CS 559 Machine Learning

Lecture 4: Logistic Regression and PCA

Ping Wang
Department of Computer Science
Stevens Institute of Technology

STEVENS
INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®

Review of Last Lecture

- Generative methods vs Discriminative methods
- Linear classifiers: Linear Discriminant Functions, Least Square Classification, Fisher's Linear Discriminant, The Perceptron Algorithm

Today’s Lecture

- Evaluation Metrics for Classification
- Probabilistic Generative Models
- Logistic Regression
- Principal Component Analysis

Evaluation Metrics for Classification

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, interpretability, scalability, etc.
- The contingency table or confusion matrix:

		Actual Class	
Predicted Class		Positive	Negative
	Positive	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	True negative (TN)

Metrics for Performance Evaluation

	Actual Class		
Predicted Class		Positive	Negative
	Positive	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	True negative (TN)

- Most widely-used metric: Accuracy= $\frac{TP+TN}{TP+TN+FP+FN}$
- Limitation:
  - For a 2-class problem, $N_0 = 9990, N_1 = 10$.
  - If the model predicts everything to be C_0, the accuracy is $\frac{9990}{10000} = 99.9\%$.
  - This is misleading since the model does not detect any example from C_1.

Cost of Classification

- Consider the cost of misclassification
- Example: cancer vs. non-cancer

Cost Matrix	Actual Class		
Predicted Class		Positive	Negative
	Positive	-1	1
	Negative	100	0

Model 1	Actual Class		
Predicted Class		Positive	Negative
	Positive	150	60
	Negative	40	250

Accuracy=80%
Cost=3190

Model 2	Actual Class		
Predicted Class		Positive	Negative
	Positive	250	5
	Negative	45	200

Accuracy=90%
Cost=4255

Cost-Sensitive Classification

- So far, we haven't considered costs during training.
- Most learning algorithms do not perform cost-sensitive learning.
- Taking the costs into the training procedure will be an algorithm specific task.
  - Consider the cost when making the predictions and minimize the expected cost.
  - Only predict high-cost class when the model is very confident about the prediction.

Metrics for Performance Evaluation

	Actual Class		
Predicted Class		Positive	Negative
	Positive	True positive (TP)	False positive (FP)
	Negative	False negative (FN)	True negative (TN)

- Recall=$\frac{TP}{TP+FN}$, the fraction of relevant instances that were retrieved.
- Precision=$\frac{TP}{TP+FP}$, the fraction of relevant instances among the retrieved instances.
- F-measure is the harmonic mean between recall and precision.

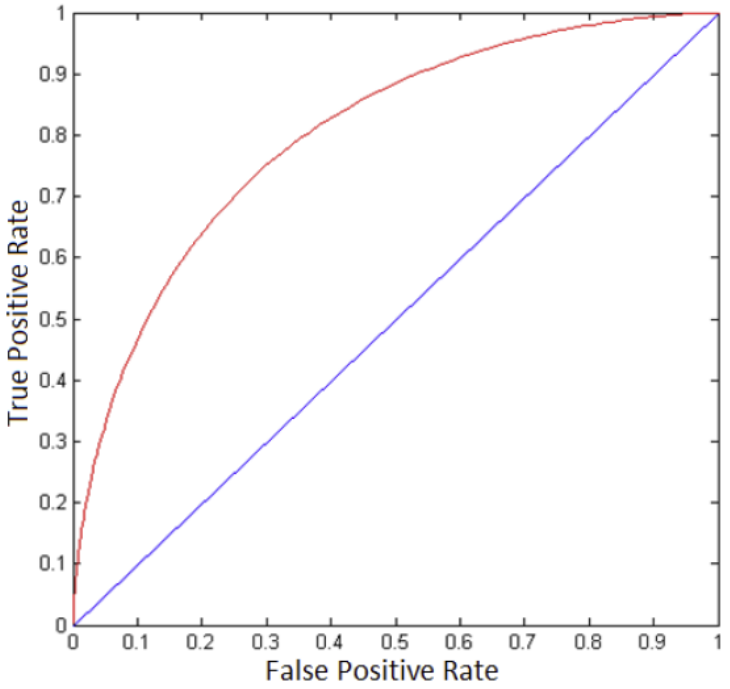
$$F-measure = \frac{2Recall \times Precision}{Recall + Precision} = \frac{2TP}{2TP + FP + FN}$$

ROC Curve

- An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds.
- This curve plots two parameters:
  - True positive rate on y-axis: $\mathrm{TPR} = \frac{TP}{TP+FN}$
  - False positive rate on x-axis: $\mathrm{FPR} = \frac{FP}{FP+TN}$
- Performance of each classifier represented as a point on the ROC curve.
  - Changing the threshold of algorithm changes the location of the point.
  - Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.

ROC Curve

- (TPR, FPR)
  - (0,0): declare everything to be negative class
  - (1,1): declare everything to be positive class
  - (1,0): Ideal/perfect classification
- Diagonal line represents random guessing. Points above the diagonal represent good classification results.
- AUC: Area Under the ROC Curve
  - Measures the entire two-dimensional area under the ROC curve
  - Provides an aggregate measure of performance across all possible classification thresholds.
  - Ideal: AUC=1; Random guess: AUC =0.5



How to Construct an ROC Curve

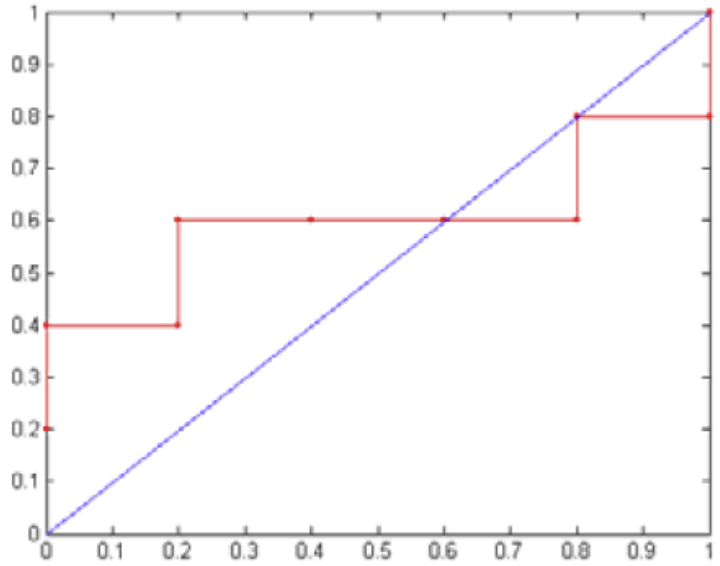
1. Use a classifier to produce posterior probability for each test instance P(+|A)
2. Sort the instances according to P(+|A) in decreasing order
3. Apply threshold at each unique value of P(+|A)
4. Count the number of TP, FP, TN, FN at each threshold
5. TPR = TP/(TP+FN); FPR = FP/(FP + TN)

Instance	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

How to Construct an ROC Curve

	Class	+	-	+	-	-	-	+	-	+	+	
Threshold >=		0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
	TP	5	4	4	3	3	3	3	2	2	1	0
	FP	5	5	4	4	3	2	1	1	0	0	0
	TN	0	0	1	1	2	3	4	4	5	5	5
	FN	0	1	1	2	2	2	2	3	3	4	5
→	TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
→	FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

ROC Curve



Instance	P(+|A)	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

Probabilistic Generative Models

Probabilistic Generative Models

- We now turn to a probabilistic approach to classification.
- Show how models with linear decision boundaries arise from simple assumptions about the distribution of the data.
- Adopt the generative approach: Solve the inference problem of estimating the class-conditional densities $p(x|C_k)$ for each class C_k.
- Infer the prior class probabilities $p(C_k)$.
- Use Bayes' theorem to find the class posterior probabilities:

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{p(x)}$$

where $p(x) = \sum_k p(x|C_k)p(C_k)$

Probabilistic Generative Approach: Two Classes

- Two-class case:

$$p(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(x|C_2)p(C_2)} = \frac{1}{1+e^{-a}} = \sigma(a)$$

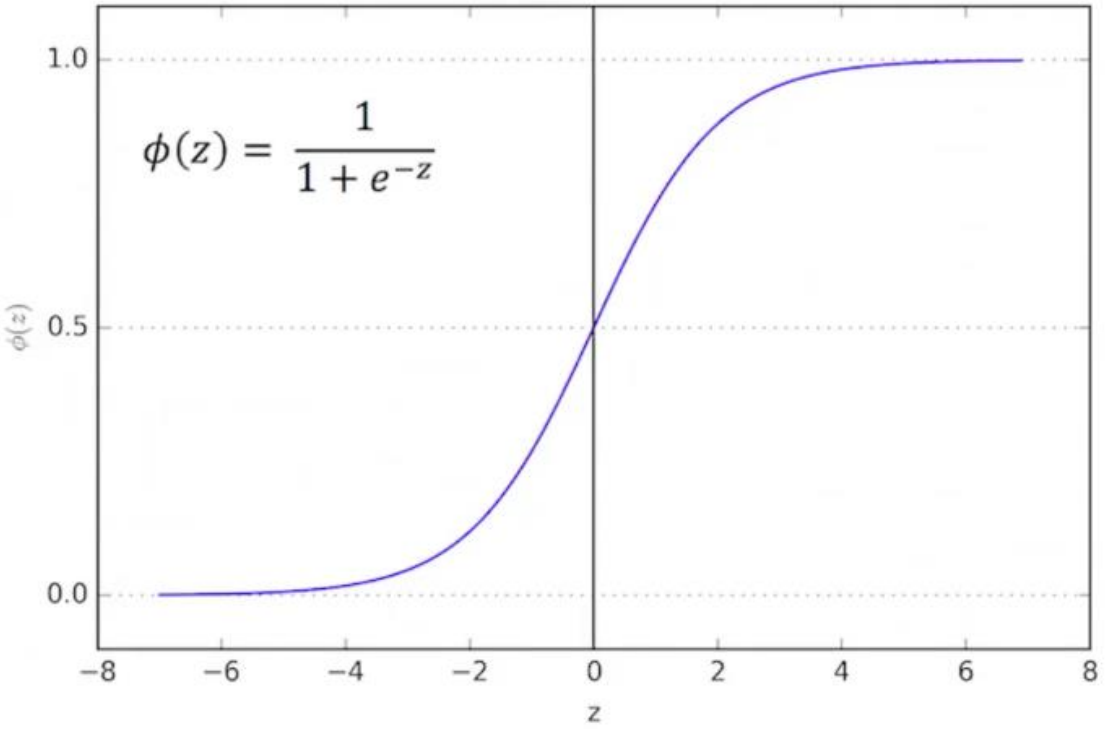
where

$$a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$$

$\sigma(a) = \frac{1}{1+e^{-a}}$ (Logistic Sigmoid Function)

Logistic Sigmoid Function

- It tends to 1 as $z \to \infty$
- It tends to 0 as $z \to -\infty$
- It is bounded between 0 and 1.



Probabilistic Generative Approach: Multiple Classes

- Multiple ($K > 2$) classes case:

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{\sum_{j=1}^{K} p(x|C_j)p(C_j)} = \frac{e^{a_k}}{\sum_{j=1}^{K} e^{a_j}}$$

where

$$a_k = \ln p(x|C_k)p(C_k)$$

$$\sigma(a) = \frac{e^{a_k}}{\sum_{j=1}^{K} e^{a_j}} \text{ (softmax function)}$$

Probabilistic Generative Models

- Let's assume the class-conditional densities are Gaussian with the same covariance matrix:

$$p(x|C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)}$$

- Two-class case: we can show the following result:

$$p(C_1|x) = \sigma(w^T x + \omega_0)$$

$$w = \Sigma^{-1}(\mu_1 - \mu_2)$$

$$\omega_0 = -\frac{1}{2}\mu_1^T \Sigma^{-1} \mu_1 + \frac{1}{2}\mu_2^T \Sigma^{-1} \mu_2 + ln \frac{p(C_1)}{p(C_2)}$$

- How to get the result?

Probabilistic Generative Models

$$p(C_1|x) = \sigma(a) = \frac{1}{1+e^{-a}} \qquad p(x|C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)}$$

$$a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$$

$$= \ln p(x|C_1) - \ln p(x|C_2) + \ln \frac{p(C_1)}{p(C_2)} \quad \text{(Replace } p(x|C_k)\text{)}$$

$$= -\frac{1}{2}(x-\mu_1)^T \Sigma^{-1}(x-\mu_1) + \frac{1}{2}(x-\mu_2)^T \Sigma^{-1}(x-\mu_2) + \ln \frac{p(C_1)}{p(C_2)}$$

Probabilistic Generative Models

$$a = -\frac{1}{2}(x-\mu_1)^T\Sigma^{-1}(x-\mu_1) + \frac{1}{2}(x-\mu_2)^T\Sigma^{-1}(x-\mu_2) + \ln\frac{p(C_1)}{p(C_2)}$$

$$= -\frac{1}{2}x^T\Sigma^{-1}x + \frac{1}{2}\mu_1{}^T\Sigma^{-1}x + \frac{1}{2}x^T\Sigma^{-1}\mu_1 - \frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1$$

$$+\frac{1}{2}x^T\Sigma^{-1}x - \frac{1}{2}\mu_2{}^T\Sigma^{-1}x - \frac{1}{2}x^T\Sigma^{-1}\mu_2 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + \ln\frac{p(C_1)}{p(C_2)}$$

$$= \boldsymbol{\mu_1{}^T\Sigma^{-1}x - \mu_2{}^T\Sigma^{-1}x - \frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + ln\frac{p(C_1)}{p(C_2)}}$$

$$= \underbrace{(\mu_1-\mu_2)^T\Sigma^{-1}}_{w^T}x \underbrace{-\frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + ln\frac{p(C_1)}{p(C_2)}}_{\omega_0}$$

The quadratic terms in x have cancelled, leading to a linear function of x.

Probabilistic Generative Models

- We have shown:

$$p(C_1|x) = \sigma(w^T x + \omega_0)$$

$$w = \Sigma^{-1}(\mu_1 - \mu_2)$$

$$\omega_0 = -\frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2^T\Sigma^{-1}\mu_2 + ln\frac{p(C_1)}{p(C_2)}$$

- Decision boundary is linear in input space:

$$p(C_1|x) = p(C_2|x) = 0.5$$

$$\Rightarrow \frac{1}{1+e^{-(w^T x+\omega_0)}} = 0.5 \Rightarrow w^T x + \omega_0 = 0$$

Probabilistic Generative Models

- Multiple ($K > 2$) classes case:

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{\sum_{j=1}^{K} p(x|C_j)p(C_j)} = \frac{e^{a_k}}{\sum_{j=1}^{K} e^{a_j}}$$

$$a_k = \ln p(x|C_k)p(C_k)$$

- Decision boundary will occur when two of the posterior probabilities are equal, which will be defined by linear functions of x:

$$p(C_k|x) = \frac{e^{w_k^T x + \omega_{k0}}}{\sum_j e^{w_j^T x + \omega_{j0}}}$$

$$w_k = \Sigma^{-1}\mu_k$$

$$\omega_{k0} = -\frac{1}{2}\mu_k^T \Sigma^{-1}\mu_k + \ln p(C_k)$$

Maximum Likelihood Solution

- Now we have a parametric functional form for the class-conditional densities:

$$p(x|C_k) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu_k)^T \Sigma^{-1}(x-\mu_k)}$$

- We can estimate the parameters and the prior class probabilities using maximum likelihood.
  - The case of two classes, each having a Gaussian class-conditional density with a shared covariance matrix.
  - Training data: $\{x_n, y_n\}, n = 1, \ldots, N$

$y_n = 1$ denotes class C_1 (N_1 data samples);

$y_n = 0$ denotes class C_2 (N_2 data samples);

Prior class probability: $p(C_1) = \gamma,\ p(C_2) = 1 - \gamma$

Maximum Likelihood Solution

- For a data point x_n from class C_1, we have $y_n = 1$ and therefore:

$$p(x_n, C_1) = p(x|C_1)p(C_1) = \gamma \mathcal{N}(x_n|\mu_1, \Sigma)$$

- For a data point x_n from class C_2, we have $y_n = 0$ and therefore:

$$p(x_n, C_2) = p(x|C_2)p(C_2) = (1-\gamma)\mathcal{N}(x_n|\mu_2, \Sigma)$$

- Assuming observations are drawn independently, the likelihood function is given as follows, where $y = (y_1, \dots, y_N)^T$.

$$p(y|\gamma, \mu_1, \mu_2, \Sigma) = \prod_{\boldsymbol{n=1}}^{\boldsymbol{N}} [p(x_n, C_1)]^{\boldsymbol{y_n}} [p(x_n, C_2)]^{1-y_n}$$

$$= \prod_{\boldsymbol{n=1}}^{\boldsymbol{N}} [\gamma N(x_n|\mu_1, \Sigma)]^{\boldsymbol{y_n}} [(1-\gamma)N(x_n|\mu_2, \Sigma)]^{1-y_n}$$

Maximum Likelihood Solution

- We want to find the values of the parameters that maximize the likelihood function, i.e., fit a model that best describes the observed data.

$$p(y|\gamma, \mu_1, \mu_2, \Sigma) = \prod_{n=1}^{N} [\gamma \mathcal{N}(x_n|\mu_1, \Sigma)\,]^{y_n} [(1-\gamma)\mathcal{N}(x_n|\mu_2, \Sigma)\,]^{1-y_n}$$

- As usual, we consider the log of the likelihood:

$$\ln p(y|\gamma, \mu_1, \mu_2, \Sigma) = \sum_{n=1}^{N} [y_n \ln \gamma + y_n \ln \mathcal{N}(x_n|\mu_1, \Sigma)$$
$$+(1-y_n)\ln(1-\gamma) + (1-y_n)\ln \mathcal{N}(x_n|\mu_2, \Sigma)]$$

Maximum Likelihood Solution: Parameter γ

$$\ln p(y|\gamma, \mu_1, \mu_2, \Sigma) = \sum_{n=1}^{N} [y_n \ln \gamma + y_n \ln \mathcal{N}(x_n|\mu_1, \Sigma)$$
$$+(1 - y_n) \ln(1 - \gamma) + (1 - y_n) \ln \mathcal{N}(x_n|\mu_2, \Sigma)]$$

- We first maximize the log-likelihood with respect to γ (set derivate to 0).

$$\gamma = \frac{1}{N} \sum_{n=1}^{N} y_n = \frac{N_1}{N} = \frac{N_1}{N_1 + N_2}$$

- The maximum likelihood estimate of γ is the fraction of points in class C_1.
- For multi-class: ML estimate for $p(C_k)$ is given by the fraction of points in the training set in C_k.

Maximum Likelihood Solution: Parameter μ

$$\ln p(D|\gamma, \mu_1, \mu_2, \Sigma) = \sum_{n=1}^{N} [y_n \ln \gamma + y_n \ln \mathcal{N}(x_n|\mu_1, \Sigma)$$
$$+(1 - y_n) \ln(1 - \gamma) + (1 - y_n) \ln \mathcal{N}(x_n|\mu_2, \Sigma)]$$

- We then maximize the log-likelihood with respect to μ_1 (set derivate to 0).

$$\mu_1 = \frac{1}{N_1} \sum_{n=1}^{N} y_n x_n$$

- The maximum likelihood estimate of μ_1 is the sample mean of all inputs x_n in class C_1.
- Similarly, the maximum likelihood estimate of μ_2 is given by

$$\mu_2 = \frac{1}{N_2} \sum_{n=1}^{N} (1 - y_n) x_n$$

Maximum Likelihood Solution: Parameter $\boldsymbol{\Sigma}$

- Maximize the log-likelihood with respect to the covariance matrix Σ (set derivate to 0), we obtain the estimate Σ_{ML}

$$\Sigma_{ML} = \frac{N_1}{N} S_1 + \frac{N_2}{N} S_2$$

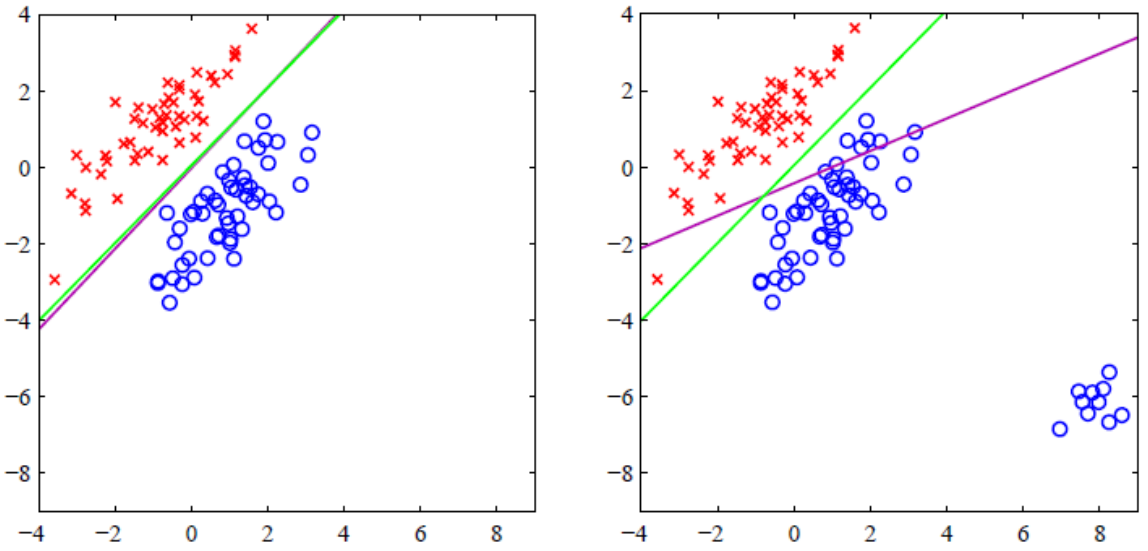
where

$$S_1 = \frac{1}{N_1} \sum_{x_n \in C_1} (x_n - \mu_1)(x_n - \mu_1)^T \quad S_2 = \frac{1}{N_2} \sum_{x_n \in C_2} (x_n - \mu_2)(x_n - \mu_2)^T$$

- The maximum likelihood estimate of the covariance is given by the weighted average of the sample covariance matrices associated with each of the classes.
- The results extend to K classes.

Summary So Far

- We assumed $p(x|y=1) \sim N(\mu_1, \Sigma)$ and $p(x|y=0) \sim N(\mu_2, \Sigma)$, and two class-probabilities $p(y=1)$ and $p(y=0)$.
- This is called a generative model, as we have written down a full joint model over the data.
- We saw that violations of the model assumption can lead to "bad" decision boundaries.



Figures from Bishop PRML, 44a and b

Probabilistic Generative Models: Parameters

- How many parameters did we estimate to fit Gaussian class-conditional densities (the generative approach)?
- Suppose d is the dimension of the input space.

$$p(C_1) \Rightarrow 1$$

$$\text{2 mean vectors} \Rightarrow 2d$$

$$\Sigma \Rightarrow d + \frac{d^2 - d}{2} = \frac{d^2 + d}{2}$$

$$Total = 1 + 2d + \frac{d^2 + d}{2} = O(d^2)$$

Logistic Regression

Logistic Regression

$$P(C_1|x) = \sigma(w^T x + \omega_0) = f(x)$$

- We use maximum likelihood to determine the parameters of the logistic regression model.

$$\{x_n, y_n\}, n = 1, \dots, N$$

$y_n = 1$ denotes class C_1; $y_n = 0$ denotes class C_2

- We want to find the values of w that maximize the posterior probabilities associated to the observed data.
- Likelihood function:

$$L(w) = \prod_{n=1}^{N} p(C_1|x_n)^{y_n}(1 - p(C_1|x_n))^{1-y_n}$$

$$= \prod_{n=1}^{N} f(x_n)^{y_n}(1 - f(x_n))^{1-y_n}$$

Logistic Regression

$$P(C_1|x) = \sigma(w^T x + \omega_0) = f(x)$$

- We consider the negative logarithm of the likelihood (Cross Entropy):

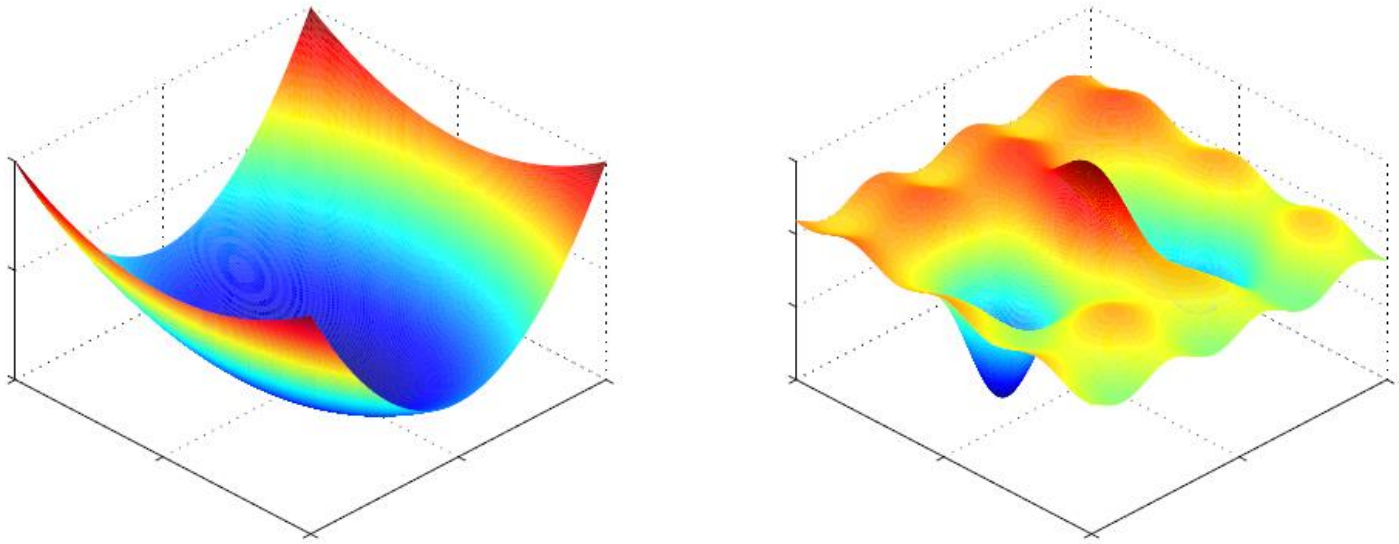
$$\varepsilon(w) = -\ln L(w) = -\ln \prod_{n=1}^{N} p(C_1|x_n)^{y_n} \left(1 - p(C_1|x_n)\right)^{1-y_n}$$

$$= -\sum_{n=1}^{N} (y_n \ln f(x_n) + (1 - y_n) \ln(1 - f(x_n)))$$

- Thus:

$$\max L(w) = \min \varepsilon(w)$$

The Cost-function for Logistic Regression is Convex.



- Fact: The negative log-likelihood is convex - this makes life much easier.
- There are no local minima to get stuck in, and there is good optimization techniques for convex problems.

Logistic Regression: Gradient

$$P(C_1|x) = \sigma(w^T x + \omega_0) = f(x)$$

- We compute the derivative of the error function with respect to w.

$$\frac{\partial \varepsilon(w)}{\partial w} = \frac{\partial}{\partial w}[-\ln \prod_{n=1}^{N} p(C_1|x_n)^{y_n}\big(1 - p(C_1|x_n)\big)^{1-y_n}]$$

- The derivative of the logistic sigmoid function:

$$\frac{\partial}{\partial w}\sigma(a) = \frac{\partial}{\partial a}\frac{1}{1+e^{-a}} = \frac{e^{-a}}{(1+e^{-a})^2} = \frac{1}{1+e^{-a}}\frac{e^{-a}}{(1+e^{-a})}$$
$$= \frac{1}{1+e^{-a}}\left(1 - \frac{1}{1+e^{-a}}\right) = \sigma(a)(1-\sigma(a))$$

Logistic Regression

- The derivative of the error function with respect to w is:

$$\nabla_w \, \varepsilon(w) = \sum_{n=1}^{N} (f(x_n) - y_n) x_n$$

Homework: show the detailed steps to calculate this derivation.

- Then, the parameter can be updated by:

$$w^{t+1} \coloneqq w^t - \eta \nabla_w \, \varepsilon(w)$$

Probabilistic Discriminative Models: parameters

- Two-class case:

$$P(C_1|x) = \sigma(w^T x + \omega_0) = f(x)$$

$$P(C_2|x) = 1 - P(C_1|x)$$

- This model is known as Logistic Regression.
- Assuming $x \in \mathbb{R}^d$, how many parameters do we need to estimate?

$$d + 1$$

Summary So Far

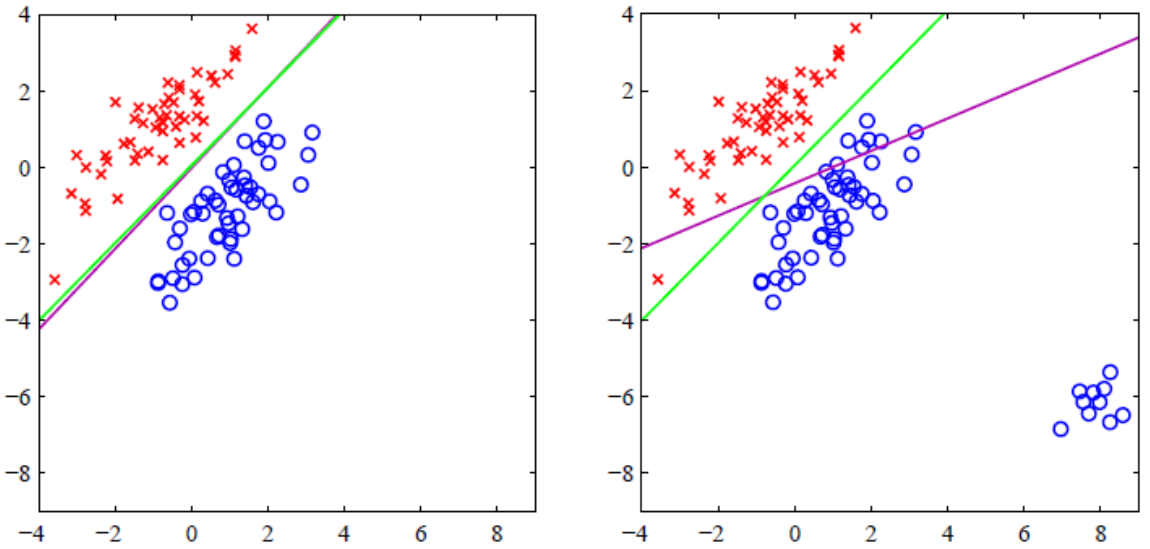
- From the previous introduction, we know that
$$P(y = 1|x) = \sigma(a(x))$$
where $\sigma(a) = 1/(1 + \exp(-a))$ and $a(x) = w^T x + \omega_0$.
- Notation is simpler if we use 0 and 1 as class labels, so we define $y_n = 1$ as the label for the positive class, and $y_n = 0$ a label for the negative class.
- In other words, $y|x \sim Bernoulli(\sigma(a(x))$.
- The parameters of $a(x) = w^T x + \omega_0$ can be learnt by minimizing the negative log-likelihood:
$$L(w) = -\sum_n \{y_n \log p(y_n|x_n, w) + (1 - y_n) \log(1 - p(y_n|x_n, w))\}$$

Summary So Far

- This is a discriminative approach to classification, as we only model the labels, and not the inputs.
- Decision rule and function shape of $p(y|x)$ will be the same for the generative and the discriminative model, but the parameters were obtained differently.

Maximum Likelihood Estimation of Logistic Regression

- Logistic regression is a much better algorithm than the algorithms we discussed last week.
- Need to optimize log-likelihood numerically.
- People typically minimize the negative log-likelihood $\mathcal{L}$ rather than maximizing the log-likelihood.
- To numerically minimize the negative log-likelihood, we need its gradient (and maybe its hessian).



Figures from Bishop PRML, 44a and b

Multiclass Logistic Regression

- Multiclass case:

$$p(C_k|x) = \frac{e^{w_k^T x+\omega_{k0}}}{\sum_j e^{w_j^T x+\omega_{j0}}} = f_k(x)$$

- We use maximum likelihood to determine the parameters:

$$\{x_n, y_n\}, n = 1, \dots, N$$

$y_n = (0, \dots, 1, \dots, 0)$ denotes class C_k

- We want to find the values of $w_1, \dots, w_k$ that maximize the posterior probabilities associated to the observed data likelihood function:

$$L(w_1, \dots, w_k) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(C_k|x_n)^{y_{nk}} = \prod_{n=1}^{N}\prod_{k=1}^{K} f_k(x_n)^{y_{nk}}$$

Multiclass Logistic Regression

$$L(w_1, \dots, w_k) = \prod_{n=1}^{N} \prod_{k=1}^{K} p(C_k|x_n)^{y_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} f_k(x_n)^{y_{nk}}$$

- Consider the negative logarithm of the likelihood (cross-entropy error):

$$\varepsilon(w_1, \dots, w_k) = -\ln L(w_1, \dots, w_k) = -\sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \ln f_k(x_n)$$

$$\min_{w_j} \varepsilon\big((w_j)\big)$$

- The gradient of the error function w.r.t one of the parameter vectors:

$$\frac{\partial}{\partial w_j} \varepsilon(w_1, \dots, w_k) = \frac{\partial}{\partial w_j} [-\sum_{n=1}^{N} \sum_{k=1}^{K} y_{nk} \ln f_k(x_n)]$$

Multiclass Logistic Regression

$$\frac{\partial}{\partial w_j}\varepsilon(w_1, \dots, w_k) = \frac{\partial}{\partial w_j}[-\sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk} \ln f_k(x_n)]$$

- The derivatives of the softmax function:

$$\frac{\partial}{\partial a_k} f_k = \frac{\partial}{\partial a_k}\frac{e^{a_k}}{\sum_j e^{a_j}} = \frac{e^{a_k}\sum_j e^{a_j} - e^{a_k}e^{a_k}}{\left(\sum_j e^{a_j}\right)^2} = f_k - f_k^2 = f_k(1 - f_k)$$

- Thus, for $j \neq k$

$$\frac{\partial}{\partial a_j} f_k = \frac{\partial}{\partial a_j}\frac{e^{a_k}}{\sum_j e^{a_j}} = \frac{-e^{a_k}e^{a_j}}{\left(\sum_j e^{a_j}\right)^2} = -f_k f_j$$

- Compact expression (I_{kj} are the elements of the identity matrix)

$$\frac{\partial}{\partial a_j} f_k = f_k(I_{kj} - f_j)$$

Multiclass Logistic Regression

$$\nabla_{w_j}\varepsilon(w_1, \dots, w_k) = \sum_{n=1}^{N}\left(f_{nj} - y_{nj}\right)x_n$$

- It can be shown that ε is a convex function of w. Thus, it has a unique minimum.
- For a batch solution, we can use the Newton-Raphson optimization technique.
- Online solution (SGD):

$$w_j^{t+1} = w_j^t - \eta\nabla_{w_j}\varepsilon_n(w) = w_j^t - \eta\left(f_{nj} - y_{nj}\right)x_n$$

Principle Component Analysis (PCA)

Dimensionality Reduction

- Many dimensions are often interdependent (correlated).
  - Solution 1: reduce the dimensionality of problems;
  - Solution 2: transform interdependent coordinates into significant and independent ones.
- PCA transforms the original input space into a lower dimensional space, by constructing dimensions that are linear combinations of the given features.

Eigenvalues and Eigenvectors

- For an $n \times n$ square matrix A, e is an eigenvector with eigenvalue λ if:

$$Ae = \lambda e$$

or

$$(A - \lambda I)e = 0$$

- If $(A - \lambda I)$ is invertible, the only solution is $e = 0$ (trivial).

Eigenvalues and Eigenvectors

- For non-trivial solutions

$$\det(A - \lambda I) = 0$$

- This is called the “characteristic polynomial”.
- Solutions are not unique because if e is an eigenvector ae is also an eigenvector.

Eigenvalues and Eigenvectors: Example

- For a 2×2 matrix:

$$A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}$$

- Given

$$\det(A - \lambda I) = \begin{vmatrix} a_{11} - \lambda & a_{12} \\ a_{21} & a_{22} - \lambda \end{vmatrix} = (a_{11} - \lambda)(a_{22} - \lambda) - a_{12}a_{21} = 0$$

- We get:

$$a_{11}a_{22} - a_{12}a_{21} - (a_{11} + a_{22})\lambda + \lambda^2 = 0$$
$$1 \cdot 4 - 2 \cdot 2 - (1 + 4)\lambda + \lambda^2 = 0$$
$$\Rightarrow \lambda = 0 \; and \; \lambda = 5$$

Eigenvalues and Eigenvectors: Example

- The eigenvector for the first eigenvalue $\lambda = 0$ is:

$$\begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x + 2y \\ 2x + 4y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- One solution for both equations is $x = 2;\ y = -1.$
- The eigenvector for the first eigenvalue $\lambda = 5$ is:

$$\begin{bmatrix} -4 & 2 \\ 2 & -1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} -4x + 2y \\ 2x - y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

- One solution for both equations is $x = 1;\ y = 2.$

Eigenvalues and Eigenvectors: Properties

- The product of the eigenvalues is the determinant of A: det(A)
- The sum of the eigenvalues = trace(A)
- The eigenvectors are pairwise orthogonal

Principal Component Analysis

- It is also called Karhunen-Loeve transformation
  - PCA transforms the original input space into a lower dimensional space, by constructing dimensions that are linear combinations of the given features;
  - The objective is to consider independent dimensions along which data have largest variance (i.e., greatest variability).

Principal Component Analysis

- PCA involves a linear algebra procedure that transforms a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components;
- The first principal component accounts for as much of the variability in the data as possible;
- Each succeeding component (orthogonal to the previous ones) accounts for as much of the remaining variability as possible.

Principal Component Analysis

- So: PCA finds n linearly transformed components, $s_1, s_2, \ldots, s_n$, so that they explain the maximum amount of variance;
- We can define PCA in an intuitive way using a recursive formulation.

Principal Component Analysis

- Suppose data are first centered at the origin (i.e., their mean is 0)
- We define the direction of the first principal component, say, w_1 as follows:

$$w_1 = \arg\max_{\|w\|=1} E[(w^T x)^2]$$

  where w_1 is of the same dimensionality d as the data vector x.
- Thus: the first principal component is the projection on the direction along which the variance of the projection is maximized.

Principal Component Analysis

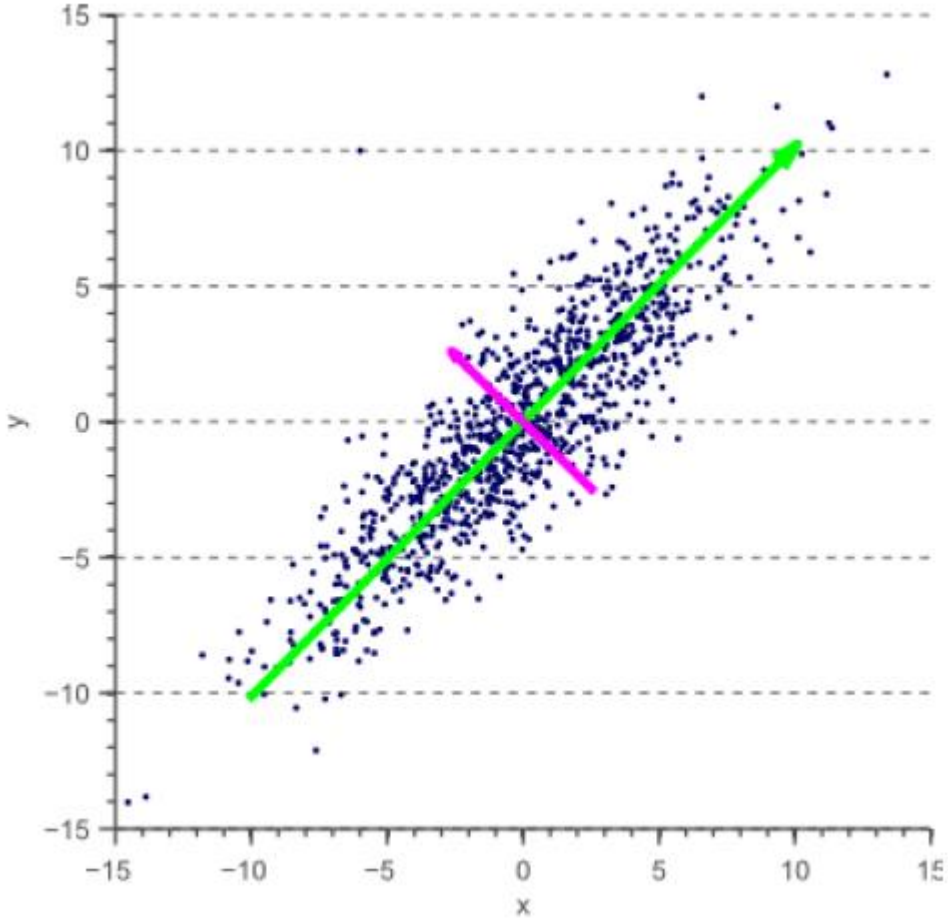
- Having determined the first $k-1$ principal components, the k^{th} principal component is determined as the principal component of the data residual:

$$w_k = \arg\max_{\|w\|=1} E\left[w^T\left(x - \sum_{i=1}^{k-1} w_i w_i^T x\right)\right]^2$$

- The principal components are then given by:

$$s_i = w_i^T x$$

Illustration of Principal Component Analysis



Green: first principal component of a two-dimensional dataset;
Pink: second principal component

PCA: Geometric interpretation

PCA rotates the data (centered at the origin) in such a way that the maximum variability is visible (i.e., aligned with the axes.)

PCA: How to compute the principal components

- Let w be the direction of the first principal component, with $\|w\| = 1$.
- $s_i = w^T x_i$ is the projection of x_i along w, and we can consider it as the new coordinate in the 1D subspace of the first principal component.
- $\bar{s} = \frac{1}{N}\sum_{i=1}^{N} s_i = \frac{1}{N}\sum_{i=1}^{N} w^T x_i$
- Variance of data along w:

$$\frac{1}{N}\sum_{i=1}^{N}(s_i - \bar{s})^2 = \frac{1}{N}\sum_{i=1}^{N}\left(w^T x_i - \frac{1}{N}\sum_{j=1}^{N} w^T x_j\right)^2$$

- The variance of data along direction w can be represented with the sample covariance matrix: $w^T \Sigma w$. How?

PCA: How to compute the principal components

$$\frac{1}{N}\sum_{i=1}^{N}(s_i-\bar{s})^2=\frac{1}{N}\sum_{i=1}^{N}\left(w^Tx_i-\frac{1}{N}\sum_{j=1}^{N}w^Tx_j\right)^2$$

$$=\frac{1}{N}\sum_{i=1}^{N}\left[w^T(x_i-\frac{1}{N}\sum_{j=1}^{N}x_j)\right]^2$$

$$=\frac{1}{N}\sum_{i=1}^{N}[w^T(x_i-\bar{x})]^2$$

$$=\frac{1}{N}\sum_{i=1}^{N}[w^T(x_i-\bar{x})(x_i-\bar{x})^Tw]$$

$$=w^T\left\{\sum_{i=1}^{N}[(x_i-\bar{x})(x_i-\bar{x})^T]\right\}w=w^T\Sigma w$$

Sample covariance matrix

Unbiased estimation of the covariance:

$$var(X)=\frac{1}{N-1}\sum_{n=1}^{N}(X_n-\bar{X})^2$$

$$Cov(X,Y)=\frac{1}{N-1}\sum_{n=1}^{N}(X_n-\bar{X})(Y_n-\bar{Y})$$

PCA: How to compute the principal components

- Our objective is to find w such that $w = \arg\max_{w} w^T \Sigma w$ with constraint $w^T w = 1$.
- By introducing one Lagrange multiplier λ, we obtain the unconstrained optimization problem:

$$w = \arg\max_{w} [w^T \Sigma w - \lambda(w^T w - 1)]$$

- Take the derivative with respect to w and set to 0, we can get

$$2\Sigma w - 2\lambda w = 0$$
$$\Rightarrow \Sigma w = \lambda w$$

- Therefore, the problem can be reduced to an eigenvalue problem.

PCA: How to compute the principal components

The solution w is the eigenvector of Σ corresponding to the largest eigenvalue λ:

$$\Sigma w = \lambda w$$

Summary of PCA

- The computation of w_i is accomplished by solving an eigenvalue problem for the sample covariance matrix (assuming data have 0 mean):

$$\Sigma = E[xx^T]$$

- The eigenvector associated with the largest eigenvalue corresponds to the first principal component; the eigenvector associated with the second largest eigenvalue corresponds to the second principal component; and so on...
- Thus: w_i is the eigenvector of Σ that corresponds to the i^{th} largest eigenvalue of Σ.

PCA: In Practice

- The basic goal of PCA is to reduce the dimensionality of the data. Thus, one usually chooses:

$$n \ll d$$

- But how do we select the number of components n?

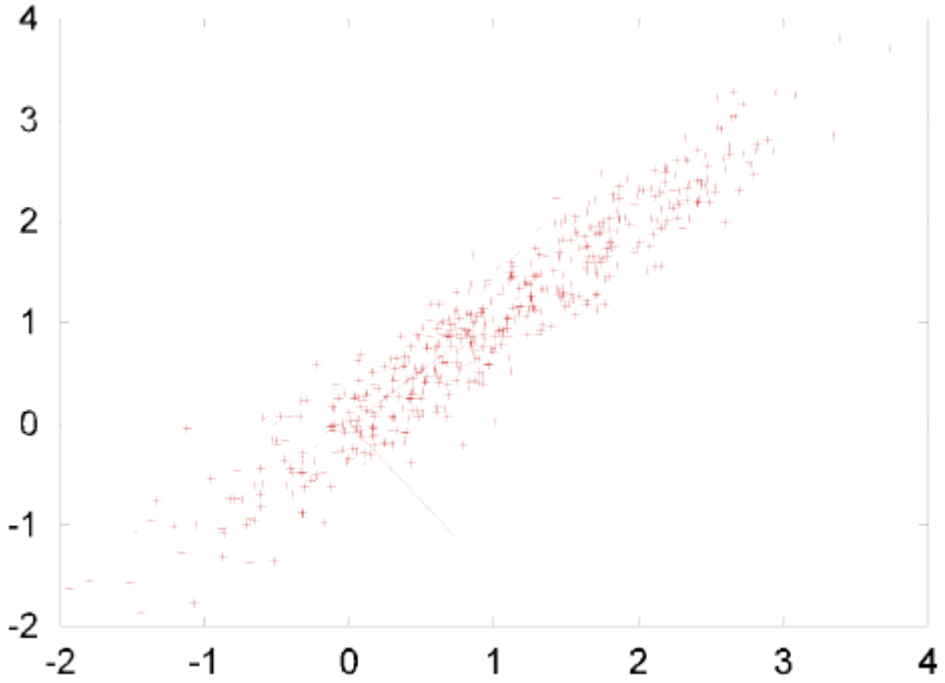
Determining the number of component

- Plot the eigenvalues: each eigenvalue is related to the amount of variation explained by the corresponding axis (eigenvector);
- If the points on the graph tend to level out (show an “elbow” shape), these eigenvalues that are usually close enough to zero can be ignored;
- In general: Limit the variance accounted for.

Determining the number of component

- A typical eigenvalue spectrum and its division into two orthogonal subspaces.
- Critical information lies in low dimensional subspaces.

Determining the number of component



$$\lambda_1 = 1.98, \lambda_2 = 0.05$$

Determining the number of component

- $x_i \in \mathbb{R}^d, i = 1, \ldots, N$
- $w_1, w_2, \ldots, w_d$: d eigenvectors (principal component directions)
- $\|w_i\| = 1$ (the w_is are orthogonal vectors)
- Representation of x_i in eigenvector space:

$$z_i = (w_1^T x_i)w_1 + (w_2^T x_i)w_2 + \ldots + \left(w_d^T x_i\right)w_d$$

- Suppose we retrain the first k principal components:

$$z_i^k = (w_1^T x_i)w_1 + (w_2^T x_i)w_2 + \ldots + \left(w_k^T x_i\right)w_k$$

- Then,

$$z_i - z_i^k = \left(w_{k+1}^T x_i\right)w_{k+1} + \cdots + \left(w_d^T x_i\right)w_d$$

Determining the number of component

$$(z_i - z_i^k)^T (z_i - z_i^k) =$$

$$\left[(w_{k+1}^T x_i) w_{k+1} + \ldots + (w_d^T x_i) w_d\right]^T \left[(w_{k+1}^T x_i) w_{k+1} + \ldots + (w_d^T x_i) w_d\right]$$

$$= w_{k+1}^T (w_{k+1}^T x_i)^2 w_{k+1} + \ldots + w_d^T (w_d^T x_i)^2 w_d$$

(note $w_i^T w_j = 0 \forall i \neq j$ since w_i and w_j are orthogonal vectors)

$$= (w_{k+1}^T x_i)^2 w_{k+1}^T w_{k+1} + \ldots + (w_d^T x_i)^2 w_d^T w_d$$

$$= (w_{k+1}^T x_i)^2 + \ldots + (w_d^T x_i)^2$$

$$= (w_{k+1}^T x_i)(x_i^T w_{k+1}) + \ldots + (w_d^T x_i)(x_i)^T w_d$$

$$= w_{k+1}^T (x_i x_i^T) w_{k+1} + \ldots + w_d^T (x_i x_i^T) w_d$$

Determining the number of component

$$\frac{1}{N}\sum_{i=1}^{N}(z_i - z_i^k)^T(z_i - z_i^k) =$$

Mean Square Error

$$\frac{1}{N}\sum_{i=1}^{N}\left[w_{k+1}^T(x_i x_i^T)w_{k+1} + \ldots + w_d^T(x_i x_i^T)w_d\right]$$

$$= w_{k+1}^T\left[\frac{1}{N}\sum_{i=1}^{N}(x_i x_i^T)\right]w_{k+1} + \ldots + w_d^T\left[\frac{1}{N}\sum_{i=1}^{N}(x_i x_i^T)\right]w_d$$

$$= w_{k+1}^T\Sigma w_{k+1} + \ldots + w_d^T\Sigma w_d$$

(Note: $\Sigma w_{k+1} = \lambda_{k+1}w_{k+1}, \ldots, \Sigma w_d = \lambda_d w_d$)

$$= w_{k+1}^T\lambda_{k+1}w_{k+1} + w_d^T\lambda_d w_d$$

$$= \lambda_{k+1} + \ldots + \lambda_d$$

Determining the number of component

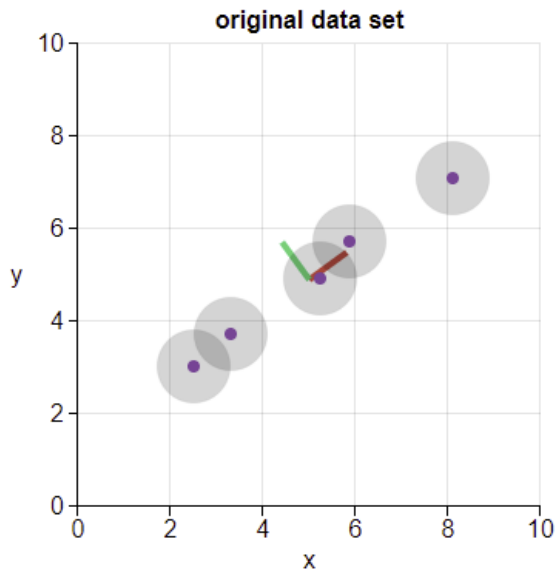
$$\frac{1}{N}\sum_{i=1}^{N}\left(z_i - z_i^k\right)^T\left(z_i - z_i^k\right) = \lambda_{k+1} + \cdots + \lambda_d$$

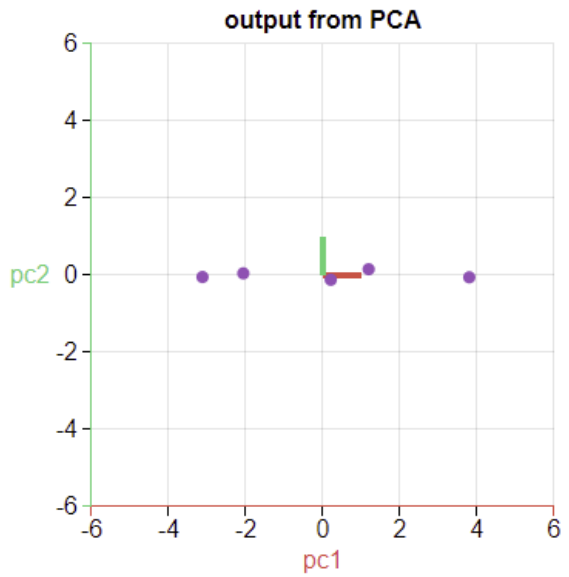
- The mean square error of the truncated representation is equal to the sum of the remaining eigenvalues.
- In general: choose k so that 90-95% of the variance of the data is captured, which is defined as the cumulative explained variance or cumulative energy content.

PCA: Steps

- Input: data matrix
- Output: a set of transformed coordinates
- Steps:
  1. Calculate the empirical mean, and get the covariance matrix
  2. Find the eigenvectors and eigenvalues of the covariance matrix
  3. Rearrange the eigenvectors and eigenvalues
  4. Compute the cumulative explained variance for each eigenvector
  5. Select a subset of the eigenvectors as basis vectors

PCA: Illustration Example 1





PCA is useful for eliminating dimensions. Below, we've plotted the data along a pair of lines: one composed of the x-values and another of the y-values.

If we're going to only see the data along one dimension, though, it might be better to make that dimension the principal component with most variation. We don't lose much by dropping PC2 since it contributes the least to the variation in the data set.

Illustration example with two dimensions. Source: https://setosa.io/ev/principal-component-analysis/

PCA: Illustration Example 2

Try it interactively through the link below to better understand PCA.

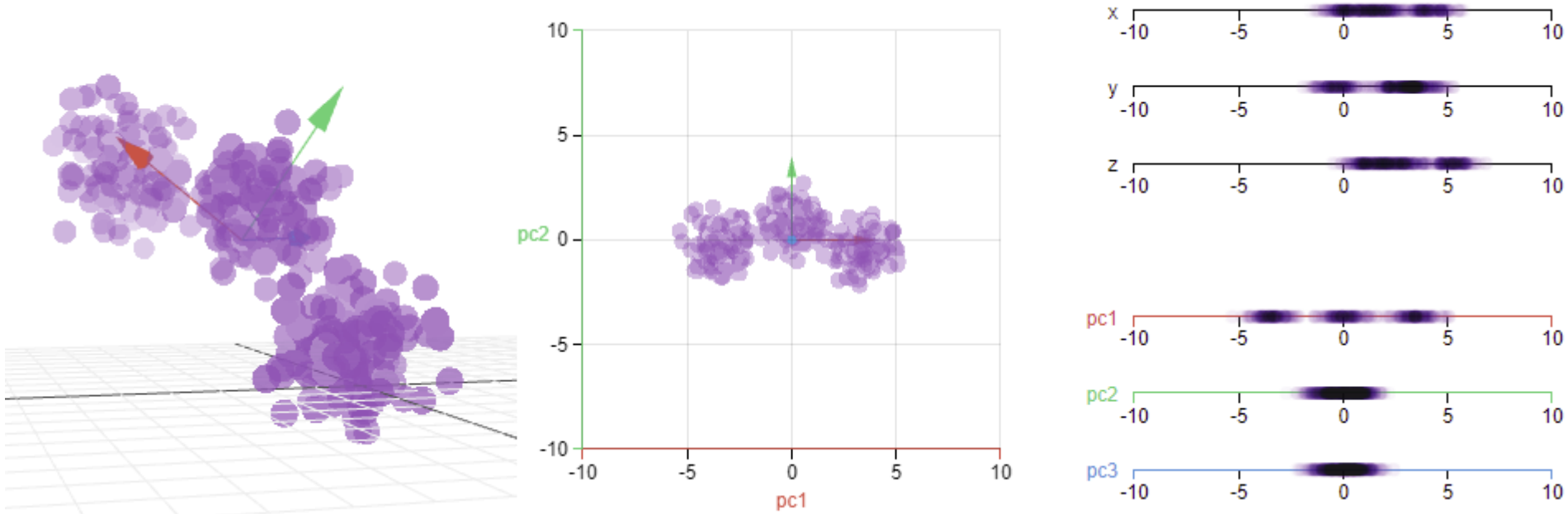


Illustration example with three dimensions. Source: https://setosa.io/ev/principal-component-analysis/

PCA: Advantages

- Optimal linear dimensionality reduction technique in the mean-square sense;
- Reduce the curse-of-dimensionality;
- Computational overhead of subsequent processing stages is reduced;
- Noise may be reduced;
- A projection into a subspace of a very low dimensionality, e.g. two dimensions, is useful for visualizing the data.

Readings

1. https://see.stanford.edu/materials/aimlcs229/cs229-notes1.pdf
2. https://web.stanford.edu/~jurafsky/slp3/5.pdf

Summary of Today's Lecture

- Evaluation Metrics for Classification
- Probabilistic Generative Models
- Logistic Regression
- Principal Component Analysis