S)\frac{29}{36} = \frac{1}{36} + \frac{29}{36}P(W) \\
&P(W) = \frac{7}{13}
\end{aligned}$$

One could also use the Markov chain to solve this problem.

2.35 Dice Game: Seen all even numbers before the first odd number

Problem: A fair 6-sided die is repeatedly rolled until an odd number appears. What is the probability that every even number appears at least once before the first occurrence of an odd numbers?

Solution:
Denote B_i be the event that the previous sequence doesn't have the number $i \in \{2, 4, 6\}$ and A be the

event that the whole sequence ends with a single odd number at the end.

$$P(B|A) = P(B_2 + B_4 + B_6|A) =$$
$$P(B_2|A) + P(B_4|A) + P(B_6|A) - (P(B_2B_4|A) + P(B_2B_6|A) + P(B_4B_6|A))$$
$$= \sum_{k=4}^{\infty} 3((\frac{2}{3})^{k-1} - (\frac{1}{3})^{k-1})$$
$$P(\bar{B}|A) = \sum_{k=4}^{\infty} 1 - 3((\frac{2}{3})^{k-1} - (\frac{1}{3})^{k-1})$$
$$P(\bar{B}) = P(\bar{B}|A)P)(A) = \sum_{k=4}^{\infty} \{1 - 3((\frac{2}{3})^{k-1} - (\frac{1}{3})^{k-1})\} \frac{1}{2^k} = \frac{1}{20}$$

2.36 Coin Problem: Unfair coin

Problem: You are given 1000 coins. Among them, 1 coin has heads on both sides. The other 999 coins are fair coins. You randomly choose a coin and toss it 10 times. Each time, the coin turns up heads. What is the probability that the coin you choose is the unfair one?[2]

Solution:
Let U be the unfair coin and F be the fair coin.

$$P(U|10h) = \frac{P(U, 10h)}{P(10h|U)P(U) + P(10h|F)P(F)} = \frac{\frac{1}{1000}}{\frac{1}{1000} + \frac{999}{1000}\frac{1}{2^{10}}} = \frac{1024}{2023} \approx 0.5$$

2.37 Coin Problem: Generate fair events by an unfair coin

Problem: How to generate even odds using an unfair coin? Solution:
Let $P(t) = \rho$, $P(h) = 1 - \rho$, toss the coin twice,

$$p(tt) = \rho^2, p(ht) = p(th) = \rho(1-\rho), p(hh) = (1-\rho)^2$$

Now $p(ht)$ $p(th)$ are two events with same probability.

2.38 Coin Problem: Check the unfair coin

Problem: You select a coin at random from the bag and toss it five times. It comes up heads three times. What is the probability that it was the coin that was biased towards tails? How many times do you need to toss the coin that is biased towards tails before it comes up with a majority of tails with probability greater than 99/100?

Solution:
let $p(h) = a$, then $P(t) = 1 - a$, you want to check if $a > 0.5$. If it is a fair coin the p-value is $0.5^3 = 0.0125$.

A Bayesian Posterior Test Assume we have n tosses. $g(p) \sim U(0, 1)$

$$f(a|h = k, t = n-k) = \frac{p(h = k|n = t + k, a)g(a)}{\int_0^1 p(h = k|n = t + k, p)g(p)dp}$$
$$p(h = k|n = t + k, r) = \binom{n}{k} a^k (1-a)^{n-k}$$
$$\Rightarrow f(a|h = k, t = n-k) = \frac{\binom{n}{k} a^k (1-a)^{n-k}}{\int_0^1 \binom{n}{k} p^k (1-p)^{n-k} dp}$$
$$= \frac{1}{beta(h+1, t+1)} a^k (1-a)^{n-k}$$
$$= \frac{(n+1)!}{k!(n-k)!} a^k (1-a)^{n-k}.$$

For this question, if it is a biased coin toward tails, then

$$p(biased\ toward\ tail) = \int_0^{0.5} \frac{4!}{0!3!} a^3\ da = 6.25\%$$

By a given probability $p(t) = r$, for n tosses,

$$p(majority\ tails) = \sum_{k=\frac{n}{2}\ or\ \frac{n+1}{2}}^{n} \binom{n}{k} r^k (1-r)^{n-k} = 0.99.$$

2.39 Coin Problem: Flip fair/unfair coin

Problem: What is the expectation of filliping a unbiased coin until you see a head/tail? What if the it is a biased coin with $P(H) = p$? [1]

Solution:
Easy geometric distribution. For an unbiased coin $E(H) = E(T) = 0.5 + 0.5(1 + E) \Rightarrow\ E = 2$.
For a biased coin $E(H) = p + (1-p)(1 + E(H)) \Rightarrow E(H) = \frac{1}{p}$.

2.40 Coin Problem: Flip coin to get 2 heads

Problem: What is the expected toss number to get 2 consecutive heads by filliping a fair coin?[1]

Solution:

$$\begin{aligned} E(\cdots HH) &= \underbrace{\frac{1}{4}2}_{HH} + \underbrace{\frac{1}{4}(2 + E(\cdots HH))}_{HT} + \underbrace{\frac{1}{2}(1 + E(\cdots HH))} \\ E(\cdots HH) &= 6. \end{aligned}$$

Problem: What if the coin is biased with $P(H) = p$?

Solution:

$$\begin{aligned} E(\cdots HH) &= \underbrace{2p^2}_{HH} + \underbrace{p(1-p)(2 + E(\cdots HH))}_{HT} + \underbrace{(1-p)(1 + E(\cdots HH))}_{T} \\ E(\cdots HH) &= \frac{1+p}{p^2}. \end{aligned}$$

2.41 Coin Problem: Flip coin with no consecutive heads

Problem: What is the probability to get a sequence of length n with no consecutive heads?[1]

Solution:
For a n sequence, there are 2^n permutations, let a_n be the number of sequences with no consecutive heads.

1 if a_n begins with a T, then $a_n = a_{n-1}$

2 if a_n begins with a H, then the following toss must be a T, thus $a_n = a_{n-2}$

- Therefore $a_n = a_{n-1} + a_{n-2}$, it is the Fibonacci Number.

How to solve this? Assume $a_n = ar^n$

$$
\begin{aligned}
& r^2 - r - 1 = 0 \\
& \Rightarrow r_1 = \frac{1+\sqrt{5}}{2}, r_2 = \frac{1-\sqrt{5}}{2} \\
& \Rightarrow a_n = C_1 r_1^n + C_2 r_2^n = C_1(\frac{1+\sqrt{5}}{2})^n + C_2(\frac{1-\sqrt{5}}{2})^n \\
& a_1 = 2 \ \{H, T\}, a_2 = 4 \ \{HT, TH, TT\} \\
& \Rightarrow C1 = \frac{3+\sqrt{5}}{2\sqrt{5}}, C2 = -\frac{3-\sqrt{5}}{2\sqrt{5}} \\
& \Rightarrow p = \frac{r_1^{n+2} - r_2^{n+2}}{2^n \sqrt{5}}
\end{aligned}
$$

2.42 Coin Problem: Coin Flip Game

Problem: Two gamblers are playing a coin toss game. Gambler A has $(n+1)$ fair coins; B has n fair coins. What is the probability that A will have more heads than B if both flip all their coins? [2]

Solution:
Let's consider for a game for both A and B have same number of coins n, thus we have the following partitions:

E1 : A and B have the same heads.

E2 : A has more heads than B.

E3 : B has more heads than A.

Thus:

$$
\begin{aligned}
P(E2) &= P(E3) \\
P(E1) + P(E2) + P(E3) &= 1
\end{aligned}
$$

For A has $n+1$ coins, the partitions for A has more heads than B is $P(E1)\frac{1}{2} + P(E2) = \frac{1}{2}(P(E1) + P(E2) + P(E3)) = \frac{1}{2}$

2.43 Coin Problem: generate identical probability with fair/unfair coin

Problem: You want to pick 3 items with the same probability, how to do this with a fair coin? What is your expectation of the tosses?

Solution:
Toss twice, the four events are HT, TH, HH, TT, with the same probability 0.25, the strategy is pick any one of the four events and skip it if you tossed that event, then use the remaining 3 to choose the corresponding item.

$$
\begin{aligned}
E &= \frac{3}{4}2 + \frac{1}{4}(2+E) \\
\Rightarrow E &= \frac{8}{3}
\end{aligned}
$$

Or this is a simple geometric distribution the mean is $\frac{1}{\frac{3}{4}} = \frac{8}{3}$.

Problem: How about an unfair coin?

Solution:
Denote $P(H) = p, P(T) = (1-p)$, the idea is to make permutations that three distinct events will have a identical probability.

1 $\{HTT, THH, HTH\}$ has the same probability $p(1-p)^2$

2 $\{HHTT, HTHT, HTTH\}$ has the same probability $p^2(1-p)^2$

2.44 Coin Problem: Toss 4 coins

Problem: You have four unbiased coins and you could toss each coin once, for every *head* you will get one dollar, what is your expected return?

Solution:

$$E(x) = (\binom{4}{1} + 2\binom{4}{2} + 3\binom{4}{3} + 4)\frac{1}{2^4} = 2$$

Problem: Given a second chance to play the game, what is your strategy and what is the expectation?

Solution:
The expectation of the previous game is 2 thus if you toss less than 3 heads you may want to play the game again.
The probability of tossing heads less than 3 is $p(h < 3) = \frac{1+4+6}{2^4} = \frac{11}{16}$. if play a second game:

$$E(x) = \frac{4}{16}3 + \frac{1}{16}4 + \frac{11}{16}2 = \frac{19}{8} = 2.375$$

Thus this strategy worth $\frac{3}{8} = 0.375$ dollars. (A `python` test with 10^5 trials gives us a solution of `2.375315` which is consist with our estimation.)

2.45 Poker Problem: Pick card until you get specific one

Problem: Picking from a 52 cards poker set with replacement until you have card K, what is the expectation and medium of the picking times x?

Solution:
This is a geometric distribution problem with

$$\begin{aligned} E(x = n) &= \sum_{n=1}^{\infty} n(\frac{51}{52})^{n-1}\frac{1}{52} \\ &= \frac{1}{52}52^2 = 52 \end{aligned}$$

The median m of a distribution means: PDF:

$$\begin{aligned} p(x = n) &= (1-p)^{n-1}p \\ P(x \le n) &= \sum_{x=1}^{n}(1-p)^{x-1}p = 1 - (1-p)^n = 1 - (\frac{51}{52})^n \end{aligned}$$

$$P(x < m) = \frac{1}{2}, P(x > m) = \frac{1}{2}$$

Then,

$$\begin{aligned} P(x < m) = 1 - (1-p)^m &= \frac{1}{2} \\ \Rightarrow m = -\frac{1}{\log(1-p)} &= -\frac{1}{\log(\frac{51}{52})} \approx 35.69 < 52 \end{aligned}$$

Another trivial way is that you find this is a geometric distribution which has mean $\frac{1}{p=\frac{1}{52}}$ and median $-\frac{1}{\log_2(1-p)}$.

2.46 Poker Problem: Poker Game

Problem: A casino offers a simple card game. There are 52 cards in a deck with 4 cards for each jack queen king ace value $2 \sim A$. Each time the cards are thoroughly shuffled (so each card has equal probability of being selected). You pick up a card from the deck and the dealer picks another one without replacement. If you have a larger number, you win; if the numbers are equal or yours is smaller, the house wins, like in all other casinos, the house always has better odds of winning. What is your probability of winning?[2]

Solution:
There are 13 different card numbers in poker, the probability of picking a number n is $\frac{1}{13}$ and the wining probability for n is $P(n) = \frac{1}{13}\frac{4(n-2)}{52-1}$, Thus:

$$P(N) = \sum_{n=2}^{n=14(A)} \frac{4(n-2)}{13 \cdot 51} = \frac{4}{13 \cdot 51} \cdot (13 \cdot 6) = \frac{8}{17}$$

By using symmetry, we know that the probability of picking the same value card is $\frac{1}{17}$ and thus your card is greater than the house is $\frac{1}{2}(1 - \frac{1}{17}) = \frac{8}{17}$.

2.47 Poker Problem: Meet the first Ace

Problem: Keep picking cards until you see the first Ace, What is expectation of the picks?

Solution:
The distribution of the the pile will be $[]A[]A[]A[]A[]$, therefore there are 5 slots for every card you could pick and the valid slot is the first one.

$$P = \sum_{1}^{48} \frac{1}{5} = \frac{48}{5}$$

2.48 Poker Problem: Poker Permutation

Problem: Poker is a card game in which each player gets a hand of 5 cards. There are 52 cards in a deck. Each card has a value and belongs to a suit. There are 13 values, $2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A$, and four suits, What are the probabilities of getting hands with four-of-a-kind (four of the five cards with the same value)? Hands with a full house (three cards of one value and two cards of another value)? Hands with two pairs?[2]

Solution:
The probability of getting four same values in one hand is:

$$p = \frac{\binom{13}{1} \cdot \binom{48}{1} \cdot}{\binom{52}{5}} = \frac{624}{2598960} = \frac{1}{4165}$$

Full house:

$$p = \frac{\binom{13}{1} \cdot \binom{4}{3} \cdot \binom{12}{1} \cdot \binom{4}{2}}{\binom{52}{5}} = \frac{6}{4165} = 0.0014$$

Two pairs:

$$p = \frac{\binom{13}{2} \cdot \binom{4}{2} \cdot \binom{4}{2} \cdot 44}{\binom{52}{5}} = \frac{198}{4165} = 0.0475$$

2.49 Circle Problem: Points in one semicircle

Problem: Given N points drawn randomly on the circumference of a circle, what is the probability that they are all within a semicircle?[2]

Solution:
Let's start from $n = 1$, $P(n = 1) = 1$. $P(n = 2) = 1$, for $n = 3$, the probability of the third point lies in the semicircle which contains the previous two points are $P(n = 3) = \frac{1}{2}$. Intuitively, no matter how the points are located on the clock, if they all lie on a semicircle, set the left most point as the leading point, rotate the clock and relocate the leading point at 12, then all the points left will lie on $0 \sim 6$. The probability that such left $n-1$ points lie on the semicircle is $\frac{1}{2^{n-1}}$, there are n points which could be chosen as the leading point. Therefore,

$$P(n) = \frac{n}{2^{n-1}}$$

2.50 Circle Problem: Circle separated by 3 lines

Problem: One straight line could separate a circle into two parts, assume 3 straight lines could separate a circle into N parts, what is the expectation of N?

Solution:
Hard question, the key point is to find how many intersections between lines. Let's start from the very beginning. For lines $k = 2$, if the intersections inside the circle $i = 0$, then $N = 3$, $i = 1$, then $N = 4$.
Now, what is the probability that two lines could intersect in a circle?
One line could separate a circle into two parts A, B, if the second line could have intersection with the first line, then the two endpoints of second line must lie separately on parts A and B. Assume the probability that endpoint lies in part A is p_a, then $p_b = 1 - p_a$. Probability two lines will intersect by symmetry will be:

$$p = 2\int_0^1 p_a(1-p_a)dp_a = p_a^2|_0^1 - \frac{2}{3}p_a^3|_0^1 = \frac{1}{3}$$

Now we have 3 lines and $\binom{3}{m}$ permutations for m intersections.

$$N = (\frac{2}{3})^3 \cdot 4 + 3\frac{1}{3}(\frac{2}{3})^2 \cdot 5 + 3(\frac{1}{3})^2\frac{2}{3} \cdot 6 + (\frac{1}{3})^3 \cdot 7 = 5$$

2.51 Circle Problem: Intersections in a circle

Problem: Suppose you are walking on the circle randomly with a degree, and go to the next point with your degree direction, the path of every step is a straight line with two ends on circle, you take n steps to come back to your starting point ,what is the expectation of the intersections of all the path?
(Example, the unit circle is $x^2 + y^2 = 1$, you start at $(1, 0)$, if you get a random degree of 180, then your next end point will be $(-1, 0)$ and the straight path is the x-axis)

Solution:
From previous question we know that the probability of two lines in a circle have intersection is $\frac{1}{3}$. For consecutive paths we don't have intersection. The total possible paths have intersection is :

$$\binom{n}{2} - n = \frac{n(n-3)}{2}$$

Therefore the expected intersection with linear expectation as $\frac{n(n-3)}{2\times 3} = \frac{n(n-3)}{6}$.

Check for $n = 3$, the answer is clearly 0.

Problem: Is the expectation still valid if you don't get back to origin within n steps?

Solution:
Yes, draw a picture and you will see the two problems are equivalent.

2.52 Picking Series: Sum pick from distribution first exceeds y

Problem: Let x_i be i.i.d. random variables which follows the uniform distribution $[0, 1]$, what is the expectation of the number of picks of x_i to satisfy their sum first exceeds y?

Solution:
Let N be the satisfied picks,

$$E(N) = N \cdot P(N)$$

$$\begin{aligned}
P(x_1 + x_2 + \cdots + x_N \le y) &= \int_0^y \int_0^{y-x_N} \cdots \int_0^{y-x_N-x_{N-1}-\cdots x_2} \; dx_1 \cdots \; dx_{N-1} \; dx_N \\
&= \int_0^y \int_0^{y-x_N} \cdots \int_0^{y-x_N-x_{N-1}-\cdots x_3} (y - x_N - x_{N-1} - \cdots - x_2) \; dx_2 \cdots \; dx_{N-1} \; dx_N \\
&= \int_0^y \int_0^{y-x_N} \cdots \int_0^{y-x_N-x_{N-1}-\cdots x_4} \frac{(y - x_N - x_{N-1} - \cdots x_3)^2}{2} \; dx_3 \cdots \; dx_{N-1} \; dx_N \\
&= \frac{y^N}{N!}
\end{aligned}$$

Taylor expansion of e^y :

$$e^y = 1 + y + \frac{y^2}{2} + \cdots + \frac{y^N}{N!} = \sum_{N=0}^{\infty} \frac{y^N}{N!}$$

Let $S_N = \sum_{i=1}^{N} x_i$, then for the first $S_N > y$, we have $P(N) = P(S_{N-1} \le y, S_N > y)$.

$$P(S_{N-1} \le y) = P(S_{N-1} \le y, S_N > y) + \big\{P(S_{N-1} \le y, S_N \le y) = P(S_N \le y)\big\}$$

$\Rightarrow$

$$\begin{aligned}
P(N) &= \frac{y^{(N-1)}}{(N-1)!} - \frac{y^N}{N!} \\
E(N) &= \sum_2^{\infty} N\left(\frac{y^{(N-1)}}{(N-1)!} - \frac{y^N}{N!}\right) \\
&= \sum_2^{\infty} (N-1)\left(\frac{y^{(N-1)}}{(N-1)!}\right) + \left(\frac{y^{(N-1)}}{(N-1)!}\right) - N\left(\frac{y^N}{N!}\right) = ye^y + (e^y - 1) - (ye^y - 1) = e^y.
\end{aligned}$$

The expectation pick is $E(N) = e^y$

2.53 Picking Series: Monotonic increasing sequence from a uniform distribution

Problem: What is the probability of picking a monotonic increasing sequence of variables $x_i \sim U(0, 1)$ one by one with size n?

Solution:

$$\begin{aligned}
P(N \ge n) &= P(x_1 < x_2 < x_3 \cdots < x_n \cdots) \\
&= \frac{1}{n!}
\end{aligned}$$

Which means only the first n elements are sorted, the probability is $\frac{1}{total\ permutation}$.

$$\Rightarrow P(N = n) = P(N \ge n) - P(N \ge n + 1) = \frac{1}{n!} - \frac{1}{(n+1)!} = \frac{n}{(n+1)!}$$

Problem: What is the expected length of an iid sequence that is monotonically increasing when drawn from a uniform [0,1] distribution?

Solution:
Let $y \geq 1$ be a discrete random variable, then

$$\begin{aligned}\int_0^\infty \Pr(Y \geqslant y)\mathrm{d}y &= \int_0^\infty \int_y^\infty f_Y(z)\mathrm{d}z\ \mathrm{d}y \\ &= \int_0^\infty \int_0^z f_Y(z)\mathrm{d}y\ \mathrm{d}z \\ &= \int_0^\infty f_Y(z) \int_0^z 1\ \mathrm{d}y\ \mathrm{d}z \\ &= \int_0^\infty z f_Y(z)\mathrm{d}z \\ &= \mathrm{E}[Y]\end{aligned}$$

$$E(n) = \sum_{n=1}^{\infty} P(N \geq n) = e - 1$$

2.54 Picking Series: Random height from street

Problem: Randomly pick someone from the street and you find his/her height is X, then you keep picking people randomly from the street until the one you picked is taller than X, what is the expected number $E(n)$ of picking times?

Solution:
The answer is infinity! We know the heights of people should follow a normal distribution as $X \sim \mathcal{N}(0, 1)$, if we randomly pick someone with height X, then $P(x = X) = f(X)$, now this problem become a geometric distribution problem, either the next person we picked is taller than X or shorter than X, with a probability $1 - F(x < X)$ Thus $E(n|x = X) = \frac{1}{1-F(X)}$

$$\begin{aligned}E(n) = \sum_X (E(n|x = X)p(x = X)) = \int \frac{f(x)}{1 - F(x)}dx = \int \frac{1}{1 - F(x)}dF(x) \\ = -\log(1 - F(x))\ |_{F(x)=0}^{F(x)=1} = \infty\end{aligned}$$

2.55 Picking Series: Pick first smaller than previous one

Problem: Keep picking number $x_i \sim U(0, 1)$ until the number you picked is smaller than the previous one, what is the expectation of the picking time?

Solution:

$$P(n) = P(x_1 < x_2 < x_3 \cdots < x_{n-1} > x_n)$$

Still, think about you get a sequence with length n, probability of picking $(x_n < x_{n-1}) = \frac{n-1}{n}$ since x_{n-1} is the largest number in this sequence, similarly with previous problems, $P(x_1 < x_2 < x_3 \cdots < x_n) = \frac{1}{(n-1)!}$ becasue you only have one case out of $(n-1)!$ permutations. Therefore

$$\begin{aligned}P(n) &= \frac{n-1}{n}\frac{1}{(n-1)!} = \frac{n-1}{n!} \\ \Rightarrow E(n) &= \sum_{n=2}^{\infty} \frac{n-1}{n!}n = \sum_{n=2}^{\infty} \frac{1}{(n-2)!} = e\end{aligned}$$

Or image the valid permutation is pick a number in the end which is not the largest, there $n-1$ ways, the previous permutation is only the sorted array, so the probability is $\frac{n-1}{n!}$.

2.56 Random Walk: Random walk martingale

Problem: Prove a random walk with identical probability is a martingale

Solution:

$$S_{n+1} = \begin{cases} S_n + 1 \text{ with probability } 1/2 \\ S_n - 1 \text{ with probability } 1/2 \end{cases} \tag{2.12}$$

Proof.

$$E(S_{n+1}) = \frac{1}{2}(S_n + 1) + \frac{1}{2}(S_n - 1) = S_n \square$$

□

Also for $E(S_n^2 - n)$

Proof.

$$E(S_{n+1}^2 - (n+1)) = S_n^2 - n \square$$

□

2.57 Random Walk: Drunk man

Problem: A drunk man is at the 17th meter of a 100 -meter-long bridge. He has a 50% probability of staggering forward or backward one meter each step. What is the probability that he will make it to the end of the bridge (the 100 th meter) before the beginning (the 0 th meter)? What is the expected number of steps he takes to reach either the beginning or the end of the bridge?[2]

Solution:
This problem equivalent to solve $E(S_{83})$ if we mark the current state 17 as S_0, assume p_a is the probability the drunk falls in 100 and $p_b = 1 - p_a$ is the probability he will fall in 0 thus we have two martingale equations as:

$$E(S_N) = 83p_a - (1 - p_a)17 = 0 \Rightarrow p_a = 0.17$$
$$E(S_N^2 - N) = E(p_a 83^2 + (1 - p_a)17^2) - E(N) = S_0^2 - 0 = 0 \Rightarrow E(N) = 1441$$

2.58 Random Walk: Ants on stick

Problem: 100 ants are on one stick with length of 1 meter. Each ant is either traveling left or right with constant speed 1 meter per minute. When two ants meet, they bounce off each other and change direction. The ant will fall if it reaches one of the two ends of the stick.

1 Over ALL possible initial configurations, what is the longest amount of time that you would need to wait until there is no ants on the stick?

- **Solution:**
  For each meeting ants, just think that the ant will switch the role with the ant it will meet. Then actually there is no any ant changed directions. The longest time is just $t = \frac{l}{v} = 1$.

2 What is the average time it takes for n ants?

- **Solution:**
  suppose one ant is in the position x, the probability is $\frac{1}{l}$, and all other $n-1$ ants are in left of this ant, they all go left as we based on question 1, then the guaranteed empty time $E(t) = \int_0^n \frac{x}{v} n(\frac{x}{l})^{n-1} \frac{1}{l} dx = \frac{n}{n+1} \frac{l}{v}$.

2.59 Strategy: Number Pick Strategy

Problem: Alice and Bob alternately choose one number from one of the following nine numbers: $\frac{1}{16}, \frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1, , 2, 4, 8, 16$, without replacement. Who ever get three numbers that multiply to one wins the game. Alice starts first, what's her wining strategy?[1]

Solution:
Image you are play in a 3×3 board and each column, row, and diagonal is a winning path, then this goes to a Tic Tac Toe game. Alice may not have a winning strategy but she will never lose, this could also be considered as a winning strategy. $\begin{bmatrix} 2^3 & 2^{-4} & 2^1 \\ 2^{-2} & 2^0 & 2^2 \\ 2^{-1} & 2^4 & 2^{-3} \end{bmatrix}$.

2.60 Strategy: Escape Strategy

Problem: Bob is imprisoned by aliens in a large circular field surrounded by a fence. Outside the fence is a viscious alien that can run four times as fast as Bob, but is constrained to stay near the fence, he can quickly scale the fence and escape. Can he get to a point on the fence ahead of the alien? [1]

Solution:

- Can Bob run straight to the fence?
  - The guard is an intelligent creature so it will stay in the closet position on fence to Bob, therefore the initial distance between bob and guard id $d(G, B) \leq r$.
  - The shortest time Bob will take if run straightly is $t_1 = \frac{r}{v}$, then the path guard will run using the same time period is $s = t_1 4v = 4r > \pi r$, The answer is no.
- Now the idea is how to make the guard and Bob not in the same radius or make the distance being larger than r? Let's assume Bob is in the position B with a distance $xr,\ 0 \leq x \leq 1$ to the circle, then the guard is in the same radius and has a distance $r(1-x)$ to Bob.
  - What is the angle difference if run rotational(by circle)?
    - Denote the initial position as $\{G, B, O\}$, where O is the center of the circle, the distance is $d(G, B) = (1-x)r, d(B, O) = xr$, the three points are in the same line(collinear).
    - For a fixed time $\Delta t = 1$, $s_B = v, s_G = 4v$.
    - Then the angle difference in a unit time is $\frac{v}{xr} - \frac{4v}{r}$, if the angle difference is positive, that means Bob will always run faster angles than the guard, and then finally there will be a time that the position is $\{G, O, B\}$ with distance is $d(G, O) = r, d(O, B) = xr$.
  - Now with this new circle, Bob could run directly to the fence with a distance of $(1-x)r$, the escape condition is $\frac{r(1-x)}{v} \times 4v < \pi r$.

sum up for x:

$$\frac{v}{xr} - \frac{4v}{r} > 0$$
$$\frac{r(1-x)}{v} \times 4v < \pi r$$
$$\Rightarrow 1 - \frac{\pi}{4} < x \leq 1$$

sum up for the strategy:

1 Choose a position with valid ratio x.

2 Run rotationally until reach the collinear position $\{G, O, B\}$.

3 Escape!

2.61 Strategy: Gaming Bet

Problem: Two teams are playing a BO7 game, you would like to bet 100 on A wining the series, and you could only bet on individual game, the wining probability of two teams are equal(0.5), what's your betting strategy?

Solution:

- Dynamic Programming, denote $P(i, j)$ be the payoff on team A has won i games and lost j games. $0 \leq i \leq 4, 0 \leq j \leq 4$. $P(4, j \leq 3) = 100$, $P(i \leq 3, 4) = -100$.
- Let $B(i, j)$ be the bets we need to place at each state (i, j). $B(3, 3) = 100$.
- The transaction function:

$$
\begin{aligned}
P(i+1, j) &= P(i, j) + B(i, j) \text{ (If A wins)} \\
P(i, j+1) &= P(i, j) - B(i, j) \text{ (If A loses)} \\
\Rightarrow P(i, j) &= \frac{1}{2}(P(i+1, j) + P(i, j+1)) \\
B(i, j) &= \frac{1}{2}(B(i+1, j) - B(i, j+1))
\end{aligned}
$$

A python approach:

```python
p = [[0 for i in range(5)] for j in range(5)]
b = [[0 for i in range(4)] for j in range(4)]
for i in range(4):
    p[-1][i] = 100
    p[i][-1] = -100
for i in range(3, -1, -1):
    for j in range(3, -1, -1):
        p[i][j] = 0.5*(p[i+1][j] + p[i][j+1])
        b[i][j] = 0.5*(p[i+1][j] - p[i][j+1])
for bet in b:
    print(bet)
# [31.25, 31.25, 25.0, 12.5]
# [31.25, 37.5, 37.5, 25.0]
# [25.0, 37.5, 50.0, 50.0]
# [12.5, 25.0, 50.0, 100.0]
```

Python

2.62 Strategy: Bash Game

Problem: You and Bob are playing a game, there are n pieces in a chessboard, each player could take $1 \sim m$ pieces every round, the player who take the last piece will be the winner, you have the option to choose who will play first, what's your wining strategy?

Solution:
Strategy: if `n % (m + 1) != 0`, the first player win, else choose to be the second player.

Proof.

- if $n < m + 1$, then you take all the pieces in the first round
- if $n = m + 1$, you will lose if you choose to be the first player.
- if $n = k(m + 1) + r$
  - $r = 0$, for every round, if you pick x pieces first, the next player will pick $m + 1 - x$ to make $n = (k - 1)(m + 1)$, and finally he will pick the last piece, you lose.

- $r \neq 0$, you pick r, then the game turns to case $r = 0$ and you are the second player, you win.

$\square$

A leetcode practice: 292. Nim Game

2.63 Digit Estimate: How many digits do 125^{100} have?

$$125^{100} = 5^{300} = 10^{300}(\frac{1}{2})^{300} = \frac{10^{300}}{1024^{30}} = \frac{10^{210}}{1.024^{30}} < \frac{10^{300}}{10^{90}} = 10^{210}.$$

$$1.024^{30} = \sum_{i=0}^{i=30} \binom{30}{i} 0.024^i$$

$$\text{The convergence rate} = \frac{\binom{30}{j+1}0.024^{j+1}}{\binom{30}{j}0.024^j} = 0.024\frac{30-j}{j+1} < 30 \times 0.024 = 0.72$$

$$\Rightarrow \binom{30}{i} 0.024^i \leq 0.72^i$$

$$\Rightarrow \sum_{i=0}^{30} 0.72^i < \frac{1}{1-0.72} = 3.57 < 10$$

$$\Rightarrow \frac{10^{210}}{3.57} < 125^{100} = \frac{10^{210}}{1.024^{30}} < 10^{210}$$

Therefore the digit number is 210.

2.64 Digit Estimate: 100^{th} digit

Problem: What is the 100^{th} digit to the right of the decimal point in the decimal representation of $(1+\sqrt{2})^{3000}$?

Solution:

$$(x+y)^n = \sum_{k=0}^{n} x^k y^{n-k}$$

$$(1+\sqrt{2})^n + (1-\sqrt{2})^n = \sum_{k=0}^{n}(\sqrt{2})^{n-k} + \sum_{k=0}^{n}(-\sqrt{2})^{n-k} = 2\sum_{k=2i, 0\leq i \leq \frac{n}{2}}^{n}(\sqrt{2})^{n-k} \text{ is an integer.}$$

$(1-\sqrt{2})^{3000} = (\sqrt{2}-1)^{3000} < (2)^{-3000} < 10^{-100}$ so the 100^{th} digit equals to 9.

2.65 Simulation: Uniform distribution from disc

Problem: How to generate points uniformly in a disc?

Solution:

Rejection Method

1 Generate $x \in [-1, 1], y \in [-1, 1]$.

2 if $x^2 + y^2 > 1$, repeat step 1.

- The probability for each point lie in the circle is $\frac{\pi}{4}$.

Polar Coordinates

1 Generate $R \in [0,1], \theta \in [0, 2\pi]$.

2 let $x = \sqrt{R}\cos(\theta), y = \sqrt{R}\sin(\theta)$

- Why we want $\sqrt{R}$?
  - If with $x = r\cos(\theta), y = r\sin(\theta)$, the circumference of a circle with radius r is $2\pi r$, the probability that a point lie in this circumference is $f(r) = \frac{2\pi r}{\pi 1^2} = 2r$, which is a lienar function of r and is not not uniformly distributed.
  - $f(r) = 2r \rightarrow F(r) = r^2$
  - Now let $r^2 = R \Leftrightarrow r = \sqrt{R}$ we have $F(r) = \int 2rdr = \int 2\sqrt{R}\frac{1}{2\sqrt{R}}dR = R \rightarrow f(R) = 1$.

2.66 3×3×3 Cube

Problem: Suppose you have a 3×3×3 original white cube and then paint the surface with red color, randomly pick one block from the cubic and roll it, the five surfaces you could see are white, what is the probability that the block you pick is all white block?

Solution:
There is only one block with all white surfaces, 6 blocks with one red surface, and the rest of them are either with 2 or 3 surfaces. For a block with $5W1R$ surfaces, the probability that you roll it and the red surface face down is $P(5W|5W1R) = \frac{1}{6}$.

$$P(6W|5W) = \frac{P(5W|6W)P(6W)}{P(5W|6W)P(6W) + P(5W|5W1R)P(5W1R)} = \frac{\frac{1}{27}}{\frac{1}{27} + \frac{1}{6}\frac{6}{27}} = \frac{1}{2}$$

2.67 Rain on weekends

Problem: The probability it will rain on Saturday is p and on Sunday is q, what is the probability it will on weekends? What is the probability it will rain one day?

Solution:
If the two events are independent then the probability is $1 - (1-p)(1-q) = p + q - pq$.
Now let $X_i = \mathbf{I},\ \mathbf{I} \in \{0,1\}$ be the event that it will rain on weekend i. $P(X_1 = 1) = p, P(X_2 = 1) = q$.
The partitions of the collection of subsets are $P(X_1 = 1, X_2 = 1)$(rain on both days), $P(X_1 = 1, X_2 = 0), P(X_1 = 0, X_2 = 1)$ and $P(X_1 = 0, X_2 = 0)$.
Let $P(X_1 = 1, X_2 = 1) = a$.

$$\begin{aligned}
P(X_1 = 0, X_2 = 1) &= q - a \\
P(X_1 = 1, X_2 = 0) &= p - a \\
P(X_1 = 0, X_2 = 0) &= 1 - p - q + a \\
0 \leq P(X_1 = \mathbf{I}, X_2 = \mathbf{I}) &\leq 1
\end{aligned}$$

Thus:

$$max(p + q - 1, 0) \leq a \leq min(p, q)$$

The probability it will rain on weekend is $1 - P(X_1 = 0, X_2 = 0) = p + q - a$.
Example, if $p = 0.5, q = 0.6$, $p + q - a \in \{0.6, 1\}$ The probability that will only rain on one day is $p + q - 2a$.

2.68 Drunk passenger

Problem: A line of 100 airline passengers are waiting to board a plane. They each hold a ticket to one of the 100 seats on that flight. For convenience, let's say that the n -th passenger in line has a ticket for the seat number n. Being drunk, the first person in line picks a random seat (equally likely for each seat). All of the other passengers are sober, and will go to their proper seats unless it is already occupied; In that case, they will randomly choose a free seat. You're person number 100. What is the probability that you end up in your seat (i.e., seat #100) ?

Solution:
This is a very interesting question. There a a lot of discussions online and here assume the drunk choose a random seat $1 < n < 100$, then is question turns out to be passenger n will not choose the correct seat, and what is the probability that 100 is not be chosen. In another perspective, passenger n now is the drunk. This problem is equivalent to if a passenger n $(1 < n < 100)$ find his/her seat was occupied by the drunk, he/her will ask the drunk to leave and the drunk will choose another seat. Finally the drunk only have two seats to choose: seat 1 and seat $k = 100$, therefore, the probability of you seat in the correct position is $\frac{1}{2}$.

2.69 Creature extinction

Problem: A creature has the same probability of dying, keeping the same, split in to 2, and split into 3, what is the probability the creature that will extinct?

Solution:
Assume extinction is independent.

$$\begin{aligned}
&p = \frac{1}{4} + \frac{1}{4}p + \frac{1}{4}p^2 + \frac{1}{4}p^3 \\
&p^3 + p^2 - 3p + 1 = 0 \\
&(p-1)(p^2 + 2p - 1) = 0 \\
&p = -1 + \sqrt{2} \approx 0.41
\end{aligned}$$

2.70 Two sticks

Problem: There are two sticks with lengths l_a and l_b, you have a ruler which will have an error $\epsilon \sim \mathcal{N}(0, \sigma)$ for each measurement, please give a strategy to measure l_a and l_b with a minimal variance of errors.[3]

Solution:
Strategy: measure $S = l_a + l_b + \epsilon_s$ and $D = l_a - l_b + \epsilon_d$.

Proof.

$$\begin{aligned}
l_a &= \frac{1}{2}(S + D) + \frac{1}{2}(\epsilon_s + \epsilon_d) \\
l_b &= \frac{1}{2}(S - D) - \frac{1}{2}(\epsilon_s - \epsilon_d) \\
Var(\epsilon_a) &= Var(\frac{1}{2}(\epsilon_s + \epsilon_d) = \frac{1}{4}(\epsilon^2 + \epsilon^2) = \frac{\sigma^2}{2} \\
Var(\epsilon_b) &= Var(\frac{1}{2}(\epsilon_s - \epsilon_d) = \frac{1}{4}(\epsilon^2 + \epsilon^2) = \frac{\sigma^2}{2}
\end{aligned}$$

2.71 Broken stick

Problem: If a stick is broken into to pieces, what is the expectation of the smaller one? What is the average ratio of the smaller piece to the larger?[3]

Solution:
Let the length of the stick be 1. $f(x) = 1$ and there are two possible sides of the smaller part.

$$E(x) = 2\int_0^{\frac{1}{2}} x \; dx = \frac{1}{4}$$

The ratio:

$$E(r) = 2\int_0^{\frac{1}{2}} \frac{x}{1-x} \; dx = 2\int_0^{\frac{1}{2}} \frac{1}{1-x} - 1 \; dx = 2(-\ln(1-x) - x)|_0^{\frac{1}{2}} = 2\ln(2) - 1$$

2.72 Climb stairs

Problem: A rabbit sits at the bottom of a staircase with n stairs. The rabbit can hop up only one or two stairs at a time. How many different ways are there for the rabbit to ascend to the top of the stairs?[2]

Solution:
This is a classical dynamic programming problem, let $f(n), n > 0$ be the number of ways the rabbit could achieve the n^{th} floor. $f(1) = 1$, $f(2) = 1$,

$$f(n) = f(n-1) + f(n-2), \; n > 2$$

Thus this is a Fibonacci Sequence, for a coding example, you may practice on leetcode70.

2.73 Tennis Tournament

Description: A tennis tournament is arranged for 2^n players. It is organised as a knockout tournament, so that only the winners in any given round proceed to the next round. Opponents in each round except the final are drawn at random, and in any match either player has a probability $\frac{1}{2}$ of winning. Two players are chosen at random before the start of the first round.

Problem: Find the probabilities that they play with each other: Assume the two players are player A and player B.

1: **In the first round**

Solution:
Let n be the round they will meet, then

$$p(n = 1) = \frac{1}{2^n - 1}$$

2: **In the final round**

Solution:
The probability A and B win the first round and not meet in first round is

$$p = p(n \neq 1)p(A_{win})p(B_{win}) = \frac{2^n - 2}{2^n - 1} \cdot \frac{1}{4} = (\frac{1}{2})\frac{2^{n-1} - 1}{2^n - 1}$$

One could prove by induction that the probability A and B win the k^{th} round and not meet is:

$$p = (\frac{1}{2})\frac{2^{n-k} - 1}{2^{n-k+1} - 1}$$

Thus the probability that they will meet in the final round is:

$$p(n = final) = \prod_{k=1}^{n-1} (\frac{1}{2}) \frac{2^{n-k} - 1}{2^{n-k+1} - 1} = (\frac{1}{2^{n-1}}) \frac{1}{2^n - 1}$$

From the answer we could find that $p = \frac{1}{\binom{2^n}{2}}$, one could directly get the solution by understanding that the probabilities of one randomly picked pair contests the **any match** are equal= $(\frac{1}{2^{n-1}})\frac{1}{2^n-1}$. As all the picks have the same probability to play the final game.
For example for the first round, there are 2^{n-1} match slots and the probability of meeting each other in a specific slot is $p = (\frac{1}{2^{n-1}})\frac{1}{2^n-1}$

3: **In the tournament.**

Solution:
probability of meeting in the k^{th} round is:

$$p = (\frac{1}{2^{k-1}}) \frac{2^n - 1}{2^{n-k+1} - 1} \cdot (\frac{1}{2^{n-k} - 1}) = \frac{1}{2^{k-1}} \frac{1}{2^n - 1}$$

sum up we have:

$$p = \frac{1}{2^n - 1} \sum_{k=1}^{n} \frac{1}{2^{k-1}} = \frac{1}{2^{n-1}}$$

Also, remind the probability of meeting at every match is same and there are $\frac{(1-2^n)}{1-2} = 2^n - 1$ matches, thus the probability is $(2^n - 1) \cdot \frac{1}{2^{k-1}} \frac{1}{2^n-1} = \frac{1}{2^{n-1}}$.

2.74 Job letters

Problem: You're sending job applications to 5 firms: Morgan Stanley, Lehman Brothers(dead), UBS, Goldman Sachs, and Merrill Lynch. You have 5 envelopes on the table neatly typed with names and addresses of people at these 5 firms. You even have 5 cover letters personalized to each of these firms. Your 3 -year-old son tried to be helpful and stuffed each cover letter into each of the envelopes for you. Unfortunately she randomly put letters into envelopes without realizing that the letters are personalized. What is the probability that all 5 cover letters are mailed to the wrong firms?[2]

Solution:
This problem is a classic example for the Inclusion-Exclusion Principle. In fact, a more general case is an example in Ross' textbook First Course in Probability. Let's denote by $E_i, i = 1, \cdots, 5$ the event that the i -th letter has the correct envelope. Then $P\left(\bigcup_{i=1}^5 E_i\right)$ is the probability that at least one letter has the correct envelope and $1 - P\left(\bigcup_{i=1}^5 E_i\right)$ is the probability that all letters have the wrong envelopes. $P\left(\bigcup_{i=1}^5 E_i\right)$ be calculated using the Inclusion-Exclusion Principle: $P\left(\bigcup_{i=1}^5 E_i\right) = \sum_{i=1}^5 P(E_i) - \sum_{i_i<i_2} P(E_{i_1}E_{i_2}) + \cdots + (-1)^6 P(E_1E_2\cdots E_5)$

$$\sum_{1}^{5} P(E_i) = 1, \sum_{i_1<i_2} P(E_{i_1}E_{i_2}) = 10\frac{1}{20} = \frac{1}{2}$$

$$\sum_{i_1<i_2<i_3} P(E_{i_1}E_{i_2}E_{i_3}) = 10\frac{1}{60} = \frac{1}{6}, \sum_{i_1<i_2<i_3<i_4} P(E_{i_1}E_{i_2}E_{i_3}E_{i_4}) = \frac{1}{24}$$

$$1 - P\left(\bigcup_{i=1}^{5} E_i\right) = \frac{1}{2!} - \frac{1}{3!} + \frac{1}{4!} - \frac{1}{5!} = \frac{11}{30}$$

2.75 Same birthday

Problem: How many people do we need in a class to make the probability that two people have the same birthday more than $1/2$? **(For simplicity, assume 365 days a year.)[2]**

Solution:
let n be the number of students. There are 365^n permutations. The probability n people have all distinct birthdays is:

$$p(n) = \begin{cases} 1, & \text{if } n > 365 \\ \frac{365 \cdot 364 \cdots (365-n+1)}{365^n} & \text{if } n <= 365 \end{cases}$$

$p(n) = \frac{365 \cdot 364 \cdots (365-n+1)}{365^n} < \frac{1}{2} \Rightarrow n = 23.$

2.76 Cubic ending

Problem: Let x be an integer between 1 and 10^{12}, what is the probability that the cubic of x ends with 11? [2]

Solution:
Let $x = 10a + b$,

$$x^3 = 100a^3 + 300a^2b + 30ab^2 + 100ab + b^3$$

Therefore the end digit for x^3 only depends on $b^3, 30ab^2 \Rightarrow b = 1, a = 7$, thus x ends up with 71, $p = 1\%$.

2.77 Both children are boys 1

Problem: A company is holding a dinner for working mothers with at least one son. Ms. Jackson, a mother with two children, is invited. What is the probability that both children are boys?[2]

Solution:
let $x = n$ be the event there are n boys is a family.

$$P(x = 2|x \geq 1) = \frac{P(x_2 = 1, x_1 = 1)}{p(x \geq 1)} = \frac{1}{3}$$

2.78 Both children are boys 2

Problem: Your new colleague, Ms. Parker is known to have two children. If you see her walking with one of her children and that child is a boy, what is the probability that both children are boys?

Solution:
Here the sample space has changed, you randomly saw a woman among all women, thus the probability that the second child is a boy is $\frac{1}{2}$. In problem one, your are picking one from those who has at least one boy.

2.79 Sex ratio

Problem: Assume the probability of having a boy and girl is equal and all couples in the society won't stop giving birth to babies until they have a girl. What will the ratio human sex ratio be finally?

Solution:

Let n be the total number of children a family will end up with.

$$E(boy) = \sum_{1}^{\infty}(n-1)\frac{1}{2^n}$$
$$E(girl) = \sum_{1}^{\infty}\frac{1}{2^n} = 1$$

$$\sum_{1}^{\infty} x^n (x<1) = \frac{1}{1-x} \Rightarrow \sum_{1}^{\infty} n \cdot x^{n-1} = \frac{1}{(1-x)^2} \Rightarrow \frac{1}{2}\sum_{1}^{\infty} n \cdot (\frac{1}{2})^{n-1} = 0.5\frac{1}{(\frac{1}{2})^2} = 2$$
$$\Rightarrow E(boy) = \sum_{1}^{\infty}(n-1)\frac{1}{2^n} = \sum_{1}^{\infty} n\frac{1}{2^n} - \sum_{1}^{\infty}\frac{1}{2^n} = 2-1 = 1$$

Amazingly the ratio will be 1 : 1.

2.80 Dart game

Problem: Jason throws two darts at a dartboard, aiming for the center. The second dart lands farther from the center than the first. If Jason throws a third dart aiming for the center, what is the probability that the third throw is farther from the center than the first? Assume Jason's skillfulness is constant.[2]

Solution:
Let's split the area from best to worst to be A, B, C. We want to calculate:

$$P(3_{rd} > 1_{st} | 2_{rd} > 1_{st})$$

There fore the events for$P(3_{rd} > 1_{st}, 2_{rd} > 1_{st})$ to be:

$$A\ B\ C$$
$$A\ C\ B$$

$P(2_{rd} > 1_{st})$ to be:

$$A\ B\ C$$
$$A\ C\ B$$
$$B\ C\ A$$

Thus $P(3_{rd} > 1_{st}|2_{rd} > 1_{st}) = \frac{2}{3}$. For a n throw, the probability that n^{th} throw to be the best is $\frac{1}{n}$, and thus the probability it is not the best throw is $\frac{n-1}{n}$.

2.81 Same birthday

Problem: At a movie theater, a whimsical manager announces that she will give a free ticket to the first person in line whose birthday is the same as someone who has already bought a ticket. You are given the opportunity to choose any position in line. Assuming that you don't know anyone else's birthday and all birthdays are distributed randomly throughout the year (assuming 365 days in a year), what position in line gives you the largest chance of getting the free ticket?[2]

Solution:
At position i where $i > 0$, let $p(i)$ be the probability that nobody shares the same birthday before i and i^{th} person have the same birthday with someone ahead.

$$p(n) = \frac{365 \cdot 364 \cdot \cdots \cdot (365-n+2)}{365^{n-1}} \cdot \frac{n-1}{365},\ p(n)_{n\geq 365} = 1.$$

$$\left.\begin{array}{r} P(n-1) = \frac{365}{365} \times \frac{364}{365} \times \cdots \times \frac{365-(n-3)}{365} \times \frac{n-2}{365} \\ P(n) = \frac{365}{365} \times \frac{364}{365} \times \cdots \times \frac{365-(n-2)}{365} \times \frac{n-1}{365} \\ P(n+1) = \frac{365}{365} \times \frac{364}{365} \times \cdots \times \frac{365-(n-2)}{365} \times \frac{365-(n-1)}{365} \times \frac{n}{365} \\ P(n) > P(n+1) \Rightarrow \frac{n-1}{365} > \frac{365-(n-1)}{365} \times \frac{n}{365} \end{array}\right\} \Rightarrow \left.\begin{array}{l} n^2 - 3n - 363 < 0 \\ n^2 - n - 365 > 0 \end{array}\right\} \Rightarrow n = 20$$

2.82 Monty Hall problem

Problem: Monty Hall problem is a probability puzzle based on an old American show Let's Make a Deal. The problem is named after the show's host. Suppose you're on the show now, and you're given the choice of 3 doors. Behind one door is a car; behind the other two, goats. You don't know ahead of time what is behind each of the doors. You pick one of the doors and announce it. As soon as you pick the door, Monty opens one of the other two doors that he knows has a goat behind it. Then he gives you the option to either keep your original choice or switch to the third door. Should you switch? What is the probability of winning a car if you switch?

Solution:
The only probability that you win without switching is that you choose the right door at the beginning, which is $\frac{1}{3}$, if you use the switching strategy, the case you will win is you choose the door with goat which is $\frac{2}{3}$, so switch!

2.83 The Flippant Juror

Problem: A three-man jury has two members each of whom independently has probability p of making the correct decision and the third member who flips a coin for each decision(majority rules). A one-man jury has probability p of making the correct decision. Which jury has the better probability of making the correct decision?[3]

Solution:
For a three-man jury, the event of making the correct decision is the number of people making correct decision $n \geq 2$.

$$p(n \geq 2) = p(n=2) + p(n=3) = \frac{1}{2}p(1-p)2 + \frac{1}{2}p^2 + \frac{1}{2}p^2 = p$$

Thus those two juries are identical.

2.84 Bomb Game

Problem: There are 100 dollars in a box, however there sis 50% probability that the box contains a bomb which follows a uni from distribution and will explode in 100 days, what is the price if some one wish to buy this box at day n? Assume you could only get the money if there is no bomb in the box.

Solution:
Denote NB be the event that there is no bomb in the box, S be the event that the box is still safe in date n.

$$P(NB|S) = \frac{P(S|NB)P(NB)}{P(S)} = \frac{P(S|NB)P(NB)}{P(S|NB)P(NB) + P(S|B)P(B)} = \frac{\frac{1}{2}}{\frac{1}{2} + \frac{1}{2}(\frac{100-n}{100})} = \frac{100}{200-n}$$

$$\Rightarrow E = 100P(NB|S) = \frac{10000}{200-n}$$

2.85 3 Box Pick

Problem: There are 3 boxes and each contains 2 balls, box 1 has 2 blue balls, box 2 has 2 red balls, and box 3 has 1 red ball and 1 blue ball. What is the probability that the second ball in the same box is also blue given the first ball you picked is blue.

Solution:

$$P(2B|1B) = \frac{P(1B|2B)P(2B)}{P(1B)} = \frac{\frac{1}{3}}{\frac{1}{3} + \frac{1}{3}\frac{1}{2}} = \frac{2}{3}$$

2.86 The Sock Drawer

Problem: A drawer contains red and black socks. When two socks are drawn at random, the probability that both are red is $\frac{1}{2}$ [4]

a How small can the number of socks in the drawer be?
Solution:

- Let red socks be a, black socks be b, and toal socks be $n = a + b$, we have

$$\frac{a}{n}\frac{a-1}{n-1} = \frac{1}{2} \Rightarrow 2(a^2 - a) = n^2 - n, a \in (0, n)$$
$$\Rightarrow a = 3, n = 4.$$

b How small if the black sock is even?
Solution:

$$\begin{aligned}
&b = (n-a)\%2 = 0 \\
&2((n-b)^2 - (n-b)) = n^2 - n \\
&\Rightarrow n^2 - 4nb + 2b^2 - n + 2b = 0 \\
&\Rightarrow n^2 - (4b+1)n + 2(b^2 + b) = 0,\ b\%2 = 0 \\
&\Rightarrow n = a + b = \frac{(4b+1) \pm \sqrt{(4b+1)^2 - 8(b^2+b)}}{2}
\end{aligned}$$

you then loop b and find the satisfied solution, however, this takes time. From (a), we may notice that:

$$\begin{aligned}
&(\frac{a}{a+b})^2 > \frac{1}{2} > (\frac{a-1}{a+b-1})^2 \Leftrightarrow (\frac{a}{a+b}) > \frac{1}{\sqrt{2}} > (\frac{a-1}{a+b-1}) \\
&\Rightarrow \frac{1}{\sqrt{2}}(a+b) < a < \frac{1}{\sqrt{2}}(a+b-1) + 1 \\
&\Rightarrow \frac{1}{\sqrt{2}}b < (1 - \frac{1}{\sqrt{2}})a < \frac{1}{\sqrt{2}}b + 1 - \frac{1}{\sqrt{2}} \\
&\Rightarrow (\sqrt{2}+1)b < a < (\sqrt{2}+1)b + 1
\end{aligned}$$

Then you loop $b \in [2, 4, 6, \cdots, 2k]$ and find all valid a, check the the smallest a, b which satisfies our equation n from previous step. The answer is:

$$b = 6, n = \frac{25 + \sqrt{289}}{2} = 21 \Rightarrow a = 15$$

2.87 Successive Wins

Problem: To encourage Elmer's promising tennis career, his father offers him a prize if he wins (at least) two consecutive tennis sets in a row in a three-set series to be played with his father and the club champion alternately: father-champion-father or champion-father-champion, according to Elmer's choice. The champion is a better player than Elmer's father. Which series should Elmer choose?[3]

Solution:
When come up with those questions like dart game we mentioned previously, intuitively we don't want to play against champion more than father because champion is a better player. However, interviewer won't ask such trivial questions.
The best idea is to enumerate all the possible situations.
Denote the probability Elmer beats his father is p_f and beats the champion is p_c, the wining situation are:

$$win = \{WWF, WWW, FWW\}$$

Therefore

$$\begin{aligned}
win(fcf) &= p_f p_c(1-p_f) + p_f p_c p_f + (1-p_f)p_c p_f = p_f p_c(2-p_f) \\
win(cfc) &= p_c p_f(1-p_c) + p_f p_c p_f + (1-p_c)p_f p_c = p_f p_c(2-p_c) \\
p_f > p_c &\Rightarrow win(cfc) > win(fcf)
\end{aligned}$$

Thus we choose champion-father-champion.

2.88 Find the lowest floor to break the egg

Problem: You have two identical eggs which will break if dropped from a certain height, given the egg will definitely break if dropped from the 100th floor, what is the minimum trails you could try to determine the lowest floor that the egg will break?

Solution:
Let's think the problem in another way, given n trials you could try, determine the highest floor you could reach to check the break floor.
Strategy:

- Try on floor n,
- if it breaks, we only left with one ball, we have to try form the bottom to up to determine the break floor,(the worst case is the breaking floor is on $(n-1)$th where we still have $(n-1)$ times to try).
- if it not breaks, then we have $n-1$ trials available, next we try $(2n-2)$th floor, if it breaks, we have one egg and try bottom to up.
- The process repeats if the egg won't break on the floor we try.
- the height is $n + (n-1) + \cdots + 1 = \frac{n(1+n)}{2}$

$$argmin_n \frac{n(1+n)}{2} \geq 100 \Rightarrow n = 14.$$

Problem: What if we have k eggs? This is also a hard leetcode problem 887. Super Egg Drop.

Solution:
A dynamic programming approach, let $f_k(n)$ be the maximum floor we could test by given k eggs and n trials.

- if you only have one last egg and then you would try on the current lowest floor and it breaks, the final floor you could test is $f_{k-1}(n-1)+1$, thus we have the bottom floors of the egg.

- if the egg didn't break, then we treat the $f_{k-1}(n-1)+2$ as the new floor, and we have $n-1$ trials left with k eggs, thus we have the upper part.
- $f_k(n) = \underbrace{f_{k-1}(n-1)}_{bottompart} + \underbrace{1}_{currentfloor} + \underbrace{f_k(n-1)}_{upperpart} = n + \sum_{j=1}^{n-1} f_{k-1}(j)$

2.89 Ant Path

Problem: An ant is in a corner of $10 \times 10 \times 10$ cubic and want to go to the opposite corner, what is the shortest path for it?[1]

Solution:
Don't let the cubic distract you, the shortest path is always the straight line, since the ant could only walk on the cubic, unfold the cubic and you will find the path is the longest hypotenuse of the right triangle as $10\sqrt{5}$.

2.90 Train Running Oppositely

Problem: At your subway station, you notice that of the two trains running opposite directions which are supposed to arrive with the same frequency, the train in on direction comes first 80% of the time and the train in another direction comes first only 20% of the time, what do you think could be happening?[1]

Solution:

1 The assumption is wrong as the frequencies are different.

2 If the frequency is same, the coming time actually doesn't matter, if you split the unite time period as 10, let B arrive in 2 and A arrive in 8, both of them will only arrive once per 10 minutes, then this satisfies the conditions.

2.91 Probability of being a subset

Problem: Given a set X with n elements, choose two subsets A and B, what is the probability that A is a subset of B?[1]

Solution:
A mathematical approach: if we pick a set B with k elements, we have $\binom{n}{k}$ permutations, then if A is a subset of B, A have 2^k combinations.

$$P(A \subseteq B) = \sum P(A \subseteq B_k|B_k)P(B_k) = \sum_{k=0}^{n} \frac{2^k}{2^n}\frac{\binom{n}{k}}{2^n} = \frac{1}{4^n}\sum_{k=0}^{n} 2^k \binom{n}{k} = \frac{1}{4^n}\sum_{k=0}^{n} 2^k 1^{n-k}\binom{n}{k} = (\frac{3}{4})^n$$

A intuitive approach by symmetry: A randomly picked element x could be in the following sets.

1 $x \in A \cap B$

2 $x \in A, x \not\subseteq B$

3 $x \not\subseteq A, x \in B$

4 $x \not\subseteq A, x \not\subseteq B$

case $1, 3, 4$ consists with the condition that $A \subseteq B$, we need every x in n to have the same probability, then $p = (\frac{3}{4})^n$.

2.92 Random guess number

Problem: Alice writes two distinct real numbers between 0 and 1 on two sheets of paper. Bob selects one of the sheets randomly to inspect it. He then has to declare whether the number he sees is bigger or smaller of the two. Is there anyway Bob can expect to be correct better than random guess($p_{correct} > 0.5$)?[1]

Solution:
Strategy:

- Denote the two numbers Alice write to be $0 \le a_1 < a_2 \le 1$, a_1, a_2 are i.i.d, so they are almost surely different.
- After pick one number A wich is written by Alice, Bob randomly sample a $B \sim U(0,1)$
- If $B < A$ then state A is the bigger number, else A is the smaller number.

Proof.

$$p_{correct} = p(B < a_2|a_2)p(a_2) + p(B > a_1|a_1)p(a_1) = \frac{1}{2}a_2 + \frac{1}{2}(1 - a_1) = \frac{1}{2} + \frac{1}{2}(a_2 - a_1) > \frac{1}{2}.$$

□

2.93 Alternative Sign Sum

Problem: For every subset of $\{1, 2, 3, \cdots, 2013\}$, arrange the numbers in the increasing order and take the sum with alternating signs. The resulting integer is called the weight subset. Find the sum of weights of all the subsets of $\{1, 2, 3, \cdots, 2013\}$[1].

Solution:
It will be better to think it in a recursive way. For example $S(\{1, 2, 3\}) = 1 - 2 + 3$, $S(\{2, 3\}) = 2 - 3$, for any subset w with out element 1, there must be a paired subset $w \cup \{1\}$ and $S(w \cup \{1\} + w) = 1$, thus $S = 2^{2012}$.

2.94 Shake Hands Problem

Problem: Mr. and Mrs. Jones invite four other couples over for a party. at the end of the party, Mr. Jones asks everyone else how many people they shook hands with, and finds that everyone gives a different answer. No one shook hands with his/her spouse and with the same person twice. How many people did Mrs. Jones shake hands with?

Solution:
There are 10 people in total, and the largest number of people one could shake hands with is $10 - 2 = 8$, the 9 numbers of shake hands is exactly form $0 - 8$.

- p_8 shacked all the others' hands expect his/her spouse, p_8, p_0 are couple.
- p_7 shacked hands with all others except for his/her spouse, this could either be p_0 or p_1, since p_0 was married with p_8, therefore, p_7, p_1 are couple.
- Finally we have p_4 should be married with another people who is also p_4, considered the other nine people have distinct hand shake times, Mrs. Jones must shacked hand for 4 times.

Chapter 3

Monte Carlo Simulations & Stochastic Calculus

3.1 How to generate π using Monte Carlo method? What is the std of this method?[1]

Solution:
Consider a square as $[-1,1] \times [-1,1]$. The Monte Carl method states that you randomly throwing N points in this square, assume the number of points lie in the circle $x^2 + y^2 \leq 1$ is A, then

$$\frac{A}{N} = \frac{\pi}{4}$$

Let $U_1, U_2, \cdots$ be a sequence of i.i.d variables uniformly distributed in $[-1,1] \times [-1,1]$, denote by $1_{D(0,1)}$ the indicator function of the unite disk $D(0,1)$, i.e.,

$$1_{D(0,1)}(x,y) = \begin{cases} 1, \text{if } (x,y) \in D(0,1) \\ 0, \ otherwise \end{cases}$$

Let

$$\begin{aligned}
&X_i = 1_{D(0,1)}(U_i), \ \forall i \geq 1. \\
&E(X_i) = \frac{\pi}{4} \\
&E(X_i^2) = 1^2 \times \frac{\pi}{4} = \frac{\pi}{4} \\
&Var(X_i) = \frac{4\pi - \pi^2}{16}
\end{aligned}$$

note for N large enougth:

$$\lim_{N \to \infty} \frac{\sum_i^N X}{N} = \frac{\pi}{4}$$

Thus:

$$Var(\frac{\sum_i^N X}{N}) = \frac{1}{N} \frac{4\pi - \pi^2}{16}$$

By throwing as many as N points and calculate the variance, it should approach $Var(\frac{\sum_i^N X}{N})$.

3.2 How to generate independent samples of standard normal distribution

- Box-Muller Method
- Acceptance-Muller
- Inverse Transform Sampling

3.3 Brownian motion

3.3.1 Definition

Brownian motion is the random motion of particles suspended in a medium (a liquid or a gas)[5]. A Wiener process is a real valued continuous-time stochastic process defined as $W(t \geq 0)$, and is a Brownian motion if:

- W(0) = 0
- for any $W(t_{n+1}) - W(t_n)$ are independent.
- $W(t_{n+1}) - W(t_n) \sim \mathcal{N}(0, t_{i+1} - t_i)$.
- $E[W(t)] = 0, E[W(t)^2] = t, W(t) \sim \mathcal{N}(0,1)$
- $E[W(t+s) \mid W(t)] = W(t); \quad \operatorname{cov}(W(s), W(t)) = s$
- $Y(t) = W(t)^2 - t$ is a martingale.
- $Z(t) = \exp\left\{\lambda W(t) - \frac{1}{2}\lambda^2 t\right\}$, where λ is any constant and $W(t)$ is a Brownian motion, is a martingale. (Exponential martingale).

3.4 Ito's lemma

Assume X_t is an Itô drift-diffusion process that satisfies the stochastic differential equation

$$dX_t = \mu_t dt + \sigma_t dW_t$$

If $f(t,x)$ is a twice-differentiable scalar function, its expansion in a Taylor series is

$$df = \frac{\partial f}{\partial t}dt + \frac{\partial f}{\partial x}dx + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}dx^2 + \cdots .$$

Substituting X_t for x and therefore $\mu_t dt + \sigma_t dW_t$ for dx gives

$$df = \frac{\partial f}{\partial t}dt + \frac{\partial f}{\partial x}(\mu_t dt + \sigma_t dW_t) + \frac{1}{2}\frac{\partial^2 f}{\partial x^2}\left(\mu_t^2 dt^2 + 2\mu_t\sigma_t dt dW_t + \sigma_t^2 dW_t^2\right) + \cdots$$

quadratic variance of a Wiener process), and collecting the dt and dB terms, we obtain

$$df = \left(\frac{\partial f}{\partial t} + \mu_t\frac{\partial f}{\partial x} + \frac{\sigma_t^2}{2}\frac{\partial^2 f}{\partial x^2}\right)dt + \sigma_t\frac{\partial f}{\partial x}dW_t$$

as required.

Part II

Data Science

Chapter 4

A/B Test & Hypothesis Test

4.1 Introduction

How the establish a statistical test for the hypothesis?

1 Propose the null hypothesis H_0 and the alternative hypothesis H_1

2 Choose test method(Wald, LRT, Score)

3 Choose the rejection parameter, usually for $\alpha = 0.05$

4 Run your test and get the corresponding p value

5 Make a decision based on the results

4.2 Likelihood ratio test

- H_0: parameter $\theta \in \Theta_0$ vs H_1: parameter $\theta \in \Theta_1$
- $llr(\theta_0) = l_{H_0} - l_{H_1} = l_{\theta_0} - l_{\theta_1} \equiv log(\frac{max_{H_0} L(\theta)}{max_{H_1} L(\theta)}) = max_{H_0} l(\theta) - max_{H_1} l(\theta)$
- check if $-2llr(\theta_0) > \chi^2_{1,1-\alpha}$

Example:
For a sample size of 100 which follows Binomial distribution, if 60 people are tested as positive for one test, run llr to check $\pi_0 = 0.5$ with $\alpha = 0.05$.

Solution:

$$
\begin{aligned}
&f(k = 60) = \binom{100}{60} \pi^{60} (1 - \pi)^{40} \\
\Rightarrow & l(\pi|k = 60) = 60 \log(\pi) + 40 \log(1 - \pi) \\
\Rightarrow & l'(\pi|k = 60) = \frac{60}{\pi} + \frac{40}{1 - \pi} = 0 \\
\Rightarrow & \hat{\pi} = 0.6 \Rightarrow -2llr = -2(60 \log(0.5) + 40 \log(0.5) - 60 \log(0.6) + 40 \log(0.4)) = 4.0271 > P(\chi^2_{1,0.95}) = 3.84.
\end{aligned}
$$

Thus we reject the null hypothesis.

Remark There are other testing methods such as Wald and Score, however llr should be the most popular and straightforward way to run the test. For testing method like Score which requires prerequisite about Fisher Information, I don't think it is necessary to review it here.

4.3 Degrees of freedom

Degrees of freedom is the number of values in the final calculation of a statistic that are free to vary.

4.4 p-value

The p-value is the probability of obtaining test results at least as extreme as the results actually observed.

- The level of statistical significance is often expressed as a p-value between 0 and 1.
- The smaller p-value we have, the stronger evidence we could have to reject null hypothesis.
- p-value could tell something, but not everything as it still could have false positive case, many statisticians doesn't rely on or believe p-value in their research.
- when the variables are correlated, the p-value migt be smaller and thus we can't not reject the null hypothesis. (Example, correlation between residuals in linear regression). 6.3.2

4.5 Standard Error

- The standard error (SE) of a statistic (usually an estimate of a parameter) is the standard deviation of its sampling distribution.[6]
- if we are picking form an known distribution, the standard error should be equal to the std of the distribution:

$$Var(\sum_{1}^{n} X_i) = nVar(X_i)$$

$$\Rightarrow se = \frac{\sqrt{n} std(X_i)}{\sqrt{n}} = std(X_i)$$

- In t-test, since we don't know weather if the samples are from the same distribution, we should still divide by $\sqrt{n}$ and $\sigma_{\bar{x}} = \frac{\sigma(\mathbf{X})}{\sqrt{n}}$.

4.6 Confidence interval

Confidence interval is a inference or estimate computed from the observed data. CI gives a range of values fro an unknown parameter(mean, coefficient). 95% confidence interval implies that the observed true data range is likely to lie with 95% confidence, or 95% of the data points will lie in this confidence interval. Lower and Upper Confidence Limits:

$$\bar{x} \pm (T^*_{df} \text{ or } Z^*_{df}) \times (SE_{\bar{x}} = \frac{\hat{\sigma}}{\sqrt{n}})$$

4.7 T test

- Usually to test if the mean of two data sets are the same.

1 One sample t test: $t = \frac{Z}{s} = \frac{\bar{X} - \mu}{\hat{\sigma}/\sqrt{n}}$

2 Equal sample sizes and variance, Two sample t test(pooled variances).

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_p \sqrt{\frac{2}{n}}}, s_p = \sqrt{\frac{s^2_{X_1} + s^2_{X_2}}{2}}$$

3 Unequal size, pooled variance:

$$t = \frac{\bar{X}_1 - \bar{X}_2}{S_p\sqrt{\frac{1}{n_A} + \frac{1}{n_B}}}, \; S_p = \sqrt{\frac{s_{X_A}^2 + s_{X_B}^2}{2}},$$

4 Two sample t test(separate variances):$t = \frac{\bar{X}_1-\bar{X}_2}{\sqrt{\frac{S_A^2}{n_A}+\frac{S_B^2}{n_B}}}$.

5 Equal or unequal sample sizes, similar variances $\left(\frac{1}{2} < \frac{s_{X_1}}{s_{X_2}} < 2\right)$.

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_p \cdot \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}, \; s_p = \sqrt{\frac{(n_1 - 1)\, s_{X_1}^2 + (n_2 - 1)\, s_{X_2}^2}{n_1 + n_2 - 2}}$$

6 Equal or unequal sample sizes, unequal variances ($s_{x_1} > 2s_{x_2}$ or $s_{x_2} > 2s_{x_1}$),

$$t = \frac{\bar{X}_1 - \bar{X}_2}{s_{\bar{\Delta}}}, \; s_{\bar{\Delta}} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

4.8 Z test

When distribution parameters (variance) are unknown, a Student's t-test should be conducted instead.

- Usually test for those with known variance.
- $Z = \frac{\bar{X}-\mu_0}{\sigma}$

4.9 Chi square test

Check is there is statistical significance between given distribution and expected distribution.(calculate by subgroup means)

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

4.10 A/B Test

A/B testing is a testing methodology in verifying weather a new module, a new function, or a new product is effective based on users' experience.

- Not only to find which version of the product is better, but also check the statistical significance between two versions.

4.10.1 How to determine a successful A/B test?

- Use statistical significance to check if your method is better than the previous model.
- **statistical significance** shows how unlikely the result will occurred given the null hypothesis.

4.10.2 Type I/II Error, Test Power.

- Type I error: **Reject the null hypothesis when it should not be rejected.** The probability when Type I error happens is the significance value α, for example 0.5.
(False Positive: find an important email in your spam mail box.)
- Type II error: **Fail to reject the null hypothesis when it should be rejected.** The probability of it is β.
(False Negative: find an spam email in your regular mail box.)
- **Statistical power** if the probability to reject the null hypothesis when it should be rejected, as $1-\beta$.
- Usually statistical power increases as sample size increases. A common value for $1-\beta = 0.8$.

4.10.3 Tradeoff Between Type I and Type II errors

- If we require very strong evidence to reject null hypothesis, this may lead to a high type II error. High precision, low recall.
- If we require very weeker evidence to reject null hypothesis to reduce to type II error, this may lead to a high type I error. low precision, high recall.

4.10.4 How to determine your A/B test sample?

Sample size for comapring two means, for example, how long time period (days) we need to track for a improvement.

- for each group size n,

$$n = \frac{\left(\sigma_1^2+\sigma_2^2\right)\left(z_{1-\alpha/2}+z_{1-\beta}\right)^2}{\Delta^2}$$
$$\Delta = |\mu_1-\mu_2|,\ \textbf{group 1} \sim (\mu_1,\sigma_1^2),\ \textbf{group 2}(\mu_2,\sigma_2^2)$$

- α is the significance rate, β is the power of a test.

Sample size for each group individuals, for example, how many people we need to check the statistical significance. Sample Size Needed to Compare Two Binomial Proportions Using a Two-Sided Test with Significance Level α and Power $1-\beta$, Where One Sample (n_2) Is k Times as Large as the Other Sample (n_1) (Independent-Sample Case)

To test the hypothesis $H_0 : p_1 = p_2$ vs. $H_1 : p_1 \neq p_2$ for the specific alternative $|p_1 - p_2| = \Delta$, with a significance level α and power $1-\beta$, the following sample size is required [7]

$$n_1 = \left[\sqrt{\bar{p}\bar{q}\left(1+\frac{1}{k}\right)}z_{1-\alpha/2} + \sqrt{p_1q_1+\frac{p_2q_2}{k}}z_{1-\beta}\right]^2/\Delta^2$$
$$n_2 = kn_1$$

where p_1, p_2 = projected true probabilities of success in the two groups

$$q_1, q_2 = 1-p_1, 1-p_2$$
$$\Delta = |p_2-p_1|$$
$$\bar{p} = \frac{p_1+kp_2}{1+k}$$
$$\bar{q} = 1-\bar{p}$$

Example: **Compare two version of website which is more affordable by users.** Steps:

1 Call out your criteria/definition of success before test. What's your hypothesis.

2 Identify goals:

  - More CTR
  - More sign-up rates

3 Split your users into two groups(doesn't have to be 50% by 50%). To make sure your test pool is enough for the statistical significance.

4 Collect the experimental data and run statistical signifincant test on your hypothesis.(χ^2 test, z test) check Statistical significance in A/B testing

4.10.5 What if you can't use A/B test? Counterfactual

Refenrence: What To Do When You Can't AB Test

- **Why?** When the customer level randomization might be impossible.(Some companies has a lot of offline business, like local shops, where it is hard to randomize the customer level.)

4.10.5.1 Market Based Approaches

Example:

- Measure impacts between geographic areas when an effect is introduced.
- If we want to introduce a promo to a new area, we can't do A/B test.
- New area: Treatment region vs Exist business area (Control region).
- Feed the control region data in a regression model and use the new features in new area to predict the result first.
- Measure the results between the predicted values and the exact values in treatment regions.

4.10.5.2 Time-Series Based Approaches: Google's Causal Impact

Example,

- Measure impacts between the data can't be split geographically.
- Promos to a specific product, then the customer randomize may be inaccurate.
- Measure the impact at a specific time series right after this promotion is activated.
- Control group:(time series of combinations of different product, less sensitive to the promo product.)
- Treatment group:(time series of combinations of different product includes promo product).
- Use predictive model such as Google's Causal Impact to get the confidence interval as compare with them.

Chapter 5

Data Science & Product Q & A

5.1 SQL

There is a leetcode tage called SQL70, you could pratice on it but I have to say that: I hate SQL.

5.2 Keys for Product Sense

1 Understand the crucial goals of the product form business side and customer side.(write down all the goals you could image.)

2 Design metrics to evaluate the goals, or the results of the product.(who is the business side, who will be the customer?)

3 Explore more meaningful features, in this part, check the marketing report and compare and analysis from other similar successful product might be helpful.(Why they want purchase your item?)

4 How you evaluate your features?(Hypothesis test and metrics design, AB test).

5.3 Product

- **Clarify your questions/definition**
- **Avoid miscommunication**
- **Think in the perspective of user**

5.4 Case Study

5.4.1 How to investigate Friend requests are down 10%

1 Definition and clarification.

  - Definition of friend requests.(How to add friend.) Clarify the time period of the drop? Month to month or quarter to quarter.

2 Content

  - Global or regional?
  - Is this process related to a specific algorithm? Mostly recommendation algorithms.(Then think about the update or rolling back to the lasted version.)
  - Is this the only decline? If other metrics also declined then think the correlation with other metrics.

3 Hypothesis

- Issue in recommendation system.
- Consider more fake accounts was recently created.
- Difference between browser and mobile and check the geography on that.

Part III

Machine Learning & Deep Learning

Chapter 6

Machine Learning & Statistical Learning

6.1 Data Preprocess

This chapter follows the great book 百面机器学习[8].

6.1.1 Why we need Feature Scaling?

Scaling such as **normalization** and **standardization** plays a crucial role in data pre-processing. There two motivations for that, see Wikipedia

1 Since the range of values of raw data varies widely, in some machine learning algorithms, objective functions will not work properly without normalization. For example, many classifiers calculate the distance between two points by the Euclidean distance. If one of the features has a broad range of values, the distance will be governed by this particular feature. Therefore, the range of all features should be normalized so that each feature contributes approximately proportionately to the final distance. (In deep learning, normalization will help to convergence by gradient descent quickly.)

2 Another reason why feature scaling is applied is that gradient descent converges much faster with feature scaling than without it.

Some scaling method:

$$X_i^{norm} = \frac{X_i - X_{min}}{X_{max} - X_{min}} \sim [0, 1], \ \textbf{min-max scaling}$$

$$z = \frac{x - \mu}{\sigma}, \ \textbf{Z-score Normalization}$$

Note: Feature scaling is important for models use gradient descent, however, for models like **Tree Model**, which relies on the increment of the entropy, feature scaling may not be useful.

6.1.2 Categorical Variable

A categorical variable is a variable that can take on one of a limited, and usually fixed, number of possible values, assigning each individual or other unit of observation to a particular group or nominal category on the basis of some qualitative property.[9]

6.1.3 Ordinal Encoding

Ordinal Encoding usually process the categorical data with ranking information and assign such data with new integer values. For example, translate size(Large, Medium, Small) to $(3, 2, 1)$.

6.1.4 One-hot Encoding

One-hot Encoding deals with the categorical data which don't have ordinary relationship, for example the blood type(A,B,C,AB).
The way to process such data is to translate it to a sparse matrix with vectors like $[1, 0, 0, 0]$, $[0, 1, 0, 0]$, $[0, 0, 1, 0]$, $[0, 0, 0, 1]$
Note:

1 sparse matrix could save space.

2 Carefully choose the useful features to reduce the dimensions of the encoding result. Curse of dimensionality is always as issue in machine learning. For example, for KNN, it is hard to measure the distance in high dimension space and for logistic regression, due to the increment of the parameters, model may suffer over-fitting.

6.1.5 Binary Encoding

Blood type example: $[A, B, C, AB] \to [1, 2, 3, 4] \to [(001), (010), (011), (100))](Binary\ numbers)$. Both one-hot encoding and binary encoding are actually mapping functional which maps the categorical data into new $[0, 1]$ features. We could see that the dimensions of *binary encoding* is usuallt smaller than $one - hotencoding$.

6.1.6 Feature Crosses

- Use tree model to do the dimension reduction, define each path as a one-hot code. (page 31 [8])
- Please also check GDBT for more details.

6.1.7 Imbalanced data

For a classification problem, if we have one class set which only consists a very small portion of the whole data set, the classifier may not be able to learn enough information form that data set, which leads to a poor model.
For example, in a credit card fraud problem, suppose we have 99 regular accounts and 1 fraud accounts, then if the model is good at figuring the regular accounts but poorly in those fraud detection, this should be a failed machine learning model. Even the precision is 99%, but the specificity is 0.

Methods

- Oversampling.(SMOTE)
  1 For any data point, find the k^{th} nearest points.
  2 Randomly choose one point among those k points
  3 $x_{new} = x_{old} + rand()(x_k - x_{old})$
- Undersampling, rather than increase the "smaller" class, we could also decrease the "larger" class to make the distributions of two classes "equal".
- Bagging
- Boosting
- Cross validation

6.2 Model Evaluation

Different models may require different evaluation indicators.

6.2.1 Accuracy

$$Accuracy = \frac{n_{correct}}{n_{total}}$$

Question: Luxury advertiser A wish to target their ads to luxury customers through a third party data company B. B got the data and used statistical model to classify the customer group. The accuracy is 95%. But the advertiser still casts most of its ads to non-luxury customers, why?

The accuracy is a straightforward way to evaluate the model, however, the weakness is also obvious: if there is only 1% portion of the group are luxury customers and the model will still have a 99% accuracy if it classifies all group luxury customers to non-luxury customers. (For example, the credit card fraud).
Then one possible reason is that the luxury customers of **A** only take up a small part of the total customers, and the predicted luxury customers may still be the non-luxury customers.

6.2.2 Precision and Recall

		True condition				
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
Predicted condition	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{TPR}{FPR}$	Diagnostic odds ratio (DOR) = $\frac{LR+}{LR-}$	F_1 score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{FNR}{TNR}$		

Figure 6.1: Figure source: Wikipedia

$$precision = \frac{tp}{tp + fp}$$
$$recall = \frac{tp}{tp + fn}$$

Precision means the fraction of predicted true positive in total predicted positive samples.
Recall means the fraction of predicted true positive of in total true positive samples.

- Check the relationship between Type I error and Type II error at 4.10.3.

Question: A website offers fuzzy search for videos and the top 5 precision results of the model is good, but the customers still can't find their desired videos, why?
In a statistical model for classification, we usually uses predicted *top N* results to calculate the precision and recall. Since the precision is good then there might be some problems in recall. Which means the model classified many positive samples to negative(FN). Which implies the website didn't find the required videos enough. For example, if there are 100 true positive samples and you uses @5, the precision could reach 100% however the recall will be 5%. To better evaluate the model we may need to see how the precision and recall performs under different (N), which is the **P-R Curve**.

6.2.3 F1 Score

F1 score shows us how accuracy of the model from 0 to 1 as:

$$F1_{score} = \frac{2Precision\ Recall}{Precision + Recall}$$

6.2.4 RMSE

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}}$$

Question: In a regression model, no matter how you choose the model, the RMSE is always higher than expected. However in real test data, the model RMSE is less than 1% for 95% of the samples, give a possible reason why this happens.

If the model works well for 95% of the samples, then the rest 5% predictions may leads to a high total error. You could reduce the RMSE in following ways:

1 Define those samples as outliers and filter it before modeling.

2 Apply more technics in modeling if you don't think those are outliers.

3 Use $MAPE = \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times \frac{100}{n}$ to evaluate the error.

6.2.5 MAE vs MSE

- MAE= $\frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}|$, l1 loss. More robust to outliers, but hard to estimate in optimization.
- MSE = RMSE^2 = $\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y})^2$, good for differentiation.

6.2.6 AUC & ROC Curve

ROC curve means the curve of **False Positive Rate, x-axis** vs **True Positive Rate, y-axis**.

$$FPR = \frac{FP}{N} = 1 - specificity$$
$$TPR = \frac{TP}{P} = recall$$
$$ROC = \frac{recall}{1 - specificity}$$

Example: 10 patients are doing cancer diagnose tests. 3 of them are diagnosed with cancer and 1 of them were misdiagnosed. What is the position of this test in a ROC curve?

$x = FPR = \frac{1}{7}, y = TPR = \frac{2}{3}$.

AUC: is the area under ROC curve, a reasonable AUC has a value between 0.5 and 1. Which means the model is better than random guess.
ROC is more widely used in sorting algorithms than P-R Curve because ROC won't change severely if the dataset changes. However, PR-Curve will change much if modeling the imbalanced data.

6.2.7 Cosine similarity

For two vectors A and B,

$$Cos(A, B) = \frac{A \cdot B}{||A||_2||B||_2} \sim [0, 1]$$
$$dist(A, B) = 1 - cos(A, B) \sim [0, 2]$$

Question: Why we use Cosine similarity instead of euclidean metric?

Cosine similarity is better than euclidean in modeling those similarity vectors with huge differences in length. Also, in modeling text, image and videos, the dimensions of the features are usually large which makes it hard to use euclidean distance. Finally, cosine similarity is able to present the correlation even in high dimensional space: **same:1, orthogonal:0, opposite:-1**. **Question: when to use Cosine similarity and when to use Euclidean distance?**

It depends, for example if $A = (0, 1)$ and $B = (1, 0)$ then $1 - cos(A, B) = 1$ which is big but $Euclidean = 0$, however, A and B have huge difference and we need to use Cosine similarity.
Another example is to check two customers activity by giving their login time and watching time vector. if $A = (1, 10)$ and $B = (10, 100)$, then their cosine similarity is small but they do have huge differences indeed, then we would like to use Euclidean distance.
For NLP problems such as comparing two word vector, we usually use cosin similarity.
Question: Is Cosine similarity a well defined metrics?

No, three conditions:

1 **Positive-definite✓:** $dist(A, B) \geq 0$

2 **Symmetry✓:** $dist(A, B) = dist(B, A)$

3 **Triangle inequality ✗: consider** $A = (1, 0), B = (1, 1), C = (0, 1)$ **Then** $dist(A, B) + dist(B, C) < dist(A, C)$

6.2.8 Overfitting and underfitting

In experiments, Overfitting means model performs well in training data but poorly in testing data. An overfitted model is a model which is over complicated(more parameters) than the original data could justify[10].
Underfitting means a model performs poorly on both training and testing data, which means the model is too simple, An under-fitted model is a model where some parameters or terms that would appear in a correctly specified model are missing[10]

Question: How to avoid overfitting?

1 Start form the original data, collect more useful data in training. However, collecting new data is hard, one could add more data by changing the current data, for example, rotate the image data. Furthermore, one could use GAN to generate more new data.

2 Reduce the model complexity. Complicated model with small data is usually the main reason. Reduction could also help to avoid modeling the noise in data. In neural network, one could delete some hidden layers, neurons. In tree model, reduce the depth and cut the tree.

3 Regularization, add more regularized constrains on parameters. For example, add penalty in the L^2 loss function:

$$C = C_0 + \frac{\lambda}{2n} \sum w_i^2$$

Where C_0 is the original loss function.

4 Ensemble meta-algorithm such as bagging (Bootstrap aggregating, Cross-Validation, random forest).

6.2.8.1 Cross-Validation

1 randomly divide the training data into K folds (Often K = 5 or K = 10).

2 For $k = 1, \cdots K$

   a Use all samples expect those in fold k to train.

   b Use samples in fold k to test, and let the number of errors be m_k.

3 The final CV error is $\frac{m_1 + \cdots m_K}{n}$, where n is the number of trails.

6.2.8.2 Bagging (bootstrap aggregating)

1 bootstrap: resample with replacement (to the same sample size as the original data).

2 When sample size n is large, each set of bootstrap samples contains about 63% of the original samples(with replicates).

$$1 - (1 - \frac{1}{n})^n = 1 - e^{-1} \approx 63\%$$

Question: How avoid underfitting?

1 Add new features.

2 Increase the complexity of the model. For example, in linear model, add new nonlinear variables. In neural network, add new layers or neurons.

3 Decrease the coefficients of the regularized model parameters in loss function.

6.2.9 Bias Variance tradeoff

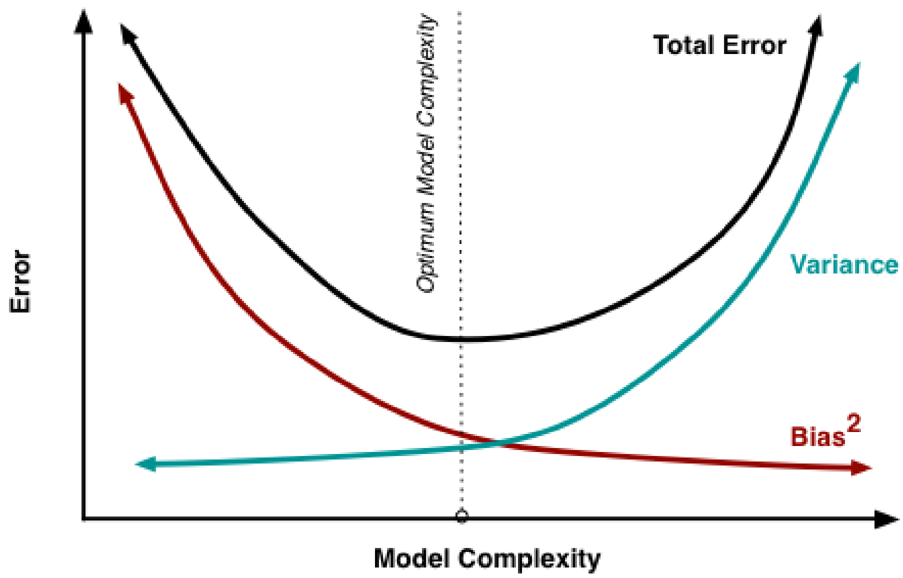
$$MSE = E[(\hat{y} - y_0)^2]$$
$$Variance = E[(\hat{y} - E(\hat{y}))^2]$$
$$Bias = E(\hat{y}) - y_0$$

Where $\hat{y}$ is the true value and y_0 is the regressed value. One could prove that:

$$MSE = Bias^2 + Variance + (\sigma^2 \sim \mathcal{N}(0,1))$$

1 Overfiiting model means large variance but small bias.

2 Underfitting model means small variance but large bias.



Thus we need to choose a proper model with the balance between bias and variance.

6.3 Linear Regression

Linear regression is a very basic, simple, but extremely important model in statistical learning. In this section, I will follow the famous classical textbook The Elements of Statistical Learning(ESL)[11].

6.3.1 what is linear regression and why we use it?

Linear regression is a kind of regression model that we assume the regression function $E(Y|X)$ is linear with the inputs X_j. Assume we have inputs X_j and we want to predict a real-valued response variable Y, we have:

$$\hat{y} = E(Y|X) = f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$$

In the definition of *Linear*, as long as the input variables are numeric, we only cares about the unknown coefficients β_j which is lienar with Y, thus we call it linear regression. The inputs X_j can be,

- numerical variables
- transformations like $\log, \tanh$
- X_j^2, X_j^3
- categorical variables, in this case it should be $\sum_{j=1}^{k} X_j \beta_j, k$ is the number of categories.
- Intersections as $X_i X_j$

- Simple, easy to describe the correlation between inputs and respond variable.
- Good for inference. (uncorrelated features)
- You don't need to use a nonlinear model to solve a simple question with large computational costs.

6.3.2 Linear Regression Assumptions

Reference: Testing the assumptions of linear regression.

- Linear relationship: there is linear relationship between dependent variable y and independent variable $\mathbf{X}$.
  - Check $\rho_{X,Y} = \text{corr}(X,Y) = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$
  - Check plots.
- Nonperfect multicollinearity: this may lead to the The Rank Deficient Problem 6.3.13.
  - check Variance inflation factor (VIF)(10: high, 5: cutoff).
- Heteroskedasticity of residual ϵ 6.3.7, this will lead to inference issue for coefficients $\hat{\beta}$.
  1. versus the independent variables.
  2. versus time.

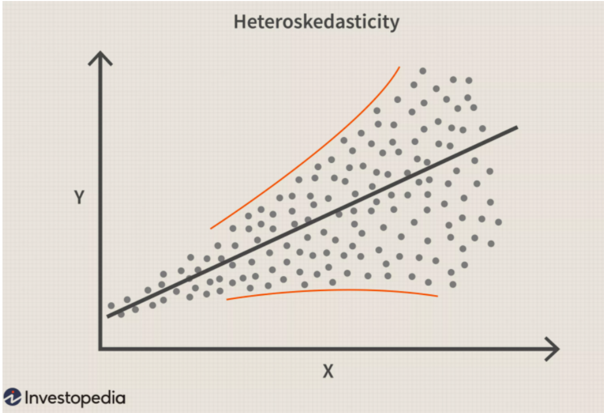


Figure 6.2: Figure source: image by Julie Bang

- No Autocorrelation for residuals
  - Issue in inference of the standard errors.

$$e_t = \rho e_{t-1} + \omega_t, \ 0 < \rho \leq 1, \omega_i \sim \mathcal{N}(0, \sigma^2)$$
$$\text{Cov}(e_t, e_{t-1}) = \text{Cov}(\rho e_{t-1} + \omega_t, e_{t-1}) = \rho \left(\frac{\sigma^2}{1-\rho^2} \right), \ \text{Cor}(e_t, e_{t-1}) = \rho$$

  - Serial correlation is also sometimes a byproduct of a violation of the linearity assumption–as in the case of a simple (i.e., straight) trend line fitted to data which are growing exponentially over time.
  - If correlated, estimated standard errors will tend to underestimate the true standard errors.(By optimization theory, estimate under another related distribution.) Which will lead to a narrower confidence interval and smaller p value.
  - Durbin-Watson Test to check. Autocorrelation Plot.
- Normally distributed residuals
  - If this assumptions don't hold, the inferences will use the incorrect probabilities from distributions.
  - y and $\mathbf{X}$ don't need to be normal in the assumption. But we usually normalize y and all the features $\mathbf{X}$ for a better interpretable model (mean is 0, good to get the close form and decrease the variace of $\hat{\beta} = (\mathbf{X}'\mathbf{X})^{-1}\sigma^2$).
  - Outliers may leads to Non-normal residual.
  - Use p=p plot to check the residual.

6.3.3 Ordinary Least Square(OLS)

- We have training data $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$, with every $x_i = (x_{i1}, x_{i2}, \cdots x_{ip})^T$, p is the dimension of the inputs, or number of the features in practice.
- The OLS algorithm estimates the coefficients β to minimize the following:

$$\begin{aligned}\arg\min RSS(\beta) &= \sum_{i}^{N}(y_i - f(x_i))^2 \\ &= \sum_{i=1}^{N}\left(y_i - \sum_{j=1}^{N} x_{ij}\beta_j - \beta_0\right)^2\end{aligned}$$

- This is a convex function thus ensures we will have the global minimum.
- Now represent the regression function by metrics, let training set $\mathbf{X} \in R^{n\times(p+1)}$, $\mathbf{y} \in R^{n\times 1}$, we have:

$$\begin{aligned}&f(x) = f(\mathbf{X}) = \mathbf{X}\beta = \hat{\mathbf{y}} \\ &RSS(\beta) = (y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta) \\ &\frac{\partial RSS(\beta)}{\partial \beta} = -2\mathbf{X}^T\mathbf{y} + 2\mathbf{X}^T\mathbf{X}\beta = 0 \\ &\Rightarrow \hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\ &\hat{y} = \underbrace{\mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T}_{\mathbf{H}:hat}\mathbf{y}\end{aligned}$$

- The final answer $\hat{\mathbf{y}}$ is a Orthogonal, Projection on the dimensional space of $\mathbf{X}$. And the residual is orthogonal to the space of $\mathbf{X}$ as $\mathbf{X}^T(\mathbf{y} - \mathbf{X}\beta) = 0$

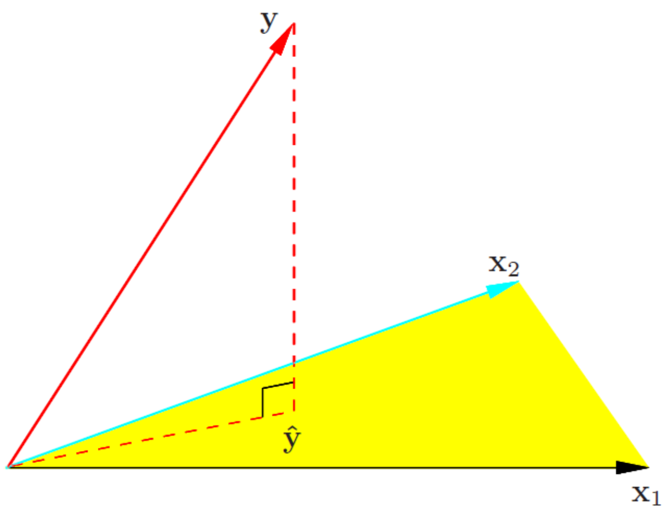


FIGURE 3.2. *The N-dimensional geometry of least squares regression with two predictors. The outcome vector $\mathbf{y}$ is orthogonally projected onto the hyperplane spanned by the input vectors $\mathbf{x}_1$ and $\mathbf{x}_2$. The projection $\hat{\mathbf{y}}$ represents the vector of the least squares predictions*

Figure 6.3: Figure source: [11]

6.3.4 Properties of OLS

$$\begin{aligned}&y = \mathbf{X}\beta + \epsilon \text{ (true model)} \\ &\hat{y} = \mathbf{X}\hat{\beta} \text{ (OLS model)} \\ &\hat{y} - y = e \equiv y = \mathbf{X}\hat{\beta} + e\end{aligned}$$

e is the residual term, and ϵ is the error term that we want to minimize.

6.3.4.1 OLS estimator β with correlation.

$$\hat{\beta}_{y\sim\mathbf{X}} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} = \frac{Cov(x,y)}{Var(x)} = corr(x,y)\frac{\sigma(y)}{\sigma(x)}$$
$$\hat{\beta}_{\mathbf{X}\sim y} = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} = \frac{Cov(x,y)}{Var(y)} = corr(x,y)\frac{\sigma(x)}{\sigma(y)}$$
$$\Rightarrow \hat{\beta}_{y\sim\mathbf{X}}\hat{\beta}_{\mathbf{X}\sim y} = corr(x,y)^2.$$

6.3.4.2 OLS is an unbiased model.

True Model:

$$y = \mathbf{X}\beta + \epsilon$$
$$\epsilon = y - \mathbf{X}\beta$$
$$\hat{\beta} = \arg\min(\epsilon^2)$$
$$\Rightarrow \hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T(\mathbf{X}\beta + \epsilon)$$
$$\Rightarrow \hat{\beta} = \boldsymbol{I}\beta + \left(\boldsymbol{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\epsilon$$
$$\Rightarrow E(\hat{\beta}) = \beta$$

6.3.4.3 Residual e is uncorrelated with $\mathbf{X}$

$$(\mathbf{X}'\mathbf{X})\hat{\beta} = \mathbf{X}'y$$
$$y = (\mathbf{X}\hat{\beta} + e)$$
$$\Rightarrow (\mathbf{X}'\mathbf{X})\hat{\beta} = \mathbf{X}'(\mathbf{X}\hat{\beta} + e)$$
$$\Rightarrow \mathbf{X}'e = 0.$$

- The observed values of $\mathbf{X}$ are uncorrelated with residuals.
- if $\mathbf{X}'$ is all ones, then $\sum e = 0$ as $E(e) = 0$.

6.3.4.4 Prove that the regression line must pass $\bar{\mathbf{X}}, \bar{y}$

$$\bar{e} = E(e) = \frac{y - \mathbf{X}\hat{\beta}}{n} = \bar{y} - \bar{\mathbf{X}}\hat{\beta}$$

6.3.4.5 $\hat{y}$ are uncorrelated with residual e

$$\hat{y}'e = (\mathbf{X}\hat{\beta})'e = \hat{\beta}'\mathbf{X}'e = 0$$

- One could also find that $\bar{\hat{y}} = \bar{y}$.

6.3.5 Variance

Problem: What is the variance-covariance matrix of $\hat{\beta}$?

$$\begin{aligned}
&\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\
&\Rightarrow Var(\hat{\beta}) = \sigma^2 \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{X}\left(\mathbf{X}^T\mathbf{X}\right)^{-1} \\
&= \sigma^2 \left(\mathbf{X}^T\mathbf{X}\right)^{-1}
\end{aligned}$$

The variance σ^2 is estimated by

$$\hat{\sigma}^2 = \frac{1}{N-p-1}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

The $N-p-1$ rather than N in the denominator makes $\hat{\sigma}^2$ an unbiased estimate of $\sigma^2 : \mathrm{E}\left(\hat{\sigma}^2\right) = \sigma^2$.

6.3.6 Variance-covariance Matrix of OLS

$$\begin{aligned}
\hat{\beta} = \mathbf{X}(\mathbf{X}'\mathbf{X}^T)^{-1}y = \mathbf{X}(\mathbf{X}'\mathbf{X}^T)^{-1}(\mathbf{X}\beta+\epsilon) &\Rightarrow \hat{\beta}-\beta = \mathbf{X}(\mathbf{X}'\mathbf{X}^T)^{-1}\epsilon \\
E\left[(\hat{\beta}-\beta)(\hat{\beta}-\beta)'\right] &= E\left[\left(\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\epsilon\right)\left(\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\epsilon\right)'\right] \\
&= E\left[\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\epsilon\epsilon'\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\right] \\
&= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'E\left[\epsilon\epsilon'\right]\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1} \\
&= \left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{I}\sigma^2\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1} \\
&= \sigma^2\left(\mathbf{X}'\mathbf{X}\right)^{-1}
\end{aligned}$$

$$E\left[(\hat{\beta}-\beta)(\hat{\beta}-\beta)'\right] = \begin{bmatrix}
\operatorname{var}\left(\hat{\beta}_1\right) & \operatorname{cov}\left(\hat{\beta}_1,\hat{\beta}_2\right) & \ldots & \operatorname{cov}\left(\hat{\beta}_1,\hat{\beta}_k\right) \\
\operatorname{cov}\left(\hat{\beta}_2,\hat{\beta}_1\right) & \operatorname{var}\left(\hat{\beta}_2\right) & \ldots & \operatorname{cov}\left(\hat{\beta}_2,\hat{\beta}_k\right) \\
\vdots & \vdots & \vdots & \vdots \\
\operatorname{cov}\left(\hat{\beta}_k,\hat{\beta}_1\right) & \operatorname{cov}\left(\hat{\beta}_k,\hat{\beta}_2\right) & \ldots & \operatorname{var}\left(\hat{\beta}_k\right)
\end{bmatrix}$$

6.3.7 Heteroskedasticity

- We mentioned the var-cov matrix of β above and assume

$$E[\epsilon'\epsilon] = \mathbf{I}\sigma^2$$

- Here we assume $\forall Var(\epsilon_i|\mathbf{X}) = \sigma^2$, However this is not always true, which is called Heteroskedasticity. For example:

$$\begin{aligned}
E\left(\epsilon\epsilon' \mid X\right) &= \begin{bmatrix}
E\left[\epsilon_1^2 \mid X\right] & E\left[\epsilon_1\epsilon_2 \mid X\right] & \ldots & E\left[\epsilon_1\epsilon_n \mid X\right] \\
E\left[\epsilon_2\epsilon_1 \mid X\right] & E\left[\epsilon_2^2 \mid X\right] & \ldots & E\left[\epsilon_2\epsilon_n \mid X\right] \\
\vdots & \vdots & \vdots & \vdots \\
E\left[\epsilon_n\epsilon_1 \mid X\right] & E\left[\epsilon_n\epsilon_2 \mid X\right] & \ldots & E\left[\epsilon_n^2 \mid X\right]
\end{bmatrix} \\
&= \begin{bmatrix}
\sigma_1^2 & 0 & \ldots & 0 \\
0 & \sigma_2^2 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots \\
0 & 0 & \ldots & \sigma_n^2
\end{bmatrix} \neq \sigma^2\mathbf{I}.
\end{aligned}$$

- If Heteroskedasticity happens, the inference of β will be inaccurate as we are estimating $\hat{\beta} \sim \mathbf{N}\left[\beta, \sigma^2\left(X'X\right)^{-1}\right]$.

How to solve Heteroskedasticity

1 Weighted Least Squares: find some term poportional to that.

2 Robust standard errors.

6.3.8 The Gauss–Markov Theorem

- OLS has the smallest variance among all linear unbiased models.
- consider $\theta = a^T\beta$, with predictions as $f(x_0) = a_0^T\beta$

$$\begin{aligned} for\ \forall\hat{\theta} &= a^T\hat{\beta} = a^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \\ E(a^T\hat{\beta}) &= a^T\hat{\beta} = a^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{X}\beta \\ &= a^T\beta \end{aligned}$$

- The Gauss-Markov theorem states that if we have any other linear estimator $\tilde{\theta} = \mathbf{c}^T\mathbf{y}$ that is unbiased for $a^T\beta$, that is, $\mathrm{E}\left(\mathbf{c}^T\mathbf{y}\right) = a^T\beta$, then

$$\operatorname{Var}\left(a^T\hat{\beta}\right) \leq \operatorname{Var}\left(\mathbf{c}^T\mathbf{y}\right)$$

- A proof from Wikipedia

Proof. Let $\beta = Cy$ be another linear estimator of β with $C = (X'X)^{-1}X' + D$ where D is a $K \times n$ non-zero matrix. As we're restricting to unbiased estimators, minimum mean squared error implies minimum variance. The goal is therefore to show that such an estimator has a variance no smaller than that of $\widehat{\beta}$, the OLS estimator. We calculate;

$$\begin{aligned} \mathrm{E}[\tilde{\beta}] &= \mathrm{E}[Cy] \\ &= \mathrm{E}\left[\left((X'X)^{-1}X' + D\right)(X\beta + \varepsilon)\right] \\ &= \left((X'X)^{-1}X' + D\right)X\beta + \left((X'X)^{-1}X' + D\right)\mathrm{E}[\varepsilon] && \mathrm{E}[\varepsilon] = 0 \\ &= \left((X'X)^{-1}X' + D\right)X\beta \\ &= (X'X)^{-1}X'X\beta + DX\beta \\ &= (I_K + DX)\beta \end{aligned}$$

Therefore, since β is unobservable, $\tilde{\beta}$ is unbiased if and only if $DX = 0$. Then:

$$\begin{aligned} \operatorname{Var}(\tilde{\beta}) &= \operatorname{Var}(Cy) \\ &= C\operatorname{Var}(y)C' \\ &= \sigma^2 CC' \\ &= \sigma^2\left((X'X)^{-1}X' + D\right)\left(X(X'X)^{-1} + D'\right) \\ &= \sigma^2\left((X'X)^{-1}X'X(X'X)^{-1} + (X'X)^{-1}X'D' + DX(X'X)^{-1} + DD'\right) \\ &= \sigma^2(X'X)^{-1} + \sigma^2(X'X)^{-1}(DX)' + \sigma^2 DX(X'X)^{-1} + \sigma^2 DD' && DX = 0 \\ &= \sigma^2(X'X)^{-1} + \sigma^2 DD' && \sigma^2(X'X)^{-1} = \operatorname{Var}(\widehat{\beta}) \\ &= \operatorname{Var}(\widehat{\beta}) + \sigma^2 DD' \end{aligned}$$

Since DD' is a positive semidefinite matrix, $\operatorname{Var}(\tilde{\beta})$ exceeds $\operatorname{Var}(\widehat{\beta})$ by a positive semidefinite matrix. □

- remark of the bais variance tradeoff.

Proof.

$$\begin{aligned}\text{MSE}(\tilde{\theta}) &= \text{E}(\tilde{\theta}-\theta)^2 \\ &= E(\tilde{\theta}^2) - E(\tilde{\theta})^2 + E(\tilde{\theta})^2 - 2\theta E(\tilde{\theta}) + \theta^2 \\ &= \text{Var}(\tilde{\theta}) + \underbrace{[\text{E}(\tilde{\theta})-\theta]^2}_{squared\ bias}\end{aligned}$$

□

6.3.9 Inference

- Null hypothesis: $\beta_j = 0$.
- Z score for $\hat{\beta}_j$: $z_j = \frac{\hat{\beta}_j}{\hat{\sigma}\sqrt{v_j}}$, where v_j is the j th diagonal element of $\left(\mathbf{X}^T\mathbf{X}\right)^{-1}$.
- if the sample size increases, we could replace v_j to the common sample size variance as t-distribution convergences to normal-distribution.
- standard error of $\beta_j = \frac{\sqrt{Var(\beta_j)}}{1} = v_j^{\frac{1}{2}}\hat{\sigma}$
- Confidence interval: $\left(\hat{\beta}_j - z^{(1-\alpha)}v_j^{\frac{1}{2}}\hat{\sigma},\quad \hat{\beta}_j + z^{(1-\alpha)}v_j^{\frac{1}{2}}\hat{\sigma}\right)$
- We use the F statistic to check if some of the coefficients could be dropped. Null hypothesis: No statistical significance between those two models.

$$F = \frac{(\text{RSS}_0 - \text{RSS}_1)\,/\,(p_1-p_0)}{\text{RSS}_1\,/\,(N-p_1-1)}$$

where RSS_1 is the residual sum-of-squares for the least squares fit of the bigger model with p_1+1 parameters, and RSS_0 the same for the nested smaller model with p_0+1 parameters, having p_1-p_0 parameters constrained to be 0.

6.3.10 R-Squared

$R^2 \in [0,1]$ is the proportion of the variation in the dependent variable that is predictable from the independent variable(s).

$$\text{R}^2 = 1 - \frac{\text{Unexplained Variation}}{\text{Total Variation}} = 1 - \frac{SS_{\text{res}} = \sum_i (y_i - \hat{y})^2}{SS_{\text{tot}} = \sum_i (y_i - \bar{y})^2}$$

A matrix form: Let $\boldsymbol{r}_i = Corr(y, x_i), \boldsymbol{r}_{i,j} = Corr(x_i, x_j)$

$$\boldsymbol{r}_{\mathbf{y},\mathbf{x}} = \begin{bmatrix} r_{y,x_1} \\ r_{y,x_2} \\ \vdots \\ r_{y,x_m}\end{bmatrix} \quad \boldsymbol{r}_{\mathbf{x},\mathbf{x}} = \begin{bmatrix} r_{1,1} & r_{1,2} & \cdots & r_{1,m} \\ r_{2,1} & r_{2,2} & \cdots & r_{2,m} \\ \vdots & \vdots & \ddots & \vdots \\ r_{m,1} & r_{m,2} & \cdots & r_{m,m}\end{bmatrix} \Rightarrow R^2 = \boldsymbol{r}_{\text{y,x}}^{\text{T}}\boldsymbol{r}_{\text{x,x}}^{-1}\boldsymbol{r}_{\text{y,x}}$$

6.3.11 Partitioned Regression and the Frisch-Waugh-Lovell Theorem

Reference OLS in Matrix Form.

Suppose we have two sets of independent variables, and the true model is:

$$y = \mathbf{X}_1\beta_1 + \mathbf{X}_2\beta_2 + \epsilon$$

- This form have $\mathbf{X}_1, \mathbf{X}_2$ because the dimensional may be different.
  For example: $y \subset \mathbf{R}^{n\times 1}, \mathbf{X}_1 \subset \mathbf{R}^{n\times k}, \beta_1 \subset \mathbf{R}^{k\times 1}, \mathbf{X}_2 \subset \mathbf{R}^{n\times p}, \beta_2 \subset \mathbf{R}^{p\times 1}$.

The estimated model is:

$$y = \mathbf{X}_1\hat{\beta}_1 + \mathbf{X}_2\hat{\beta}_2 + e$$

In order to isolate the coefficients $\hat{\beta}_2$ and to estimate $\hat{\beta}_1$, we have the close normal equation:

$$\begin{bmatrix} \mathbf{X}_1'\mathbf{X}_1 & \mathbf{X}_1'\mathbf{X}_2 \\ \mathbf{X}_2'\mathbf{X}_1 & \mathbf{X}_2'\mathbf{X}_2 \end{bmatrix} \begin{bmatrix} \hat{\beta}_1 \\ \hat{\beta}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{X}_1'y \\ \mathbf{X}_2'y \end{bmatrix}$$

$$\begin{aligned}
(\mathbf{X}_1'\mathbf{X}_1)\,\hat{\beta}_1 + (\mathbf{X}_1'\mathbf{X}_2)\,\hat{\beta}_2 &= \mathbf{X}_1'y \\
(\mathbf{X}_1'\mathbf{X}_1)\,\hat{\beta}_1 &= \mathbf{X}_1'y - (\mathbf{X}_1'\mathbf{X}_2)\,\hat{\beta}_2 \\
\hat{\beta}_1 &= (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'y - (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2\hat{\beta}_2 \\
\hat{\beta}_1 &= (\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\left(y - \mathbf{X}_2\hat{\beta}_2\right)
\end{aligned}$$

6.3.12 Problems

Problem: 1: for dependent variables X_1, X_2 and the response variable y, given $R^2(y \sim X_1) = a$, $R^2(y \sim X_2) = b$, what is the lower bound and upper bound for $R^2(y \sim (X_1, X_2))$?

Solution:
I was asked this question in a hedge fund interview, I got rejected right after I guessed the wrong answer. There are some discussions on stackoverflow which I think may be over-complicated since you suppose to answer this in an interview.
Let's think about it in a more tricky but simpler way. Remember if we regress on one dependent variable X_i, we are projecting y on the space of X_i, the worst case is that X_1, X_2 are identical, therefore $\min(R^2(y \sim (X_1, X_2))) = max(a, b)$. Now if y lies perfectly in the hyperplane spanned by X_1, X_2, $R^2 = 1$, thus

$$max(a, b) \le R^2 \le 1$$

For example in $1d$ case, let $X_2 = y - X_1$, then $R^2(y \sim (X_1, X_2)) = R^2(y \sim y) = 1$.

Problem: 2: How will $\hat{\beta}_1$ influenced by $\hat{\beta}_2$?

Solution:

- If $\mathbf{X}_2$ is independent with y, then $\hat{\beta}_2 = 0$, this is samle with OLS on only $\mathbf{X}_1$.
- Look up on the term $\underbrace{(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2}_{=\hat{\beta}\sim(\mathbf{X}_2\sim\mathbf{X}_1)}\hat{\beta}_2$, therefore, if $\mathbf{X}_1, \mathbf{X}_2$ are uncorrelated or orthogonal, this will still be the OLS on only $\mathbf{X}_1$.
- Otherwise, the $\hat{\beta}_1$ will be changed as the closed form above.

Problem: 3: What will happen if you copy the original data used for OLS to enlarge the dataset?

Solution:
Interesting, quant fund really like to ask those non sense problems, but let's see what will happen, simply think you are doubling the data, new OLS data $Y_{new} = \{Y, Y\}, X_{new} = \{\mathbf{X}, \mathbf{X}\}$ The new OLS becomes

$$\hat{\beta}_{ols} = \left(\begin{pmatrix} \mathbf{X} \\ \mathbf{X} \end{pmatrix}^T \begin{pmatrix} \mathbf{X} \\ \mathbf{X} \end{pmatrix}\right)^{-1} \begin{pmatrix} \mathbf{X} \\ \mathbf{X} \end{pmatrix}^T Y_{new}$$

$$Y_{new} = \begin{pmatrix} Y \\ Y \end{pmatrix} = X_{new}\,\beta + \epsilon$$

- $\left(\begin{pmatrix} \mathbf{X} \\ \mathbf{X} \end{pmatrix}^T \begin{pmatrix} \mathbf{X} \\ \mathbf{X} \end{pmatrix}\right)^{-1} \begin{pmatrix} \mathbf{X} \\ \mathbf{X} \end{pmatrix}^T \begin{pmatrix} Y \\ Y \end{pmatrix} = \left(2\mathbf{X}^T\mathbf{X}\right)^{-1} 2\mathbf{X}^T Y = \left(\mathbf{X}^T\mathbf{X}\right)^{-1} \mathbf{X}^T Y = \hat{\beta}_{ols},\ \ \hat{\beta}_{ols}$ will not change.
- The new variance of $\hat{\beta}_{ols} = \sigma^2(X_{new}^T X_{new})^{-1} = \frac{\sigma^2}{2}(\mathbf{X}'\mathbf{X})^{-1}$, will become smaller.
- The confidence interval is related to its standard error so will also have a shrink rate of $\frac{1}{\sqrt{2}}$. But is that true? No, because we have duplicated data! therefore this leads to autocorrealtion of residuals and we can't use the same covariance matrix for residuals, the result should be smaller than the shrinkage rate.

6.3.12.1 The Residual Maker and the Hat Matrix

$$\begin{aligned} e &= y - \mathbf{X}\hat{\beta} \\ &= y - \mathbf{X}\left(X'\mathbf{X}\right)^{-1}\mathbf{X}'y \\ &= \left(I - \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\right)y \\ &= My \end{aligned}$$

- The residual maker M is a square matrix and is idempotent: $M^2 = MM = M$.

$$\begin{aligned} MM &= \left(I - X\left(X'X\right)^{-1}X'\right)\left(I - X\left(X'X\right)^{-1}X'\right) \\ &= I^2 - 2X\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}' + \mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}'\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}' \\ &= I - 2\mathbf{X}\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}' + X\left(\mathbf{X}'\mathbf{X}\right)^{-1}\mathbf{X}' \\ &= I - \mathbf{X}\left(X'\mathbf{X}\right)^{-1}\mathbf{X}' \\ &= M \\ &\Rightarrow M\mathbf{X} = 0, Me = e. \end{aligned}$$

Insert back to the previous equations:

$$\begin{aligned} &\hat{\beta}_1 = \left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\left(y - \mathbf{X}_2\hat{\beta}_2\right) \\ &\left(\mathbf{X}_2'\mathbf{X}_1\right)\hat{\beta}_1 + \left(\mathbf{X}_2'\mathbf{X}_2\right)\hat{\beta}_2 = \mathbf{X}_2'y \\ \Rightarrow \hat{\beta}_2 &= \left[\mathbf{X}_2'\left(I - \mathbf{X}_1\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\right)\mathbf{X}_2\right]^{-1}\mathbf{X}_2'\left(I - \mathbf{X}_1\left(\mathbf{X}_1'\mathbf{X}_1\right)^{-1}\mathbf{X}_1'\right)y \\ &= \left(X_2'M_1\mathbf{X}_2\right)^{-1}\left(\mathbf{X}_2'M_1y\right) \\ &= \left(X_2'M_1M_1\mathbf{X}_2\right)^{-1}\left(\mathbf{X}_2'M_1M_1y\right) \end{aligned}$$

- M_1 is the residual maker of $\mathbf{X}_1$.
- $M_1\mathbf{X}_2 = (\mathbf{I} - \mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1')\mathbf{X}_2 = \mathbf{X}_2 - \mathbf{X}_1(\mathbf{X}_1'\mathbf{X}_1)^{-1}\mathbf{X}_1'\mathbf{X}_2 = e \sim (\mathbf{X}_2 \sim \mathbf{X}_1)$, OLS residual on $\mathbf{X}_2 \sim \mathbf{X}_1$.

Let

$$\begin{aligned} \hat{\beta}_2 &= \left(\mathbf{X}_2^{*'}\mathbf{X}_2^*\right)^{-1}\mathbf{X}_2^{*'}y^* \\ &\Leftrightarrow (OLS)\ y^* = \mathbf{X}_2^*\beta_2 + \epsilon \end{aligned}$$

where $\mathbf{X}_2^* = M_1\mathbf{X}_2 = e_{(\mathbf{X}_2\sim\mathbf{X}_1)}$ and $y^* = M_1y = e_{(y\sim\mathbf{X}_1)}$. Therefore:

$$\begin{aligned} \hat{\beta}_2 &= \left(e_{(\mathbf{X}_2\sim\mathbf{X}_1)}\mathbf{X}_2\right)^{-1}e_{(\mathbf{X}_2\sim\mathbf{X}_1)}e_{(y\sim\mathbf{X}_1)} \\ &\Leftrightarrow (OLS)\ e_{(y\sim\mathbf{X}_1)} = e_{(\mathbf{X}_2\sim\mathbf{X}_1)}\beta_2 + \epsilon \end{aligned}$$

Frisch-Waugh-Lovell Theorem: In the OLS regression of vector y on two sets of variables, $\mathbf{X}_1$ and $\mathbf{X}_2$, the subvector $\hat{\beta}_2$ is the set of coefficients obtained when the residuals from a regression of y on $\mathbf{X}_1$ alone are regressed on the set of residuals obtained when each column of $\mathbf{X}_2$ is regressed on $\mathbf{X}_1$.

6.3.13 The Rank Deficient Problem

You may notice that there might be an issue if we are soling the linear regression problem by matrix. What if $\mathbf{X}$ is not fully ranked ($X_i = aX_j$)? Which means the matrix $\mathbf{X}$ is singular.

- $\mathbf{X}$ in sigular doesn't mean we cant not project $\mathbf{y}$ into the space, now we have many ways represent the projection. You could also use other method like deleting the correlated matrix or use ridge regression.
- if the number of the features p exceed the training cases n as $p >> n$, which means you meet curse of dimensionality, as each sample could lead to a very specific regression value(Overfitting), then you could apply some feature selection methods(Lasso, PCA).

6.3.14 Ridge Regression

In order to solve the rank deficient problem and also regularize the weights β, we here introduce a prior distribution for β as$p\left(\boldsymbol{\beta} \mid \tau^2\right)=\prod_{j=1}^p \mathscr{N}\left(\beta_j \mid 0, \tau^2\right)$:

$$
\begin{aligned}
p\left(\boldsymbol{\beta} \mid \boldsymbol{x}, \boldsymbol{y}, \tau^2, \sigma^2\right) & \propto \sum_{i=1}^n \log \left[\left(\frac{1}{2 \pi \sigma^2}\right)^{1/2} \exp \left\{\frac{-1}{2 \sigma^2}\left(y_i-\boldsymbol{\beta}^T \boldsymbol{x}_i\right)^2\right\}\right]+\sum_{j=1}^p \log \mathscr{N}\left(\beta_j \mid 0, \tau^2\right) \\
& =\frac{1}{n} \sum_{i=1}^n\left(y_i-\boldsymbol{\beta}^T \boldsymbol{x}_i\right)^2+\frac{\sigma^2}{\tau^2} \sum_{p=1}^j \beta_j^2
\end{aligned}
$$

Now we have our standard formula for ridge regression:

$$
\begin{aligned}
& \hat{\boldsymbol{\beta}}=\underset{\boldsymbol{\beta}}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n\left(y_i-\hat{y}_i\right)^2+\lambda \sum_{j=1}^p \beta_j^2 \\
& \Rightarrow \hat{\boldsymbol{\beta}}=\left(X^T X+\lambda I_p\right)^{-1} X^T \boldsymbol{y}
\end{aligned}
$$

- The term λI_p helps to make the matrix invertible by adding some small values to the diagonal terms, and thus resolve the rank deficient issue.
- λ is a penalty parameter for β, with larger λ, more β will shrink to 0(Underfitting), with larger λ, then the OLS is more sensitive to outliers.(Overfitting) (similar with SVM)

6.3.15 Lasso Regression

Another important shrinkage method, but the penalty term is the $l1$ absolute value sum of weights as:

$$
\begin{aligned}
\hat{\beta}^{\text{lasso}} & =\underset{\beta}{\operatorname{argmin}}\left\{\frac{1}{2} \sum_{i=1}^N\left(y_i-\beta_0-\sum_{j=1}^p x_{i j} \beta_j\right)^2+\lambda \sum_{j=1}^p\left|\beta_j\right|\right\}, \\
s.t. & \sum_{j=1}^p\left|\beta_j\right|<t
\end{aligned}
$$

- Making t extremely small will lead to more β become 0.
- If t is chosen to be larger than $t_0=\sum_1^p|\hat{\beta}_j^{ols}|$, then lasso regression is no difference with OLS, otherwise, if t was set to for example $\frac{t_0}{2}$, then the parameters will also decrease by about 50%.

6.3.16 Lasso vs Ridge

- Ridge regression does a proportional shrinkage, use SVD(similar with PCA, review ESL for details).
- Lasso translates each coefficient by a constant factor λ, truncating at zero.(soft thresholding)
- In a lasso contour, it has corners. if the solution occurs at corner then one β will be equal to 0, when $p>2$, the diamond $|\beta_1|+|\beta_2|+\cdots+|\beta_p| \leq t$ has more corners and are more possible to shrink the parameters to 0_s.
- Why we use lasso? Because lasso ($p=1$) is the smallest p to make the constrain region convex, convace problem is hard in optimization.

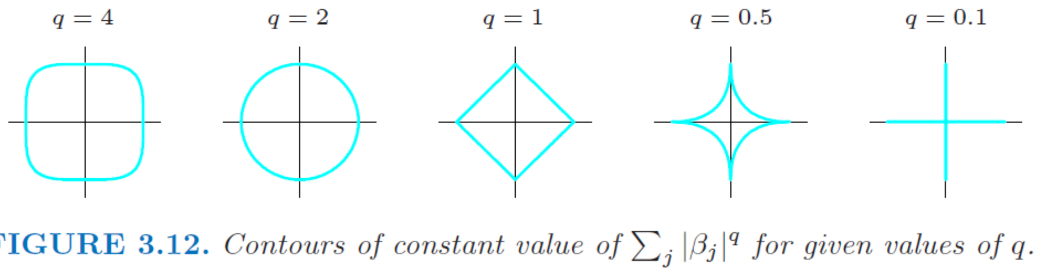


Figure 6.4: Figure source: [11]

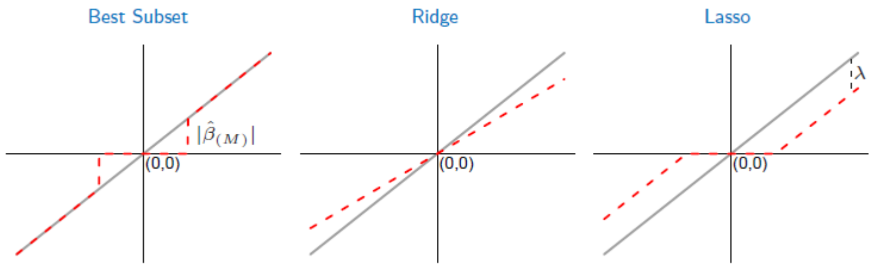


TABLE 3.4. *Estimators of β_j in the case of orthonormal columns of $\mathbf{X}$. M and λ are constants chosen by the corresponding techniques;* sign *denotes the sign of its argument (± 1), and x_+ denotes "positive part" of x. Below the table, estimators are shown by broken red lines. The 45° line in gray shows the unrestricted estimate for reference.*

Estimator	Formula
Best subset (size M)	$\hat\beta_j \cdot I(\lvert\hat\beta_j\rvert \geq \lvert\hat\beta_{(M)}\rvert)$
Ridge	$\hat\beta_j/(1+\lambda)$
Lasso	$\text{sign}(\hat\beta_j)(\lvert\hat\beta_j\rvert - \lambda)_+$

Figure 6.5: Figure source: [11]

6.3.17 Logistic Regression

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable.

6.3.17.1 What is the difference and similarities between logistic regression with linear regression?

1 Logistic regression deals with the classification problem whereas linear regression usually works on regression problems.

2 Linear regression tries to estimate the regression value:

$$\hat{y} = w^T X$$

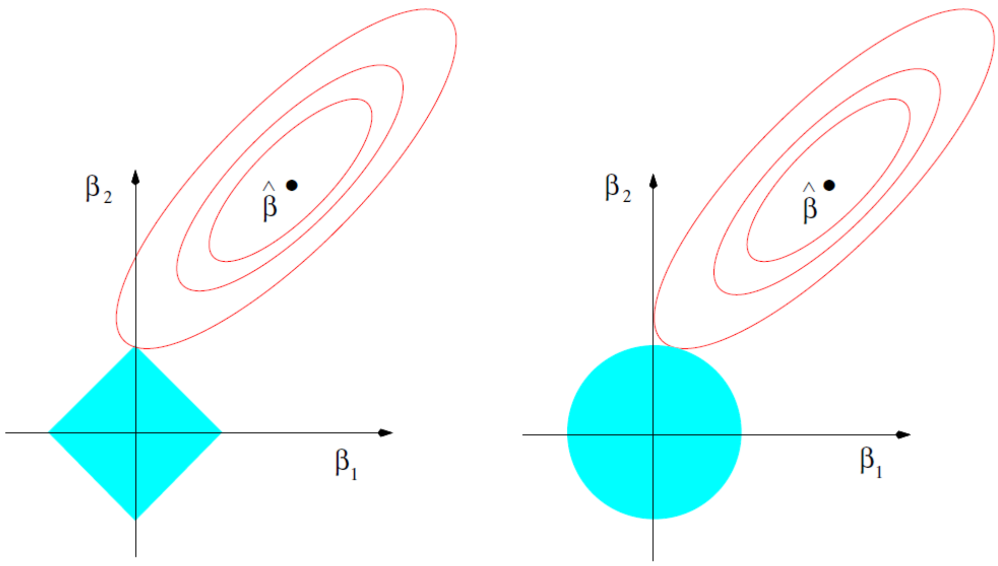


FIGURE 3.11. *Estimation picture for the lasso (left) and ridge regression (right). Shown are contours of the error and constraint functions. The solid blue areas are the constraint regions $|\beta_1| + |\beta_2| \leq t$ and $\beta_1^2 + \beta_2^2 \leq t^2$, respectively, while the red ellipses are the contours of the least squares error function.*

Figure 6.6: Figure source: [11]

Logistic regression comes from the odds ratio as:

$$\log(\frac{p}{1-p}) = w^T X$$
$$\frac{p}{1-p} = e^{w^T X}$$
$$p = \frac{e^{w^T X}}{1+e^{w^T X}},\ p = P(y=1|x)$$
$$y = \begin{cases} 1, if\ p > \theta \\ 0, if\ p \leq \theta \end{cases}$$

3 The response variable y in logistic regression is discrete however usually in linear regression it is continuous. Given w, x, logistic regression could be regarded as Generalized Linear Model where y follows Bernoulli distribution. In linear regression we assume y follows normal distribution.

4 Both uses MLE to estimate the variable w, linear regression uses least square and logistic regression uses likelihood function as: $L(w) = \prod_{i=1}^N P(y_i|x_i; w) = \prod_{i=1}^N (\pi(x_i))^{y_i}(1-\pi(x_i))^{1-y_i}$.

- For linear regression we use MSE as lost functions, for logistic regression we use log loss as $Loss = -\sum_1^N (y_i \log(p_i) + (1-y_i)\log(1-p_i))$ for binary case and $Loss = -\sum_1^N y_i \log(p_i)$ for multiclasses, $y_i = 1$ if the sample belongs to class i else 0.

6.3.17.2 How to apply logistic regression in multivariate classification?

Use softmax regression.

$$h_w(x) = \begin{bmatrix} p(y=1 \mid x; w) \\ p(y=2 \mid x; w) \\ \vdots \\ p(y=k \mid x; w) \end{bmatrix} = \frac{1}{\sum_{j=1}^k e^{w_j^T x}} \begin{bmatrix} e^{w_1^\top x} \\ e_2^{w_2^\top x} \\ \vdots \\ e^{w_k^T x} \end{bmatrix}$$

For example, in a binary classification problem,

$$\begin{aligned} h_w(x) &= \frac{1}{e^{w_1 x} + e^{w_2 x}} \begin{bmatrix} \mathrm{e}^{w_1^{\mathrm{T}} x} \\ \mathrm{e}^{w_2^{\mathrm{T}} x} \end{bmatrix} \\ &= \frac{1}{e^0 + e^{w_2^T - w_1^T x}} \begin{bmatrix} \mathrm{e}^0 \\ \mathrm{e}^{(w_2^T - w_1^T) x} \end{bmatrix} \end{aligned}$$

Which is same with the binary logistic regression. When one sample may belong to more than two classes, we could generate k logistic classifier, and use the i^{th} model to check if it belongs to i^{th} class.

6.4 Time Series Analysis

This is a note for the corresponding chapters in [12].

6.4.1 Moving Average

Assume the time series data at t is related to its previous n steps, we select $k \leq n$, the k order moving average methods states that:

$$\bar{x}_{t,1} = \frac{1}{k}\sum_{t=1}^{k} x_t,\ \bar{x}_{t,2} = \frac{1}{k}\sum_{t=2}^{k+1} x_t, \ldots,\ \bar{x}_{t,n-k+1} = \frac{1}{k}\sum_{t=n-k+1}^{n} x_t$$
$$\Rightarrow \hat{\mu}_t = \frac{1}{k}\sum_{t=T-k+1}^{T} x_t$$

- The estimated value at t is just the simple mean of itself and it's previous k steps.
- For forecasting, the moving average indicates that the $t+1$ step value is the mean of its previous k time step values.
- Moving average could also be used as a smoothing method in machine learning models.
- a second level smoothing average leads to $\bar{x}_3 = \frac{1}{2}x_2 + \frac{1}{2}x_3 = \frac{1}{2}(\frac{1}{2}x_1 + \frac{1}{2}x_2) + \frac{1}{2}(\frac{1}{2}x_2 + \frac{1}{2}x_3)$.
- MV could remove period effects if computed with the length of periodicity as known. For example, one could remove the periodicity of a year data by take a yearly MV.

6.4.2 Exponentially weighted moving averages (EWMA)

A regular version of MV model will be of the form:

$$\hat{\mu}_t = \bar{x}_t = \sum_{t=t-k+1}^{t} \alpha_t x_t,\ 2 \leq k \leq n$$

A symmetric version of the MV model will be:

$$\hat{\mu}_t = \bar{x}_t = \sum_{t=t-q}^{t+q} \alpha_t x_t, 2 \leq q \leq n-1,\ \sum_{t=t-q}^{t+q} \alpha_t = 1$$

If we take $\alpha = \frac{1}{k}$ then this is sample k order of MV model.
The binomial exponential smoothing aims to select the α geometrically, for example, for $q = 1$, we could have $\alpha = [\frac{1}{4}, \frac{1}{2}, \frac{1}{4}]$. The weighted form for moving average is

$$\begin{aligned}
&S_t = \hat{\mu}_t = \bar{x}_t = \alpha(1-\alpha)^0 x_t + \alpha(1-\alpha)^1 x_{t-1} + \alpha(1-\alpha)^2 x_{t-2} + \ldots + \alpha(1-\alpha)^{k-1} x_{t-k+1}\\
&S_t = \alpha x_t + (1-\alpha) S_{t-1}\\
&S_t \text{ is often used for forecasting } \hat{x}_{t+1}\\
&\hat{x}_{t+1} = \alpha x_t + (1-\alpha) S_{t-1} = \alpha x_t - \alpha S_{t-1} + S_{t-1}\\
&= \alpha\left(x_t - S_{t-1}\right) + S_{t-1}\\
&= \alpha\left(x_t - \hat{x}_t\right) + S_{t-1} = \alpha\epsilon_t + S_{t-1}.
\end{aligned}$$

- $\sum_{i=0}^{\infty} \alpha(1-\alpha)^i = 1$.
- The weighted coefficients follows a binomial distribution.
- $\operatorname{var}(S_t) = \alpha\sigma_x^2 \frac{\left(1-(1-\alpha)^{2t}\right)}{2-\alpha}$
- The key idea for EWMA is regard the closer data as more important data and the further data are less correlated data, for example you could pick $\alpha = 0.9$. This technique is also widely used in Deep Learning optimization fields, for example the Adam optimizer.

6.4.3 Trend Analysis and Seasonality

- Time series data with trends and seasonality is usually considered as non-stationary.
- Model should not learn much from those signals with clear trends and seasonality.

6.4.3.1 Trend Analysis

- Many time series have trends.
- Remove the trends would be helpful to analysis the data.
- Autocorrelation analysis is helpful in detecting the trends.

Autocorrelation in trends:

- When have a trend, the autocorrelation for small lags such r_1, r_2 will be positive and larger.

How to remove trend?

- Curve-fitting, OLS, then use $y_t = x_t - f(x_{t-1})$
- filtering, MV
- differencing, use new $y_t = x_t - x_{t-1}$ to remove linear trend.

Example, moving average in trend analysis

$$\begin{aligned}
\textbf{Forecast equation } & x_t = S_t + b_t \\
& S_t = \alpha x_t + (1-\alpha) S_{t-1} \\
\textbf{Let } & S_{t-1} \rightarrow S_{t-1} + b_{t-1} \\
\textbf{Level equation } & \Rightarrow S_t = \alpha x_t + (1-\alpha)(S_{t-1} + b_{t-1}) \\
\textbf{double exponential smoothing (Trend equation): } & b_t = \gamma\left(s_t - s_{t-1}\right) + (1-\gamma) b_{t-1}, 0 \leq \gamma \leq 1
\end{aligned}$$

Here we add a new term b_{t-1} to curve the trend change, b_t could be a constant for a function related to t.

6.4.3.2 Seasonal Analysis

- Seasonality means a time series data shows periodic features, i.e. monthly, yearly.
- In autocorrelation, r will also have seasonal performance and will be lager in seasonal periodic.(12h, 24h)

How to remove seasonality?

- Moving average, for example, if the period is 12, we could remove them by $s_{t,12} = \frac{0.5x_{t-6}+x_{t-5}+\ldots+x_{t-1}+x_t+x_{t+1}\ldots+x_{t+5}}{12}$
- Holt-Winters' method:
- differencing, $y(t) = x(t) - x(t - period)$.
- Modeling, fit a polynomial f and then differencing.
  - When constant seasonality: Holt-Winters' additive method
  - When changing proportional to time: Holt-Winters' multiplicative method.

6.4.4 Autoregressive model

$$x_t = \alpha_1 x_{t-1} + \ldots \alpha_p x_{t-p} + \epsilon_t$$

- The current prediction x_t is related to its previous p steps data.
- A Markov Process is a first order autoregressive process. For $p = 1, \alpha = 1$, $x_t = x_{t-1} + \epsilon_t$, which means this is a simple 1d random walk.

For a one lag AR model.

$$x_t = \alpha\left(\alpha x_{t-2} + \epsilon_{t-1}\right) + \epsilon_t$$
$$x_t = \alpha^k x_{t-k} + \alpha^{k-1}\epsilon_{t-k+1} + \ldots + \alpha\epsilon_{t-1} + \epsilon_t, \text{ or } x_t = \sum_{i=1}^{k} \alpha^i x_{t-i} + \epsilon_t$$
$$\Rightarrow \operatorname{var}(x_t) = \operatorname{var}\left(\epsilon_t + \alpha\epsilon_{t-1} + \alpha^2\epsilon_{t-2} + \ldots\right) = \operatorname{var}(z)(1 + \alpha^2 + \alpha^4 + \cdots) \to \frac{\operatorname{var}(z)}{1-\alpha}, \ \forall |\alpha| < 1.$$

- α is actually the covariance of x_t, x_{t+k} as $\gamma(k) = \operatorname{cov}(x_t, x_{t+k}) = \sigma_z^2 \sum_{i=0}^{\infty} \alpha^i \alpha^{k+i}, k > 0$.
- In order to fit an AR model to an observed dataset, we seek to minimize the MSE using the smallest number of terms that provide a satisfactory fit to the data.

6.4.5 ARMA, ARIMA models

- Combing AR and MV models, we have AR, ARIMA models.

ARMA

$$\begin{aligned} \textbf{AR model: } & x_t = \alpha_1 x_{t-1} + \ldots \alpha_p x_{t-p} + \epsilon_t \\ \textbf{MA model: } & x_t = \beta_0\epsilon_t + \beta_1\epsilon_{t-1} + \ldots \beta_q\epsilon_{t-q} \\ \textbf{ARMA model: } & x_t = \alpha_1 x_{t-1} + \ldots \alpha_p x_{t-p} + \epsilon_t + \beta_1\epsilon_{t-1} + \ldots \beta_q\epsilon_{t-q} \end{aligned}$$

How to understand:

- The p terms of AR are most related previous terms.
- The q terms of MA represent the average variation of random variation over q previous terms.

- The form assume the time series is stationary.
- You need to remove the trend and seasonality first. This leads to ARIMA model.

ARIMA (Box-Jenkins)

- Combing differencing of a non-stationary time series with ARMA model.

1 Difference the time series until it is stationary (remove the seasonality and trend).

2 Feed aN ARMA model into the data.

3 ARIMA have 3 variables p, q, d, d means the order of differentiation.

6.5 Principal component analysis (PCA)

PCA is an unsupervised machine learning method for dimensionality reduction.

6.5.1 What is principle component? How to define and solve PCA problems?

PCA aims to find the so called principle component which could present the most important feature of the original data, and we use those components to analysis the data.

- Suppose $X_{p\times 1}$ is a p-dimensional random variable with mean $\mu_{p\times 1}$ and variance covariance matrix $\Sigma_{p\times p}$, we want to find a projection direction $w_{p\times 1}$ so that $Var(w^T X)$ is maximized. w is defined as:

$$w^T \cdot w = 1$$

- **Why we want maximum $Var(w^T X)$?**

  Intuitively the Variance measures how much the data is dispersed in the w direction. In signal processing, signals are believed to have larger variance than noise. That's why we want maximum the variance.

- The solution of this problem w is called principle components. w_1 is defined as the first component, we can further define the second principal component w_2 as the projection vector that maximizes var $\left(w_2^T X\right)$ subject to $w_2 \perp w_1$. By the same way, we can define $w_3, w_4, \cdots, w_p$.

- **How to calculate Σ?**

  1 Centralize the columns X to $\hat{X}$, $\hat{X}_i = X_i - \mu$. Thus after the projection, the mean of the projected value is 0:

$$w^T \cdot \hat{X} = w^T \cdot (\frac{1}{n}\sum \hat{X}_i) = 0$$

  2 The variance will be:

$$\begin{aligned} Var(w^T \cdot \hat{X}) &= \frac{1}{n}\sum(w^T \cdot \hat{X}_i)^2 = Var(w^T \cdot \hat{X}) = \frac{1}{n}\sum(w^T \cdot \hat{X}_i)^T(w^T \cdot \hat{X}_i) \\ &= \frac{1}{n}\sum(w^T \cdot \hat{X}_i\hat{X}_i^{\,T} w) = w^T \cdot \Sigma \cdot w \end{aligned}$$

3 Now the problem becomes an optimization problem:

$$\begin{aligned} max: \ & w^T \cdot \Sigma \cdot w \\ & w^T \cdot w = 1 \end{aligned}$$

$$\begin{aligned} max: \mathscr{J} &= w^T \cdot \Sigma \cdot w - \lambda(w^T \cdot w - 1), \ for\ \forall \lambda > 0 \\ \frac{\nabla \mathscr{J}}{w} &= 2w \cdot \Sigma - 2w\lambda = 0 \\ &\Rightarrow \lambda \textbf{ is the eigenvalue of } \Sigma \end{aligned}$$

4 Calculate the eigenvalues of Σ as $\lambda_1 \geq \lambda_2 \geq \cdots \lambda_n$ with corresponding eigenvectors $w_1, w_2, \cdots w_n$.

5 The final projection data will be $w_i^T \hat{X}$. Note $Var(w_i^T X) = w_i^T \lambda_i w_i = \lambda_i$, then a k-dimensional projection will be:

$$X_k = \begin{bmatrix} w_1^T \hat{X} \\ w_2^T \hat{X} \\ \vdots \\ w_k^T \hat{X} \end{bmatrix}$$

- The portion of information is defined as:

$$\frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^n \lambda_i}$$

6.5.2 Prove principle components are orthogonal to each other.

Let $\lambda_i \neq \lambda_j$ be any twp arbitrage eigenvalues of Σ,

$$< \Sigma\lambda_i, \lambda_j > = < \lambda_i, \Sigma^T \lambda_i >$$
$$\Rightarrow \lambda_a < \lambda_i, \lambda_j > = \lambda_b < \lambda_i, \lambda_j >, for \ \forall \ \lambda_a \neq \lambda_b$$
$$\Rightarrow (\lambda_a - \lambda_b) < \lambda_i, \lambda_j > = 0$$
$$\Rightarrow < \lambda_i, \lambda_j > = 0. \square$$

6.5.3 Show the similarities between linear regression and PCA.

For a $1d$ projection, PCA maps the original data into a Cartesian coordinate system (x, y-axis), and tries to minimise to distance between (x_i, y_i) to the model line.
OLS minimise the distance between points $\sum ||y_i - \hat{y}_i||^2$. (The distance on the y -axis.), The straight line in the projection of the data in the direction of the corresponding eigenvector. For more details, please review page (95)[8].

6.6 Decision Tree & Random Forest

Decision Trees are a non-parametric supervised learning method used for both classification and regression tasks.

6.6.1 What are the criterion of splits method in tree based models?

Key idea: Loss of diversity

$$\Delta V = V - \frac{m_L}{m} V_L - \frac{m_R}{m} V_R$$

Here m, m_l, m_r are the nodes in parent node, left daughter and right daughter.
V is often chosen to be Gini Index or Entropy.

Entropy

$$V = H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log_2 \frac{|C_k|}{|D|} = -\sum_{k=1}^{K} p_k \log p_k$$

- When $p_k = \frac{1}{K}$, H gets the maximum value and ΔV reaches to smallest value. Which means the system still have much uncertainty. On the other hand, if $p_k = 1$ and others are 0, the system gains the maximum information.
- For feature A, the conditional entropy to sample D will be

$$H(D \mid A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} \left(-\sum_{k=1}^{k} \frac{|D_{ik}|}{|D_i|} \log_2 \frac{|D_{ik}|}{|D_i|} \right)$$

  D_i represents the sample of $i^t h$ value in feature A, and D_{ik} means the the sample belongs to class k with i^{th} value of feature A.
- Finally the entropy growth is:

$$g(D, A) = H(D) - H(D \mid A)$$

Gini Index

$$G = 1 - \sum_{k=1}^{K} p_k^2$$

C4.5

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}$$

$$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \log_2 \frac{|D_i|}{|D|}$$

6.6.2 Problems of large tree model?

1 Overfitting

2 Difficult to interpret.

6.6.3 What are the pruning rules for tree?

1 Stop when the number of samples that reach a node is smaller than a threshold.

2 Stop when the gain of purity is smaller than a threshold. (set a threshold for the gain of entropy).

3 Post-pruning: let the tree grow big, then try to prune it bottom-up to find the tree with the smallest cost: $S = r + k \cdot M$, where S is the total cost, r is the misclassification rate, k is the penalty parameter (a constant), and M is the total number of nodes (or number of splits) of a tree.

4 Reduced error pruning: Starting from bottom, each node is replaced with its majority class. Keep the replacement if the prediction accuracy is within threhold.

6.6.4 Random Forest

Random Forest is a summation of multiple decision trees (as is called "Forest"), and there are two ways in making those trees:

- Randomly select features of the data.
- Randomly select data points from the original data.

That how "Random" is simulated.

6.6.4.1 Advantages

- Overcome overfitting as we use "smaller" trees(reduce the variance).
- Can work on data with missing values.
- Usually increase the accuracy of the model.

6.6.5 How to deal with missing values in desicion trees?

1 Use random forest to pick the trees with out the missing feature.

2 For those tress with the missing feature, just try its left and right trees see which will lead to more information gain.

6.6.6 Classification & Regression Tree

Use vote Majority rule for classification trees and take mean for regression trees.

6.6.7 Gradient Boost Tree

- Intuition: when the model works poorly in some difficult samples, we construct a classifier which will focus more in those samples. (Adaboost, MART)
- Example on Toward Data Science.
- Pros:
  - see GBDT
  - Help overcome the overfitting problem(add the learning rate as another hyperparameter).
  - A nonlinear transformation in regression and classification.
  - Works well for large dataset with dense features.(decrease the memory).
  - More reasonable in handing categorical features and missing values.
  - Different of loss functions as: mse, logloss, huberized hinge loss, etc.
- Cons:
  - Weaker in high dimensional data or sparse data than SVM and Neural Network.
  - Might not intereptable in NLP problems.

6.7 Support Vector Machine

SVM is a supervised learning algorithm.
Intuition:

- $w^T x + b = 0$ is a hyperplane, and w is the direction that the perpendicular to the hyperplane.
- All points above the hyperplane as $w^T x + b > 0$ are in group 1 and points below the hyperplane are in group 2 as $w^T x + b < 0$.
- Normalize the hyperplane as $w^T x + b = 1$ and $w^T x + b = -1$, then we want to maximise the margin, which equivalents to max $\frac{2}{||w||} \Leftrightarrow \min \frac{1}{2}||w||^2$.

Now the optimization problem becomes (Linear seperatable case):

$$
\begin{aligned}
&\min \frac{1}{2}||w||^2, \\
&s.t.\ y_i(w^T x_i + b) \geq 1,\ y_i = 1\ if\ y_i \in C_1, y_i = -1\ if\ y_i \in C_2
\end{aligned}
$$

Use KKD condition ans turns to problem into a linear programming problem:

$$
\begin{aligned}
\min L_p &= \frac{1}{2}||w||^2 + \sum_i^n \lambda_i(1 - y_i(w^T x_i + b)) \\
\frac{\partial L_p}{\partial w} &= 0 \Rightarrow w = \sum_{i=1}^n \lambda_i y_i x_i \\
\frac{\partial L_p}{\partial b} &= 0 \Rightarrow \sum_{i=1}^n y_i \lambda_i = 0
\end{aligned}
$$

Substituting back to the original function we have the dual problem:

$$
\begin{aligned}
\max L_D &= -\frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \lambda_i \lambda_j y_i y_j x_i x_j + \sum_{i=0}^n \lambda_i \\
s.t.\ &\sum_{i=1}^n y_i \lambda_i = 0
\end{aligned}
$$

Algorithm:

- Solve the dual problem for $\lambda_i, i = 1, \ldots, n$.
- Plug λ_i to the KKT conditions to get b. b is determined by $\left\{y_i \left[\sum_j \lambda_j y_j x_j^T x_i + b\right] - 1\right\} = 0, y_i^2 = 1$
  Let x_l be a support vector from Class 1, then $b = y_l - \sum_j \lambda_j y_j x_j^T x_l$
- The prediction of x^{new} is $\hat{y} = \text{sign}\left[\sum_i \lambda_i y_i x_i^T x^{\text{new}} + b\right]$.
- The solution is always the global optimum.

Slack variabe linearly non-separable case

$$
\begin{aligned}
&\min \frac{1}{2}||w||^2 + C\sum_{i=1}^n \xi_i \\
&s.t.\ y_i(w^T x_i + b) \geq 1 - \xi_i,\ for\ \forall \xi_i \geq 0
\end{aligned}
$$

- Larger C means smaller margin and less number of support vectors.

Support vectors are the points lie on the hyperplane.

6.7.1 Why we want use Kernel trick?

Samples that are nonlinearly separable may become linearly separable if we map the data onto a high dimensional space.

6.7.2 Pros of using SVM

- Always could reach the global optimization with a fast calculation.
- Workable on nonlinear case by using kernel trick.

6.7.3 How to use SVM to solve multi-class classification task?

- One-versus-one
  - class i vs. class j majority votes by $\binom{K}{2} = \frac{K(K-1)}{2}$ classifiers. (need hyperplane to separate between every two classes).
- One-versus-all
  - class i vs. all other classes largest probability score of K classifiers

6.7.4 Do their exist a set of parameters which makes the training error of SVM to be 0?

Use a Gaussian kernel: $e^{\frac{-|x-z|^2}{r^2}}$, assume all the training data points are in different locations. Let $\lambda_i = 1$ and $b = 0$ we have:

$$
\begin{aligned}
f(x) &= \sum_{i=1}^{m} \lambda_i y_i K(x_i, x) + \sum_{i}^{m} b \\
&= \sum_{i=1}^{m} y_i K(x_i, x) \\
&= \sum_{i=1}^{m} y_i e^{\frac{-|x-x_i|^2}{r^2}} \\
||f(x_j) - y_j|| &= \sum_{i=1, i!=j}^{m} y_i e^{\frac{-|x_j-x_i|^2}{r^2}}, \ for\ \forall\ i \neq j. \\
&\leq \sum_{i=1, i!=j}^{m} e^{\frac{-|x_j-x_i|^2}{r^2}} \leq (m-1) e^{\frac{-\epsilon^2}{r^2}},\ ||x_i - x_j|| \geq \epsilon \\
Let\ r^2 &= \frac{\epsilon^2}{\log(m)} \Rightarrow \sim \leq \frac{m-1}{m} < 1
\end{aligned}
$$

Therefore, for any training data, the difference between the predicted value and true value is less than 1, thus for $y_j = 1, f(x_j) > 0$, $y_j = -1, f(x_j) < 0$, which makes the training error to be 0.

6.7.5 If the training error is 0, is this classifier a solution of SVM?

In the above problem we have seen there exists a Gaussian kernel which could make the training error to be 0. Now, let's see if it satisfies the conditions of SVM. Let $b = 0$, we have

$$\begin{aligned}
f(x) &= \sum_{i=1}^{m} \lambda_i y_i K(x_i, x) \\
y_j f(x_j) &= y_j \sum_{i=1}^{m} \lambda_i y_i K(x_i, x_j) \\
&= y_j \sum_{i=1, i\neq j}^{m} \lambda_i y_i K(x_i, x_j) + \lambda_j y_j y_j K(x_j, x_j) \\
&= \sum_{i=1, i\neq j}^{m} \lambda_i y_i y_j K(x_i, x_j) + \lambda_j
\end{aligned}$$

If we want to have $y_j f(x_j) > 1$, then let $r^2 \to 0 \Rightarrow K(x_i, x_j) \to 0$, also let $\lambda_j \to \infty$, we have $y_j f(x_j) > 1$.

6.7.6 Will the slack variables help to reduce the training error to be 0?

No, slack variables try to make a balance between the number of support vectors and errors, for example, $C = 0, w = 0$, satisfies the condition but the error is not 0.

6.7.7 Regression

$$\begin{aligned}
&\min \frac{1}{2}||w||^2 + C\sum_{i=1}^{n} \xi_i \\
&s.t. \; |y_i - \langle w, x_i\rangle - b| \leq \varepsilon
\end{aligned}$$

6.8 K-means

K-means is a unsupervised learning algorithm. k-means is a clustering algorithm.

- Clustering belongs to unsupervised learning; that is, you only need to cluster the input data without labeling.
- Classification is a supervised learning algorithm, you know the input and output data classes. Typically, we have binary classification and multiclassification tasks.

Aim: find K clusters which corresponds to the loss function:

$$SSE = \sum_{k=1}^{K} \sum_{i \in C_k} \|x_i - m_k\|_2^2$$
$$x_i : i^{th} \text{ point that belongs to cluster } C_k$$
$$m_k : \frac{1}{n_k} \sum_{i \in C_k} x_i, \text{mean of points in } C_k$$
$$K : \text{number of total data points}$$

6.8.1 Algorithms

- Select (often randomly) K samples as the initial centroids $m_k, k = 1, \ldots, K$.
- Repeat
  - Assign each sample to the nearest centroid.
  - Recompute centroids by the new class labels. until the centroids do not change, or equivalently, the class labels do not change.

The loss function is quadratic function, thus it **guarantees to converge**.

6.8.2 Cons

- Local Minimums. (The algorithms will give us a global minimum for a given k, however, one could not promise the class is strictly belongs to those k clusters.)
- Doesn't work well if two samples' sizes vary much from each other.(Consider one class is surrounded by another class.)

6.8.3 Advantages

- Fast algorithm, expecially for large dataset with a `O(NKt)` complexity.

6.8.4 Optimization: Gap Statistic

Do the following procedure for $K = 1, 2, \ldots, K_{\max}$[13]

a Do K-means to cluster the samples to K clusters and get SSE_K

b Generate a set of n samples that are uniformly distributed in $\mathbb{R}^p$ space that have the same range as the original data. This dataset is viewed as under the null hypothesis that there is only one cluster in the data.

c Do K-means on this new set of data and get SSE_K^*.

d Repeat (b) and (c) for B times, and denote their SSE to be $SSE_K^{(b)*}, b = 1, \ldots, B$.

e Define $\text{Gap}(K) = \frac{1}{B}\sum_{b=1}^{B} \log SSE_K^{(b)*} - \log SSE_K$

Select K that maximizes $\text{Gap}(K)$.

6.9 Linear Discriminate Analysis

LDA is a supervised learning algorithm.

6.9.1 Proof

Most prediction models estimate $P(Y|X)$. LDA uses Bayesian rule and multivariate normal distribution with equal covariance to estimate get the decision bounds as:

$$\begin{aligned}\hat{P}(Y=k \mid X=x) &= \frac{\hat{P}(X=x \mid Y=k)\hat{P}(Y=k)}{\hat{P}(X=x)} \\ &= \frac{\hat{P}(X=x \mid Y=k)\hat{P}(Y=k)}{\sum_{j=1}^{K}\hat{P}(X=x \mid Y=j)\hat{P}(Y=j)} \\ &= \frac{N(\mu_k,\Sigma)\pi_k}{\sum_{j=1}^{K}\hat{P}(X=x \mid Y=j)\hat{P}(Y=j)} \\ &\sim N(\mu_k,\Sigma)\pi_k\end{aligned}$$

$$\begin{aligned}\hat{P}(X=x \mid Y=k) = f_k(x) &= \frac{1}{(2\pi)^{K/2}|\mathbf{\Sigma}|^{1/2}}e^{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)} \\ \Rightarrow \hat{P}(Y=k \mid X=x) &\sim \pi_k e^{-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)} \\ log(\hat{P}(Y=k \mid X=x)) &\sim \log\pi_k - \frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)\end{aligned}$$

Let n_k be samples in Class k as $\sum_{k=1}^{K} n_k = n$. X^k be data matrix for samples in Class k . X^k is of dimension $n_k \times p$. Let $\tilde{X}^k$ be the centralized X^k.

Discriminant rule :

$$k = \underset{1\le k\le K}{\arg\max} h_k(x).$$

Gaussian discriminant functions:

$$h_k(x) = \log\pi_k - \frac{1}{2}\log(\det\Sigma_k) - \frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)$$

If estimated by data, use n_k/n, $\hat{\mu}_k$: the mean of observations in Class k, $\hat{\Sigma}_k = \frac{1}{n_k-1}\left(\tilde{X}^k\right)^T\tilde{X}^k$ In this case, $h_{k_1}(x) - h_{k_2}(x)$ is a quadratic function wrt x. So the decision boundary is quadratic. This method is called "Quadratic Discriminant Analysis" (QDA). - We sometimes assume Σ_k are the same for all k 's. Then we estimate the common

$$\hat{\Sigma} = \frac{1}{n-K}\sum_{k=1}^{K}\left(\tilde{X}^k\right)^T\tilde{X}^k$$

and use it to replace $\hat{\Sigma}_k$ in the formula of $h_k(x)$. $h_{k_1}(x) - h_{k_2}(x)$ is a linear function wrt x if we neglect the quadratic terms. This method is so called "Linear Discriminant Analysis" (LDA), other wise we call this "Quadratic Discriminant Analysis" (QDA).

6.9.2 Example: Poisson Distribution

The LDA predict aim function:

$$\hat{y_0} = \operatorname{argmax}_y \hat{P}(Y=y \mid X=x_0)$$

where: $\hat{P}(Y=k) = \hat{\pi}_k$ Beysian Law:

$$\hat{P}(Y=k \mid X=x) = \frac{\hat{P}(X=x \mid Y=k)\hat{P}(Y=k)}{\sum_j \hat{P}(X=x \mid Y=j)\hat{P}(Y=j)}$$

let $P(X = x \mid Y = k) = f_k(x)$ which follows the Poisson distribution:

$$f_k(x) = e^{-\lambda} \frac{\lambda^x}{x!}$$

We have:

$$\hat{P}(Y = k \mid X = x) = \frac{f_k(x)\hat{\pi}_k}{\hat{P}(X = x)}$$

The denominator is independent with k, therefore:

$$\begin{aligned} P(Y = k \mid X = x) &= C f_k(x) \pi_k \\ &= C\pi_k e^{-\lambda_k} \frac{(\lambda_k)^x}{x!} \\ &= C' \pi_k e^{-\lambda_k} (\lambda_k)^x \\ \log(P(Y = k \mid X = x)) &= \log(C) + \log(\pi_k) - (\lambda_k) + x \log(\lambda_k) \end{aligned}$$

Question: What is the discriminant rule for Class i and Class j ? When $K = 3, \pi_1 = 0.2, \pi_2 = 0.3, \pi_3 = 0.5, \lambda_1 = 20$, $\lambda_2 = 50, \lambda_3 = 70$, write out the decision rules between the three classes explicitly.

The aim function turns to:

$$\begin{gathered} \operatorname{argmax}_x \log(\pi_k) - (\lambda_k) + x \log(\lambda_k) \\ h_i(x) = \log(\pi_i) - (\lambda_i) + x \log(\lambda_i) \end{gathered}$$

separation boundary function:

$$h_i(x) - h_j(x) = \log\left(\frac{\pi_i}{\pi_j}\right) - (\lambda_i - \lambda_j) + x \log\left(\frac{\lambda_i}{\lambda_j}\right) = 0$$

Decision:

$$y = \operatorname{argmax}_i (h_1(x), h_2(x), h_3(x))$$

Then the boundaries are:

$$\begin{gathered} x1 = \arg_x (h_1 = h_2)(x) = 32.3 \\ x2 = \arg_x (h_2 = h_3)(x) = 57.9 \end{gathered}$$

for $x < 33, y = 1$, for $33 < x < 58, y = 2$, else the $y = 3$

6.10 Word embedding method

Word embedding is an unsupervised learning problem. this section follows the lecture of Hung-yi Lee's machine learning class.

6.10.1 1-of N encoding & on hot encoding

Use a large vector to represent the word, for example $apple = [1, 0, 0, 0, 0]$, however the word might be very large and information is hard to extract. Thus you use **word classes** which merge similar words into one class.
Thus we need word embedding:

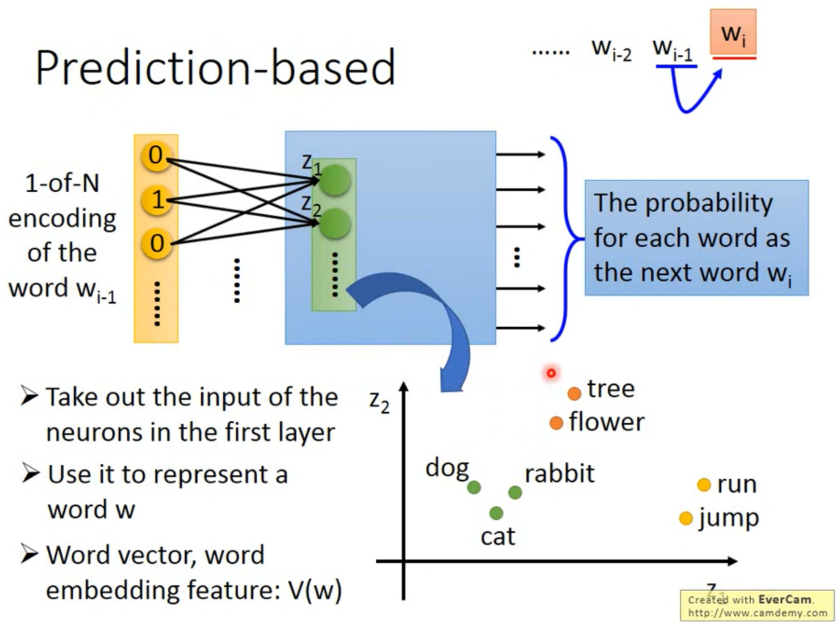
- Project the words into a high dimensional space.
- An ideal projection space should cluster similar words as one group.

word information:

- machine could learn the meaning of the words by reading a lot of documents without supervision.
- The meaning of the word is related to its previous and following words.(Like verb), For example, `[Mom loves me], [Dad loves me]`, then the machine may infer that `Mom` and `Dad` are similar words or have connections.

How to exploit the context?

- Count based
  - if two words w_i and w_i frequently co-occur, $V(w_i)$ and $V(w_j)$ would close to each other.
    * Glove Vector($N_{i,j}$ `Number of times` w_i `and` w_j `in the same document`): $N_{i,j} = V(w_i) \cdot V(w_j)$
- Prediction based:
  - Given previous word w_{i-1}, predict the next word w_i.



  -
  - if the word class is 1000, the NN will predict the probabilities of each word out of 1000 will be the next word.
  - For example, When training: $w_i =$ `Dad`, $w_{i+1} =$ `Love`.

  - You could also input multiply previous words in to the NN and predict the next word where the NN should share the parameters, that's why RNN is widely used in practice.
  - Continuous bag of word model(CBWM): use w_{i-1}, w_{i+1} to predict w_i.
  - Skip-gram: use w_i to predict its nearby words.

6.10.2 TF-IDF

Ideal: the frequency of words shown in a document may present its features, for example marketing emails may contain `discount, low-price`.

- Term frequency: $tf(t,d) = \frac{f_{t,d}}{\sum_{t' \in d} f_{t',d}}$, $f_{t,d}$ is the raw count if a term in a document.
- Inverse document frequency: measure of how much information the word provides:

$$idf(t,D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

N : total number of documents in the corpus $N = |D|$

$|\{d \in D : t \in d\}|$is the number of documents where term t appears.

6.11 Deep Learning triva

6.11.1 Regularization

- $L1$ regularization (Lasso):

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}w_j\right)^2}_{L} + \lambda\sum_{j=1}^{p}|w_j|$$

  By adding the L_1 regularization term (Lasso), less important feature will shrink with 0 coefficients, this helps in feature selection.

- L_2 regularization cost function as (similar with ridge regression):

$$\underbrace{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p} x_{ij}w_j\right)^2}_{L} + \lambda\sum_{j=1}^{p}w_j^2$$

$$w_{i+1} = \underbrace{w_i - \frac{\lambda}{2m}w_i}_{\textbf{weight decay}} - \frac{\partial L}{\partial w}.$$

  By adding the L_2 regularization term the loss function may help decrease during training because of weight decay.

- Dropout: The hidden unit has some specific probability to be neglected during training thus could help from overfitting.

6.11.2 Vanishing / Exploding gradients

$$\frac{\partial \mathbf{w}^{(T)}}{\partial \mathbf{w}^{(1)}} = \frac{\partial \mathbf{w}^{(T)}}{\partial \mathbf{w}^{(T-1)}} \cdots \frac{\partial \mathbf{w}^{(2)}}{\partial \mathbf{w}^{(1)}}$$

If $\frac{\partial \mathbf{w}^{(i)}}{\partial \mathbf{w}^{(i-1)}}$ is big or extremely small, the gradient will explode/ vanish.
How to deal with that:

- set $Var(w_i) = \frac{1 \text{ or } 2}{n}$,
- Xavier $w_i = \texttt{bp.random.randn(shape)} \times \tanh\sqrt{\frac{1}{n^{l-1}}}$, $w_{i,j} \sim \mathcal{N}(0, \frac{1}{n^{l-1}})$.

6.11.3 Mini-batch Gradient descent

For stochastic gradient descent, we only randomly process one training sample, Mini-Batch combines the batch method with stochastic gradient descent which we pick a batch $t : Xt, Yt$, for example, a batch t includes 1000 samples out of 5 millions samples. Pseudo code for one epoch:

```
for t in range(Num_of_batches)
  A, caches = forward_prop(X{t}, Y{t})
  cost = compute_cost(A, Y{t})
  grads = backward_prop(A, caches)
  update_parameters(grads)
```

6.11.4 Exponentially Weighted Averages

For the dataset with noise we may want to smooth it. The EWA(Exponentially Weighted Average) formula is:

$$v(t) = \beta v(t-1) + (1-\beta)\theta(t)$$

- $\beta = 0.9$ will average last 10 entries.
- $\beta = 0.98$ will average last 50 entries.
- $\beta = 0.5$ will average last 2 entries.
- as $\frac{1}{1-\beta}$

Vt is some smoothen value at point t, and $\theta(t)$ is the data at time t. Consider for $v(1)$, $v(0) = 0$ thus lead $v(1)$ to be considerable smaller than its real value, which will also infuluence the latter $v(t > 1)$, we apply a Bias correction in exponentially weighted averages as:

$$v(t) = \frac{\beta v(t-1) + (1-\beta)\theta(t)}{1-\beta^t}$$

6.11.5 Gradient Descent with Momentum

- Idea: use exponentially weighted averages to update the gradient in deep learning, which leads to a faster learning scheme than regular gradient descent.

Psudo code:

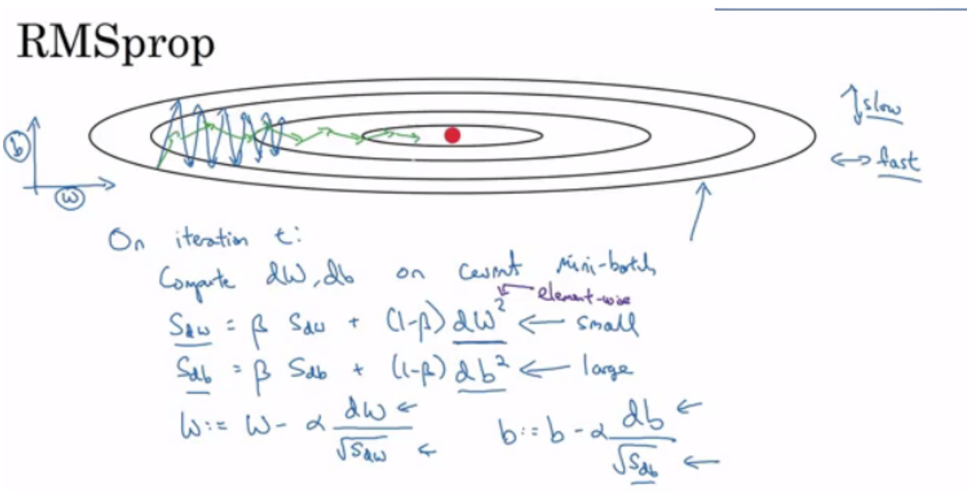
```
vdW = 0, vdb = 0
for i in range(Iterations):
  vdW = beta * vdW + (1 - beta) * dW
  vdb = beta * vdb + (1 - beta) * db
  W = W - learning_rate * vdW
  b = b - learning_rate * vdb

```

We usually pick $\beta = 0.9$.

6.11.6 RMSprop

As we see in the lecture picture on Andrew Ng's course, we apply a method called RMSprop to help the gradient move slower in the vertical variable and keep the fast learning speed in horizontal variable. (By Jeff Hinton)



As:

$$W = W - \alpha \frac{dW}{(\sqrt{s_{dW}} + \epsilon)}$$

- add $\epsilon = 10^{-8}$ to ensure a non-zero denominator.
- RMSprop allows to increase your learning rate.

6.11.7 Adam optimizer(Adaptive Moment Estimation.)

Adam[14] optimizer is a method which combines RMSprop and Momentun.

```
vdW = 0, vdW = 0
sdW = 0, sdb = 0
for t in range(Iterations):
  vdW = (beta1 * vdW) + (1 - beta1) * dW     # momentum
  vdb = (beta1 * vdb) + (1 - beta1) * db     # momentum

  sdW = (beta2 * sdW) + (1 - beta2) * dW^2   # RMSprop
  sdb = (beta2 * sdb) + (1 - beta2) * db^2   # RMSprop

  vdW = vdW / (1 - beta1^t)       # bias correction
  vdb = vdb / (1 - beta1^t)       # bias correction

  sdW = sdW / (1 - beta2^t)       # bias correction
  sdb = sdb / (1 - beta2^t)       # bias correction

  W = W - learning_rate * vdW / (sqrt(sdW) + epsilon)
  b = B - learning_rate * vdb / (sqrt(sdb) + epsilon)

```

Usually take $\beta_1 = 0.9, \beta_2 = 0.999$.

6.11.8 Learning Rate Decay

Some tuning method for learning rate α:

$$\alpha^{i+1} = \frac{\alpha^i}{1 + ri_{epoch}}$$
$$\alpha^{i+1} = (0.95^{i_{epoch}})\alpha^i$$
$$\alpha^{i+1} = \frac{k\alpha^i}{\sqrt{i_{epoch}}} \ or \ \frac{k\alpha^i}{\sqrt{t}}$$

Here r the decay rate, you could choose as 1, k is some constants.

6.12 NLP interview questions

6.12.1 Sequence to sequence model

- A Seq2Seq model is a model that takes a sequence of items (words, sentence, images, letters, time series, etc) and outputs another sequence of items., it uses a Encoder - decoder structure.
- IN CV, it uses a CNN - RNN structure.
- In NLP, usually a LSTM - LSTM structure is used.
- Input: $(x_1, x_2, \cdots, x_n)$, feed into encoder we got $(h_1, h_2, \cdots, h_n) = encoder(x_1, x_2, \cdots, x_n)$, feed into decoder we will have the final output $(y_1, y_2, \cdots, y_k) = decoder(h_1, h_2, \cdots, h_n)$.

Issue in Seq2seq model:

- The model will trend to loss the long time dependency if the input is large/long.
- The model is hard to catch the important part when the input is fixed with a specific size.

6.12.2 Attention

The attention layer is a neural network structure which mimics cognitive attention, it will help the neural network focus more on the important part and filter the non-important part. Why we want use Attention compared with LSTM?

- Computational costs, if the input is large we will be needing more neurons and deep layers to 'remember' the important part.
- Easier for us to use the backpropagation since we have limited optimization methods in solving complex neural networks.

What is the key feature of self-attention?

- Learn and extract the relation between words in a input sentence.
- need normalization $\sqrt{d_k}$ because we don't want softmax to have a one hot vector when facing large $q_i k_i^T$, which will leads gradient vanish.

How self attention works?

- Attention $(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V$.

```python
class Self_Attention(torch.nn.Module):
    # input : batch_size * seq_len * input_dim
    # q : batch_size * input_dim * dim_k
    # k : batch_size * input_dim * dim_k
    # v : batch_size * input_dim * dim_v
    def __init__(self,input_dim,dim_k,dim_v):
        super(Self_Attention,self).__init__()
        self.q = torch.nn.Linear(input_dim,dim_k)
        self.k = torch.nn.Linear(input_dim,dim_k)
        self.v = torch.nn.Linear(input_dim,dim_v)
        self._norm_fact = 1 / sqrt(dim_k)


    def forward(self,x):
        Q = self.q(x) # Q: batch_size * seq_len * dim_k
        K = self.k(x) # K: batch_size * seq_len * dim_k
        V = self.v(x) # V: batch_size * seq_len * dim_v
        atten = torch.nn.Softmax(dim=1)(torch.bmm(Q,K.permute(0,2,1))) * self._norm_fact # Q * K.T() # batch_size * seq_len * seq_len
        #print('atten: ', atten)
        output = torch.bmm(atten,V) # Q * K.T() * V # batch_size * seq_len * dim_v

        return output
```

Python

6.12.3 Transformer

Why we need transformer?

- RNN: LSTM could catch long term dependency but could not be parallel Computed.
- CNN: Could parallel compute but week in catch long term dependency.
- Original attention: cares only the relationship between output and input, can't catch the relationship between inner context in input.

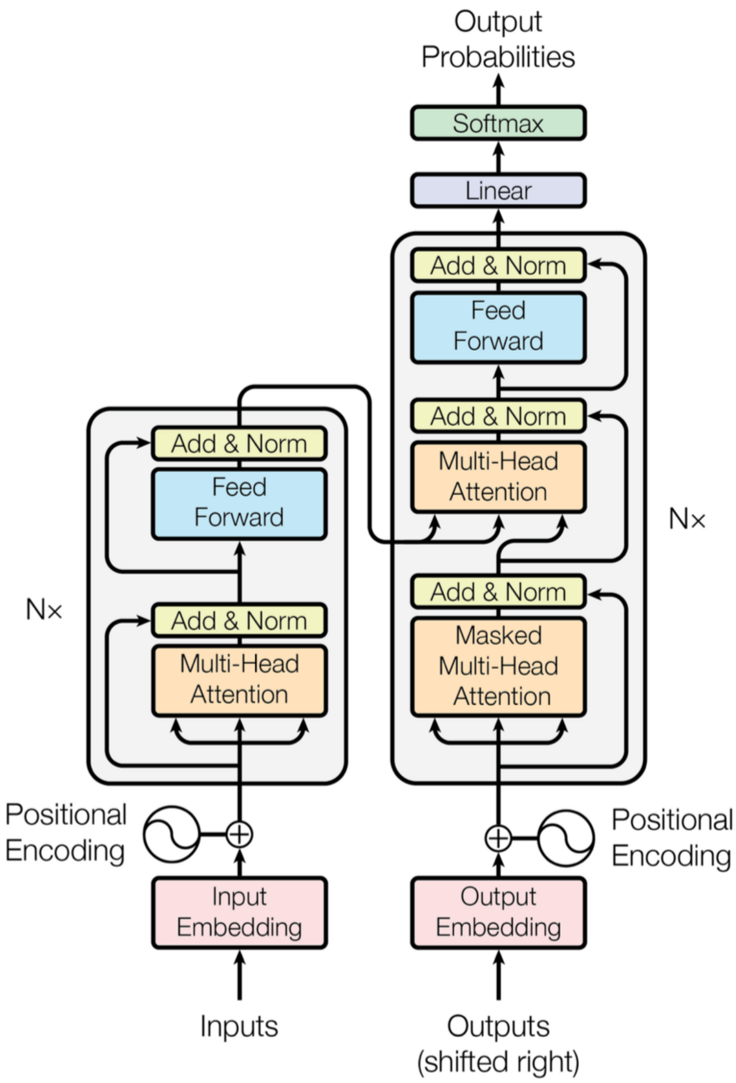


Figure 1: The Transformer - model architecture.

Encoder:

1 Word embedding

2 Positional encoding(sin, cos).

3 Multi-head attention

4 Layer-normalization + res of step 3

5 Resnet feedforward

6 Layer-normalization

Decoder:

1 Masked Multi-head Attention.
   - Why we use mask? because we only want our prediction based on previous output.
2 Add + Norm
3 Linear
4 Softmax gives the probability of predicted word
5 Output

Why we need position encoding:

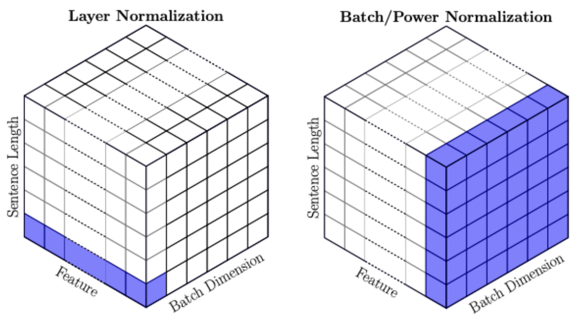
- Encode the position information of the input. Unlike RNN, which will naturally have the position feature.
- In transformer, position information uses encoding , which doesn't allow learning coefficients, but in BERT, position embedding was used because as pre-trained model, BERT task requires more information for the words and usually is applied in large datasets.

Why we need Resnet:

- To avoid gradient vanish.

Why we use layer normilization instead of batch normalization? :

- In CV we usually use batch norm because we know exactly the size of the image. In NLP, the input size is usually not fixed and layer normalization would norm in the feature scale. If we use batch norm, it might be hard for the final inference layer.
- Computational costs.
- For details, please review this paper. [15]



Part IV

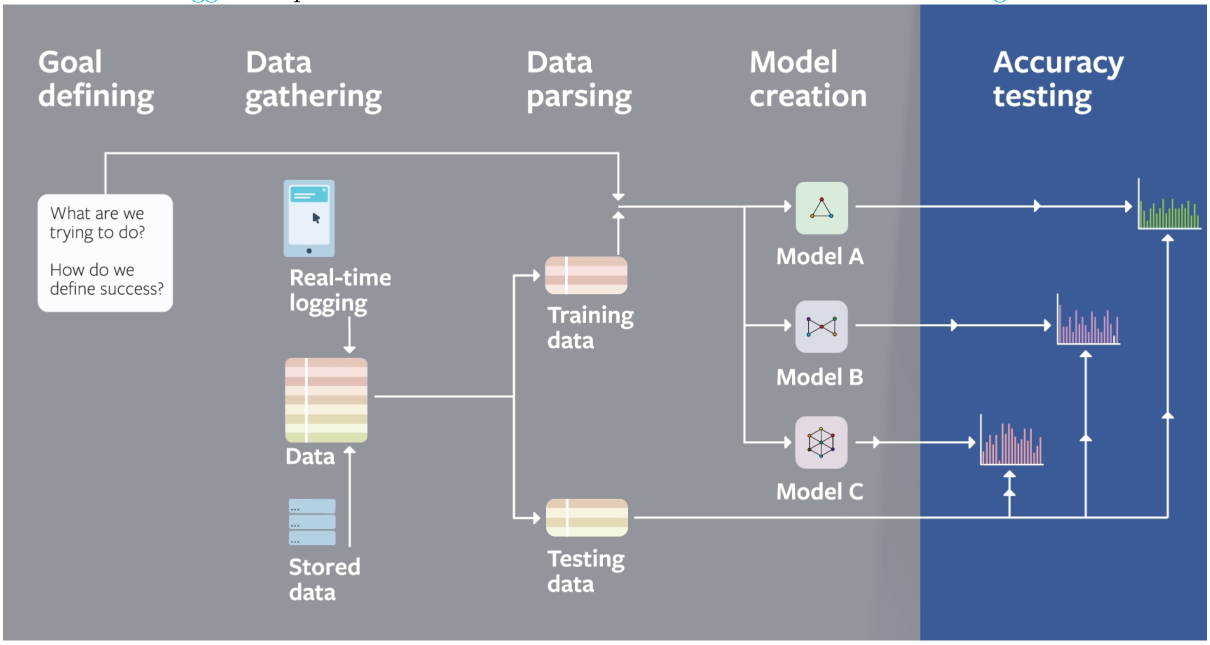
Recommendation System Design

Chapter 7

Machine Learning System Design

7.1 Introduction

This chapter introduces the part of designing a machine learning system during your interviews, which is similar like kaggle competition. I will follow the instructions on Facebook ML Design interview.



The typical design process consists of 6 following parts:

1 Problem Definition

2 Data Process

3 Evaluation

4 Features

5 Model

6 Experimentation

- Understand what is happening before your algorithm and after your algorithm which could have a huge influence on your model.
- The right side and direction is more important than your algorithm.

7.2 Problem Definition

for example for a **Business Problem** we have

- **what data to use?**
- **which feature to extract?**
- **which model to choose?**
- **what prediction results do we expect?**
- **How to improve the model based on the business/product goals?**

7.2.1 Remark

1. reasonable period of time.
    - How you quantifies your goal, for example **enjoyment**, estimate the data size and your model complexity.
2. Sparse Data
    - Preprocess the data if the data is imbalanced, which could leads to a fast iteration. For imbalanced data, consider the rare labels and work on those data.
3. features
    - model choose.
4. outcome
    - True desired outcome sensitive to the variations? (user select do not show me this, then recommendation system will calibrate.)

- Determine your goal of the project task.
- Simple is better than useless complicated.
- Define your label and training examples precisely. (edge cases)
- Don't prematurely optimize.

7.3 Data

Build and process the data is the core part of machine learning engineering. Key part:

1 **Data recency and real-time training**

2 **Training/prediction consistency**

3 **Records and sampling**

Note, in industry:

- The data is rare i.i.d

How to get the data/features?

1 Directly from the system or service that does inference.

2 Recalculate the feature along with some tables(SQL join).

How do you deal with imbalanced data?

- Resampling(for example SMOTE)

7.4 Evaluation

Now we know what kind of problem we are trying to solve and also have some raw training data. Model evaluation relies on two things

1 Offline training with logged data.(Fast and efficient)

2 Online experimentation by real time data.(Slow but sensitive and accurate)

Set up baseline model = simplest possible model to compare your model. Split your dataset into three parts

1 Training set

2 Evaluation set (Tuning)

3 Testing set to show results.

for metrics, it should be

1 Interpretable

2 Sensitive to the improvements of the model.

- Example (precision and recall)
- Check if it has statistical significance.

When the data is large it might be helpful to break the dataset into smaller groups and use metrics on those groups.

7.4.1 Remark

1 **Evaluation offline before evaluating online**

2 **Evaluate both the data you choose and statistics you calculated**

3 **Don't be bounded to evaluating and training on the same things**

4 **Understand where and why your performance comes from**

7.5 Features

Building features after is the second most important part in building our machine learning model. Feature engineering is how we extract useful information from the data and feed them to train a efficient model. Key note for features are:

- Interpretable, relevant to the outcome.
- Model architecture.
- Special cases
- Training data

Feature class:

1 Categorical
   - K out of N encoding.(gender, True/False, binary)

2 Continuous
   - Numerical, CTR

3 Derived(CTR)
   - May significantly improve your model however could also make your model hard to interpret.

7.5.1 Remark

- When design a new feature, make sure you have the specific semantic of this feature, then incorporate the change with new feature to build a new model and then deprecate the old model.
- Make sure your feature is consistent with your test and training dataset, inconsistent features such as different definition of CTR may hurt your model.
- Make sure your feature doesn't give away your model, or we call this **feature leakage.**
- **Feature coverage**: there might be large portion of missing data in the feature, such as birthday. However, many models have the power to handle missing values, such as Naive Bayesian, so you may want to ensure those features are valuable.

7.6 Model

Process for model selection:

- **How to pick a model**
- **How to tune a model**
- **How to compare models**

Key point:

- **Interpretability and ease to debug**
- **Data volume**
- **Training and prediction considerations**

Models:

- **Linear Model**
  - Pros: Easy to understand and debug, fast inference and work well in many areas.
  - Cons: suitable for small dataset and small amount of features.
  - How to deal with the nonlienar data with linear model?
    * Separate data into groups and then modeling with linear regression.
    * Pre-train the features with some linear mapping.(Use DNN to train your data and then use the last layer as your data for linear model.)

7.6.1 Remark

- **Once the data and features are fixed and the model is set, then we only need to do the following two steps:**
  1. Hyper Parameters Tuning
  2. Model Architecture Settings
     * **feature interactions for linear models**
     * **Number of leaves for tree based model**
     * **Type, number, with of layers for neural network**
  - There might be some trade-off in engineering perspective, for example, large hash memory might leads to a better model however it would decrease the serve's capability.
- When compare with other models, normalized entropy, negative log likelihood might be a good metric. If normalized entropy of a new model is greater than one we could say that model is not a valid ML model.
- **Reproducibility Principles** : track every step of your model know what data you are training on.

7.7 Experiments

The key way to ensure online and offline behaviors is the absence of feedback loops. What our ML model is optimizing for is usually called as **The Online Offline Gap**

7.7.1 Online Test

- A/B Test, split the data into two groups as Control and Test(review previous chapters)
- Minimize the time to first experiment.
  - The effect of your code changes.
  - The effect of your machine learning model.
- Forward test and backward test.
- Have a good backup plan.

7.8 Case Study

7.8.1 Design FB news feeding ranking system

Note of this paper: Practical Lessons from Predicting Clicks on Ads at Facebook. [16]
Related videos:

- On How Machine Learning and Auction Theory Power Facebook Advertising
- Ewa Dominowska - Generating a Billion Personal News Feeds - MLconf SEA 2016
- Serving a Billion Personalized News Feeds

7.8.1.1 Overview & Business Goals

What is news feed?

- Status updates
- Photos
- Videos, etc.

Goal

- Recommend the information which matters you the most.
- The recommendations should based on your control, such as like, unlike.
- Billions news and posts ranked every day.
- Construct a fast and accurate model with a good user exprience. Key Points:
  - Show the stories that related to you/(show like options)
  - Rank those stories.
  - Rank the New Content(Which you haven't clicked)
    * New friend shares same link you've seen.
    * Unseen old stories.
    * Seen stories with new comments.
  - Find a relevant and also diverse content. (Don't want to see almost everything in the same category even you are really interested in.)

7.8.1.2 Definition of Good & Measure

- Probability of click, like, comments
- Assign different weights to different events, according to significance.

7.8.1.3 Feature Engineering

- Over 100k potential features.

1 Prune to top 2k features.(Train many boost gradient trees and find out which of the feature is most important, iterate rows of features and each time eliminate the feature which is least important).

2 Under-sampling negative examples(Impression but no action) to help with # examples.
  - Under-sampling is a technique to balance uneven datasets by keeping all of the data in the minority class and decreasing the size of the majority class.

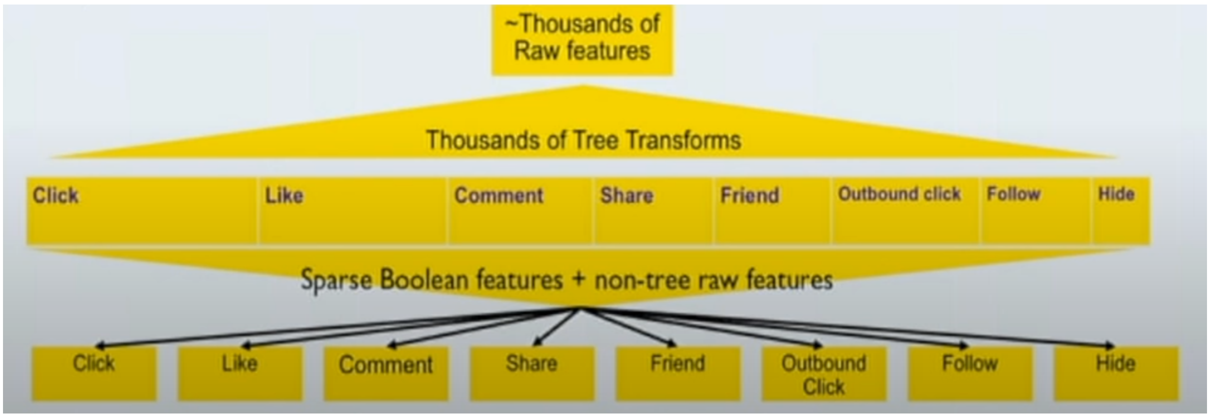
3 Do this for each feed event type: train many forests.

7.8.1.4 Model

- Use different models for different measurement. For exapmle, to test if user likes it, then we use the "like" model, also for "click" mode, etc.

Selection

- Logistic regression:
  - Quickly with a well trained model.
  - simple, fast, and easy to distributed.
- Gradient Boost Trees:
  - Not that fast than LR, but have nonlinearity and also is powerful in modeling those social related features.
- In order to combine the pros of those models, stacking might be a good idea in developing the model.
  - Use Thousands of trees to go through some features, the feed them and the features you haven't used in to logistic model to get the final prediction.
  - You could also replace the GDBT by neural network.



7.8.1.5 Measurement

Objective function.

- Metrics
  - Engagement: Click
  - Quality: Survey
  - Goal is to align ranking with personalized data.
  - Align ranking with personalized relevance.

Loss function

- use log loss for classification.
- rank by $\alpha p(like) + \beta p(comment)$

7.8.2 Machine Learning-Powered Search Ranking of Airbnb Experiences

This is an article read from the paper, for details please read Machine Learning-Powered Search Ranking of Airbnb Experiences.
Business goal:

- Provide the user a better search ranking based on their user data.

- At the beginning the dataset was small and limited. The date features we could select includes (impressions, clicks and bookings.
- **At this moment, randomly re-rank experience daily until a small dataset was collected for train the stage 1 ML model.**

Traning data collection:

- Collect search logs(i.e. clicks) who end up making books.
- Label training data.
  - Experience that were booked.
  - Experience that were not booked.

Feature engineering

- Experience duration.
- Price and Price/hour
- Category
- Reviews(comments)
- Number of bookings(with last N days)
- CTR

7.8.2.1 Baseline Model

Model selection Here use the Gradient Boost Tree, why?

- Pros:
  - Does not need to worry much about scaling the feature values, or missing values.
- Cons:
  - Not stable when the features change rapidly in fast growing market. (Thus you could change features from counts to some ratio.)

Model Evaluation

- Use hand out data that was not used for traning.
- Metrics selection
  - AUC
  - NDGG

- rank the experience based on the model scores(Probability of booking), then test where the booked experience would ranking among all experience the user clicked.(The higher the better)

Remark

- Cons
  - Offline model limited to using only Experience Features, the ranking Experience was the same for all users.
  - The output was just a complete ordering of all experience.

7.8.2.2 Personalized Machine Learning Model

Model Goal

- Add Personalization capability to improve the model performance.

Add personalized features:

- booked home location
- Trip dates
- Trip length
- Number of guests
- Domestic/International trip

Personalize based on their previous click

- Infer user interest in certain categories.
- Infer user's time-of-day availability.
- New Feature:
  - Category Intensity $= \sum_{d=d_0}^{d_{now}} = \alpha^{d-d_{now}} A_d$
  - Category Recency: Number of days that passed since the user last clicked on an Experience in that category.

Model training with personalized features

- Generated training data that contains those features by reconstructing the past based on search logs.
- leak the label, only use data before those books.

Testing the ranking model

- A/B test with 2 models, one have personalized data and another has non personalized features.

7.8.2.3 Online scoring

- **New Feature:**
  - Query Features: Would able to use the entered location, number of guests, and dates to engineer more features. For example, Use the user input location to get the Distance between Experience and Entered Location.
  - Browser language.
  - Country information, when travelling, For example, Japanese travelers prefer Classes and Workshops (e.g. Perfume making), US travelers prefer Food and Drink Experiences, while French travelers prefer History and Volunteering.

Model retraining

- 2 GBDT models, one for logged-in users, another for logged-out traffic.
- Pros:
  - Use the logged in model for far more users than before. Get more data to do the hyperparameter trainings.

7.8.2.4 Loss Function

Consider this as a binary classification problem and then use the log loss function as(will book = 1, not book = 0):

$$Loss = -\sum_{1}^{N}(y_i \log(p_i) + (1 - y_i)\log(1 - p_i))$$

The target predicted value is the probability that the user will click the searches.(Accuracy on each leaf.)

7.8.3 Machine learning in Fraud detection

This case follows the blog: From shallow to deep learning in fraud

7.8.3.1 Goal and Definitions

- Classification Problem with good information and bad(Fraud) information.
- Given a specific target with its features, use machine learning to check the probability if it is fraud information.

Note:

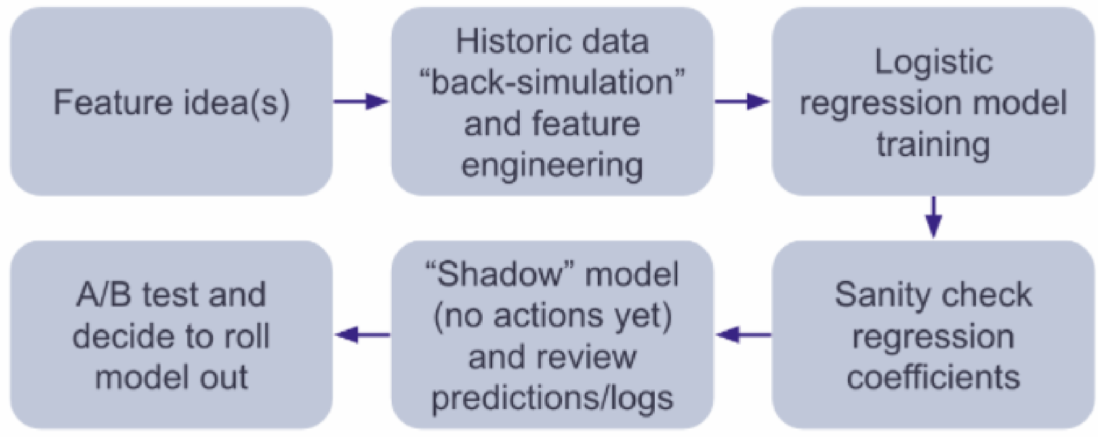
- In many industries so far, for classification problems, logistic regression is still a very powerful classifier.
  - The regression coefficients can be interpreted as how much a feature correlates with the likelihood of fraud.

7.8.3.2 Feature Engineering & data collection

- Linear methods like logistic regression sometimes are hard to deal with complicated features.

7.8.3.3 Model

Baseline Model: **Logistic regression**



New Model

- Can deal with more complicated user log information features.
- **GDBT**
  - More powerful than logistic regression.
  - The decision boundaries of logistic regression, its just the hyperplane of features. GDBT's decision boundaries are more like a high-dimensional boundary. Which allows us to operate with more complicated features.
- I think you could also use the stack method with GDBT as filter and logistic regression as final classifier to calculate the probability.

7.8.3.4 Evaluation & Measurement

Binary classification: (Fraud, not fraud), use log loss.

7.8.3.5 Remark

- Note in fraud detection, the data usually in quite imbalanced, (as the normal accounts are way more than fraud accounts, check specificity for your model). To better train your model, please review some mechanics in dealing imbalanced data.

7.8.4 Deep Learning for Recommendation System.

This is a case study of the following paper:

- Wide & Deep Learning for Recommender Systems [17]

Key concepts:

- A recommendation system can be viewed as a search ranking system.
  - Input: user query
  - Output: ranked list of items.

data:

- Usually from the user's history.
- High rank and sparse.

Models:

- Baseline: Logistic regression model.
  - Trained on binarized sparse features with on-hot encoding.
- Embedding-based features
  - factorization machines.
  - deep neural networks.
- Wide and Deep model to generate memorization (recommendations similar to old history) and generalization (new category recommendations based on historical data.)

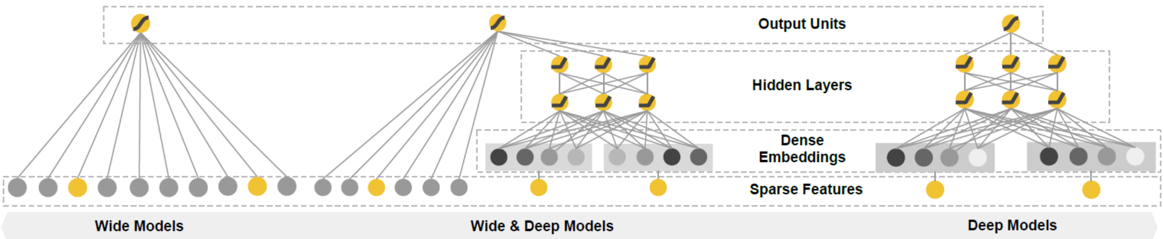


Figure 1: The spectrum of Wide & Deep models.

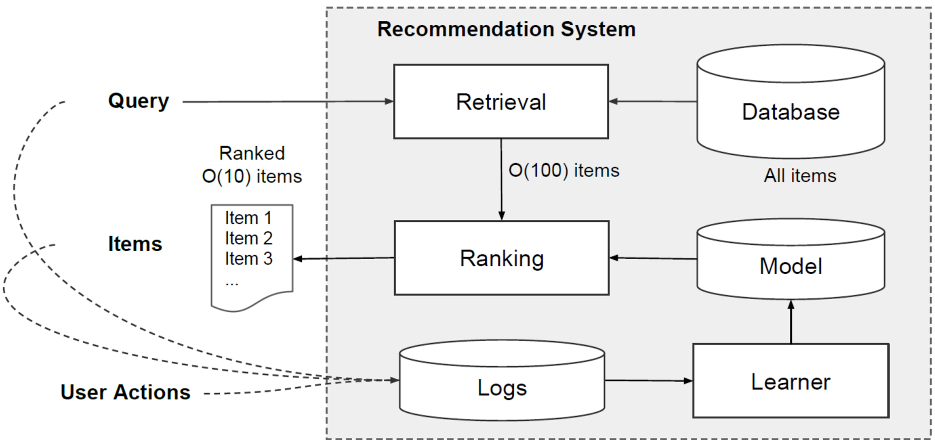


Figure 2: Overview of the recommender system.

Model Structure:

- training feed-forward neural network with embedding and linear model with feature transformations with sparse inputs.
- Simple idea! But works very well in real practice, remember we mentioned about the facebook news ranking case study, They have a very similar recommendation structure and uses the GDBT instead of DNN as their feature transformations.

Model Details:

- **Wide(linear)** component is a generalized linear model: $y = w^T x + b$, $x = [x_1, x_2, \cdots, x_d]$ is a vector of d features.
  - feature sets include the row feature and transformed features such as cross-product transformation:

$$\phi_k(\mathrm{x}) = \prod_{i=1}^{d} x_i^{c_{ki}} \quad c_{ki} \in \{0, 1\}$$

  - c_{ki} is a boolean variable, a cross-product transformation (e.g., `AND(gender=female, language=en)"`) is 1 if and only if the constituent features `(gender=female" and language=en")` are all 1, and 0 otherwise. This captures the interactions between the binary features, and adds non-linearity to the generalized linear model.
- **Deep** Component
  - Feed forward neural network.
    - For categorical features: original inputs are feature strings (language = en).
    - Convert those sparse data into a low-dimensional and dense real-valued vector.
    - Feed those low-dimensional data into the neural network as $a^{(l+1)} = f\left(W^{(l)} a^{(l)} + b^{(l)}\right)$.
- **Output**
  - Use logistic function to combine the previous output by wide and deep components. The loss function should be log loss.
- Why we stack or what is the difference between ensemble and joint training?
  - In an ensemble, individual models are trained separately.
  - Joint training trains all the models simultaneously.
  - In an ensemble, the model size is lager since we need to feed all the features into it.
  - In joint training the model combines the features as linear and also cross-product features.
  - The model uses **mini-batch stochastic optimization, back propagation, AdaGrad.**

Model Function:

$$P(Y = 1 \mid \mathbf{x}) = \sigma\left(\mathbf{w}_{wide}^T [\mathbf{x}, \phi(\mathbf{x})] + \mathbf{w}_{\text{deep}}^T \, a^{(l_f)} + b\right)$$

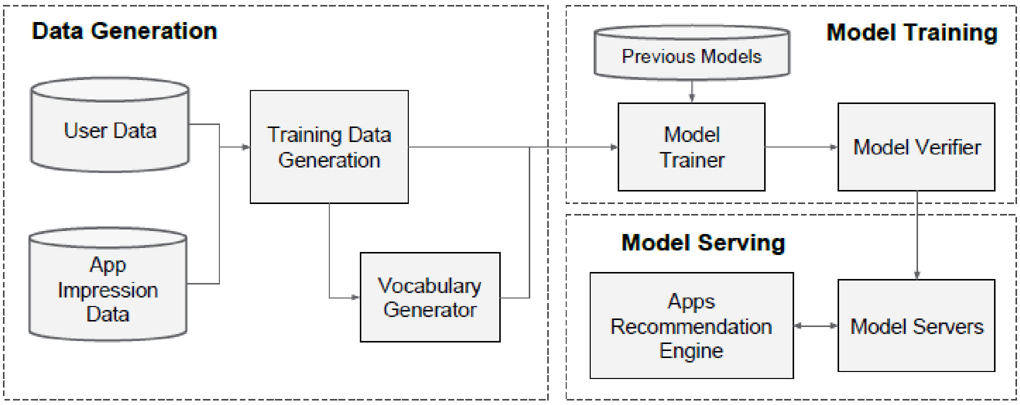


Figure 3: Apps recommendation pipeline overview.

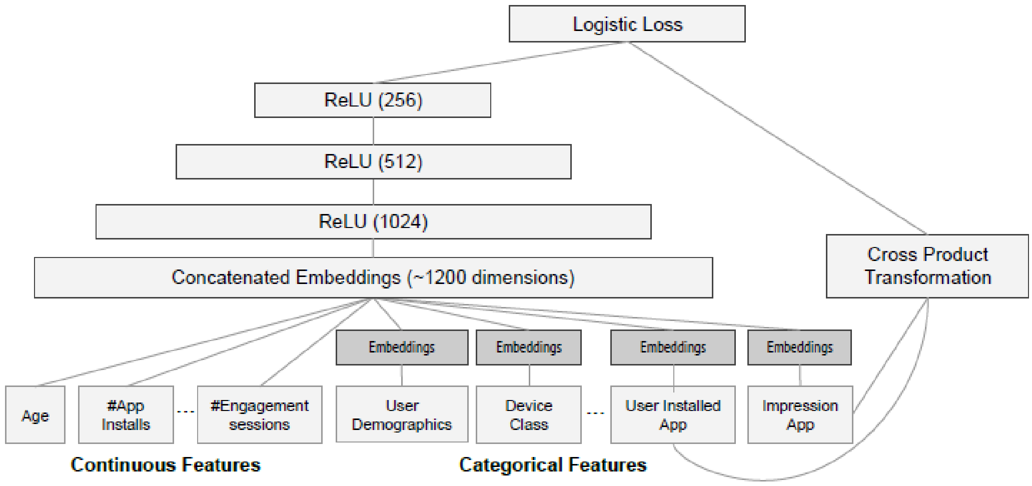


Figure 4: Wide & Deep model structure for apps recommendation.

Model Training & Serving:

- Training data:
  - Output: probability of the product the user would like to use.
  - Input: User's information data, cross-product features and app data.
- Model Prediction:
  - Server receive a set of app candidates and user data, feed to features into the model and rank them by their probabilities.

Evaluation A/B test

- Control Group: 1% of the users are randomly selected and uses the previous model.(Baeline model, highly optimized logistic regression model).
- Experiment group: 1% of the users are randomly selected and uses the deep & wide model.
- Use t test, z test, AUC to check the significance and model performance.

7.8.5 Remark

The machine learning design interview is quite unpredictable, as you don't know your interviewer's domain knowledge and how is he feeling today. Besides, there are countless methods in designing a recommendation system and the covered topics are broad! Just follow the steps and mention the classical topics in each part.(Like for features, normalization and selections, etc...). For those new grads like me, remember you are not omniscience and the so called machine learning design in interview is just armchair strategy, you will learn much and come up with so many "wired" problems in real practice.
The first machine learning design interview I had is with the big "F" company, I reviewed my interview notes again and again after the design interview and made sure that I have tried my best. However, you will find that someone had made his decision to fail you at the first glance.
So, be confident, it is okay to say you don't know for some topics, just take it easy and good luck!

Part V

Algorithms & Coding

Chapter 8

Data Structure Trivia

8.1 What is the difference between interpreted language and compiled language?

A compiled language is a programming language which are generally compiled and not interpreted.
An interpreted language is a programming language which are generally interpreted, without compiling a program into machine instructions. It is one where the instructions are not directly executed by the target machine, but instead read and executed by some other program (interpreter).

8.2 What is the difference between array and list?(data structures and memory)

A list in Python is a collection of items which can contain elements of multiple data types, which may be either numeric, character logical values, etc. It is an ordered collection supporting negative indexing. A list can be created using `[]` containing data values. Contents of lists can be easily merged and copied using python's inbuilt functions. An array is a vector containing homogeneous elements i.e. belonging to the same data type. Elements are allocated with contiguous memory locations allowing easy modification, that is, addition, deletion, accessing of elements. In Python, we have to use the array module to declare arrays. If the elements of an array belong to different data types, an exception "Incompatible data types" is thrown.(Reference: GeeksForGeeks)
Meomory in C++:
An array is a contiguous chunk of memory with a fixedsize whereas a list is typically implemented as individual elements linked to each other via pointers and does not have a fixed size.
Retrieving a specific index in an array is also much faster since the memory is contiguous. Any element in an array can be located as an offset from the address of the first element in the array. `O(1)`
A list on the other hand requires a traversal through the elements to find a specific index. `O(n)`

8.3 Heap vs Stack

Reference: GeeksForGeeks

Parameter	STACK	HEAP
Basic	Memory is allocated in a contiguous block.	Memory is allocated in any random order.
Allocation and Deallocation	Automatic by compiler instructions.	Manual by the programmer.
Cost	Less	More
Implementation	Easy	Hard
Access time	Faster	Slower
Main Issue	Shortage of memory	Memory fragmentation
Locality of reference	Excellent	Adequate
Flexibility	Fixed-size	Resizing is possible
Data type structure	Linear	Hierarchical

8.4 What is the difference between array and linked-list

1 Array could contains similar type of data whereas whereas the Linked list is considered as a non-primitive data structure contains a collection of unordered linked elements known as nodes.

2 In array the item is accessed by index, but in linked-list you need traverse from head to reach the item.(`O(1)`vs `O(N)`)

3 Deleting and insert in array may take linear time but in linked-list it is faster. (`O(N)`vs `O(1)`)

4 Array has fixed size but linked-list is flexible in rearranging the size.

5 In an array, memory is assigned during compile time while in a Linked list it is allocated during execution or runtime.

6 Elements are stored consecutively in arrays whereas it is stored randomly in Linked lists.

7 The requirement of memory is less due to actual data being stored within the index in the array. As against, there is a need for more memory in Linked Lists due to storage of additional next and previous referencing elements.

8.5 DS Operation Complexity Chart

Common Data Structure Operations

Data Structure	Time Complexity								Space Complexity
	Average				Worst				Worst
	Access	Search	Insertion	Deletion	Access	Search	Insertion	Deletion	
Array	Θ(1)	Θ(n)	Θ(n)	Θ(n)	O(1)	O(n)	O(n)	O(n)	O(n)
Stack	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Queue	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Singly-Linked List	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Doubly-Linked List	Θ(n)	Θ(n)	Θ(1)	Θ(1)	O(n)	O(n)	O(1)	O(1)	O(n)
Skip List	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n log(n))
Hash Table	N/A	Θ(1)	Θ(1)	Θ(1)	N/A	O(n)	O(n)	O(n)	O(n)
Binary Search Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)
Cartesian Tree	N/A	Θ(log(n))	Θ(log(n))	Θ(log(n))	N/A	O(n)	O(n)	O(n)	O(n)
B-Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Red-Black Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
Splay Tree	N/A	Θ(log(n))	Θ(log(n))	Θ(log(n))	N/A	O(log(n))	O(log(n))	O(log(n))	O(n)
AVL Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(log(n))	O(log(n))	O(log(n))	O(log(n))	O(n)
KD Tree	Θ(log(n))	Θ(log(n))	Θ(log(n))	Θ(log(n))	O(n)	O(n)	O(n)	O(n)	O(n)

8.6 Sorting Algorithm Complexity Chart

Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst	Worst
Quicksort	Ω(n log(n))	Θ(n log(n))	O(n^2)	O(log(n))
Mergesort	Ω(n log(n))	Θ(n log(n))	O(n log(n))	O(n)
Timsort	Ω(n)	Θ(n log(n))	O(n log(n))	O(n)
Heapsort	Ω(n log(n))	Θ(n log(n))	O(n log(n))	O(1)
Bubble Sort	Ω(n)	Θ(n^2)	O(n^2)	O(1)
Insertion Sort	Ω(n)	Θ(n^2)	O(n^2)	O(1)
Selection Sort	Ω(n^2)	Θ(n^2)	O(n^2)	O(1)
Tree Sort	Ω(n log(n))	Θ(n log(n))	O(n^2)	O(n)
Shell Sort	Ω(n log(n))	Θ(n(log(n))^2)	O(n(log(n))^2)	O(1)
Bucket Sort	Ω(n+k)	Θ(n+k)	O(n^2)	O(n)
Radix Sort	Ω(nk)	Θ(nk)	O(nk)	O(n+k)
Counting Sort	Ω(n+k)	Θ(n+k)	O(n+k)	O(k)
Cubesort	Ω(n)	Θ(n log(n))	O(n log(n))	O(n)

8.7 Merge Sort

Merge sort is a divide and conquer algorithm. It divides the target array into two halves and sort them respectively. Then merge them together to get a sorted list.
Algorithm:

- While `l<r`

1 find the middle point $m = \frac{l+r}{2}$

2 sort the left part `MergeSort(l, m, arr);`

3 sort the right part `MergeSort(m+1, r, arr);`

4 merge the two halves `merge(arr1, arr2);`

8.8 Quick Sort

review GeeksforGeeks

```
# This function takes last element as pivot, places
# the pivot element at its correct position in sorted
# array, and places all smaller (smaller than pivot)
# to left of pivot and all greater elements to right
# of pivot
def partition(arr,low,high):
    i = ( low-1 )         # index of smaller element
    pivot = arr[high]     # pivot

    for j in range(low , high):

        # If current element is smaller than the pivot
        if   arr[j] < pivot:

            # increment index of smaller element
            i = i+1
            arr[i],arr[j] = arr[j],arr[i]

    arr[i+1],arr[high] = arr[high],arr[i+1]
    return ( i+1 )

# The main function that implements QuickSort
# arr[] --> Array to be sorted,
# low  --> Starting index,
# high  --> Ending index

# Function to do Quick sort
def quickSort(arr,low,high):
    if low < high:

        # pi is partitioning index, arr[p] is now
        # at right place
        pi = partition(arr,low,high)

        # Separately sort elements before
        # partition and after partition
        quickSort(arr, low, pi-1)
        quickSort(arr, pi+1, high)
# call function
quickSort(arr,0,n-1)
```

Python

8.9 What is hash-table and what is its time complexity? How to handle the collision in hash-map?

A hash table, also known as a hash map, is a data structure that maps keys to values. It is one part of a technique called hashing, the other of which is a hash function. A hash function is an algorithm that produces an index of where a value can be found or stored in the hash table. The average complexity of insert, search and delete is `O(1)`. Some important notes about hash tables:

- Values are not stored in a sorted order.
- You mush account for potential collisions. This is usually done with a technique called chaining. Chaining means to create a linked list of values, the keys of which map to a certain index.

Collision: chaining and open addressing See freeCodeCamp

8.10 Advantages of BST over Hash Table

The time complexity of searching in a binary tree is `O(log(n))`

1 We can get all keys in sorted order by just doing Inorder Traversal of BST. This is not a natural operation in Hash Tables and requires extra efforts.

2 Doing order statistics, finding closest lower and greater elements, doing range queries are easy to do with BSTs. Like sorting, these operations are not a natural operation with Hash Tables.

3 BSTs are easy to implement compared to hashing, we can easily implement our own customized BST. To implement Hashing, we generally rely on libraries provided by programming languages.

4 With Self-Balancing BSTs, all operations are guaranteed to work in $O(\log(n))$ time. But with Hashing, $O(\log(1))$ is average time and some particular operations may be costly, especially when table resizing happens.

8.11 What is the return type of `range()` function in `python`?

The range() function is used to generate a sequence of numbers over time. At its simplest, it accepts an integer and returns a range object (a type of iterable).

Chapter 9

Algorithms

This chapter follows this great guide in Chinese: A leetcode grinding guide, star it on github!
For those who are interested in practicing Facebook problems, hope my notion note will be helpful.

9.1 Greedy

9.1.1 455. Assign Cookeis

Space `O(glog(g)) + O(slog(s))+O(s)+O(g)`
Time `O(1)`

```
def findContentChildren(self, g: List[int], s: List[int]) -> int:
    g.sort()
    s.sort()
    kid, cookie = 0, 0
    while kid<len(g) and cookie<len(s):
        if s[cookie]>=g[kid]:
            kid += 1
        cookie += 1
    return kid
```

Python

```
int findContentChildren(vector<int>& g, vector<int>& s) {
    sort(g.begin(), g.end());
    sort(s.begin(), s.end());
    int i=0, j=0;
    while(i<g.size() && j<s.size()){
        if (s[j]>=g[i]) i++;
        j++;
    }
    return i;
}
```

C++

9.1.2 135. Candy

Every time when come up with such problem which has conditions on its previous and next neighbors, you may notice that this problem is highly related to `prefix` and `suffix` traversal. Then is question requires two traversals, `prefix` satisfies the first condition and the second `suffix` traversal guaranties the second condition.
Time `O(n)`, Space `O(n)`.

```
def candy(self, ratings: List[int]) -> int:
    nums = [1]*len(ratings)
    for i in range(1, len(ratings)):
```

```python
            if ratings[i]>ratings[i-1]:
                nums[i] = nums[i-1]+1
        for i in range(len(ratings)-2, -1, -1):
            if ratings[i]>ratings[i+1]:
                nums[i] = max(nums[i+1]+1,nums[i])
        return sum(nums)
```

Python

```cpp
    int candy(vector<int>& ratings) {
        if (ratings.size()<2) return (int)ratings.size();
        int n = ratings.size();
        vector<int> nums(n,1);
        for (int i=1; i<n; i++){
            if (ratings[i]>ratings[i-1]) nums[i] = nums[i-1]+1;
        }
        for (int i=n-2; i>=0; i--){
            if (ratings[i]>ratings[i+1]) nums[i] = max(nums[i], nums[i+1]+1);
        }
        return accumulate(nums.begin(), nums.end(), 0);
    }
```

C++

Furthermore, this problem could be optimized with a Space complexity `O(1)`, see code

9.1.3 435. Non-overlapping Intervals

Time `O(nlog(n))`, Space `O(1)`

```python
    def eraseOverlapIntervals(self, intervals: List[List[int]]) -> int:
        intervals.sort(key=lambda x:x[1])
        ans, end = 0, float('-inf')
        for interval in intervals:
            if interval[0]<end: ans += 1
            else: end = interval[1]
        return ans
```

Python

```cpp
    int eraseOverlapIntervals(vector<vector<int>>& intervals) {
        int n = intervals.size();
        if (n<2) return 0;
        sort(intervals.begin(), intervals.end(),
             [](const vector<int>& a, const vector<int>&b){
                 return a[1]<b[1];
             }
            );
        int ans = 0, end = INT_MIN;
        for (auto interval:intervals){
            if (interval[0]<end){
                ans++;
            } else{
                end = interval[1];
            }
        }
        return ans;
    }
```

C++

9.1.4 605. Can Place Flowers

Time `O(n)`, Space `O(1)`.

```python
    def canPlaceFlowers(self, flowerbed: List[int], n: int) -> bool:
        ans = 0
        for i in range(len(flowerbed)):
            if not flowerbed[i] and (i==0 or flowerbed[i-1] == 0) and (i == len(
    flowerbed)-1 or flowerbed[i+1] == 0):
                ans += 1
```

```
                flowerbed[i] = 1
            if ans>=n: return True
        return ans>=n
```

Python

```
    bool canPlaceFlowers(vector<int>& flowerbed, int n) {
        int s = flowerbed.size();
        int ans = 0;
        for (int i = 0; i<s; i++){
            if (flowerbed[i]==0 && (i==0 || flowerbed[i-1]==0) && (i==s-1 || flowerbed[i
    +1]==0)){
                ans++;
                flowerbed[i] = 1;
            }
            if(ans>=n) return true;
        }
        return ans>=n;
    }
```

C++

9.1.5 452. Minimum Number of Arrows to Burst Balloons

Time `O(nlog(n))`, Space `O(1)`. Algorithm:

1 sort the items by the end time

2 traverse the points, if the next point has overlaps with the current intersection, then this means our current arrow could also shoot this balloon, else use a new arrow to shoot the next and update the intersection.

```
    def findMinArrowShots(self, points: List[List[int]]) -> int:
        if len(points)<2: return len(points)
        points.sort(key = lambda x:x[1])
        ans = 1
        end= points[0][1]
        for p in points:
            if p[0]>end:
                ans += 1
                end= p[1]
        return ans
```

Python

```
  int findMinArrowShots(vector<vector<int>>& points) {
    if (points.size() <2) return (int)points.size();
    sort(begin(points), end(points),
         [](const vector<int> &a, const vector<int> &b){
             return a[1] < b[1];
         }
        );
    int ans = 1;
    int end = points[0][1];
    for (auto p : points){
        if (p[0]>end){
            ans++;
            end= p[1];
        }
    }
    return ans;
  }
```

C++

9.2 Two Pointers

9.2.1 167 Two Sum 2

Time `O(n)`, Space `O(1)`.

```python
    def twoSum(self, numbers: List[int], target: int) -> List[int]:
        l = 0
        r = len(numbers)-1
        while l<r:
            s = numbers[l] + numbers[r]
            if s == target:
                return [l+1, r+1]
            elif s<target:
                l += 1
            else:
                r -= 1
        return []
```

Python

```cpp
    vector<int> twoSum(vector<int>& numbers, int target) {
        int l = 0, r = numbers.size()-1, s;
        while (l<r){
            s = numbers[l] + numbers[r];
            if (s == target){
                return {l+1, r+1};
            } else if(s<target){
                l++;
            }else{
                r--;
            }
        }
        return {0,0};
    }
};
```

C++

9.2.2 88. Merge Sorted Array

Coner case: `j>=0 and i<0`. Time `O(m+n)`, Space `O(1)`.

```python
    def merge(self, nums1: List[int], m: int, nums2: List[int], n: int) -> None:
        i, j, idx = m-1, n-1, m+n-1
        while i>=0 and j>=0:
            if nums1[i]>nums2[j]:
                nums1[idx] = nums1[i]
                i-=1
            else:
                nums1[idx] = nums2[j]
                j-=1
            idx -= 1
        if j>=0: nums1[:j+1] = nums2[:j+1]
```

Python

```cpp
public:
    void merge(vector<int>& nums1, int m, vector<int>& nums2, int n) {
        int idx=m+n-1,i = m-1, j=n-1;
        while(j>=0){
            nums1[idx--] = i>=0&&nums1[i]>nums2[j]? nums1[i--]:nums2[j--];
       }
    }
};
```

C++

9.2.3 142. Linked List Cycle II

One could easily solve this problem by traverse the linked list and use a hash map to store the seen node, the first repeated seen node would be the circle starting point. Time `O(n)`, Space `O(n)`.
Here we introduce the Floyd algorithm to search the node in a circular linked list:

1 initialize two pointers `slow, fast` with `slow` moves one step forward and `fast` moves two steps forward.

2 if those two pointers meets, then relocated `fast` to `head`, and now let both `fast, slow` move one step forward in loop.

3 if `fast == slow` again, then this node is the entry node of the circle.

Proof:
Suppose the length of the linked list is `m`, the entry node is at position `n`. Then The circle length is `l=m-n+1> 0`
Assume `slow` moved `s` steps when the first time they meet, `fast` moved `2S`.

$$\{(2s-n)-(s-n)\}\%l == 0$$
$$\Rightarrow s\%l==0$$

If setting `fast` to `head` and let `x` be the steps for fast to travel for the next meeting, position `fast = x`, position `slow = n +(s-n+x) % l`:

$$n +(s-n+x) \% l = x$$
$$\Rightarrow (x-n) \% l = x-n$$
$$\Rightarrow n==x$$

Time `O(n)`, Space `O(1)`.

```python
def detectCycle(self, head: ListNode) -> ListNode:
    if not head: return None
    fast, slow = head, head
    while fast and fast.next:
        fast = fast.next.next
        slow = slow.next
        if fast == slow:
            fast = head
            break
    if not fast or not fast.next: return None
    while fast:
        if fast == slow: return fast
        fast = fast.next
        slow = slow.next
    return None
```

Python

```cpp
ListNode *detectCycle(ListNode *head) {
    if (!head) return nullptr;
    ListNode* fast = head, *slow = head;
    while (fast && fast->next){
        fast = fast->next->next;
        slow = slow->next;
        if (slow == fast){
            fast = head;
            break;
        }
    }
    if (!fast || !fast->next) return nullptr; // check if circle
    while (fast){
        if (fast == slow){
            return fast;
        }
        fast = fast->next;
        slow = slow->next;
    }
    return nullptr;
}
```

C++

9.2.4 76. Minimum Window Substring

Sliding window, Time `O(n)`, Space `O(1)` expect for the output.

```python
def minWindow(self, s: str, t: str) -> str:
    tcount = collections.Counter(t)
    min_l, min_len = 0, float('inf')
    l, r = 0, 0
    require = len(t)
    while r<len(s)+1:
        if require:
            if r == len(s): break
            if s[r] in tcount:
                if tcount[s[r]]>0: require -= 1
                tcount[s[r]] -= 1
            r += 1
        else: #did find the satisfied answer
            if r - l < min_len:
                min_l = l
                min_len = r - l
            if s[l] in tcount:
                if tcount[s[l]]>=0: require += 1
                tcount[s[l]] += 1
            l += 1
    return s[min_l:min_l+min_len] if min_len != float('inf') else ''
```

Python

```cpp
string minWindow(string s, string t) {
    if (s.size() == 0 || t.size() == 0) return "";
    vector<int> remaining(128, 0);
    int required = t.size();
    for (int i = 0; i < required; i++) remaining[t[i]]++;
    // left is the start index of the min-length substring ever found
    int min = INT_MAX, start = 0, left = 0, i = 0;
    while(i <= s.size() && start < s.size()) {
        if(required) {
            if (i == s.size()) break;
            if (remaining[s[i]] > 0) required--;
            remaining[s[i]]--;
            i++;
        } else {
            if (i - start < min) {
                min = i -start;
                left = start;
            }
            if (remaining[s[start]] >= 0) required++;
            remaining[s[start]]++;
            start++;
        }
    }
    return min == INT_MAX? "" : s.substr(left, min);
}
```

C++

9.2.5 633. Sum of Square Numbers

Time O(clog(c)), Space O(1).

```python
def judgeSquareSum(self, c: int) -> bool:
    a = 0
    while a*a<=c:
        b = math.sqrt(c-a*a)
        if b == int(b): return True
        a += 1
    return False
```

Python

```cpp
bool judgeSquareSum(int c) {
    long a = 0, b = sqrt(c);
    while (a<=b){
        long s = a*a + b*b;
        if (s> c) b--;
        else if (s < c) a++;
        else return true;
    }
```

```
        return false;
    }
```

C++

9.2.6 680. Valid Palindrome II

Time O(n), Space O(1).

```
    def is_valid(self, s, start, end):
        l, r = start, end
        while l<r:
            if s[l] != s[r]: return False
            l += 1
            r -=1
        return True
    def validPalindrome(self, s: str) -> bool:
        l, r =0, len(s)-1
        while l<r:
            while l<r and s[l] == s[r]:
                l += 1
                r -= 1
            if l==r: return True
            #now s[l]!= s[r], check which one to delete
            return self.is_valid(s, l, r-1) or self.is_valid(s, l+1, r)
        return True
```

Python

```
    bool is_valid(const string& s, const int& start, const int& end){
        int l= start, r = end;
        while (l<r){
            if (s[l]!=s[r]) return false;
            l++;
            r--;
        }
        return true;
    }
    bool validPalindrome(string s) {
        int l=0, r=s.size()-1;
        while (l<r){
            while (l<r && s[l]==s[r]){
                l++;
                r--;
            }
            if (l==r) return true;
            return is_valid(s, l, r-1) || is_valid(s, l+1, r);
        }
        return true;
    }
```

C++

9.2.7 340. Longest Substring with At Most K Distinct Characters

Time O(n), Space O(1).

```
    def lengthOfLongestSubstringKDistinct(self, s: str, k: int) -> int:
        l, r = 0, 0
        dic = collections.defaultdict(int)
        ans = 0
        while r<len(s):
            dic[s[r]] += 1
            if len(dic)<=k:
                ans = max(ans, r-l + 1)
            else:
                while len(dic)>k:
                    dic[s[l]] -= 1
                    if dic[s[l]] == 0:
                        del dic[s[l]]
                    l += 1
            r += 1
```

```
        return ans
```

Python

```
    int lengthOfLongestSubstringKDistinct(string s, int k) {
        if(k==0) return 0;
        unordered_map<char, int> count;
        int l=0, r=0;
        int ans = 0;
        while (r<s.size()){
            count[s[r]]++;
            if (count.size()<=k){
                ans= max(ans, r-l+1);
            }
            else{//delete from the left
                while(count.size()>k){
                    count[s[l]]--;
                    if (count[s[l]]==0) count.erase(s[l]);
                    l++;
                }
            }
            r++;
        }
        return ans;
    }
```

C++

9.2.8 524. Longest Word in Dictionary through Deleting

Time `O()`, Space `O()`.

```
    def findLongestWord(self, s: str, d: List[str]) -> str:
        d.sort(key = lambda x:(-len(x), x))
        for word in d:
            i = 0
            for c in s:
                if i<len(word) and c == word[i]: i+=1
            if i==len(word): return word;
        return ""
```

Python

```
    string findLongestWord(string s, vector<string>& d) {
        sort(d.begin(), d.end(), [](const string& a, const string&b){
            if (a.size()==b.size()) return a<b;
            return a.size()>b.size();
        } );
        for (string word:d){
            int i= 0;
            for (char c:s){
                if (i<word.size() && word[i] == c) i++;
            }
            if (i==word.size()) return word;
        }
        return "";
    }
```

C++

9.3 Binary Search

9.3.1 69. Sqrt(x)

Time `O(log(n)`, Space `O(1)`.

```
    int mySqrt(int x) {
        if(x<2) return x;
        int l=0, r = x;
        while (l<r) {
```

```
            int mid = (r-l)/2 + l;
            if (mid>x/mid){
                r = mid;
            } else if (mid <x/mid){
                l = mid+1 ;
            } else{
                return mid;
            }
        }
        return l-1;
    }
```

C++

```
    def mySqrt(self, x: int) -> int:
        if x == 1: return 1
        # #binary search n^2-x = 0
        # a, b = 0.0, 1.0*x
        # mid = 0
        # while abs(mid*mid-x)>.2:
        #     mid = (a + b)/2
        #     if mid*mid - x>0: b = mid
        #     else: a = mid
        # return int(mid) if (int(mid)+1)**2>x else int(mid)+1
        # newton
        n = 0.1
        while abs(n*n - x) >1:
            n -= 0.5*(n - x/n)
        return int(n)
```

Python

9.3.2 34. Find First and Last Position of Element in Sorted Array

Time `O(log(n))`, Space `O(1)`.

```
class Solution:
    def binary(self, nums, target):
        l, r = 0, len(nums)
        while l<r :
            mid = (r-l)//2 + l
            if nums[mid]>= target:
                r = mid
            else:
                l = mid + 1
        return l
    def searchRange(self, nums: List[int], target: int) -> List[int]:
        l = self.binary(nums, target)
        r = self.binary(nums, target+1) -1
        return [l, r] if 0<=l<len(nums) and nums[l] == target else [-1, -1]
```

Python

```
class Solution {
public:
vector<int> searchRange(vector<int>& nums, int target) {
    int idx1 = lower_bound(nums, target);
    int idx2 = lower_bound(nums, target+1)-1;
    //cout<<idx1<<endl;
    if (idx1 < nums.size() && nums[idx1] == target)
        return {idx1, idx2};
    else
        return {-1, -1};
}

int lower_bound(vector<int>& nums, int target) {
    int l = 0, r = nums.size();
    while (l < r) {
        int mid = (r-l)/2+l;
        //cout<<l<<' '<<r<<' '<<mid<<endl;
        if (nums[mid] < target)
            l = mid+1;
        else
            r = mid;
```

```
    }
    return 1;
}
};
```

C++

9.4 Sorting

For some complicated sorting algorithms such as quick sort, I highly recommend practicing before interviews.

9.4.1 215. Kth Largest Element in an Array

Time `O(nlog(k))`, Space `O(k)`.

```python
    def findKthLargest(self, nums: List[int], k: int) -> int:
        # a heap solution
        h = []
        for n in nums:
            if len(h)==k:
                heapq.heappushpop(h, n)
            else:
                heapq.heappush(h,n)
        return h[0]
```

Python

```cpp
int findKthLargest(vector<int>& nums, int k) {
        priority_queue<int, vector<int>, greater<int> > dp;
        for(auto num:nums){
            dp.push(num);
            if(dp.size()>k) dp.pop();
        }
        return dp.top();
    }
```

C++

9.4.2 347. Top K Frequent Elements

Time `O(N+M(the distinct element numbers))log(k))`, Space `O(M + k)`.

```python
    def topKFrequent(self, nums: List[int], k: int) -> List[int]:
        C = collections.Counter(nums)
        h = []
        for key in C:
            if len(h)==k:
                heapq.heappushpop(h,(C[key], key))
            else:
                heapq.heappush(h, (C[key], key))
        return [a[1] for a in h ]
```

Python

```cpp
    using elem_t  = pair<int,int>;
    using elem_vt = vector<elem_t>;
    struct Cmp {
        bool operator()(const elem_t& lhs, const elem_t& rhs) const {
            return lhs.second > rhs.second;
        }
    };
public:
    vector<int> topKFrequent(vector<int>& nums, const int K) {
        unordered_map<int, int> mp;
        for (auto e : nums) mp[e]++;

        priority_queue<elem_t, elem_vt, Cmp> pq;
        for (auto& [k,v] : mp) {
```

```
            pq.push({k,v});
            if (pq.size() > K) pq.pop();
        }

        vector<int> res;
        while (pq.size()) {
            res.emplace_back(pq.top().first);
            pq.pop();
        }
        return res;
    }
```

C++

9.4.3 Quick Sort 912. Sort an Array

Time O(NlogN) average, Space O(N).

```python
class Solution:
    def sortArray(self, nums: List[int]) -> List[int]:
        self.quick_sort(nums, 0, len(nums) - 1)
        return nums

    def quick_sort(self, nums, lower, upper):
        if lower < upper:
            pivot = self.partition(nums, lower, upper)
            self.quick_sort(nums, lower, pivot - 1)
            self.quick_sort(nums, pivot + 1, upper)

        return

    def partition(self, nums, l, r):
        #random pick element part, one could also directly use pivot = nums[r]
        pivot_idx = random.randint(l, r)
        nums[pivot_idx], nums[r] = nums[r], nums[pivot_idx]
        pivot = nums[r]
        idx = l
        for i in range(l, r):
            if nums[i] < pivot:
                nums[idx], nums[i] = nums[i], nums[idx]
                idx += 1
        nums[idx], nums[r] = nums[r], nums[idx]

        return idx
```

Python

9.5 Reservoir Sampling

Reservoir sampling is an algorithm in picking random numbers uniformly when have a large pool of numbers.

Problem . Having a large string A with unknown length, how to select k variables with equal probabilities?

Algorithm.

1 Store the first k elements in the stream as `curr`

2 Assume we have seen the `stream` with length n, Loop i in range $[k, n]$, we randomly pick j from $[0, i]$, each index $\leq k$ has probability $\frac{k}{i}$ to be chosen. Let `curr[j] = stream[i]` if $j < k$.

□

Let's proof any of the elements has the same probability $\frac{k}{n}$.

Proof.

- By induction, obviously it works for $n = 1$. Let assume the algorithms works for case $n > 1$.
- For $n+1, n \to \infty$, we want to prove that the probability of each element to be chosen in the output array is $\frac{k}{n+1}$.
- For any element x_i in `curr` with `len(curr) = k`, it has a probability $\frac{k}{n}$ to be chosen (by induction), for $l = n+1$, to prove $P(pick\ x_i) = \frac{k}{n+1}$, thus for $x_i \in$ `curr`, $P(x_i$ stay in `curr`$|$`len(stream) = n + 1`$) = P(x_i \text{ in curr}) * P(\text{not pick } x_i) = \frac{k}{n}\frac{n}{n+1} = \frac{k}{n+1}$.

□

```
def selectKItems(stream, n, k):
  reservoir = [0]*k
  for i in range(k):
    reservoir[i] = stream[i];
  i = k
  while(i < n):
    j = random.randrange(i+1)
    if(j < k):
      reservoir[j] = stream[i]
      i+=1
  return reservoir
```

Python

Bibliography

[1] D. Stefanica, R. Radojičić, and T. Wang. *150 Most Frequently Asked Questions on Quant Interviews.* Pocket book guides for Quant interviews. FE Press, 2013.

[2] Xinfeng Zhou. *A Practical Guide to Quantitative Finance Interviews.* CreateSpace, Scotts Valley, CA, 2008.

[3] F. Mosteller. *Fifty Challenging Problems in Probability with Solutions.* Dover Books on Mathematical and Logical Puzzles, Cryptography and Word Recreations. Dover Publications, 1987.

[4] Frederick Mosteller. *Fifty challenging problems in probability with solutions.* Courier Corporation, 1987.

[5] R. Feynman. The brownian movement. *The Feynman Lectures of Physics*, 41&#45, 1964.

[6] LJ Anthony. The cambridge dictionary of statistics. *Reference Reviews*, 2003.

[7] B. Rosner. *Fundamentals of Biostatistics.* Cengage Learning, 2010.

[8] 诸葛越. 葫芦娃. 百面机器学习算法工程师带你去面试. 2018.

[9] Daniel Yates. *The practice of statistics : TI-83/89 graphing calculator enhanced.* W.H. Freeman, New York, 2003.

[10] B.S. Everitt and A. Skrondal. *The Cambridge Dictionary of Statistics.* Cambridge University Press, 2010.

[11] Jerome H Friedman. *The elements of statistical learning: Data mining, inference, and prediction.* springer open, 2017.

[12] MICHAEL SMITH. *STATISTICAL ANALYSIS HANDBOOK.* DRUMLIN SECURITY LTD, Place of publication not identified, 2018.

[13] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[14] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] Sheng Shen, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. Rethinking batch normalization in transformers. *CoRR*, abs/2003.07845, 2020.

[16] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pages 1–9, 2014.

[17] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pages 7–10, 2016.