CS 541 Homework 1

Stephen Szemis

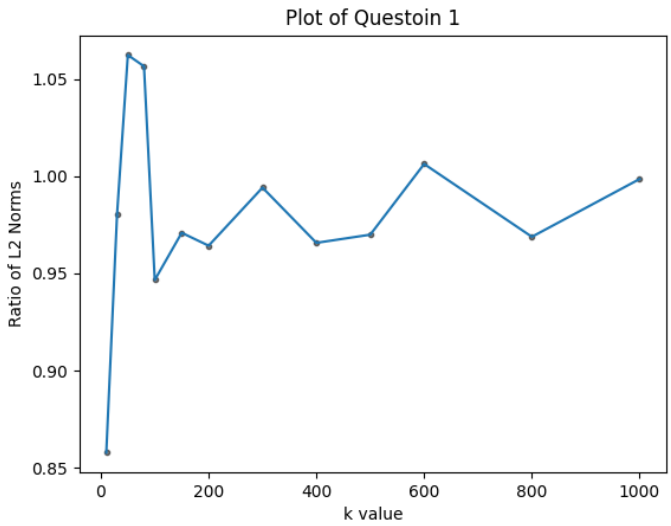
September 27, 2020

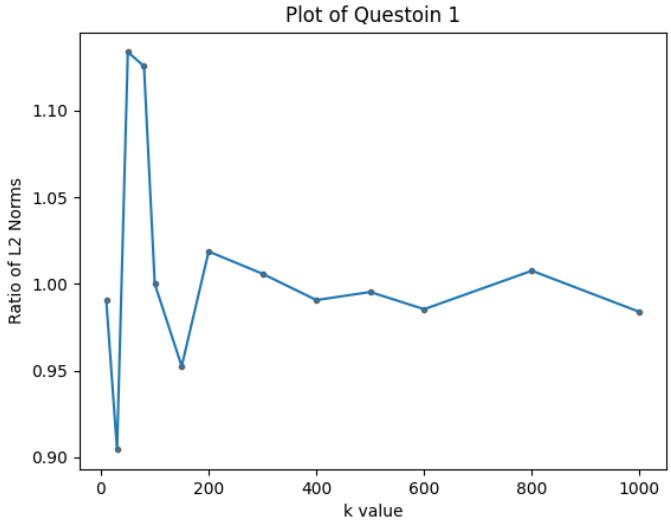
Problem 1: Random Projection for Nearest Neighbor Search
I wrote the following code in order to generate the data. See comments for explanation.

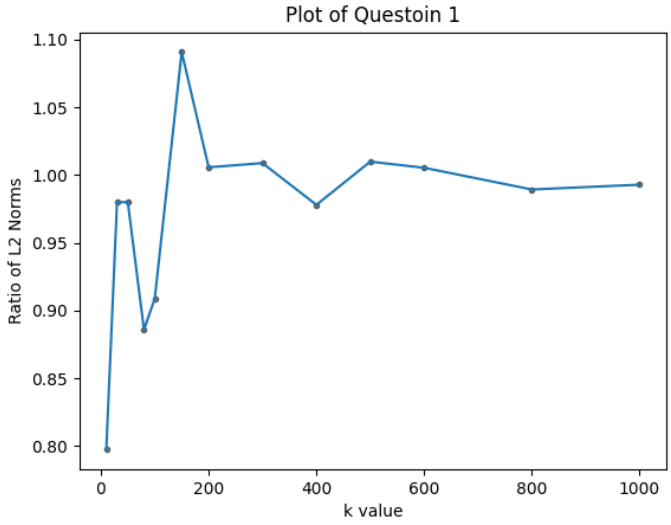
```
import numpy as np
import random as rand
import matplotlib.pyplot as ppl
# I pledge my honor that I have abided by the Stevens honor system.

# Experiments
d = 1000
# All of our k values
k_array = [10, 30, 50, 80, 100, 150, 200, 300, 400, 500, 600, 800, 1000]

# Create an empty array so we can store our L2 norm calculations
Ax_l2 = np.zeros(len(k_array))

# Create a uniform random vector of dimesion d
x = np.random.default_rng().uniform(-100, 100, d)

# Generate our A matrix and calculate the L2 of [Ax]
for i, k in enumerate(k_array):
    A = np.random.default_rng().normal(0, (1/np.sqrt(k)), (k, d))
    Ax_l2[i] = np.linalg.norm(A @ x)

# Calculate the ratios
ratio = Ax_l2 / np.linalg.norm(x)
```

```
# Plot data
ppl.scatter(k_array, ratio, s=np.pi*3, c=(0, 0, 0), alpha=0.5)
ppl.plot(k_array, ratio)
ppl.title('Plot of Questoin 1')
ppl.xlabel('k value')
ppl.ylabel('Ratio of L2 Norms')
ppl.show()
```

The data is shown below, since this data is generated randomly I've include three different runs.







As can be seen in the above graphs, the ratio between $\|\mathbf{x}\|$ and $\|\mathbf{A}\|$ approach 1 sometime around $k = 200$. This means that in general we can say that for some vector with dimension d, we can safely use random projection to lower it's dimension by about 1/5 the size, while still keeping the distance between the data similar.

Problem 2: Reliable Data Annotation

1. The condition can be described like so...

$$Pr(\sum_{i=1}^{n} y_i > 0)$$

2. Chebyshelv's Inequality is written as...

$$Pr(|X - \mathbb{E}[X]| \geq t) \leq \frac{Var[X]}{t^2}$$

Let X be our condition from part 1. We need the expected value and variance of X for Chebyshelv's.

$$\mathbb{E}[X] = \mathbb{E}\left[\sum_{i=1}^{n} y_i\right] = \sum_{i=1}^{n} \mathbb{E}[y_i] = n * (0.6 - 0.4) = 0.2n$$

$$Var[X] = Var\left[\sum_{i=1}^{n} y_i\right] = \sum_{i=1}^{n} Var[y_i] = n * (\mathbb{E}[y_i^2] - \mathbb{E}[y_i]^2) = 0.96n$$

With that information we can now start calculating the inequality.

$$Pr(|X - 0.2n| \geq t) \leq \frac{0.96n}{t^2}$$

Knowing we want a probability of 0.99 and X to be at least 1, we can write...

$$0.2n - t = 1$$
$$\frac{0.96n}{t^2} = 0.01$$

Solving the quadratic for n and t we find.

$$n \approx 2410$$

$$t \approx 481$$

3. We will use the form of the Chernoff bound.

$$Pr(Z \leq (1 - \alpha)\mu n) \leq e^{-\frac{\alpha^2 \mu n}{2}}$$

Our $\mu = 0.6$. Note: Our condition was changed from part 1, since we need to our random variables to be 0, 1 not -1, 1. Instead of looking for

the sum to be greater then zero, we now look for the sum to be greater then half of the total, or $\frac{n}{2}$. Essentially we want the probability that Z is less then half of n to be lower then 1 percent. Written formally:

$$(1-\alpha)0.6n = \frac{n}{2}$$

$$e^{-\frac{\alpha^2 * 0.6n}{2}} = 0.01$$

Solving for n and α we get...

$$\alpha = \frac{1}{6}$$

$$-\frac{\alpha^2 * 0.6n}{2} = ln(0.01)$$

$$n \approx 553$$

4. In conclusion, it is easy to see from our estimates of n that the Chernoff bound gives a much tighter approximation. This makes sense since it makes use of the inherent properties of the 0 ,1 distribution and therefore can get exponentiation on the right hand side of the probability.