CS 541-A Homework 2

Sesha Vadlamudi

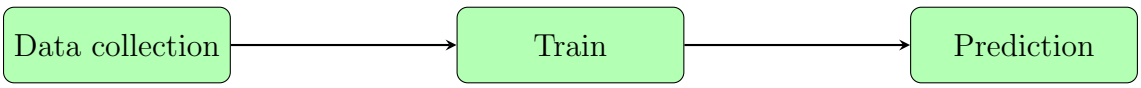
October 17, 2020

1 Lecture 1 - September 1^{st}

Overview of course and subject.
Syllabus:

- Overview of Probability
- Overview of Linear Algebra
- Unsupervised Learning
    1. Random Projection
    2. SVD
    3. PCA
    4. K-means Clustering
    5. Subspace clustering
    6. Dictionary Learning (Image/ Signal Processing)
- Low rank matrix estimation with application to recommender systems
- Computational social science
- Linear Regression, Classification

1.1 Big Picture of AI/Machine Learning



1.1.1 Data Collection

- Passive : First collect all the data and then only feed it to train the model.
- Active : You make use of a rough model to filter the raw data and detect the very informative characteristics and then this is used to label the new data to retrain the model.
- Batch Collect all the data at one point.
- Collect data points one after the other.

Sometimes batch collection may not be useful or compatible. For example, stock price prediction where the data points are collected sequentially.

1.1.2 Training

- We care about time complexity of the model.

1.1.3 Prediction

- Accuracy on unseen data.

1.2 Linear Algebra

- d-dimensional column vector x is a set of d numbers.
- almost all the data is in the form of vectors.

1.2.1 Operation

- $x, y \in R^d$ column vectors; $a, b \in R$
- $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$
- $x^T = \begin{bmatrix} x_1 & x_2 & \cdots & x_d \end{bmatrix}$
- $ax = \begin{bmatrix} ax_1 \\ ax_2 \\ \vdots \\ ax_d \end{bmatrix}$
- $x + y = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_d + y_d \end{bmatrix}$

- $\langle x, y \rangle = \sum_{i=1}^{d} x_i \cdot y_i \in R$ - This is also know as inner product or dot product

1.2.2 Vector Norms

- L_1 $Norm$ $\|x\|_1 = \sum_{i=1}^{d} |x_i|$
- L_2 $Norm$ also know as $EuclideanNorm$ $\|x\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$
- L_∞ $Norm$ also known as $MaximumNorm$ $\|x\|_\infty = \max_{1 \le i \le d} |x_i|$

1.3 Matrices

- Vector : A set of numbers
- Matrix : set of vectors

1.3.1 Properties

- $X \in R^{d \times n}$
- $aX, \ \forall a \in R$
- $aX + bY | X, Y$ have same size
- Multiplication of matrices: $X \in R^{d \times n}$ and $Y \in R^{p \times m}$
  $X \cdot Y$ is possible only when $n = p$
  $XY \in R^{d \times m}$
- $x, y \in R^d$; $x^T \cdot y \in R$; $x \cdot y^T \in R^{d \times d}$
- **Transpose**: The transpose of a matrix is a new matrix whose rows are the columns of the original. The transpose of matrix X is represented as X^T
- **Symmetric Matrix**: A matrix is said to be symmetric when $X = X^T$

1.4 Overview of Probability

- **Probability**: Measure of likelihood that an event will occur.
- **Random Variable**: A random variable is a function that maps each of the experiments outcomes to a value. It is usually represented as X.
- X is discrete.
  - X takes on value from a countable set.
  - In discrete case a probability mass function is defined.
  - $p(x) = P(X = x)$

- X is continuous.
  - X could take on any value in the distribution. Every number has a finite probability of occurring.
  - Continuous random variables admit Probability Density Functions
  - $p(X \leq x) = \int_{-\infty}^{x} p(z)\, dz$
  - Uniform Distribution
  - Normal/ Gaussian Distribution
- **Expected Value**:
  - Discrete : $E[X] = \sum xp(x)$
  - Continuous : $E[X] = \int xp(x)dx$
  - average of multiple outcomes

1.5 Markov's Inequality

Theorem: If $x > 0, P(X \geq t) \leq \frac{E[X]}{t}\ \forall t > 0$

2 Lecture 2 - September 8^{th}

2.1 Markov's inequality

2.1.1 Proof of Correctness

$$t = t' \cdot E[X] \mid t' > 0, E[X] > 0$$
$$\Rightarrow P(X \geq t'.E[X]) \leq \frac{E[X]}{t'E[X]} = \frac{1}{t'} \forall t' > 0$$
$$\Rightarrow E[X] = \int_0^{+\infty} xf(x)dx$$
$$\Rightarrow E[X] = \int_0^{t} xf(x)dx + \int_t^{+\infty} xf(x)dx$$

The first term is always greater than or equal to 0. The second term however is $\geq \int_t^{+\infty} tf(x)dx = t\int_t^{+\infty} f(x)dx$. Therefore,

$$\Rightarrow E[X]\ \geq\ 0 + t\int_t^{+\infty} f(x)dx =\ t.P(X \geq t)$$
$$\therefore\ E[X]\ \geq\ t.P(X \geq t)$$

2.1.2 Proof of Tightness

An inequality is said to be tight when for a specific value the inequality becomes equality. Let X be a random variable that can take values either 0 or 1. $P(X = 1) = \frac{1}{t}$ and therefore $P(X = 0) = 1 - P(X = 1) = 1 - \frac{1}{t}$

Markov's inequality states $P(X \geq tE[X]) \leq \frac{1}{t}$

$E[X] = \sum_{x=0}^{1} x_i \cdot P(X = x_i) = \frac{1}{t}$

$LHS = P(X \geq t.\frac{1}{t}) = P(X \geq 1) = P(X = 1) = \frac{1}{t} = RHS$

2.2 Chebyshev's Inequality

Chebyshev's inequality gives a stronger result that Markov's inequality. This is because we have more information on the random variable. In Markov's we only have the expectation of the random variable whereas in Chebyshelv's we have variance of the random variable along with it's expectation.

Theorem: Let x be a random variable. We have, $P(|X - E[X]| \geq t) \leq \frac{Var[X]}{t^2}\ \forall t > 0$

2.3 Application : Nearest Neighbor Search

- Query $q \in R^d$
- $x_1, x_2, \ldots, x_n \in R^d$
- Which x_i is closest to q according to L_2 norm
- Goal : find $i = arg\min_{1 \leq i \leq n} \|x_1 - q\|_2$

2.3.1 Random Projection

Independent of data, very fast

- Determine the new dimension $k < d$
- Generate A |each $a_{ij} \overset{i.i.d}{\sim} N(0, 1)$
  $x_i \mapsto \widetilde{x_i} = \frac{1}{\sqrt{k}} A x_i, q \mapsto \widetilde{q} = \frac{1}{\sqrt{k}} Aq$
- Hope $\|\widetilde{x_i} - \widetilde{q}\|_2 \approx \|x_i - q\|_2$ - With random projection in new space, distance is preserved or order is preserved.

2.4 Johnson-Lindenstrauss Lemma

Theorem 2.1 *For any $\epsilon \in (0,1)$ and any integer $d > 0$, let $k \geq \frac{24}{3\epsilon^2 - 2\epsilon^3} logn$ Then for any $x_i, x_j \in \{x_1, x_2, \cdots, x_n\}$ with high probability*

$$(1-\epsilon)\|a_i - x_j\|_2^2 \ \leq \ \|f(x_i) - f(x_j)\|_2^2 \ \leq \ (1+\epsilon)\|a_i - x_j\|_2^2$$

$$where\ f(x) = \frac{1}{\sqrt{k}} Ax, A \in R^{k \times d}$$

and each of A is i.i.d $N(0,1)$

The lemma states that a set of points in high dimensional space can be embedded in low dimensional space with their distances between the points nearly preserved.

2.5 Chernoff Bound

It gives exponentially decreasing bounds on tail distributions of sums of independent random variables. It is a sharper boundthan the first know Markov's inequality or Chebysev's inequality.

Theorem 2.2 *Let $Z_1, Z_2, Z_3, \cdots, Z_n$ be n independent random variables that take value in $\{0,1\}$. Let $Z = \sum_{i=1}^{n} Z_i$.For each Z_i, suppose that $Pr(Z_i = 1) \leq \eta$. Then for any $\alpha \in [0,1]$*

$$Pr(Z \geq (1+\alpha)\eta n) \leq e^{-\frac{\alpha^2 \eta n}{3}}$$

When $Pr(Z_i = 1) \geq \eta$, for any $\alpha \in [0,1]$

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

For Chernoff bound we take into account the Moment Generating Function - MGF.

3 Lecture 3 - September 15^{th}

3.1 Recap

We briefly reviewed Markov's inequality, Chebyshev's inequality, and showed with an example that the Chebyshev's inequality is more sharper than Markov's inequality. Also stated in this context with a brief note that Hoeffding's inequality gives more sharper bound than both the Markov's inequality and the Chebyshev's inequality. This is because Hoeffding's inequality makes use of moment generating functions. This essentially means that Hoeffdings inequality requires complete knowledge of the distribution. In the case of Markov's inequality we only need to know the expectation; whereas in the case of Chebyshev we need to know not only the expectation but also the variance.

3.2 Random Projection

The problem in random projection is that given query point q and n data points, $x_1, x_2, \cdots, x_n$, in a database; each of them is a d dimensional vector, find a data point x_i that is closest to the query point q.

This can be done in a straight forward way by computing the distance of q form each of the data points $x_1, x_2, \cdots, x_n$ and take that data point whose ℓ_2 distance is the smallest. Note that this can be done in $O(nd)$ time. This is because the computation of the distance between two d– dimensional vectors can be done in $O(d)$ computation and there are n such computations. We would like to reduce this computation: one approach is by reducing n (by applying probably hashing techniques) and the second approach is by reducing the dimension, d, of each of the vectors. This can be achieved by random projection.

So as to achieve this by random projection, we first need to decide on k, the new dimension to which we would like to reduce. This new dimension, k, is a function of the number of data points n in the database. Exact function can be had from the Johnson Lindenstrauss lemma. After deciding on k then construct a $k \times d$ random matrix A, each of whose entry is random element taken from the Gaussian distribution $N(0,1)$ with zero mean and unit variance. Now the new representation of the data point x_i is $\frac{1}{\sqrt{k}}Ax_i$. Note that the dimension of the new representation of the data point is k. If the new representation of x_i is $\tilde{x_i}$ then we can say that $\parallel x_i – x_j \parallel_2$ is approximately equal to the $\parallel \tilde{x_i} – \tilde{x_j} \parallel_2$. This is because we have shown earlier that $E(\parallel \tilde{x_i} – \tilde{x_j} \parallel_2^2)$ equal to $\parallel x_i – x_j \parallel_2^2$. Also by the chebyshev's in equality we know that actual value is around the expected value. So the result that the distances are approximately preserved is proved.

Note: In random projection, the matrix that we generate is independent of the data. PCA is another technique for dimensionality reduction. In contrast to random projection, PCA is dependent of data.

3.3 Subspace

Let V be a vector space of dimension d and let $\vec{x}$ and $\vec{y}$ be two vectors of V. Then the set of all vectors of the form $a\vec{x} + b\vec{y}$, where a and b are real numbers, is a subspace of V. That is the set of all linear combinations of the vectors $\vec{x}$ and $\vec{y}$ is a subspace of V. That is a subspace is closed under the linear combination operation.

3.3.1 Examples

Example 1: Let V be equal to $\vec{0}$, where $\vec{0}$ is a vector of d dimension consisting of all 0 elements, is a subspace. This is because, it is closed under the linear combination operation.

Example 2: Let V be equal to $\vec{0}, \vec{1}, a\vec{1}$, where $\vec{0}$, $\vec{1}$, and $a\vec{1}$ are d dimensional vectors. Also all the elements of vector $\vec{1}$ and $a\vec{1}$ are 1's and a's respectively. Now V is a sub space. This is because it is closed under the linear combination operation.

Example 3: V is equal to $\vec{0}, e1, e2, a * e1 + b * e2$, where $ei, i = 1, 2$ is a d dimensional vector whose i^{th} element is 1 and all its other elements are 0, is a subspace. This is because it is closed under the linear combination operation.

3.3.2 Null Space

Let M be an $n \times d$ matrix. Then the null space of M is the set of all vectors $\vec{v}$ such that $M\vec{v}$ equal to $\vec{0}$. So, for the null space, first determine all kinds of possibilities of solution vectors $\vec{v}$ and then the linear combination of all these $\vec{v}$'s is the null space of M.

3.3.3 Span / Linear combination

Suppose we have vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n$. Then the span of these vectors is the linear combination of these vectors. That is, $Span(\vec{v}_1, \vec{v}_2, \cdots v_n) = a_1\vec{v}_+ a_2\vec{v}_+ \cdots a_n\vec{v}$, for all the real values of $a_1, a_2, \cdots, a_n$.

3.3.4 Examples

Example 1: Suppose $\vec{v}_1 = 1, 0$. Then the span of $\vec{v}_1$ is the x axis or the real line.

Example 2: Suppose $\vec{v}_1$ as in the previous example and let $\vec{v}_2$ is $1, 1$. Then the span of $\vec{v}_1$ and $\vec{v}_2$ is the $x - y$ plane. That is the entire 2 dimensional space. This concept of span given in the examples can be generalized to higher dimensional examples as well.

3.3.5 Linear Independence:

Let $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n$ be n d dimensional vectors. Then the vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n$ are linearly dependent if there exists real numbers $a_1, a_2, \cdots, a_n$ such that the ℓ_2 norm of the n dimensional vector consisting of the elements $a_1, a_2, \cdots, a_n$ is non zero and the linear combination $a_1\vec{v}_1 + a_2\vec{v}_2 + \cdots a_n\vec{v}_n = 0$.

If the only solution of $a_1, a_2, \cdots, a_n$ in the vector equation $a_1\vec{v}_1 + a_2\vec{v}_2 + \cdots + a_n\vec{v}_n$ is $0, \cdots, 0)$ then the vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_n$ are said to be linearly independent.

3.3.6 Basis of subspace V:

Basis is a set of vectors in V. Let $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_r$ be a set of vectors in V. Also let for any vector $\vec{x}$ in V there is a set of real numbers $a_1, a_2, \cdots, a_r$, such that the linear combination $a_1\vec{v}_1 + a_2\vec{v}_2 + \cdots + a_r\vec{v}_r = x$ and $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_r$ are linearly independent. Then the vectors $\vec{v}_1, \vec{v}_2, \cdots, \vec{v}_r$ are said to form a basis of the subspace V. Note that basis is not unique.

Example: Consider the two dimensional real space R^2, i.e., $\vec{v} \in R^2 \forall \vec{v}$. Then $\vec{v}_1$ is $1, 0$ and $\vec{v}_2$ is $0, 1$ forms a basis. This is because any point (x_1, x_2) in the two dimensional space can be written as a linear combination $x_1.\vec{v}_1 + x_2.\vec{v}_2$ of $\vec{v}_1$ and $\vec{v}_2$. Also v_1 and v_2 are linearly independent.

Second basis for the above example: Let $\vec{v}_1$ be $1, 0$ and $\vec{v}_2$ be $1, 1$. Then $\vec{v}_1$ and $\vec{v}_2$ form a basis. This is because any point $(x1, x2)$ can be written as $(x1 - x2).\vec{v}_1 + x2.\vec{v}_2 = (x1, x2)$. Also, $\vec{v}_1$ and $\vec{v}_2$ are linearly independent.

3.3.7 Column Space:

Consider a matrix $M = [m_1, m_2, \cdots, m_n]$, where m_i denotes the $i^{th}, i = 1, 2, \cdots, n$, column of the matrix, and each column vector is a d dimensional vector. Note that M is a $d \times n$ matrix.

The column vectors $m_1, m_2, \cdots, m_n$ of the matrix M form a subspace of d dimensional space. This subspace must have a basis. This basis is called the column space of M. In fact, the span of these column vectors of the matrix M is the column space of M.

3.3.8 Row Space:

Likewise, we can define the row space of M. To define the row space of M, let $m_1, m_2, \cdots, m_d$ denote the rows of M, where each m_i, I $= 1, 2, \cdots, d$ is an n dimensional vector. These $m_1, m_2, \cdots, m_d$ vectors form a subspace of n dimensional space. This subspace must have a basis. This basis is called the row space of M. In fact the span of these row vectors of M is the row space of M.

3.3.9 Rank:

Let V be a subspace of d dimensional space. Also let for any $\vec{x} \in V$ there exists r linearly independent vectors such that $\vec{x}$ can be written as a linear combination of these r vectors then the rank of the subspace V is r. This rank is also known as the intrinsic dimension of the subspace V. r refers to the number vectors in a basis of V.

Example: Consider a three dimensional real space R^3.
Consider subspace $V = (1,0,0), (0,1,0), a.(1,0,0) + b.(0,1,0)$, for all possible values $a, b \in R$. That is V is the $x - y$ plane. Since every point in V is a linear combination of $(1,0,0)$ and $(0,1,0)$, it follows that the rank r, the intrinsic dimension of V, is 2. Note that the ambient dimension is 3.

3.3.10 Direct Sum:

Let V_1 and V_2 be subspaces of a vector space V. Then the direct sum of V_1 and V_2 is a linear combination of the form $a_1\vec{v}_1 + a_2\vec{v}_2$, where a_1 and a_2 are real numbers and $\vec{v}_1$ and $\vec{v}_2$ are vectors of the subspaces V_1 and V_2 respectively.

Example: Let the subspace V_1 be the x axis and the subspace V_2 be the y axis of two dimensional space. Direct sum of the subspaces V_1 and V_2 gives us the two dimensional $x-y$ plane.

3.3.11 Union:

Union of two subspaces V_1 and V_2 is defined as the union of the set of all vectors from V_1 and the set of all vectors from V_2. Note that it is just the union of the two sets of vectors and not the linear combinations of the sets of vectors from the two subspaces.

Example: Let the two subspaces V_1 and V_2 be the x axis and the y axis respectively of a the two dimensional subspace. Then $V_1 \cup V_2$ is just the x axis and the y axis only and not the $x - y$ plane as in the case of the direct sum and the linear combinations of vectors.

3.4 Principal Component Analysis

Goal is to reduce the dimension. We have already seen that a way of reducing the dimension is by random projection. Principal Component Analysis is yet another technique to reduce

the dimension.

Trivial Example: Consider a data set consisting of two dimensional vectors of the form $(x, y), x, y \in R$ and $x = y$. Here, you see that the second dimension of the vector is not providing any new information. So by just getting rid of the second dimension, we are not losing any information. So, by removing the second dimension from each of the data points we are implicitly projecting the data points on to the x axis. Note that the relative distances between any pair of data points is preserved in the projected space. That is if two points, say x_1 and x_2, are closer to each other compared to, say x_1 and x_3, in the original space they remain so even in the projected space.

Noisy Data: Consider a set of two dimensional vectors of the form $(x, y), x, y \in R$ and x and y differ by a very small number (noise). Here first we try to remove the noise. Once we remove the noise, we get to the ideal case, where $x = y$. then we project as in the previous example. That is, we are trying here to arrive at a low rank approximation of the given data.

Let M be a matrix with d dimensional column vectors $m_1, m_2, \cdots, m_n$. We want to find a matrix X such that the $\| X - M \|_F$ is minimum. Also the rank of $X \leq r$, where $r \leq d$.

Frobenious norm of $M = \sqrt{\sum_{1 \leq i \leq d, 1 \leq j \leq n} m_{ij}^2}$.

If we can find a matrix X such that the $\| X - M \|_F$ is minimum, it means that only r rows of X is sufficient to represent all the rows of M. That is any feature can be expressed as a linear combination of only those r features. Note that each row represents a feature and each column represents a data point.

subsectionRecommender Systems: In Recommender Systems matrix M is a rating matrix. In this matrix each row corresponds to ratings given by a particular user; columns corresponds to the ratings of an item given by the users. In this matrix most of the entries are not filled. That is this rating matrix is highly sparse. Idea here is to come up with a way of filling out the missing entries of the rating matrix. If we can somehow fill out the missing entries then based on the entries of the row corresponding to the user, we can recommend an item to that user.

So as to predict the entries of missing values of the matrix, we would like to come up with another matrix X which is close (in the sense of the error considering only the observed entries of the matrix and not taking the missing values into account) to the rating matrix M. As an optimization problem, this is as follows: Minimize the Frobenious norm of $\| (X - M)_\Omega \|_F$, where Ω denotes the index set of the observed entries of the rating matrix M, subject to the constraint that the rank of $X \leq r$, where $r < d$.

The difference between finding X in recommender systems and that of finding in PCA is that in PCA we have the matrix M with all its entries observed; whereas in the case of recommender systems we have the matrix M with only some of the entries observed.

Note: The rank constraint is non convex. Because of this only the PCA problem, wherein the Frobenious norm is used, has a closed form solution. However, recommender system problem or any other problem with other norms does not have a closed form solution.

PCA solution can be obtained by resorting to singular value decomposition of M. Let $M = USV^t$ be the singular value decomposition of M; where U is a $d \times d$ ortho-normal matrix, S is a $d \times d$ diagonal matrix whose elements are in decreasing order, and V is an $n \times d$ ortho-normal matrix. That is $u^t u = I, V^t V = I$. Computational complexity of SVD

is minimum of d^{2n}, n^{2d}.

Therefore the closed form solution X to PCA problem is obtained by taking only the first r columns of U, first $r \times r$ diagonal matrix of S, and the first r rows of V^t, and carrying out matrix multiplication of these new matrices, say U', S', and V'^t, we get to the closed form solution X to the PCA problem. In the PCA problem, the second step is the projection. Here, one may take $U'^t.U'.S'.V'^t$ which is equal to $S'.V'^t$ or just V'^t as a matrix for the projection. Choice depends on a particular application.

Now let us see how the projection gives a new representation of the data points. X is considered to be a set of data points without noise. So then $X = U'.S'.V'^t = [u_1, u_2, \cdots, u_r].C$, where $C = S'.V'^t$. The dimension of C is then $r \times n$. So $X = [u_1, u_2, \cdots, u_r].C$, where $C = \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ c_{r1} & c_{r2} & \cdots & c_{rn} \end{bmatrix}$ So the first column of X, say $\vec{x}_1$, is then the linear combination $c_{11}\vec{u}_1 + c_{21}\vec{u}_2 + \cdots + c_{r1}\vec{u}_r$. Similarly the second column of X, say $\vec{x}_2$, is then the linear combination of $c_{12}\vec{u}_1 + c_{22}\vec{u}_2 + \cdots + c_{r2}\vec{u}_r$. Also the other columns of X. Note that in all the representations of the columns of X, $\vec{u}_1, \vec{u}_2, \cdots, \vec{u}_r$ are same. The only difference in their representations is the coefficients in the linear combinations. So, assuming that $\vec{u}_1, \vec{u}_2, \cdots, \vec{u}_r$ are fixed, we can have new representation of these columns of X (i.e. the data points) using only the C matrix. The dimension of C matrix is only $r \times n$ as opposed to $d \times n$ as in the original case. $U'^t.M = U'^t.U.S.M$ is an $r \times n$ matrix. This follows because U is an ortho-normal matrix.

4 Lecture 4 - September 22^{nd}

4.1 Recap the PCA

1. Goal of PCA is to reduce the dimension. Of course, this goal can also be achieved using random projection. The main difference between these two methods is that the random projection is data independent; whereas PCA is data dependent. That is we come up with a random projection matrix based on only the original dimension d and the number of data points n, and not the actual values of the elements in the data points. This means the same random projection is applicable to another set of data points as long as the dimension and the number of points is same as that of the first dataset.

2. As mentioned, this is not the case with the PCA. In the case of PCA, the projection matrix that we compute is a function of the coordinate values of the data points.

3. let X be a $d \times n$ data matrix, where the columns $x_1, x_2, \cdots, x_n$ of X are the n data points and d denotes the number of features.

4. In PCA, we first compute SVD of X. Let the SVD of $X = U.S.V^t$; where U is a $d \times d$ orthonormal matrix, S is a $d \times d$ diagonal matrix and V^t is $d \times n$ ortho-normal matrix. That is $U^tU = I_{d\times d}, V^TV = I_{d\times d}$.

5. Diagonal elements of S are in the decreasing order of magnitude. That is $s_{11} \geq s_{22} \geq \cdots \geq s_{dd}$.

6. Then the PCA matrix A is the transpose of the first k columns of U. That is A^t is equal to the first k columns of U. Then A is the transpose of A^t. Note that the matrix A is $k \times d$. Also the product AX of the PCA matrix A and the data matrix X is the best representation of the data matrix in the k dimensional subspace. That is AX is closest to X than any MX to X, where M is any at most rank k matrix.

7. Motivation to PCA is discussed through an example. Basically PCA tries to remove the noise and gives you the inherently low rank approximation of the original data matrix.

8. Motivation from theoretical perspective is that we don't want over fitting. That is by approximating your data using PCA you prevent over fitting.

9. Learning theory gives you the necessary and sufficient conditions to prevent over fitting.

4.2 Recommender System

Consider a rating matrix. In this matrix, each row corresponds to a user and each column corresponds to an item. Let us say we have n users and p items. Then the matrix is an $n \times p$ matrix; where n and p are very large. Though there are p items, each user may end up buying a fraction of these items and may not give feedback for each of the items the user bought. So that the rating matrix will in general be highly sparse. Our goal is ambitious and would like to fill out all the remaining entries of the rating matrix. If we can do this then recommendation is straight forward. That is, for each user, sort all the ratings in the corresponding row and recommend the item with the highest rating.

To solve this problem, we start with a popular model, known as, Collaborative Filtering. Let Z denote our observed data. Then Z is an $n \times p$ large matrix and it is highly sparse. Ω is the index set of the observed data. X is the variable matrix which consists of all the entries filled. X can be thought of as the rating matrix in which all the entries are filled. The problem then is then

$$min_X \parallel (Z\text{–}X)_\Omega \parallel_F^2 \text{ such that } rank(X) \leq r$$

X may be thought of as a true preference matrix. Because of the group structure present in the column data as well as in the row data, we expect X to have low rank.

This problem is different from that of PCA. Here minimization is restricted to only the observed entries; whereas in the PCA minimization is across all the entries of the data matrix. So, the solution of PCA may not be the solution to the recommender system problem.

5 Lecture 5 - September 29^{th}

In this lecture the basics of Calculus, MLE and recommender systems were discussed. -PCA cannot find group structure of the data.

5.1 Probabilistic Model

A probabilistic model is a model that uses probability theory to model the uncertainty in the data. It is a model that provides different outcomes with different probabilities in the data. It includes elements of randomness.

- For $(i,j) \in \omega Z_{i,j} = \begin{cases} +1 & \text{with probability} f(X_{i,j)} \\ -1 & \text{with probability} 1 - f(X_{i,j}) \end{cases}$
- $f(X_{i,j}) = \frac{1}{1+e^{-X_{i,j}}}$ is a logistic function. It is also known as the sigmoid function.

5.2 Maximum Likelihood Estimation

This is a method of estimating the parameters of a probability distribution by maximizing a likelihood function, so that under the assumed statistical model the observed data is most probable.

$$\theta_{MLE} = arg\max_{\theta} P(D|\theta) = arg\max_{\theta} P(D|\theta)$$

$$= arg\max_{\theta} log \prod_{i} P(x_i|\theta) = arg\max_{\theta} \sum_{i} logP(x_i|\theta)$$

6 Lecture 6 - October 6^{th}

6.1 Logistic Regression

- $P(Y = +) = p$
- $P(Y = -) = 1 - p$
- IID assumption : $Y_1, Y_2, \cdots, Y_n$ are independent and identically generated
- Suppose let S be a sequence of observations $+, -, +, -, -, +, +, \cdots +$ be $Y_1, \cdots, Y_n$, then the $P(S) = \prod_{i=1}^{n} P(Y_i)$
- MLE: $\underset{P\in[0,1]}{max} P(S) \Rightarrow P^* \Leftrightarrow \underset{P\in[0,1]}{min} - logP(S) \Leftrightarrow \underset{P\in[0,1]}{min} \sum_{i=1}^{n} logP(Y_i)$
- We can either maximize $\underset{P\in[0,1]}{max} P(S)$ or minimize $\underset{P\in[0,1]}{min} \sum_{i=1}^{n} logP(Y_i)$

6.2 Gradient Descent

1. We calculate the slope of our model/graph by caluclating the derivative.
2. To start with, we care more about the sign of the derivative than the magnitude.
3. We pick a random initial value to begin with.
4. We calculate the derivative or partial derivatives according to our case. (Partial derivative if we have many features.)

5. Calculate the step sizes for each parameter: gradient*learning rate
6. Calculate the new parameters = old parameters - gradient*learning rate
7. Repeat steps 4-6 until the gradient is almost 0

- learning rate is represented as η
- gradient is given by $f'(w_i)$
- step size is calculated as $\eta \cdot f'(w_i)$
- So each iteration looks like this:

$$w_{i+1} = w_i - \eta \cdot f'(w_i)$$

- The complexity of Gradient Descent is $O(Tnd)$ where
  - T is the number of iterations
  - n is the number of observation
  - d is the number of dimensions
- Different functions have different convergence rates.
- Gradient Descent algorithm stops when $w_{i+1} = w_i$ and $w_{i+2} = w_i$

6.2.1 Learning Rate η

A high learning rate may overshoot the minima. For learning rates that are too high, the loss may increase and bounce around and may even cause to diverge from the minima. A very low learning rate may take too long to converge and sometimes may even get stuck at the local minima. For learning rates that are too low, the loss may decrease at a very slow pace. In an optimal learning rate range, there is a sharp drop in the loss and therefore the slope is of main interest here.

6.2.2 Backdrops of Gradient Descent

- As the algorithm approaches minimum, the steps taken are often too noisy and may cause gradient descent to oscillate in other directions. Therefore it may take long to converge at the minima.
- There are often cases where the gradient descent may not reach global minima and may get stuck at one of the local minima or a stationary point (neither minima nor maxima).
- If you start at a peak where the gradient is 0, even in this case you may never reach the global minima.
- Approaching the global minima often depends on the initial value.

Acknowledgement

I would like to thank my father, V. Ch. Venkaiah, who has not only given me guidance in understanding the subject but also for being there as the biggest support. The conversations and discussions I had with him gave me the ability to think independently and outside of the box.