Basic Machine Learning
Lecture 8
Dimensionality Reduction
&
Principal Component Analysis (PCA)

Mannes Poel

Gwenn Englebienne

Study material

- Sections 12.1 and 12.4.2 of the course book.

Discriminating/Biased ML

- Amazon recruiting engine did not like women.

Probabilistic Models

Bayes Law:

$$\underbrace{P(\mathrm{C}\mid\mathbf{x})}_{posterior}=\frac{\overbrace{p(\mathbf{x}\mid\mathrm{C})}^{likelihood}\overbrace{P(C)}^{prior}}{\underbrace{p(\mathbf{x})}_{evidence}}$$

Classify new *x* as C_1 if $P(C_1|x) > P(C_2|x)$ else x is classified as C_2
this is equivalent to
Classify new *x* as C1 if $P(x|C_1)P(C_1) > P(x|C_2)P(C_2)$ else x is classified as C2

Challenge: How to calculate/estimate $P(x|C_1)$ and $P(x|C_2)$?

Probabilistic Approach

Advantages:

- Insight in how the data is generated.
- No incremental learning involved. Just one shot parameter estimation.

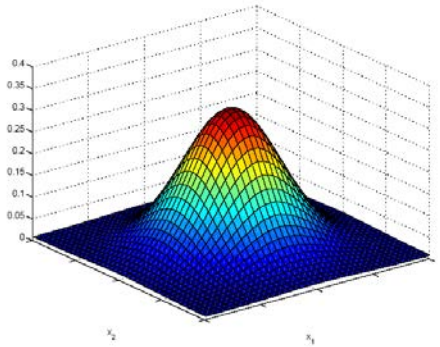
Disadvantages:

- Per class one needs to estimate $d + d(d+1)/2$ parameters. Needs more data.
- Problems with data on *low dimensional subspace.*

What is the problem?

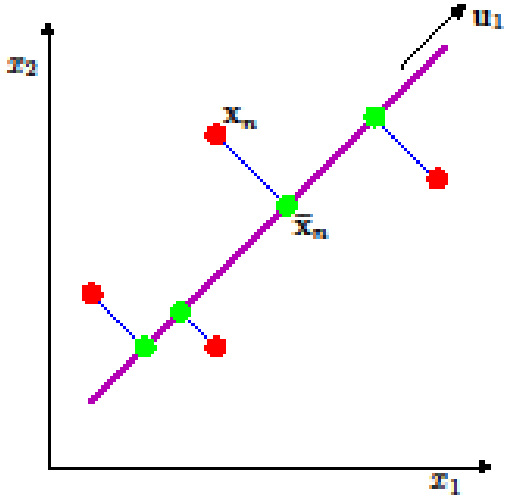
$$\mathbf{x} \sim \mathcal{N}_d(\boldsymbol{\mu}, \Sigma)$$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right]$$



Cause of the problems

Dimensionality Reduction



Why Reduce Dimensionality?

1. Removing singularities; for instance in covariance matrix Σ
2. Reduces time complexity: Less computation
3. Reduces space complexity: Less parameters, smaller feature size → less memory
4. Saves the cost of observing the feature
5. Simpler models are more robust on small datasets, for small datasets with a lot of features overfitting is a problem even for simple models.
6. More interpretable; simpler explanation
7. Data visualization (structure, groups, outliers, etc) if plotted in 2 or 3 dimensions

Feature Selection vs Extraction

- Feature selection:
  - Choosing $k<d$ important features, ignoring the remaining $d - k$.
    Subset selection algorithms
- Feature extraction:
  - Project the original x_i , i =1,...,d dimensions to new $k<d$ dimensions, z_j , j =1,...,k
    Principal components analysis (PCA), linear discriminant analysis (LDA), factor analysis (FA)

Subset Selection

Main problem: there are 2^d subsets of d features.

- Correlation based: remove one of the features with high correlation. Stop when no features with high correlations are left.
- Knowledge based: remove irrelevant features. For instance patient id.
- But relevance can be dependent on the ML model under consideration.

Subset Selection

Given a ML model *M:*

- Forward search: Add the best feature at each step
  - Set of features *F* initially Ø.
  - At each iteration, find the best new feature

    j = argmin$_i$ $E(F \cup x_i)$ (error of *M* when feature x_i is added to the feature set.
  - add x_j to F if $E\left(F \cup x_j\right) < E(F)$
  - Hill-climbing O(d^2) algorithm
- Backward search: Start with all features and remove one at a time, if possible.
- Floating search (Add *k*, remove *l*)

Principal Components Analysis (PCA)

Used for:

- Feature/Dimensionality reduction
- Lossy data compression
- Data visualization

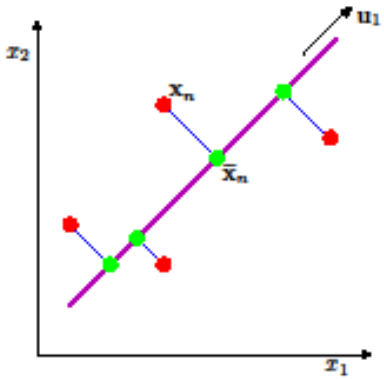
Also known as *Karhunen-Loève* transformation

Principal Components Analysis (PCA)

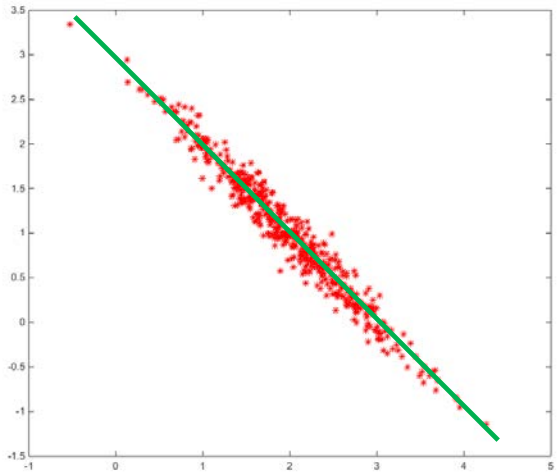
- Find a lower (one)-dimensional space *S* such that when $\{\boldsymbol{x}_n\}$is projected on *S*, information loss is minimized.
- The projection of $\{\boldsymbol{x}_n\}$ on the direction of $\boldsymbol{w}$ is: $\{\boldsymbol{w}^T\boldsymbol{x}_n\}$
- Find $\boldsymbol{w}$ such that Var($\{\boldsymbol{w}^T\boldsymbol{x}_n\}$) is maximized

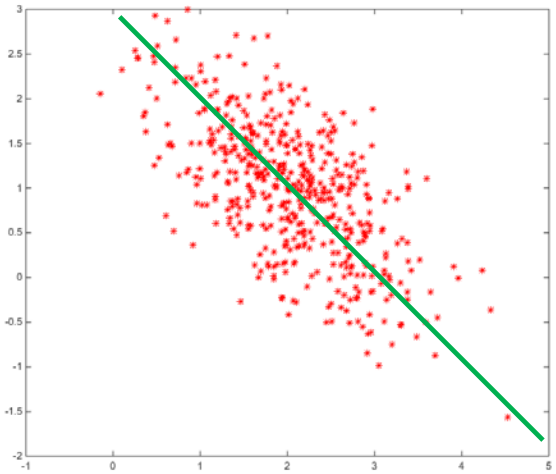
$$\mathrm{Var}(\{\boldsymbol{w}^T\boldsymbol{x}_n\}) = E[(\boldsymbol{w}^T\boldsymbol{x}_n - \boldsymbol{w}^T\overline{\boldsymbol{x}})^2]$$
$$= E[(\boldsymbol{w}^T\boldsymbol{x}_n - \boldsymbol{w}^T\overline{\boldsymbol{x}})(\boldsymbol{w}^T\boldsymbol{x}_n - \boldsymbol{w}^T\overline{\boldsymbol{x}})]$$
$$= E\left[\boldsymbol{w}^T(\boldsymbol{x}_n - \overline{\boldsymbol{x}})\left((\boldsymbol{x}_n - \overline{\boldsymbol{x}})\right)^T\boldsymbol{w}\right]$$
$$= \boldsymbol{w}^T E\left[(\boldsymbol{x}_n - \overline{\boldsymbol{x}})\left((\boldsymbol{x}_n - \overline{\boldsymbol{x}})\right)^T\right]\boldsymbol{w} = \boldsymbol{w}^T\Sigma\boldsymbol{w}$$

where Σ is the covariance of $\{\boldsymbol{x}_n\}$.



Example





- Maximize Var($\{\boldsymbol{w}^T\boldsymbol{x}_n\}$) subject to $||\boldsymbol{w}|| = 1$

$$\max_{\boldsymbol{w}} \left(\boldsymbol{w}^T \sum \boldsymbol{w} - \lambda\left(\boldsymbol{w}^{\boldsymbol{T}}\boldsymbol{w} - 1\right)\right)$$

Solution:

$$\sum \boldsymbol{w} = \alpha \boldsymbol{w}$$

that is, $\boldsymbol{w}$ is an eigenvector of $\sum$.

Choose the eigenvector, say $\boldsymbol{w}_1$ with the largest eigenvalue. This gives the maximal variance.

Second principal component:

Second Principal Component

Maximize Var($\{\boldsymbol{w}^T \boldsymbol{x}_n\}$),

Constrains: $||\boldsymbol{w}|| = 1$ and orthogonal to $\boldsymbol{w}_1$.

This gives that:

$$\sum \boldsymbol{w} = \alpha \boldsymbol{w}$$

and $\boldsymbol{w}$ is orthogonal to $\boldsymbol{w}_1$, the first principal component. Hence $\boldsymbol{w}$ is the eigenvector corresponding to the second largest eigenvalue.

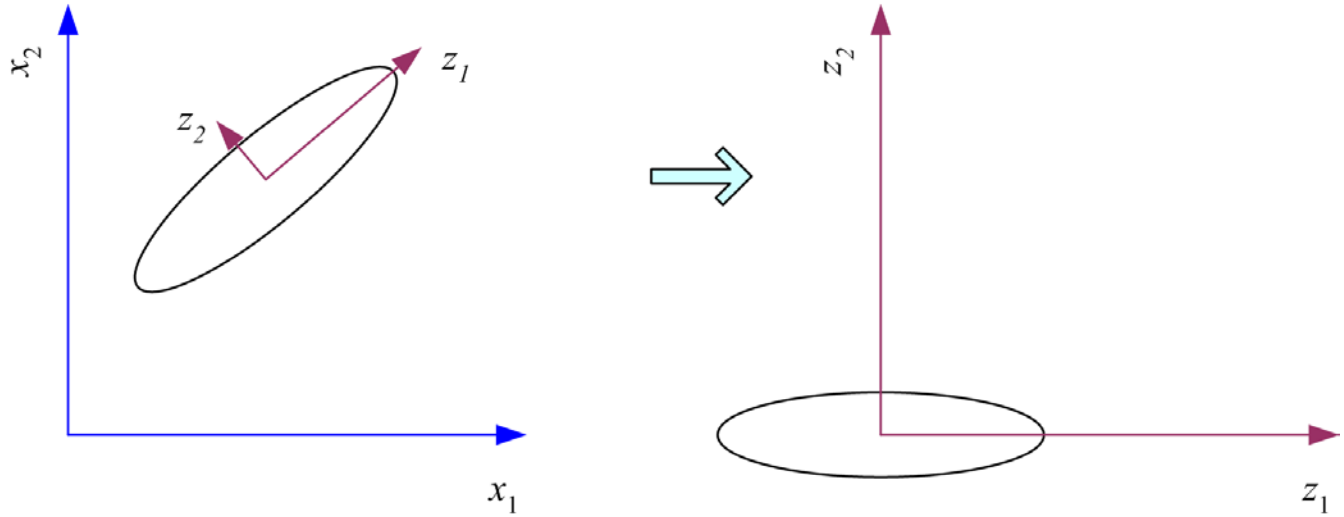
What PCA does

$$\boldsymbol{z} = \mathbf{W}^T(\boldsymbol{x} - \boldsymbol{m})$$

$\boldsymbol{z}$ are the ***new*** feature vectors, mean($\boldsymbol{z}$)=0

where the columns of **W** are the eigenvectors of $\sum$, and $\boldsymbol{m}$ is sample mean.

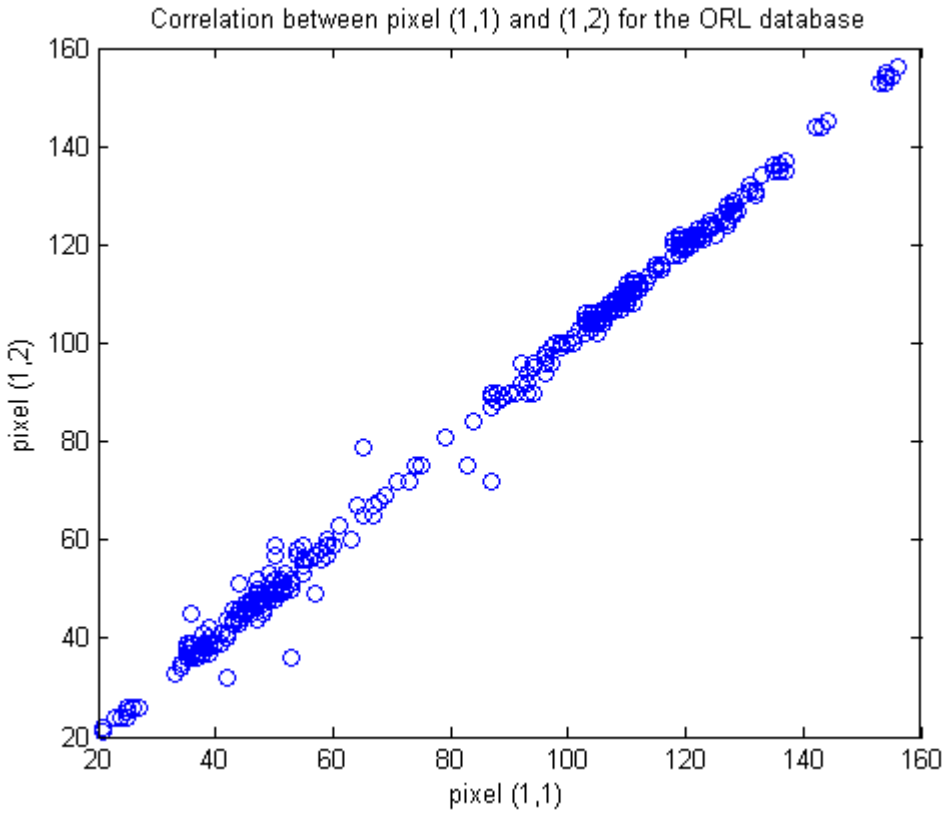
Centers the data at the origin and rotates the axes



Example of PCA: Eigenfaces

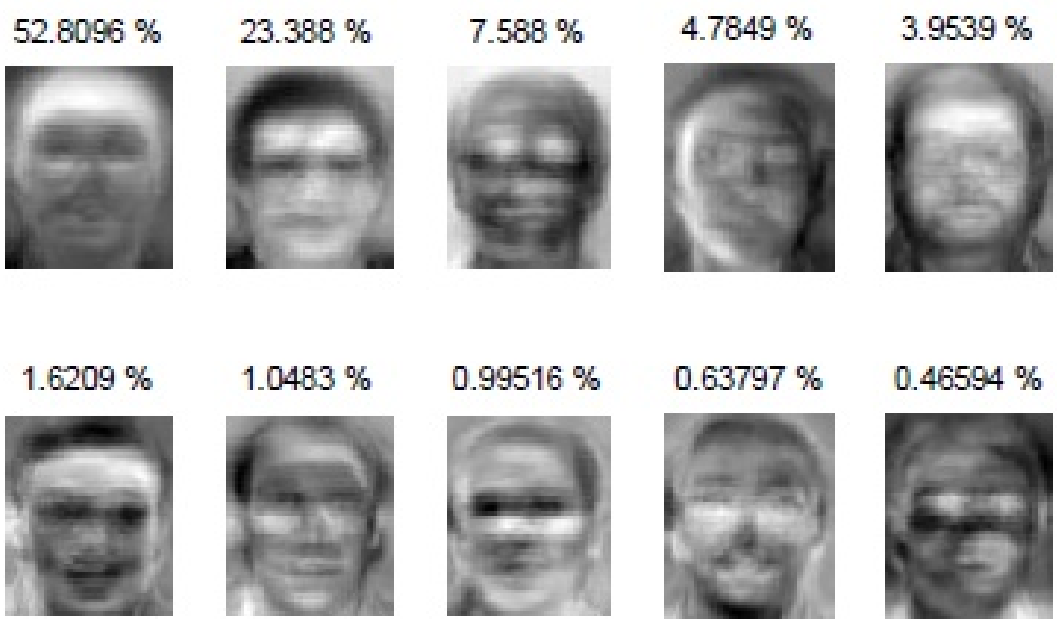
- Often applied in face recognition and reconstruction.
- Example; the OLR database.
- 46x56 greyscale image, so 2576 dimensional data but a lot redundancy

Correlation between neighbouring pixels

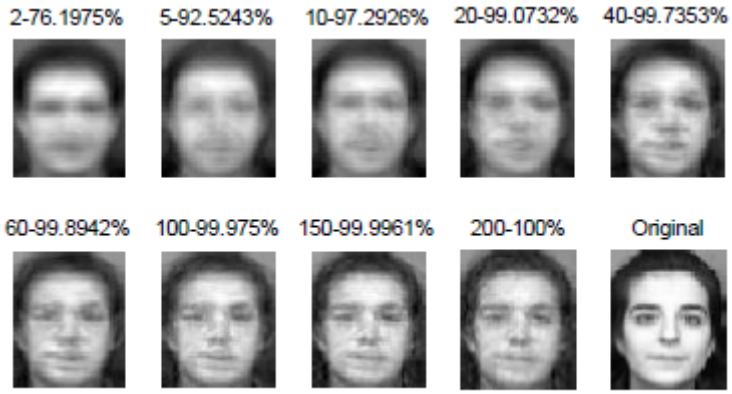


The principal components

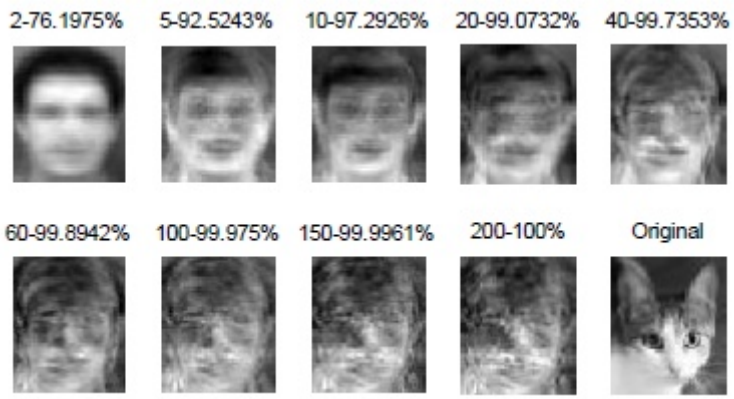
- A principal component is a direction and for this case a direction corresponds with a face (up to a scalar). The 10 most significant ones:



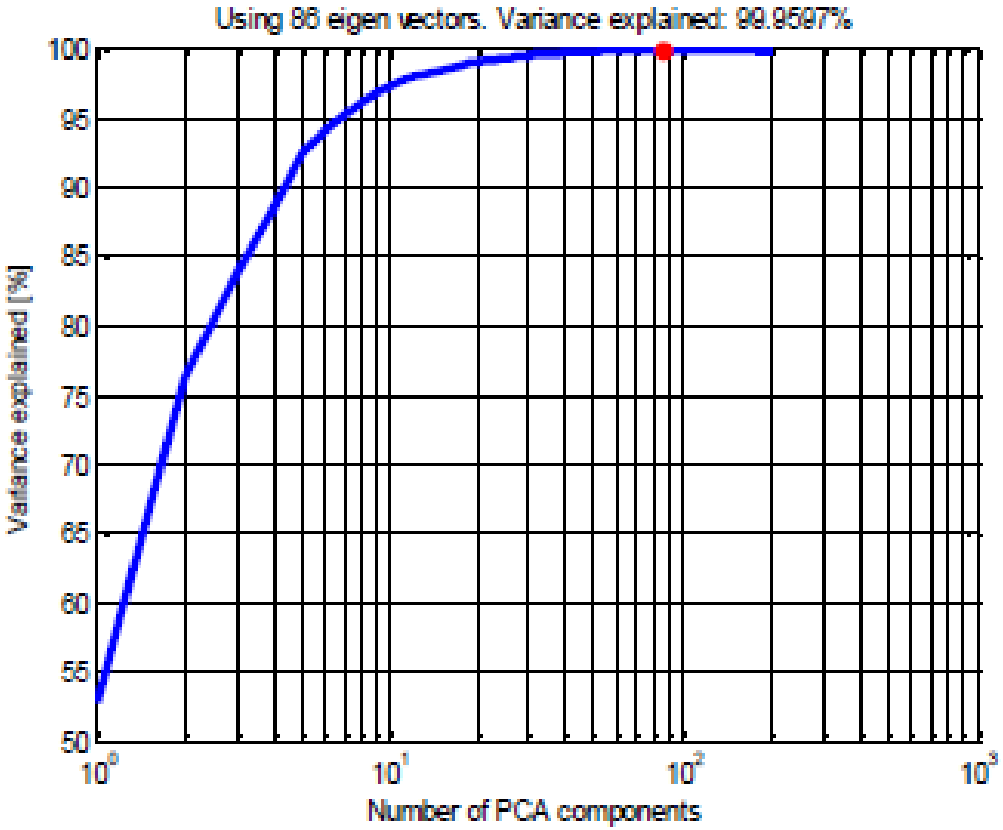
Face reconstruction



Non face reconstruction



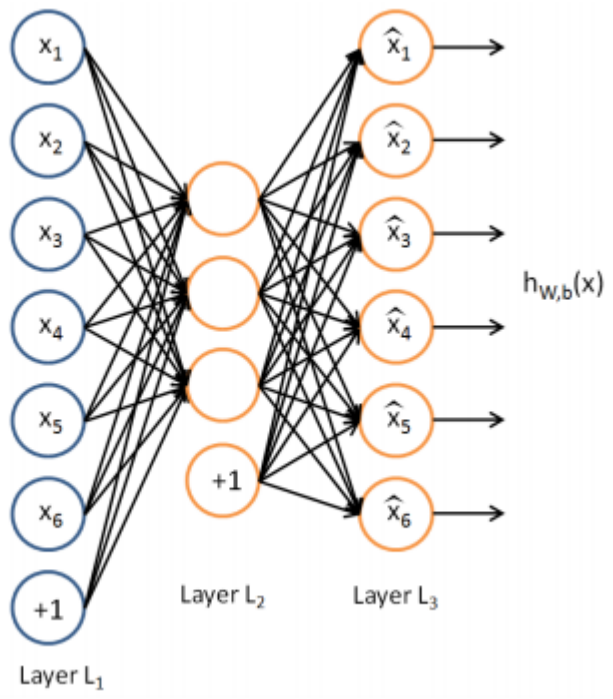
Variance explained by PCA



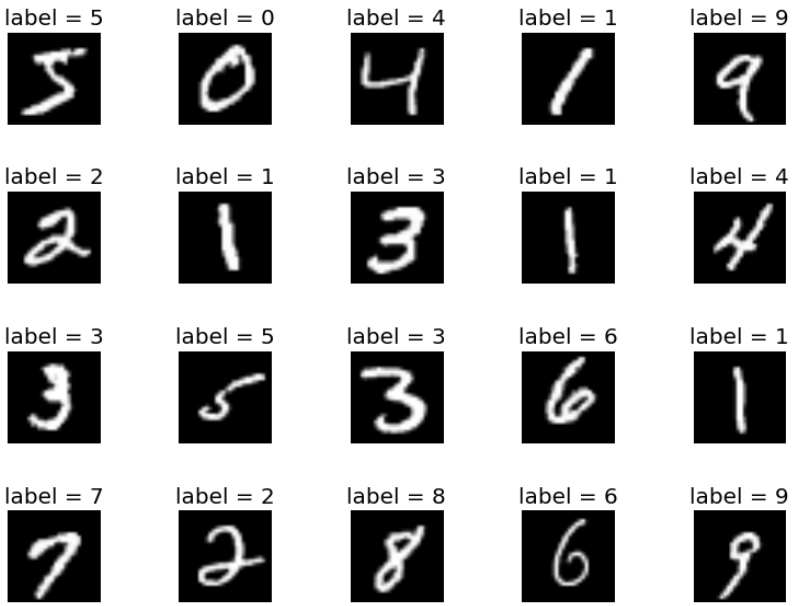
Autoassociative NNs

- Another approach to dimensionality reduction, also called autoencoders in DL.
- Squeeze the data through a NN:
  - Output layer same dimension as input layer.
  - One hidden layer with dimension (much) smaller then input/ouput dimension.
  - Target is same as input, so dataset is $D = \{(x_n, x_n) | n = 1 \ldots . N\}$

Autoencoder



Example; MNIST Data set



Digits 0 – 9 in 8x8 pixel gray image

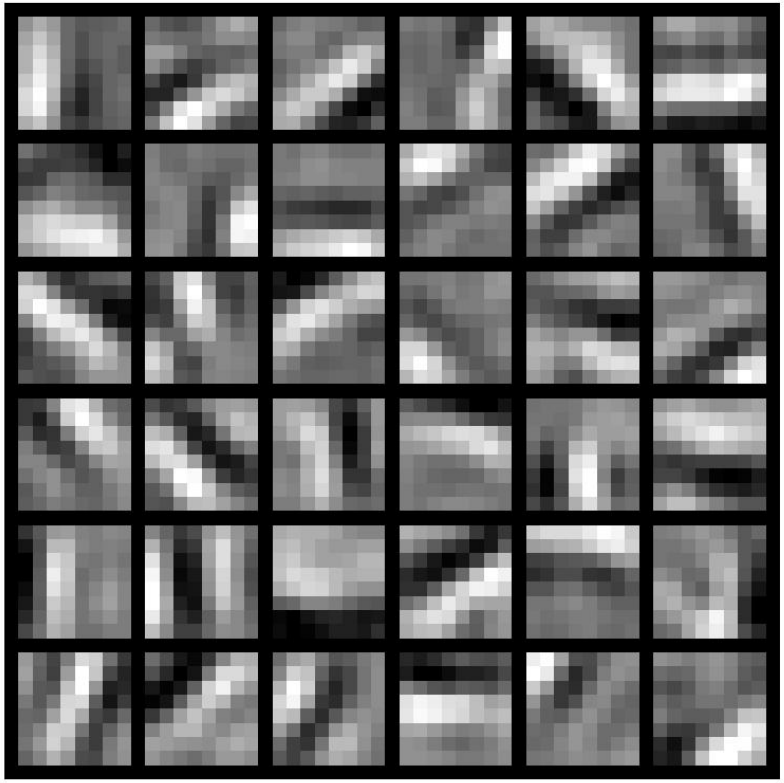
Example

Applying auto-encoder on MNIST digit dataset:

- Hidden layer of 36 neurons.
- For each neuron the input is depicted which will result in the highest output.

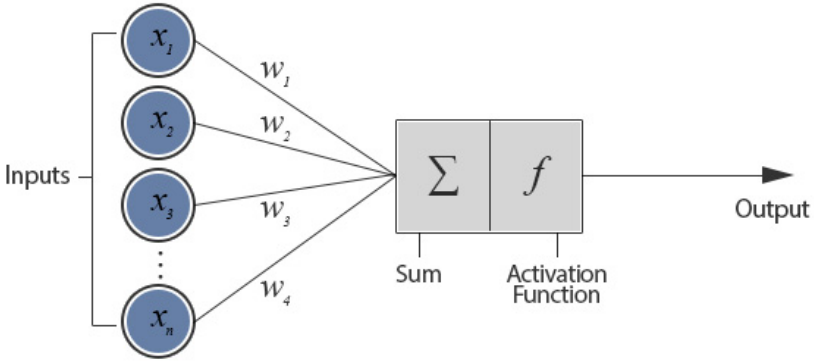
Results in the following 36 dimensions.

Classification performance improved from 89% to 95%



Input with highest output

- For which input x is the output highest?
- Assume $||x||=1$.

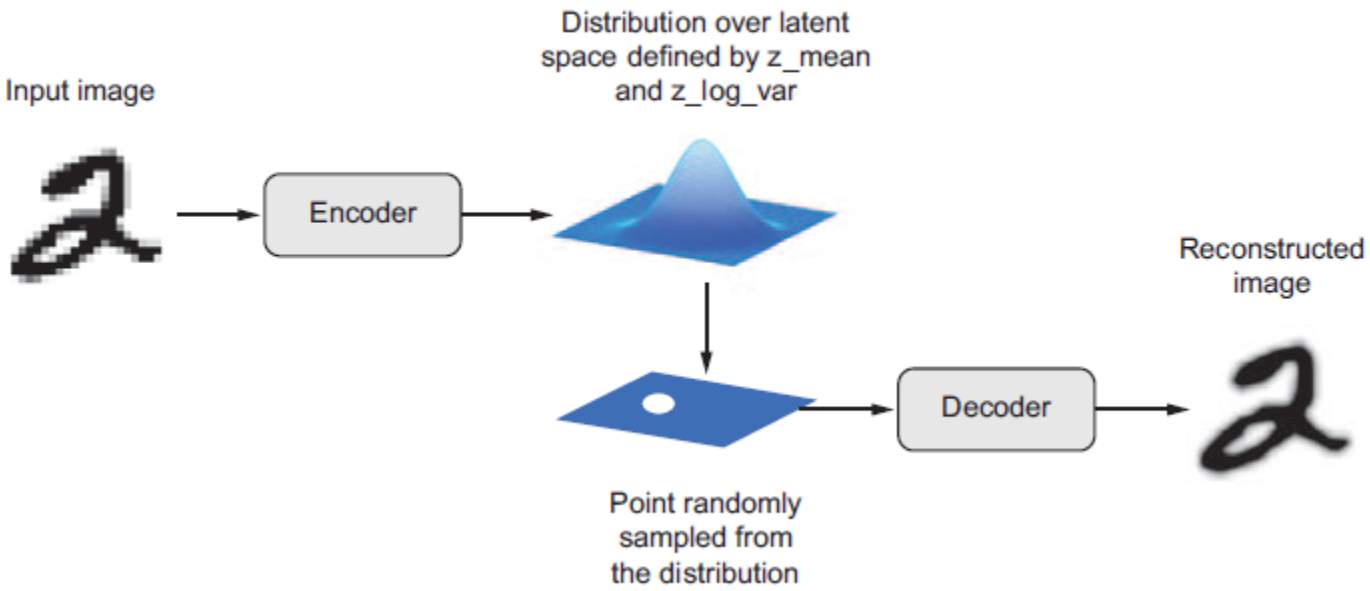


- $x=w/||w||$

Question

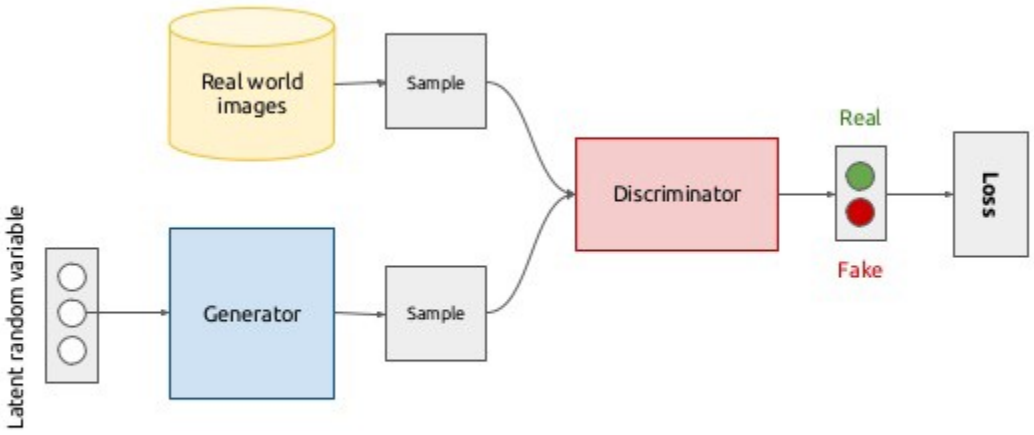
Are NN discriminative or generative models?

Variational Autoencoders



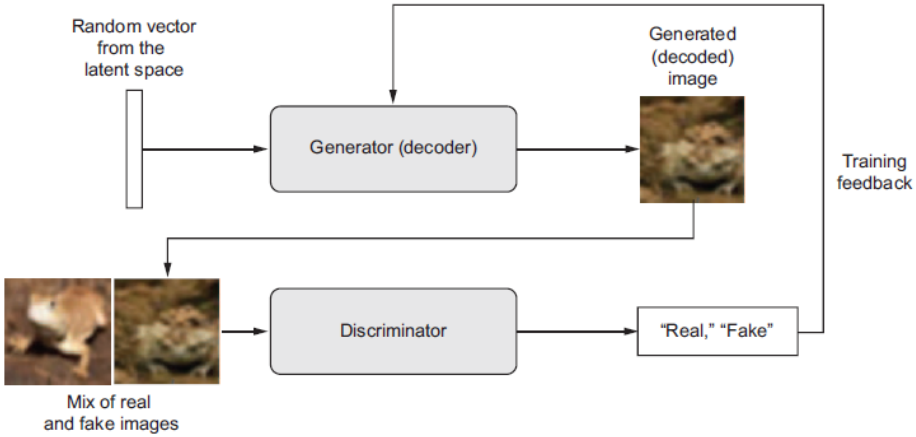
See also link

Generative Adversal Networks



GAN architecture

Example of GAN



Wrap up

- Introduction to Machine Learning
- Basic models and learning methods are covered:
  - Logistic classification
  - NN
  - SVM
  - DT
  - Probabilistic classification
- Methodology for evaluating and comparing ML models
- Dimensionality reduction: PCA & Autoencoders

What’s next

- Advanced ML course:
  - Mixture models
  - Sequence classification
  - Deep Learning
  - Project
- Deep Learning – From Theory to Practice


Are there any questions?

Parametric Estimation for p(x|C)

- Write $p(x)$ instead of $p(x|C)$
- $X = \{ x^t \}_t$ where $x^t \sim p(x)$
- Parametric estimation:

  Assume a parametric form for p *i.e.* $p(x)=p(x \mid \theta)$ and estimate θ, its sufficient statistics, using $\mathcal{X}$

  e.g., $p=\mathcal{N}(\mu, \sigma^2)$ where $\theta = \{ \mu, \sigma^2\}$: p is normally distributed with mean μ and variance σ^2

Maximum Likelihood Estimation

- Likelihood of the sample $\mathcal{X}$ given θ

$$l\,(\mathcal{X}\,|\theta) = p\,(\mathcal{X}\,|\theta) = \prod_t p\,(x^t|\theta)$$

- Log likelihood

$$\mathcal{L}(\mathcal{X}|\,\theta) = \log l\,(\mathcal{X}\,|\theta) = \sum_t \log p\,(x^t|\theta)$$

(Why should one consider a log likelihood?)

- Maximum likelihood estimator (MLE)

$$\theta^* = \operatorname{argmax}_\theta \mathcal{L}(\mathcal{X}\,|\theta)$$

Multivariate Data

- Multiple measurements (sensors)
- *d* inputs/features/attributes: *d*-variate
- *N* instances/observations/examples

$$\mathbf{X} = \begin{bmatrix} X_1^1 & X_1^2 & \cdots & X_1^N \\ X_2^1 & X_2^2 & \cdots & X_2^N \\ \vdots & & & \\ X_d^1 & X_d^2 & \cdots & X_d^N \end{bmatrix}$$

Sometimes one uses another convention:
Each row is a data point!

Multivariate Parameters

$$\text{Mean}: E[\boldsymbol{x}] = \boldsymbol{\mu} = [\mu_1, \ldots, \mu_d]^T$$

$$\text{Covariance}: \sigma_{ij} \equiv \text{Cov}(X_i, X_j)$$

$$\text{Correlation}: \text{Corr}(X_i, X_j) \equiv \rho_{ij} = \frac{\sigma_{ij}}{\sigma_i \sigma_j}$$

$$\Sigma \equiv \text{Cov}(\boldsymbol{X}) = E\left[(\boldsymbol{X} - \mu)(\boldsymbol{X} - \mu)^T\right] = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1d} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2d} \\ \vdots & & & \\ \sigma_{d1} & \sigma_{d2} & \cdots & \sigma_d^2 \end{bmatrix}$$

MLE: Parameter Estimation

$$\text{Sample mean } \mathbf{m}: m_i = \frac{\sum_{t=1}^{N} x_i^t}{N}, i = 1, \ldots, d$$

$$\text{Covariance matrix } \mathbf{S}: s_{ij} = \frac{\sum_{t=1}^{N} \left(x_i^t - m_i\right)\left(x_j^t - m_j\right)}{N}$$

$$\text{Correlation matrix } \mathbf{R}: r_{ij} = \frac{s_{ij}}{s_i s_j}$$

This is what is called a biased estimator for the covariance:
If we divide by N-1 then we have an unbiased estimator. Hence most of the time we use 1/(N-1)

Estimator for covariance

$$\text{Covariance matrix } \mathbf{S} : s_{ij} = \frac{\sum_{t=1}^{N} \left(x_i^t - m_i\right)\left(x_j^t - m_j\right)}{N}$$

This is what is called a biased estimator for the covariance:
If we divide by N-1 then we have an unbiased estimator. Hence most of the time we use 1/(N-1)

$$\text{Covariance matrix } \mathbf{S} : s_{ij} = \frac{\sum_{t=1}^{N} \left(x_i^t - m_i\right)\left(x_j^t - m_j\right)}{N-1}$$

Quiz: estimation of mean and covariance

Given a class C with example data set:

$$C=\{(2,0)^T,(0,2)^T,(2,2)^T, (0,0)^T\}$$

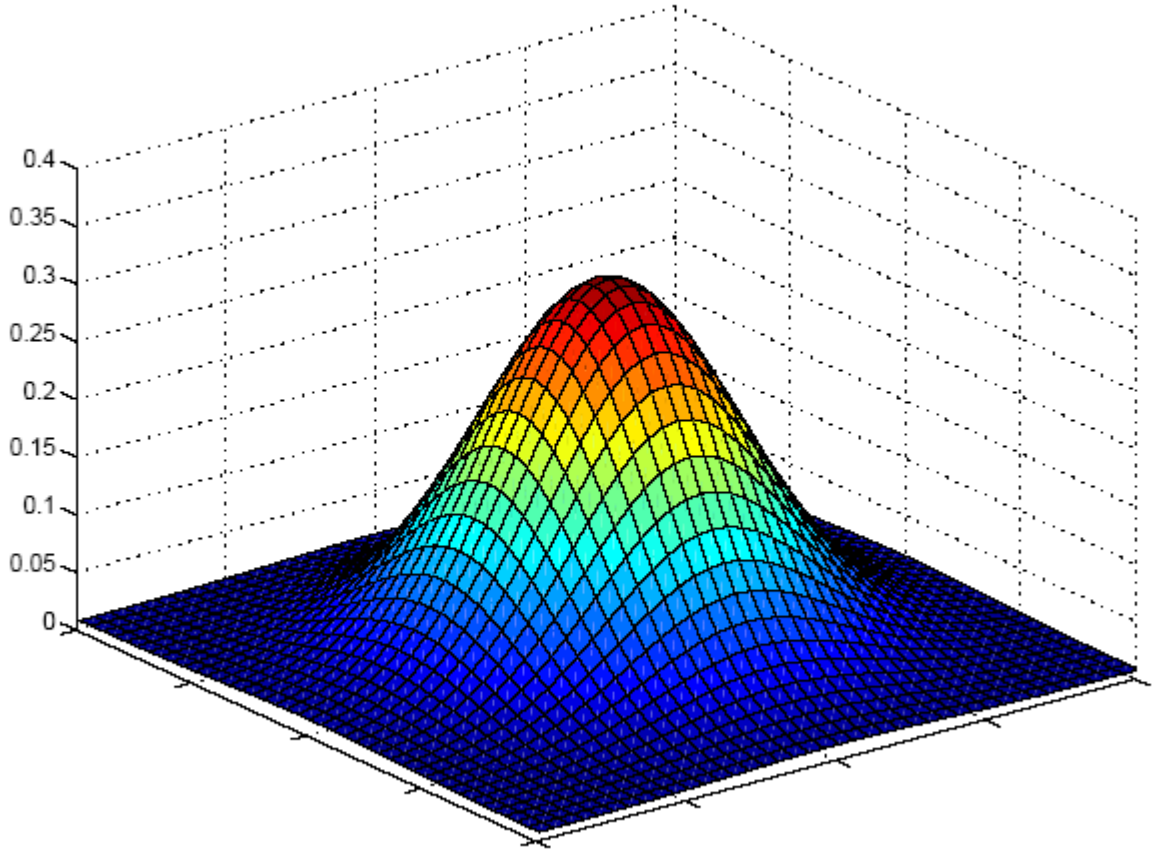
compute the mean and covariance.

Given $x=(1,2)^T$, compute the likelihood that x is generated by C, i.e. compute $P(x|C)$.

Answer

1. Estimate mean C: $m=[1,1]^T$
2. Estimate covariance C:
   $S=\begin{bmatrix} 1.33 & 0 \\ 0 & 1.33 \end{bmatrix}$
3. Compute likelihood $P(x|C)=0.082$

Multivariate Normal Distribution



$$\boldsymbol{x} \sim \mathcal{N}_d(\boldsymbol{\mu}, \Sigma)$$

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} \exp\left[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right]$$

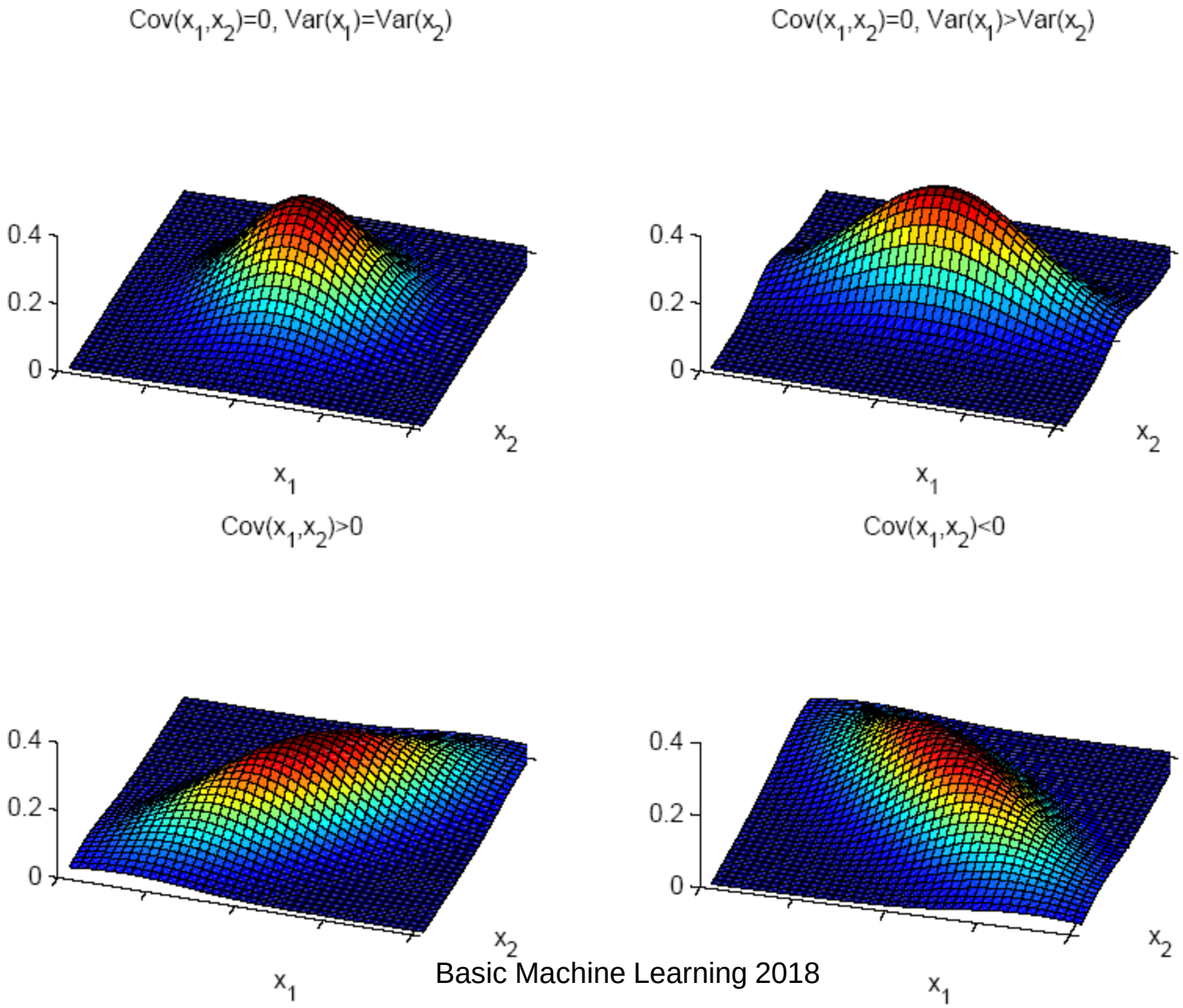
Multivariate Normal Distribution

- Mahalanobis distance: $(\boldsymbol{x} - \boldsymbol{\mu})^T \sum^{-1} (\boldsymbol{x} - \boldsymbol{\mu})$ measures the distance from $\boldsymbol{x}$ to $\boldsymbol{\mu}$ in terms of $\sum$ (normalizes for difference in variances and correlations)
- Bivariate: $d = 2$

$$\Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}$$

$$p(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left[-\frac{1}{2(1-\rho^2)} \left(z_1^2 - 2\rho z_1 z_2 + z_2^2 \right) \right]$$

$$z_i = (x_i - \mu_i) / \sigma_i$$


Cov(x1,x2)=0, Var(x1)=Var(x2)
Cov(x1,x2)=0, Var(x1)>Var(x2)
Cov(x1,x2)>0
Cov(x1,x2)<0
0.4
0.2
0
x1
x2

Independent Inputs: Naive Bayes

- If x_i are independent, offdiagonals of $\sum$ are 0, Mahalanobis distance reduces to weighted (by $1/\sigma_i$) Euclidean distance:

$$p(\mathbf{x}) = \prod_{i=1}^{d} p_i(x_i) = \frac{1}{(2\pi)^{d/2} \prod_{i=1}^{d} \sigma_i} \exp\left[-\frac{1}{2} \sum_{i=1}^{d} \left(\frac{x_i - \mu_i}{\sigma_i} \right)^2 \right]$$

- If variances are also equal, reduces to Euclidean distance

Parametric Classification

- If $p(\boldsymbol{x} \mid C_i) \sim \mathcal{N}(\boldsymbol{\mu}_i, \sum_i)$

$$p(\mathbf{x} \mid C_i) = \frac{1}{(2\pi)^{d/2} |\Sigma_i|^{1/2}} \exp\left[-\frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) \right]$$

- Discriminant functions are

$$g_i(\mathbf{x}) = \log p(\mathbf{x} \mid C_i) + \log P(C_i)$$

$$= -\frac{d}{2} \log 2\pi - \frac{1}{2} \log |\Sigma_i| - \frac{1}{2} (\mathbf{x} - \mu_i)^T \Sigma_i^{-1} (\mathbf{x} - \mu_i) + \log P(C_i)$$

Estimation of Parameters

$$\hat{P}(C_i) = \frac{\sum_t r_i^t}{N}$$

$$\boldsymbol{m}_i = \frac{\sum_t r_i^t \boldsymbol{x}^t}{\sum_t r_i^t}$$

$$\mathbf{S}_i = \frac{\sum_t r_i^t \left(\boldsymbol{x}^t - \boldsymbol{m}_i\right)\left(\boldsymbol{x}^t - \boldsymbol{m}_i\right)^T}{\sum_t r_i^t}$$

$$g_i(\boldsymbol{x}) = -\frac{1}{2}\log\left|\mathbf{S}_i\right| - \frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{m}_i\right)^T \mathbf{S}_i^{-1}\left(\boldsymbol{x} - \boldsymbol{m}_i\right) + \log \hat{P}(C_i)$$

Different $\mathbf{S}_i$

- Quadratic discriminant

$$g_i(\boldsymbol{x}) = -\frac{1}{2}\log|\mathbf{S}_i| - \frac{1}{2}\left(\boldsymbol{x}^T\mathbf{S}_i^{-1}\boldsymbol{x} - 2\boldsymbol{x}^T\mathbf{S}_i^{-1}\boldsymbol{m}_i + \boldsymbol{m}_i^T\mathbf{S}_i^{-1}\boldsymbol{m}_i\right) + \log\hat{P}(C_i)$$

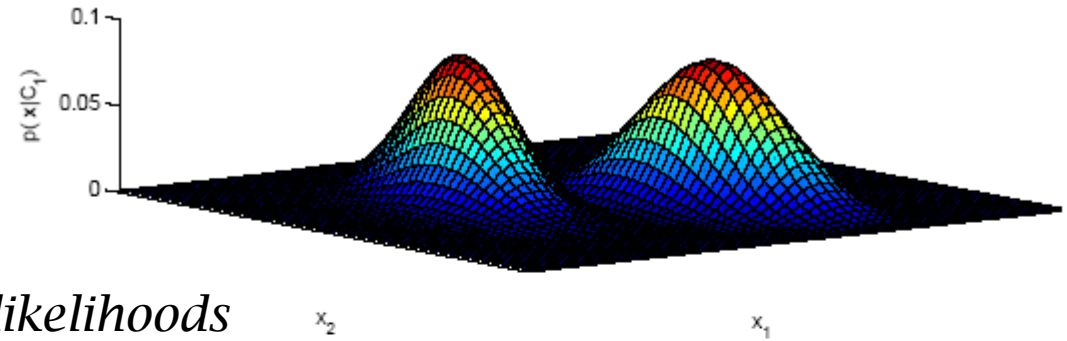
$$= \boldsymbol{x}^T\mathbf{W}_i\boldsymbol{x} + \boldsymbol{w}_i^T\boldsymbol{x} + w_{i0}$$

where

$$\mathbf{W}_i = -\frac{1}{2}\mathbf{S}_i^{-1}$$

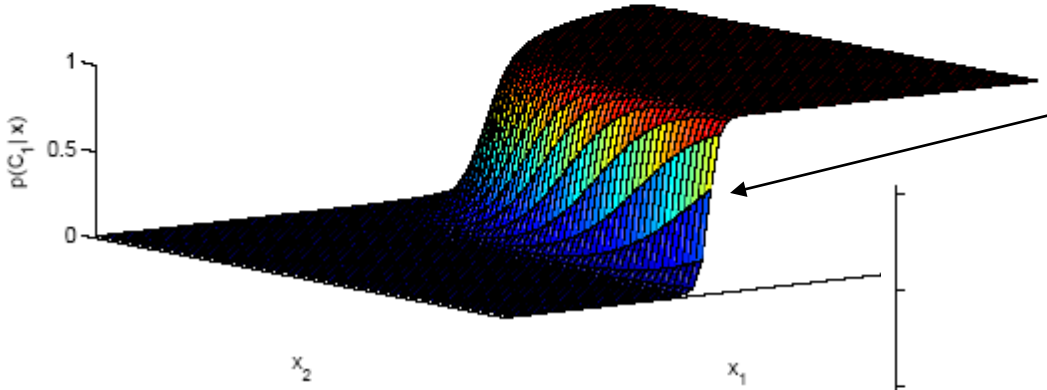
$$\boldsymbol{w}_i = \mathbf{S}_i^{-1}\boldsymbol{m}_i$$

$$w_{i0} = -\frac{1}{2}\boldsymbol{m}_i^T\mathbf{S}_i^{-1}\boldsymbol{m}_i - \frac{1}{2}\log|\mathbf{S}_i| + \log\hat{P}(C_i)$$

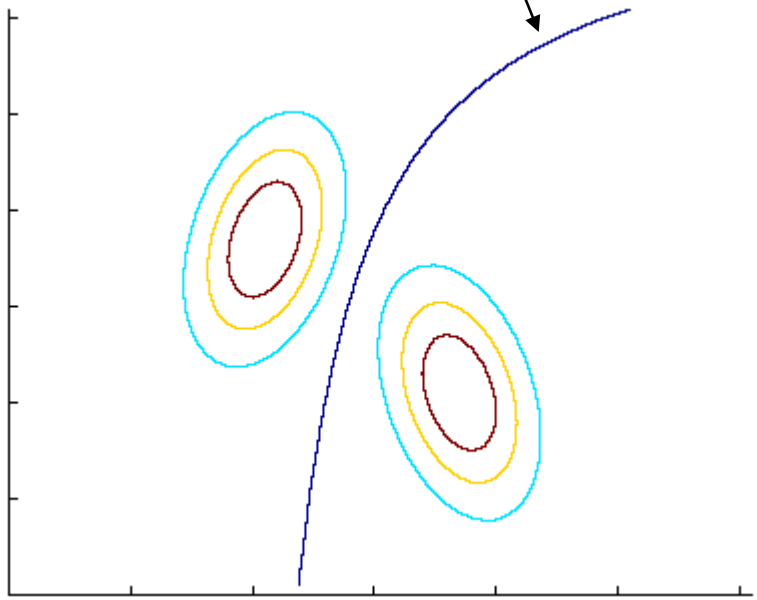


likelihoods

discriminant:
$P(C_1|\boldsymbol{x}) = 0.5$



posterior for C_1



Common Covariance Matrix **S**

- Shared common sample covariance **S**

$$\mathbf{S} = \sum_i \hat{P}(C_i)\mathbf{S}_i$$

- Discriminant reduces to

$$g_i(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{m}_i)^T \mathbf{S}_i^{-1}(\boldsymbol{x} - \boldsymbol{m}_i) + \log \hat{P}(C_i)$$

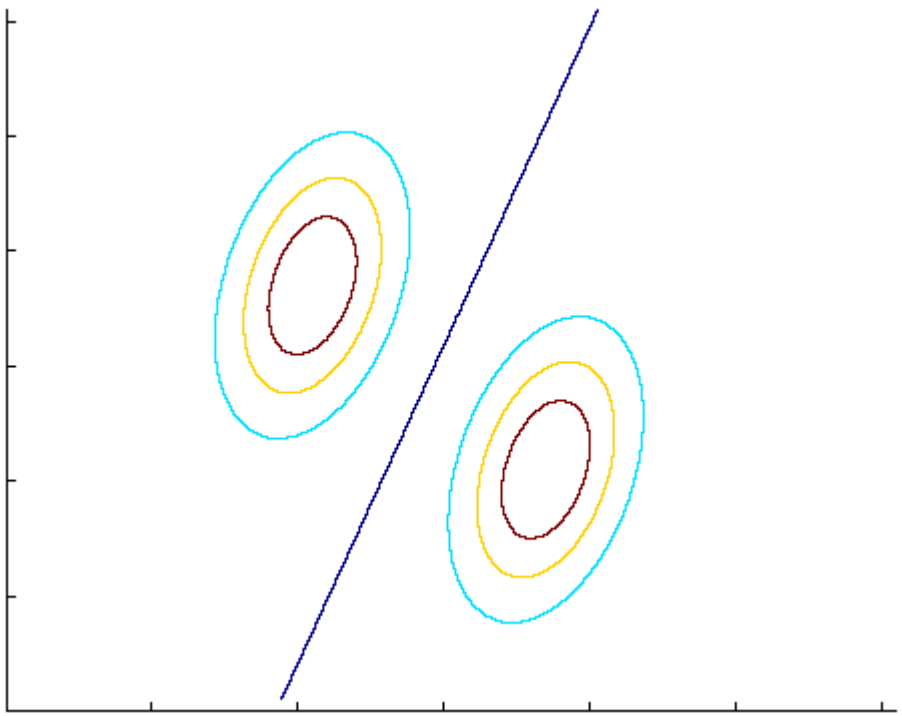
$$g_i(\mathbf{x}) = \mathbf{w}_i^T \mathbf{x} + w_{i0}$$

where

$$\mathbf{w}_i = \mathbf{S}_i^{-1}\mathbf{m}_i \quad w_{i0} = -\frac{1}{2}\mathbf{m}_i^T \mathbf{S}_i^{-1}\mathbf{m}_i + \log \hat{P}(C_i)$$

which is a linear discriminant

Common Covariance Matrix **S**



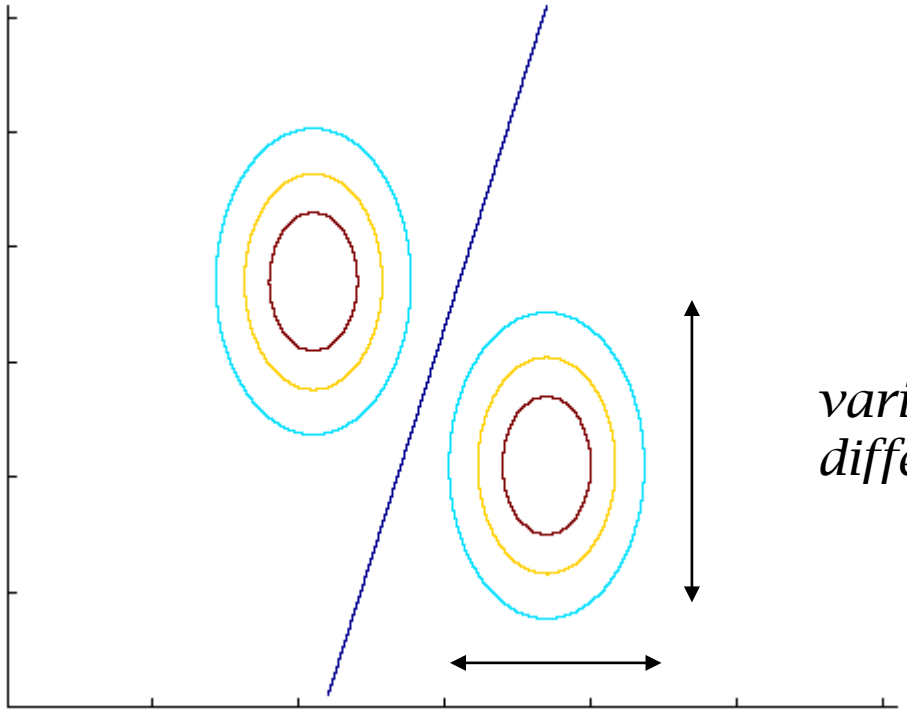
Diagonal **S**

- When x_j j = 1,..d, are independent, $\sum$ is diagonal
  p ($\mathbf{x}$|C_i) = $\prod_j p$ (x_j |C_i) (Naive Bayes' assumption)

$$g_i(\mathbf{x}) = -\frac{1}{2}\sum_{j=1}^{d}\left(\frac{x_j^t - m_{ij}}{s_j}\right)^2 + \log \hat{P}(C_i)$$

Classification based on weighted Euclidean distance (in s_j units) to the nearest mean

Diagonal **S**



variances may be different

Diagonal **S**, equal variances

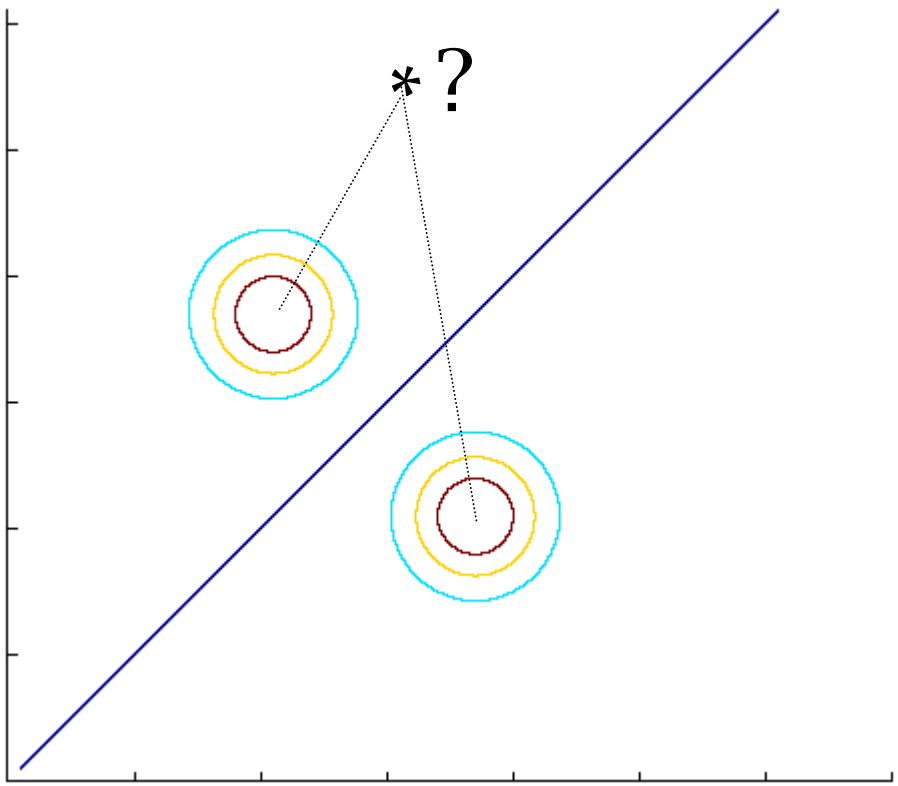
- Nearest mean classifier: Classify based on Euclidean distance to the nearest mean

$$g_i(\boldsymbol{x}) = -\frac{\left\|\boldsymbol{x} - \boldsymbol{m}_i\right\|^2}{2s^2} + \log \hat{P}(C_i)$$

$$= -\frac{1}{2s^2}\sum_{j=1}^{d}\left(x_j^t - m_{ij}\right)^2 + \log \hat{P}(C_i)$$

- Each mean can be considered a prototype or template and this is template matching

Diagonal **S**, equal variances



Model Selection

Assumption	*Covariance matrix*	*No of parameters*
Shared, Hyperspheric	$\mathbf{S}_i=\mathbf{S}=s^2\mathbf{I}$	1
Shared, Axis-aligned	$\mathbf{S}_i=\mathbf{S}$, with $s_{ij}=0$	d
Shared, Hyperellipsoidal	$\mathbf{S}_i=\mathbf{S}$	$d(d+1)/2$
Different, Hyperellipsoidal	$\mathbf{S}_i$	$K\,d(d+1)/2$

- As we increase complexity (less restricted **S**), bias decreases and variance increases
- Assume simple models (allow some bias) to control variance (regularization)