DATA 201B: Notes on PCA and KNN

H. C. Hannon

October 2020

1 The Math Behind PCA (at least some of it)

One thing to make clear is that PCA is not really a statistical learning method, as in it makes no assumptions about the data that it is given. In statistical learning, the first assumption is that you have a set of data that has been drawn iid (independent, identically distributed) from some unknown distribution. For PCA, there is no such assumption. Another thing to make clear is that PCA is not feature selection. In feature selection, you try to choose a subset of features from your data that will still represent your data well, meaning it *captures the essence* of the data well. PCA does not choose features, it creates its own set of features that will capture the maximum amount of variance from the data. To show this maximum calculation, and to also give some explanation why we find eigenvectors and eigenvalues of the covariance matrix, we will walk through finding the first principle component. First, recall the covariance of two sets of data, given $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$ and $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ then:

$$\text{Cov}(\mathbf{x}, \mathbf{y}) = \frac{1}{n}\sum_{i=1}^{n} x_i y_i - \left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)\left(\frac{1}{n}\sum_{i=1}^{n} y_i\right)$$

Notice that the right part of the expression is just the sample mean of $\mathbf{x}$ and $\mathbf{y}$, so if we normalize the data these will both be equal to 0, so the Covariance just becomes the first term. Now, for the covariance matrix, which is just the covariance of every pair of columns, we can represent the formula in the following way. Let $\mathbf{x} \in \mathbb{R}^{m \times n}$, and let $\mathbf{x}(i)$ represent the i^{th} row of that matrix. This means we have m total observations with each observation consisting of n values. Also, assume that each column has a sample mean of zero, then the covariance matrix is as follows:

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}(i)\mathbf{x}(i)^T$$

Now, we search of a one-dimensional projection for the data, as in, we search for a line which will have all data projected onto it. Let $\mathbf{w}$ be a unit vector, then given a row of data the projection is equal to:

$$\langle \mathbf{x}(i), \mathbf{w} \rangle \mathbf{w}$$

We would like the difference between our true points and the projected points to be small so we wish to minimize:

$$\|\mathbf{x}(i) - \langle \mathbf{x}(i), \mathbf{w} \rangle \mathbf{w}\|_2 = \text{some math happens} = \langle \mathbf{x}(i), \mathbf{x}(i) \rangle - \langle \mathbf{w}, \mathbf{x}(i) \rangle^2$$

If we sum over i we obtain the following thing to minimize:

$$\sum_{i=1}^{n} \|\mathbf{x}(i)\|^2 - \sum_{i=1}^{n} \langle \mathbf{w}, \mathbf{x}(i) \rangle^2$$

The first term is independent of $\mathbf{w}$, so we drop it, and we are left to maximize:

$$\langle \mathbf{w}, \mathbf{x}(i) \rangle^2$$

We maximize this because minimizing a negative thing means we should maximize the positive version of it. Taking the sum and adding $\frac{1}{m}$ to the front, we see it is the mean of the squares, and using a nice identity we see:

$$\frac{1}{m}\sum_{i=1}^{m}\langle \mathbf{w}, \mathbf{x}(i)\rangle^2 = \left(\frac{1}{m}\sum_{i=1}^{m}\langle \mathbf{w}, \mathbf{x}(i)\rangle\right)^2 + \mathrm{Var}(\langle \mathbf{w}, \mathbf{x}(i)\rangle)$$

The first term is the square of the mean, but since we normalized the data, this will be zero, so really we are left with the variance of these projections! So, finding the best $\mathbf{w}$ really comes down to maximizing the variance! In general, we want to maximize the following quantity, with a restriction that $\|\mathbf{w}\| = 1$:

$$\mathbf{w}^T \Sigma \mathbf{w}$$

We can form a lagrangian problem with this, and solve it, leaving us with the result:

$$\Sigma \mathbf{w} = \lambda \mathbf{w}$$

So, $\mathbf{w}$ is an eignevector, and λ is a corresponding eigenvalue! I skipped a lot of steps in this calculation, but the idea to get is that we are finding orthogonal vectors (at right angles to each other), that maximize the variance of the projections. PCA is nice, but note that it does not work miracles, sometimes your data is varied enough that finding these principle components just will not work.

2 K-Nearest Neighbors

A classic example of statistical learning with not much statistics behind it, just some naive assumptions. The assumption is that new datapoints must belong to the class that is most common of its closest neighbors. It is kind of a greedy algorithm in that it works with the mode, and it does not matter if there are three closest points with class 1 and four closest points with class 2, the four with class 2 will claim the new point. The problem of KNN is that the number for k is not given, and the questions of whether there is an optimal one, and if it is unique do not have good answers. We will have to do that with cross validation!