Mathematics of Big Data I

Lecture 4: PCA, SVD, SVM

September 26th, 2016

1 Principal Component Analysis

Consider a dataset $\mathcal{D}$ which lies approximately on a linear subspace. As an example of 'dimensionality reduction', we want to find this subspace so we can project $\mathcal{D}$ onto a lower dimensional dataset that accounts for this variance, $\tilde{\mathcal{D}}$. There is a detailed proof in Murphy, but the result is as follows:

1. We calculate the mean of our dataset
$$\mu = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i.$$
2. Standardize our data:
$$\tilde{\mathbf{x}_i} = \mathbf{x}_i - \mu.$$
3. Compute the covariance matrix
$$\mathbf{\Sigma} = \sum_i \tilde{\mathbf{x}}_i\tilde{\mathbf{x}}_i^\top = \frac{1}{n}\tilde{X}^\top\tilde{X}.$$
4. Compute the eigendecomposition of $\mathbf{\Sigma}$
$$\mathbf{\Sigma} = P\Lambda P^\top$$
where Λ is a diagonal matrix of eigenvalues sorted in a decreasing way.
5. Project your data onto the eigenvectors corresponding to the top k eigenvectors to get $\mathcal{D}$.

Proof: We want to create the basis which *captures the most variance of your data*. That is, we want to iteratively find $\mathbf{x}_1$ solving the following problem

$$\begin{aligned} \text{maximize: } & \mathbf{x}^\top\Sigma\mathbf{x} \\ \text{subj. to: } & \mathbf{x}^\top\mathbf{x} = 1 \end{aligned}$$

and use that as the first vector of the basis. So solve for $\mathbf{x}_1$ we first construct the Lagrangian, giving

$$\mathcal{L}(\mathbf{x},\lambda) = \mathbf{x}^\top\Sigma\mathbf{x} - \lambda(\mathbf{x}^\top\mathbf{x} - 1).$$

Taking a derivative with respect to $\mathbf{x}$ we can see

$$\nabla\mathcal{L}(\mathbf{x},\lambda) = 2\Sigma\mathbf{x} - 2\lambda\mathbf{x} = 0$$

This gives

$$\Sigma\mathbf{x} = \lambda\mathbf{x},$$

so $\mathbf{x}_1$ must be an eigenvector! It follows (check this yourself) that $\mathbf{x}_1$ must be the eigenvector with the greatest eigenvalue.

2 Singular Value Decomposition (SVD)

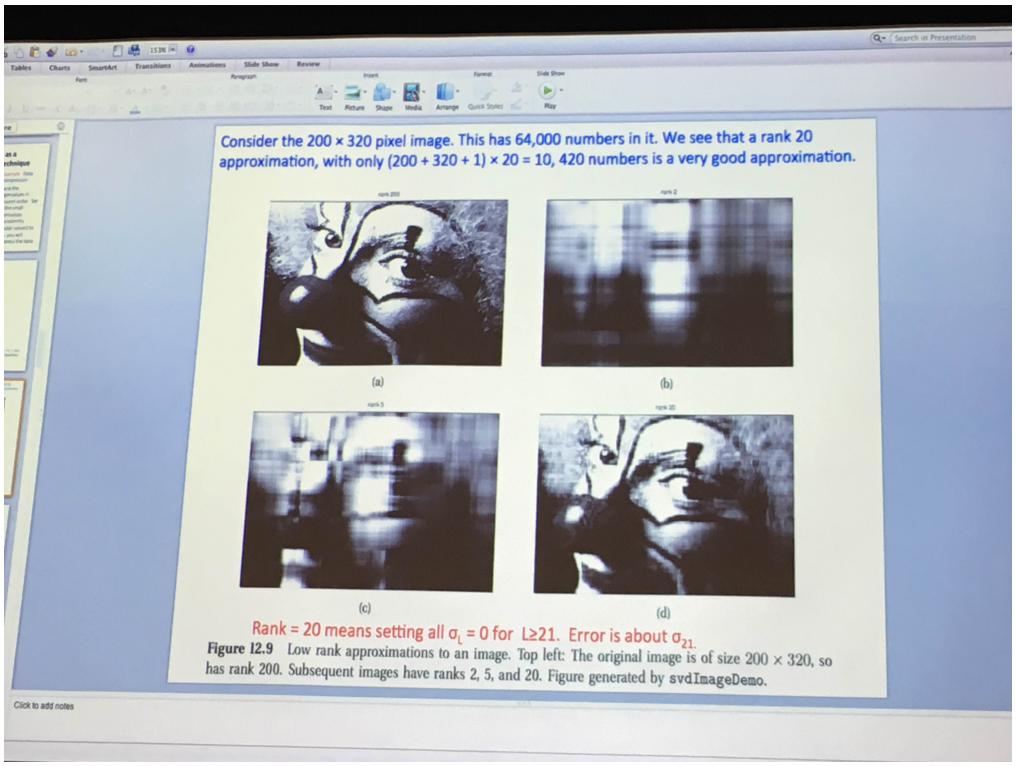
We can compose *any* matrix $A \in \mathbb{R}^{n \times m}$ into $A = USV^\top$ where $U \in \mathbb{R}^{n \times n}$, $S \in \mathbb{R}^{n \times m}$ and $V^\top \in \mathbb{R}^{m \times m}$. Also, S is a diagonal matrix of singular values $\{\sigma_i\}$. To compute this decomposition we can see that $A^\top A = VS^\top U^\top USV^\top = VS^\top SV^\top = VDV^\top$. This is clearly the eigendecomposition of $A^\top A$. Similarly we have $AA^\top = U\Lambda U^\top$, the eigendecomposition of $AA^\top$. Recall that if A is symmetric then it has a decomposition as $A = PDP^T$ where P is orthogonal and D is diagonal. However, we can get a decomposition of a matrix X is not symmetric, say it is $n \times d$. We can in fact decompose

$$X = UDV^T$$

where U is $n \times n$ orthogonal, V is $d \times d$ orthogonal, and D is $n \times d$ with elements on the (approximate) diagonal. The matrices U and V are constructed from the orthogonal decompositions of the matrices XX^T and X^TX respectively. Specifically we have that

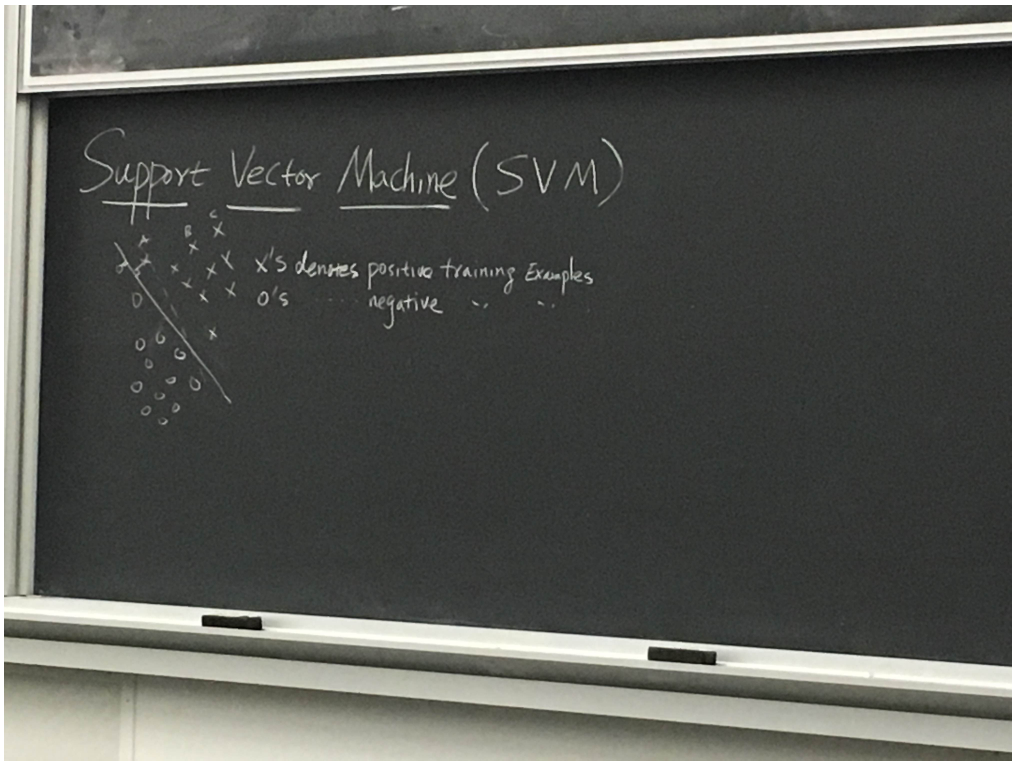
$$\begin{aligned} XX^T &= UD_1U^T \\ X^TX &= VD_2V^T \end{aligned}$$

where D_1 and D_2 in fact have the same nonzero diagonal entries $\lambda_1, \ldots, \lambda_k$ The matrix D in the singular value decomposition has diagonal entries $\sqrt{\lambda_1}, \ldots, \sqrt{\lambda_k}$. SVD can be used for dimensionality reduction in the following way: we can list our singular values in descending order and throw away dimensions with very small singular values. This way we can capture the large-scale dynamics of our data without computing every single component of the data.

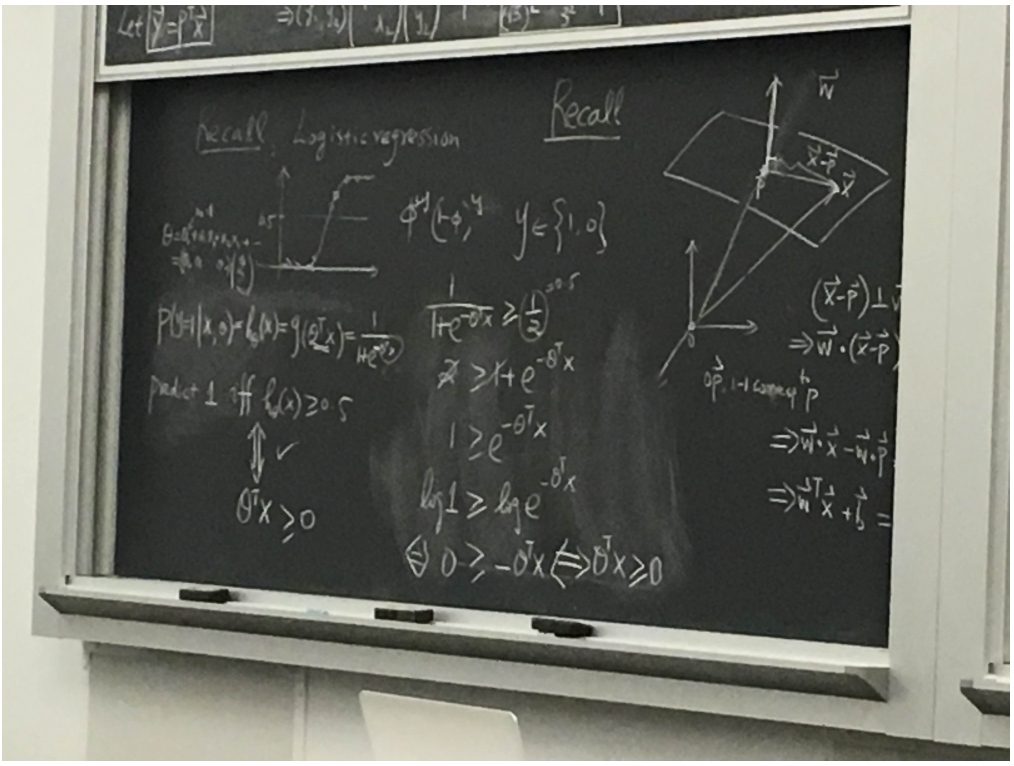


3 Support Vector Machine (SVM)

The support vector machine can be gathered from an intuitive geometric concept. The main idea here is that when we plot the data, we may want to classify our data according to some binary criterion.



We would like to find the line (or hyperplane, in higher dimensions) that separate the classes of data. One point of concern is that if we adjust the fitting line slightly, then some cases that may legitamtely be in one class will be instead put in the other class. Another way of saying this is that our data may be sensitive to the fitting. How do we model this concept geometrically? We call the **separating hyperplane** or **decision boundary** the plane that separates the two classes of data. Recall that we can describe a plane $P \subseteq \mathbb{R}^n$ by a vector $\vec{w}$ orthogonal to the plane and a point $p \in P$ in the plane. We can pick some other point $\vec{x} \in P$ and thus the vector $\vec{x} - p$ lies in the plane. Since this is perpendicular to the normal vector we have $(\vec{x} - p) \perp \vec{w}$ and therefore we can express this relationship as $\vec{w}^T \vec{x} + b = 0$ for some scalar b.



Geometric Observation: If a point is far from the separating hyperplane, we are very confident in our prediction for the classification of this data point. How do we write this down mathematically? We have

$$h_{w,b}(x) \;=\; g(w^T x + b)$$

where

$$g(z) \;=\; \begin{cases} 1, & \text{if } z \geq 0 \\ -1, & \text{if } z < 0 \end{cases}.$$

This is similar logistic regression, but we'd like to use only one function if possible. For instance, for the sigmoid function, if we want $\sigma(\theta^T x)$ to predict an answer of 1 if and only if $\sigma(\theta^T x) \geq 0.5$, this occurs if and only if $\theta^T x \geq 0$ (this can be shown by a basic computation). Just as in the Bernoulli random variable we have that $p(y;\phi) = \phi^y(1-\phi)^{1-y}$. We can use the training targets $y^{(i)} \in \{-1, 1\}$ and write

$$y^{(i)}(w^T x + b) \;\triangleq\; \hat{\gamma}^{(i)}.$$

Note that if $y^{(i)} = 1$ we want $\hat{\gamma}^{(i)}$ to be large and positive, implying that $w^T x + b$ to be large and positive. If on the other hand we have $y^{(i)} = -1$, then we want $\hat{\gamma}^{(i)}$ to be large and positive, requiring that we want $w^T x + b$ to be large and negative. In summary if $y^{(i)}(w^T x + b) > 0$ then our prediction is correct. The quantity $\hat{\gamma}^{(i)}$ is called the **functional margin**. The intutive concept here is that the normal vector w is adjusted and supported by the different data points from opposite sides of the decision boundary (in a learning algorithm, a data point will "support" the normal vector to keep the data point on the correct side of the decision boundary). A large functional margin represents a confident and correct prediction in the classification model. The larger this functional margin is, the better in terms of our prediction.

How can we find the distance between the plane and a data point? This can be done by usual vector analysis, however there is a caveat; it depends on how we represent our hyperplane. If we have a plane equation $w^T x + b = 0$, this is equivalent to $2w^T x + 2b = 0$. To avoid this ambiguity, we restrict our problem to the case that $\|w\| = 1$. The distance that a point $x^{(i)}$ is from the plane is given by $\hat{\gamma}^{(i)}$. If our origin is o, p is the point on our plane closest to $x^{(i)}$ and px the vector from p to $x^{(i)}$ we have that $\vec{op} + \vec{px} = x^{(i)}$ but then

$$p \;=\; \vec{op} \;=\; x^{(i)} - \vec{px} \;=\; x^{(i)} - \hat{\gamma}^{(i)} \frac{\vec{w}}{||\vec{w}||}$$

and in order for w to be orthogonal to this plane we have

$$\vec{w}^T \left(x^{(i)} - \hat{\gamma}^{(i)} \frac{\vec{w}}{||\vec{w}||} \right) + b \;=\; 0$$

thus we can solve for our functional margin to be

$$\hat{\gamma}^{(i)} \;=\; \frac{\vec{w}^T x^{(i)}}{||w||} + \frac{b}{||w||}.$$

The above worked for the case that $y^{(i)} = 1$. For when $y^{(i)} = -1$, we have a similar situation except that the direction is opposite from the first case. We find that we obtain the same quantity except for a minus sign. We can combine these results and in general just solve for

$$\hat{\gamma}^{(i)} \;=\; y^{(i)} \left(\frac{w^T x^{(i)}}{||w||} + \frac{b}{||w||} \right).$$

The next step is that we want to use a cost function in order to maximize our functional margin with respect to w and b. This is an example of an **optimal classifier**. The cost function we want to use simply $\gamma = \min_i \hat{\gamma}^{(i)}$.

$$\max_{w,b} \gamma$$

such that $y^{(i)}(w^T x + b) \geq \gamma$ and $||w|| = 1$. This cost function is called the **optimal margin classifier**. While we wrote down the optimization problem we wanted to solve, the problem is that this function is not convex. We need to discuss how we will choose to optimize this problem. Next time we'll see that we can cleverly change this to a convex problem.