COS 424: Interacting with Data

Lecturer: Rob Schapire & David Blei
Scribe: Janet Suzie Yoon

Lecture # 16
April 10, 2007

1 Logistic Regression and Naive Bayes (Rob)

Logistic Regression overview

The Logistic Regression (LR) model, given data $\mathbf{x} \in \mathbb{R}^n$, predicts $y \in \{-1, +1\}$ by directly modeling the probability $y = +1$ for given example x:

$$P(y = +1|\mathbf{x}, \mathbf{w}) = \sigma(\mathbf{w} \cdot \mathbf{x})$$

where $\mathbf{w}$ is the weight vector and $\sigma(z) = \frac{1}{1+e^{-z}}$. The problem of finding the weight vector $\mathbf{w}$ is solved by using maximum likelihood (i.e. minimum negative log likelihood).

$$\min - \sum_{i=1}^{m} \log(P(y_i|\mathbf{x_i}\mathbf{w})) = \min \sum_{i=1}^{m} \log(1 + exp(y_i \mathbf{w} \cdot \mathbf{x_i}))$$

Therefore Logistic Regression is a *discriminative* model. It only models $P(y|\mathbf{x})$, it does not care about the distribution over $\mathbf{x}$.

Naive Bayes overview

The Naive Bayes (NB) model, given data $\mathbf{x} \in \mathbb{R}^n$, predicts $y \in \{-1, +1\}$ by modeling

$$P(\mathbf{x}|y), P(y)$$

Naive Bayes is thus a *generative* model since it produces the probability distribution of pairs $(\mathbf{x}, y)$ (i.e. generates observed data).

Relating Logistic Regression with Naive Bayes

Let's focus our attention to the special boolean case of Naive Bayes. Suppose

$$\mathbf{x} =< x_1, x_2, ..., x_m >, x_j \in \{0, 1\}$$

Assume y is generated first, and each x_j is independently generated (see figure 1). Let

$$P(y) = \pi$$
$$P(x_j|y = +1) = \theta_{+j}$$
$$P(x_j|y = -1) = \theta_{-j}$$

Applying Bayes rule (and then a lot of algebra) we see that

$$\begin{aligned} P(y = +1|\mathbf{x}) &= P(\mathbf{x}|y = +1)P(y = +1)/P(\mathbf{x}) \\ &= \sigma\left(\log\left(\frac{\pi}{1-\pi}\right)\right) + \sum_j \log\left(\frac{1-\theta_{+j}}{1-\theta_{-j}}\right) + \sum_j x_j \log\left(\frac{\theta_{+j}\,(1-\theta_{-j})}{\theta_{-j}\,(1-\theta_{+j})}\right) \\ &= \sigma(w_0 + \mathbf{w} \cdot \mathbf{x}) \end{aligned}$$

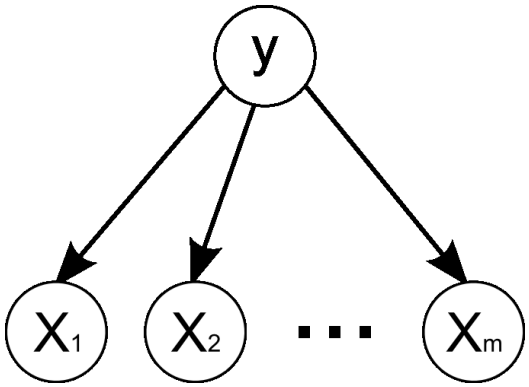


Figure 1: Naive Bayes Model

where $w_0 = \sigma\left(\log\left(\frac{\pi}{1-\pi}\right) + \sum_j \log\left(\frac{1-\theta_{+j}}{1-\theta_{-j}}\right)\right)$ and $w_j = \log\left(\frac{\theta_{+j}(1-\theta_{-j})}{\theta_{-j}(1-\theta_{+j})}\right)$. This is exactly the same form as Logistic Regression!!!!

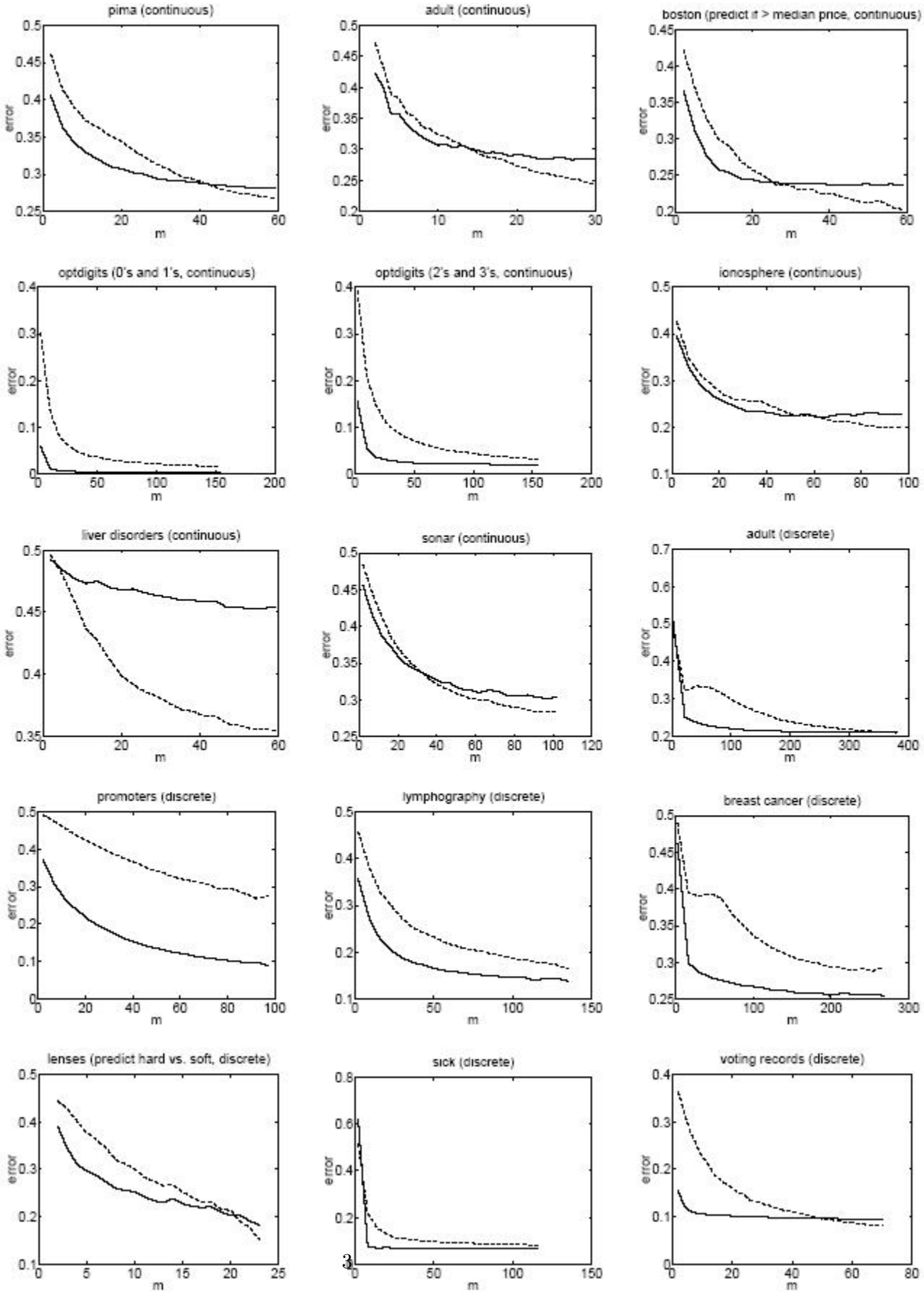
The two models produce the same results as the training set size approaches ∞ *IF* the Naive Bayes assumption holds that the x_i's are conditionally independent of one another given y. So what are the differences between the two models?

Naive Bayes comes up with $\sigma\left(w_0 + \mathbf{w} \cdot \mathbf{x}\right)$ using a restrictive generative model. Thus, to build this model, it needs to infer π, θ_{-j}, θ_{+j}. These values can be estimated from frequency counts. Thus the amount of data needed to acheive good results (i.e. the same results as when we have an infinate number of data) is relatively small. More specificially, it can be shown to be $O(\log n)$.

Logistic Regression, on the otherhand, is less restrictive. Although consistent with the Naive Bayes assumption that the x_i's are conditionally independent given y, Logistic Regression is not rigidly tied to this assumption. If data is given that violates this assumption, then Logistic Regression will adjust the weights to maximize fit to the data. The weights are chosen arbitrarily and thus require a full search over the linear space of possible models. The data requirement for this is of size $O(n)$. Therefore Logistic Regression converges slower to its asymptotic accuracy than Naive Bayes.

To conclude, Naive Bayes is the better choice for small data sets (Logistic Regression will overfit). For large data sets, the winner is Logistic Regression (Naive Bayes underfits).

In figure 2 we see how Naive Bayes and Logistic Regression compare over different data sets. Each graph is a different data set. The plots are the error verses the size of the training data. The dashed lines are Logistic Regression and the solid lines are Naive Bayes. These graphs support the claim that Naive Bayes works better over smaller data sets, and Logistic Regression works better over larger data sets.


pima (continuous)
adult (continuous)
boston (predict if > median price, continuous)
optdigits (0's and 1's, continuous)
optdigits (2's and 3's, continuous)
ionosphere (continuous)
liver disorders (continuous)
sonar (continuous)
adult (discrete)
promoters (discrete)
lymphography (discrete)
breast cancer (discrete)
lenses (predict hard vs. soft, discrete)
sick (discrete)
voting records (discrete)
error
m

2 Dimensionality Reduction (Dave)

Dimensionality Reduction is the real value analog to clustering. The goal is to reduce the representation of our data. Therefore, if we have data $\mathbf{x} \in \mathbb{R}^p$, we want to transform it to $\tilde{\mathbf{x}} \in \mathbb{R}^q$ where $q < p$.

$$\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_p \end{pmatrix} \longrightarrow \begin{pmatrix} \tilde{\mathbf{x}}_1 \\ \vdots \\ \tilde{\mathbf{x}}_q \end{pmatrix}$$

Why would we want to do this?

- find a simpler way of looking at the data (exploratory technique)
- use the simpler representation for better learning
- you think the data is simpler then it seems
- faster computation, reduced memory requirements (compression)

One way to achieve Dimensionality Reduction is Principle Components Analysis.

Principle Components Analysis (PCA)

The basic idea of PCA is to project the high dimensional data in a low dimension manifold in the original space. For an example, suppose we want to convert our $2D$ data into $1D$. Figure 3 shows our data in 2 dimensions.

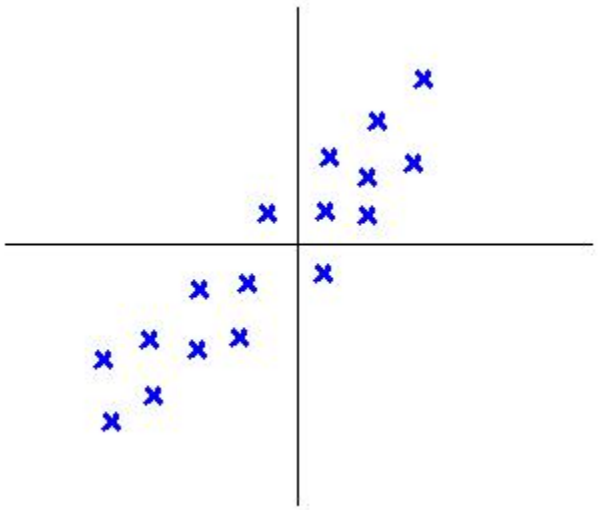


Figure 3: Data points in $\mathbb{R}^2$

A subspace in one-dimension is a line (such as the line in red in figure 4). We project the data point (x_1, x_2) on the line (figure 5). Now the data point is represented by its location on the line. Thus PCA works in the following steps:

1. Define a low dimensional manifold (in our example, the line) in the orginal space.
2. Represent each data point $\mathbf{x}$ by its projection onto this manifold.

But, how do we choose the line (subspace)?

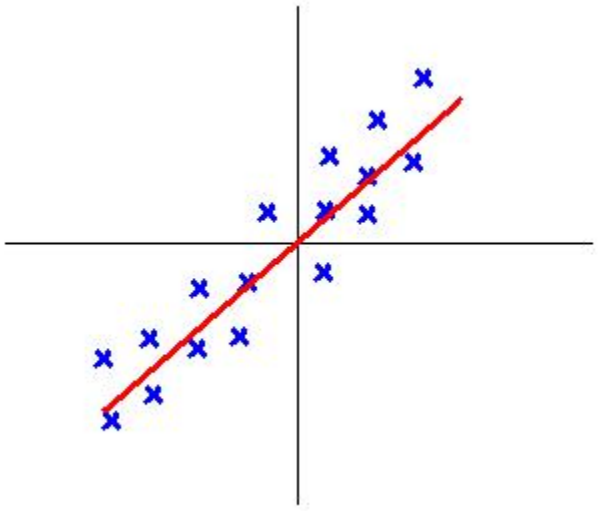


Figure 4: One dimensional subspace (the red line) in $\mathbb{R}^2$

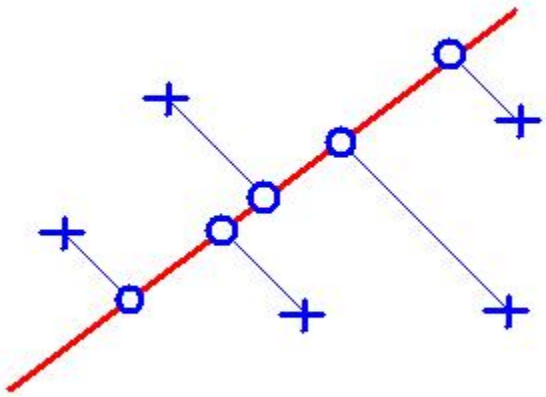


Figure 5: Projection of $2D$ data points onto a line.

Choosing the subspace

There are three equivalent methods for picking a subspace.

- **Maximize the *variance* of the projection** (Hotelling 1933). In other words, this method tries to maximize the spread of the projected data. Once more taking our $2D$ to $1D$ example, if we look in figure 6 we see the line $x = y$ has a better spread of the data points then $y = 0$ (the x axis). Thus, $x = y$ would be a better pick out of these two.

- **Minimize the *reconstruction error*** (Pearson 1901). Another method is to minimize the squared distance between the original data and its "estimate" in the low

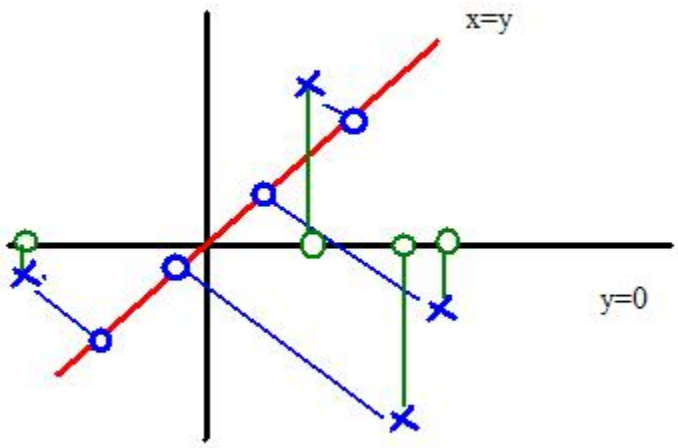


Figure 6: Variance of $y = 0$ and $y = x$

dimensional space.

- ***MLE* of the parameter in a latent variable** (Bishop 1996, Roqeig 1998, Rob!!! 2001). The projection of x onto a subspace in rank-q can be modeled by $\sum_{i=1}^{q} \alpha_i x_i + \epsilon_j$ where the α_i's are the latent variables and ϵ_j is the error. The model is fit by maximum likelihood.

Figure 7 illustrates for when $q = 2$. The right plot shows the projection x_i's onto the $2D$ subspace.

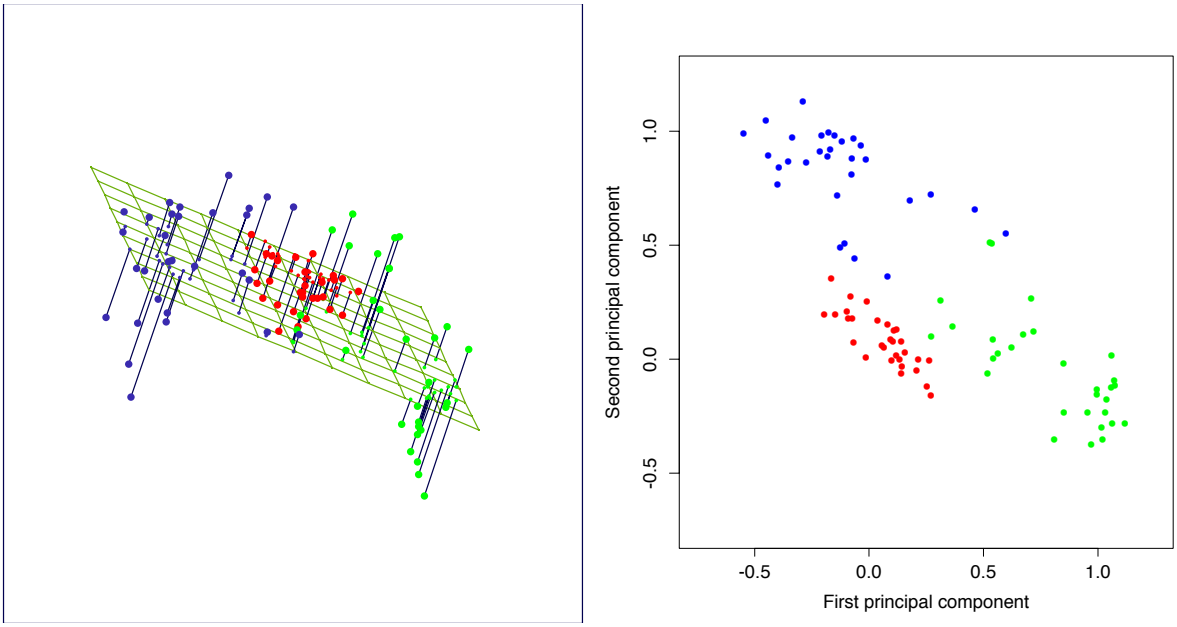


Figure 7: The best rank-two linear approximation to the half-sphere data.

Example

Figure 8 shows a sample of 130 handwritten threes, each a digitized 16×16 grayscale image. We can see noticible variations in writing style, thickness, and orientation of these samples.

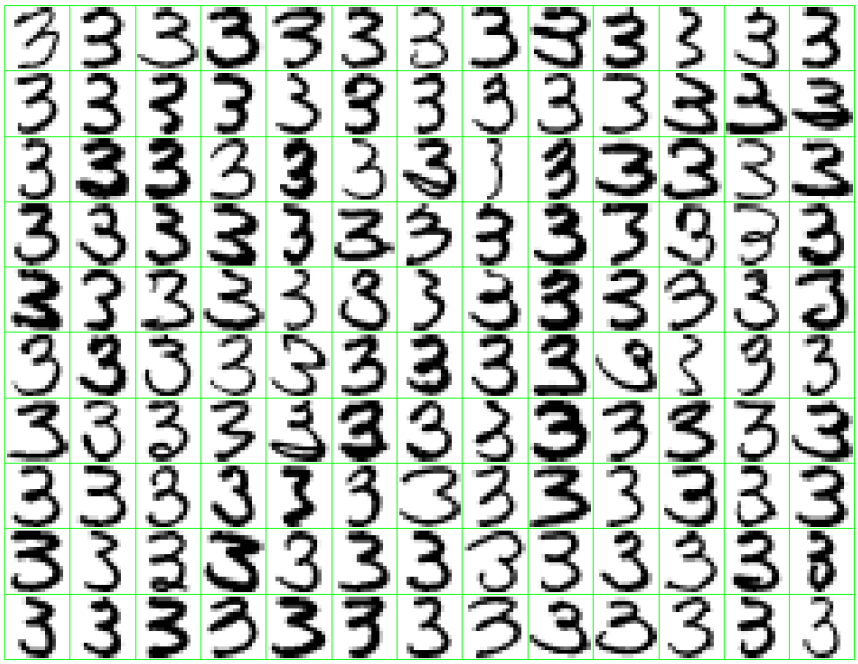


Figure 8: A sample of 130 handwritten threes shows a variety of writing styles.

Consider these images as points $x_i \in \mathbb{R}^{256}$. In other words, the images are defined by 256 components (or features). Figure 9 shows the nature of the first two principle components of the data. The grid super-imposed on the left graph is defined by the $0.05, 0.25, 0.5, 0.75$ and 0.95 quantile points. The 0.05 quantile point q is the point such that $P(X \leq q) = .05$. The circled points are the images that are close to the vertices of the grid and correspond to the grid on the right. From left to right, the lower tail of the three's lengthen and from top down, the thickness increases. These features are represented by the variables v_1 and v_2 respectively. Thus the two-component model has the form shown in Figure 10.

Data Reconstruction

Let data $\mathbf{x_1}, ..., \mathbf{x_n} \in \Re^p$. We define the reconstruction of the data projection on the subspace $\mathbb{R}^q$ back to space $\mathbb{R}^p$ as $f(\lambda)$.

$$f(\lambda) = \mu + V_q \lambda$$

where $\mu \in \mathbb{R}^p$, V_q is a $p \times q$ matrix consisting of q orthogonal unit vectors, and δ is a vector in $\mathbb{R}^q$. The value λ is the low dimenional representation of the data points and the matrix V_q is its representation on the line. Thus finding a good low-D space is basically finding good V, λ, μ values. Fitting this model to data amounts to minimizing the reconstruction error.

$$\min \sum \|\mathbf{x_i} - \mu - V_q \lambda_\mathbf{i}\|^2$$

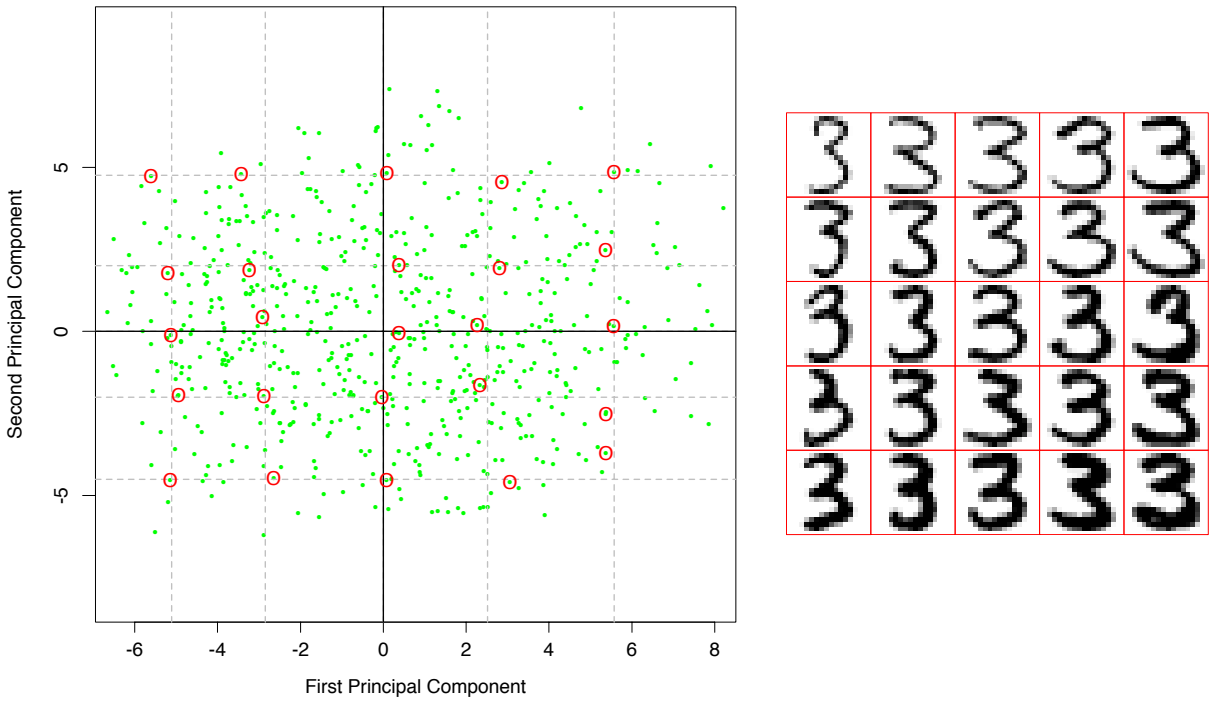


Figure 9: Projection of 3-dimensional datapoints onto a 2-dimensional subspace.

$$\hat{f}(\lambda) = \bar{x} + \lambda_1 v_1 + \lambda_2 v_2$$

$$= \quad + \lambda_1 \cdot \quad + \lambda_2 \cdot \quad .$$

Figure 10: The first two principal component diractions, v_1 and v_2 are displayed as images.