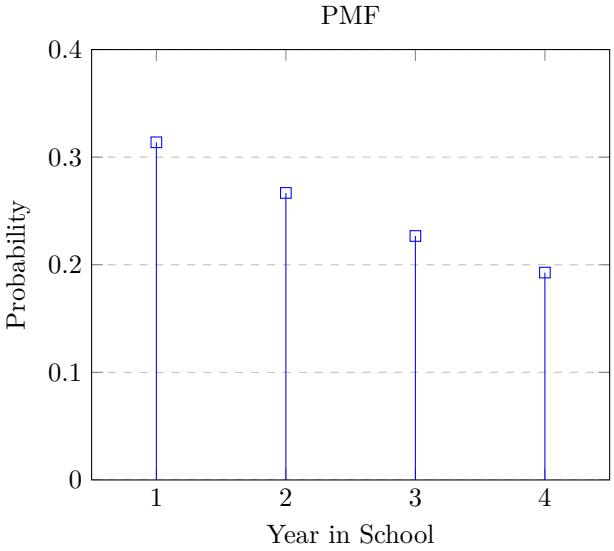
Homework 1

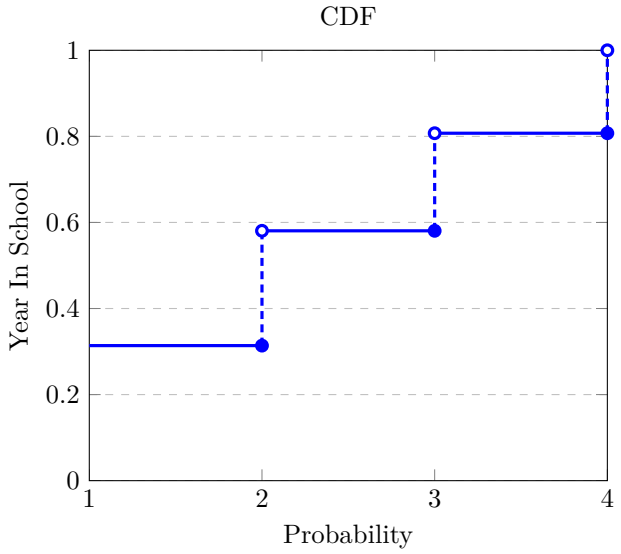
Marshall Grimmett

October 20, 2020

1 Part 1

1.1 a





1.2 b

$$\mu = \sum_{i=1}^{n} X_i P_i$$
$$= [(1 * 1000/3186.625) + (2 * 850/3186.625)$$
$$+ (3 * 722.5/3186.625) + (4 * 614/3186.625)]$$
$$= 2.2981995057466755$$

$$\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} (X_i - \mu)^2$$
$$= (1/3) * 5.162893757923545$$
$$= 1.29072$$

1.3 c

$$\mu = f(\alpha) = \sum_{i=1}^{n} \frac{X_i (1-\alpha)^{i-1}}{\sum_{i=0}^{n} \alpha^i}$$

1.4 d

Mean of University C

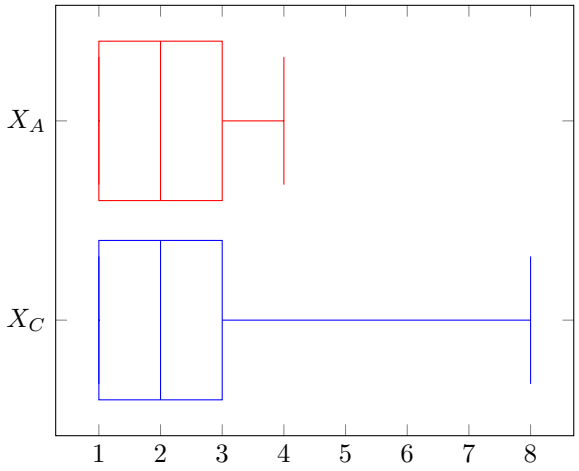
$$\begin{aligned}\mu = (1 * 1000/3376.625) + (2 * 850/3376.625) + (3 * 722.5/3376.625) \\ + (4 * 614/3376.625) + (6 * 40/3376.625) \\ + (7 * 50/3376.625) + (8 * 100/3376.625) \\ = 2.5805352978195684\end{aligned}$$

The mean is not very sensitive to these new data points. The median stays at 2 for both Universities, so it is not at all sensitive to the new data. The mode stays at year 1 for both Universities.

1.5 e

The Median, Q1, and Q3 are all the same for both Universities. Therefore the box plots should be the exact same. The only difference is the range.

University A (X_A) vs. University C (X_C)



2 Part 2

2.1 a

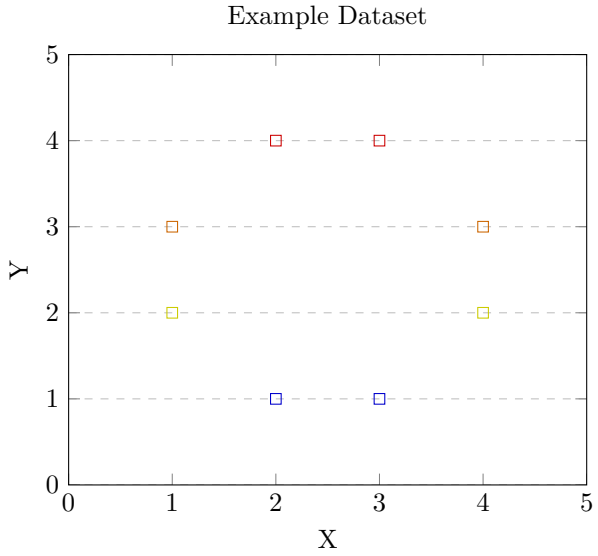
PCA fails when the data appears non-linear, such as a circle or any other shape. Another case would be if the data has no direction with significantly higher variance.

2.2 b

A way to quantify this would be to look at the data from different directions to see if there are any non-linear shapes. We can also look at the principal components to see how much they differ.

2.3 c

In the below example, there is a plot of 8 data points that form a circle. If we were to run PCA on this data and reduce it's dimension, this circle would be lost. Also, we can see that along any new axis, there is hardly any change in variance. So if we projected horizontally it would be the same as projecting vertically.



3 Part 3

3.1 a

See code.

3.2 b

Matrix Product: 0.0 second
Sum of Outer Products: 0.0050048828125 seconds
Sample Covariance: 0.4223923683166504 seconds

The Matrix product was the fastest and it happened so quickly that it recorded 0 seconds. This is because computers can handle matrix multipli-

cation very well and numpy can use smarter algorithms as well.For loops on the other hand, must happen sequentially so they cannot be optimized nearly as much as a single computation. The for loops are also have to perform a lot more computations. The sample covariance was the slowest because it had to use 3 for loops resulting in $O(n^3)$ complexity.

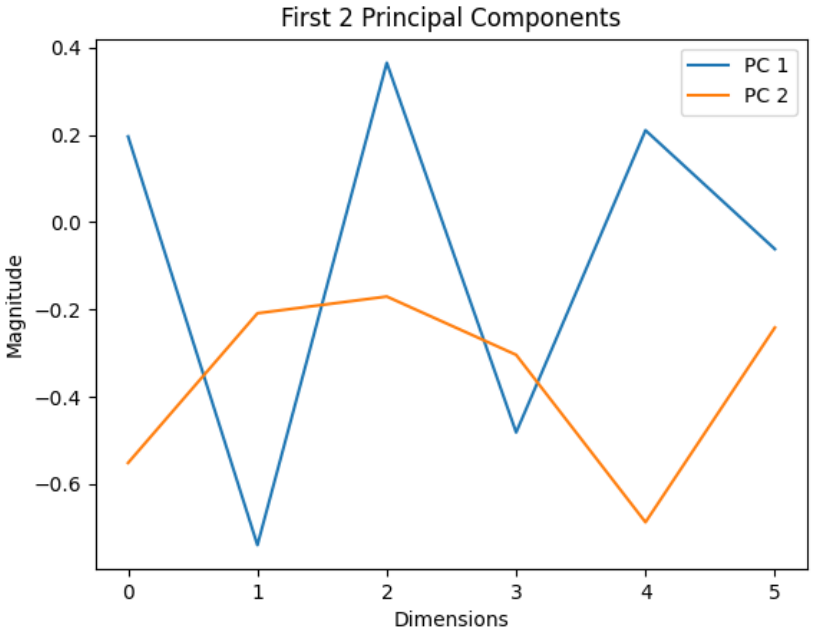
3.3 c

See code.

3.4 d

To compute r, I computed the fraction of total variance cumulatively for each component. Once the threshold has been reached, the function exits with the value r.

3.5 e



3.6 f

The reduced dimension data appears to have very high variance with one large centroid. Smaller clusters can be picked out as well as some vacancies, but

nothing significant. The retained variance is 0.31759. The top 2 principal components accounts for nearly 77% of the variance, while the top 3 account for 97%. While 3 components may be a better fit, 2 is still very sufficient. Also, looking at the plot we can see a high degree of variation in the data. So, I would say this is a good fit.