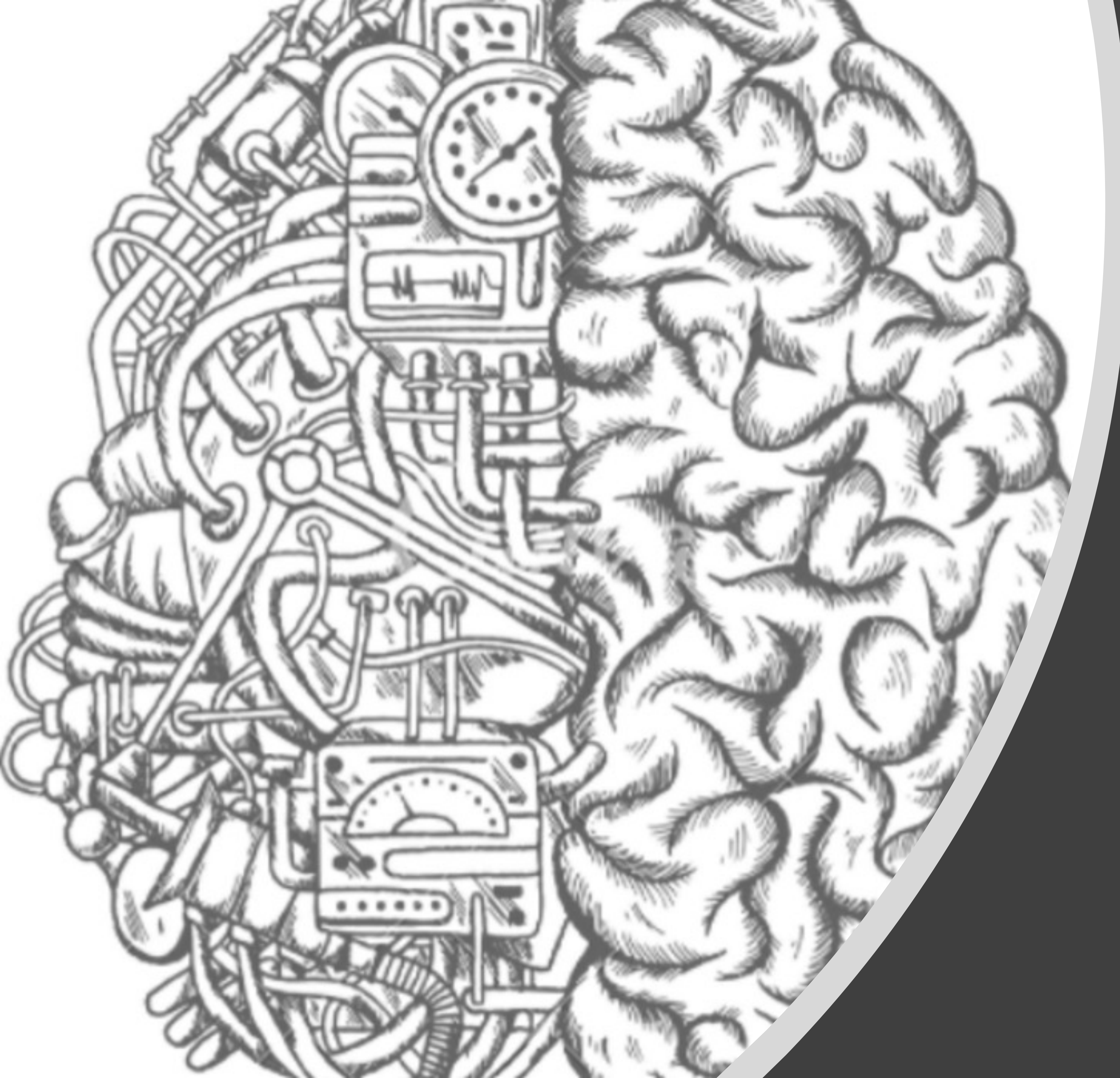


Machine Learning: Models and Applications

Lecture 7

Lecturer: Xinchao Wang

Outline

- Linear Regression
  - Regression technique
- Logistic Regression (called regression, but in fact, classification)
  - Classification technique
- Gradient Descent/Ascent
  - Descent -> min.
  - Ascent -> max.
  - Optimization technique
- Perceptron
  - Classification technique

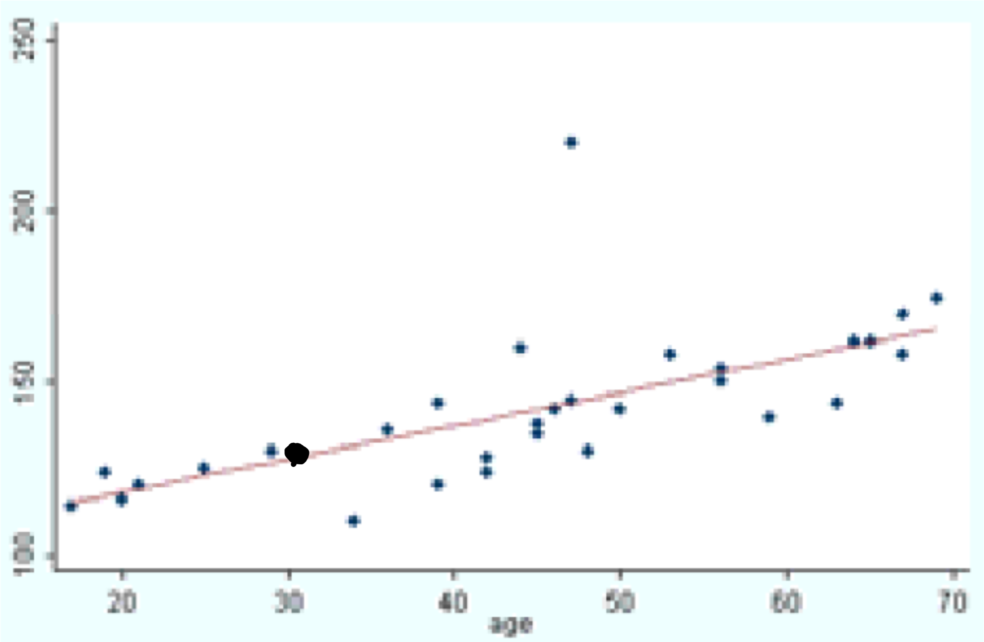
Linear Regression

Simple Linear Regression

- How does a single variable of interest relate to another (single) variable?
  - Y = outcome variable (response, dependent...)
  - X = explanatory variable (predictor, feature, independent...)
- Data: n pairs of continuous observations $(X_1,Y_1) \ldots (X_n,Y_n)$

Example

- How does systolic blood pressure (SBP) relate to age?

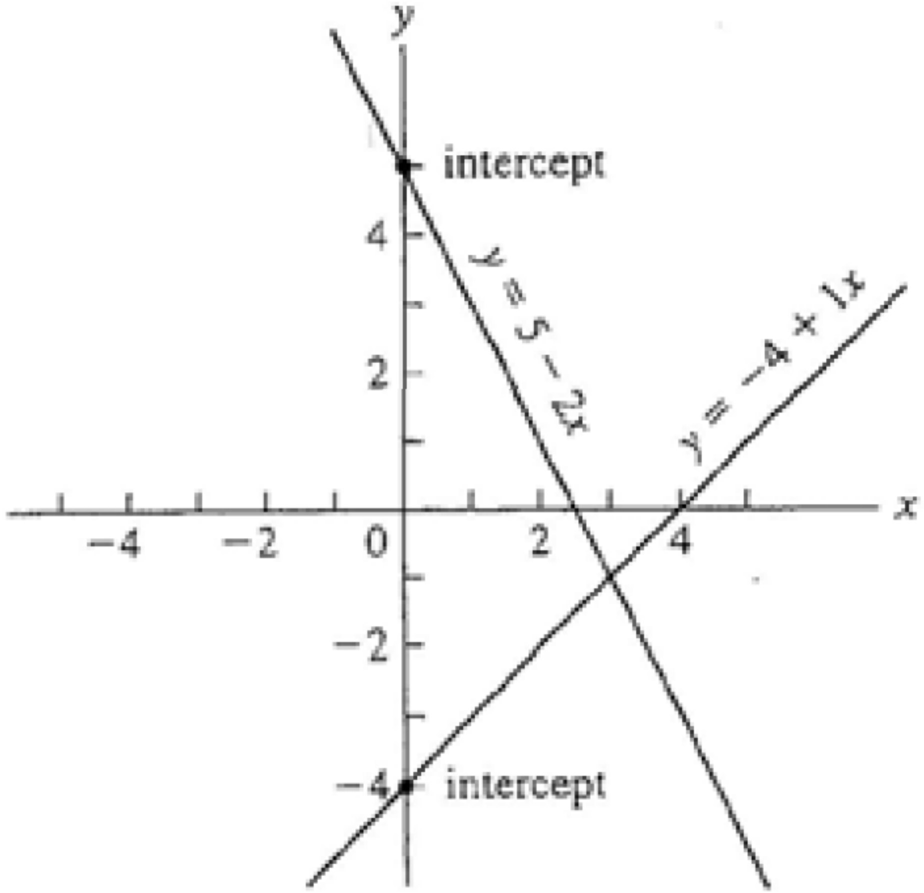


- Graph suggests that Y relates to X in an approximately linear way

Regression: Step by Step

1. Assume a linear model: $Y = \beta_0 + \beta_1 X$
2. Find the line which "best" fits the data, i.e. estimate parameters β_0 and β_1
3. Does variation in X help describe variation in Y ?
4. Check assumptions of model
5. Draw inferences and make predictions

Straight-line Plots



Assumptions of Linear Regression

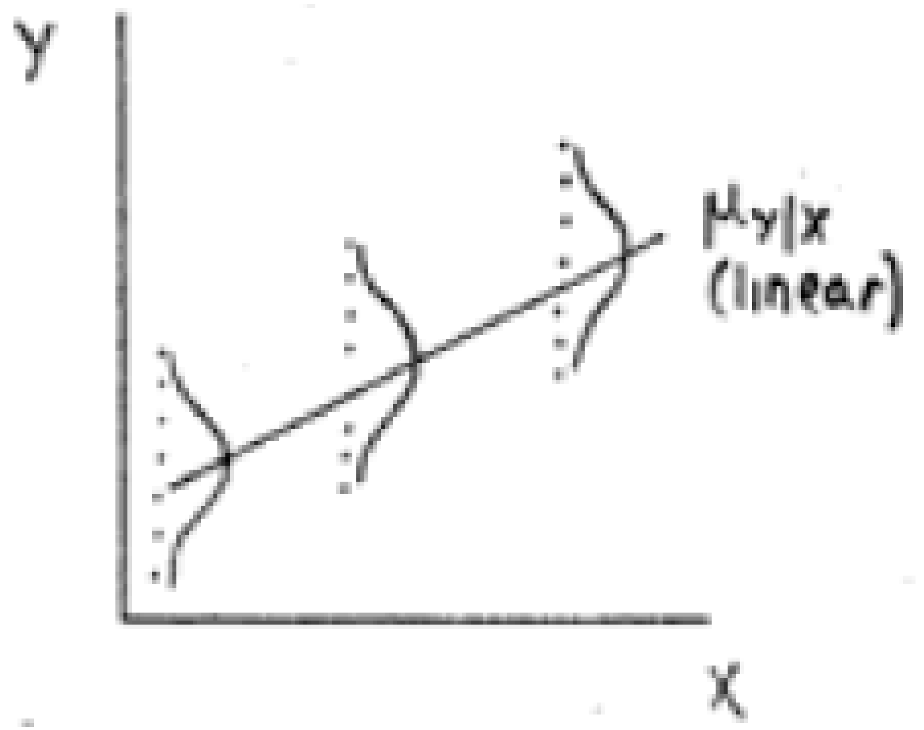
- Five basic assumptions

1. Existence: for each fixed value of X, Y is a random variable with finite mean and variance
2. Independence: the set of Y_i are independent random variables given X_i

Assumptions of Linear Regression

3. Linearity: the mean value of Y is a linear function of X

$$\mu_{Y|X} = E[Y|X] = \beta_0 + \beta_1 X$$



Assumptions of Linear Regression

4. Homoscedasticity: the variance of Y is the same for any X
5. Normality: For each fixed value of X, Y has a normal distribution (by assumption 4, σ^2 does not depend on X)

$$Y \sim N(\mu_{Y|X}, \sigma^2)$$

Formulation

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i$$

$$\varepsilon_i \sim N(0, \sigma^2) \quad \text{independent}$$

$$\Rightarrow Y_i \text{ are } N(\beta_0 + \beta_1 X_i, \sigma^2) \text{ given } X_i$$

$$\Rightarrow E(Y_i \mid X_i) = \beta_0 + \beta_1 X_i$$

$$\text{Var}(Y_i \mid X_i) = \sigma^2 = \text{variability of } Y_i \text{ about } \mu_{Y \mid X_i}$$

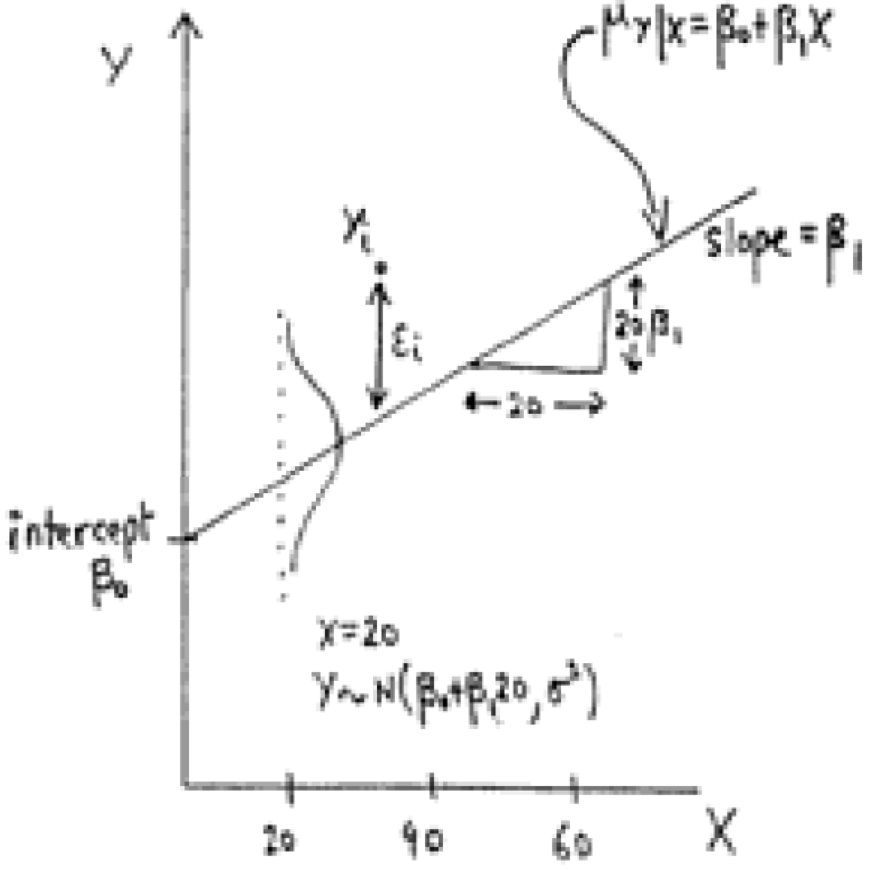
- Y_i are linear function of X_i plus some random error

$$\varepsilon_i = \text{error} \left(= Y_i - \mu_{Y \mid X_i} \right)$$

$$\hat{\varepsilon}_i = Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_i \quad \text{is called the "residual" for } Y_i$$

$$\hat{\varepsilon}_i = Y_i - \hat{Y}_i$$

Linear Regression



Estimating β_0 and β_1

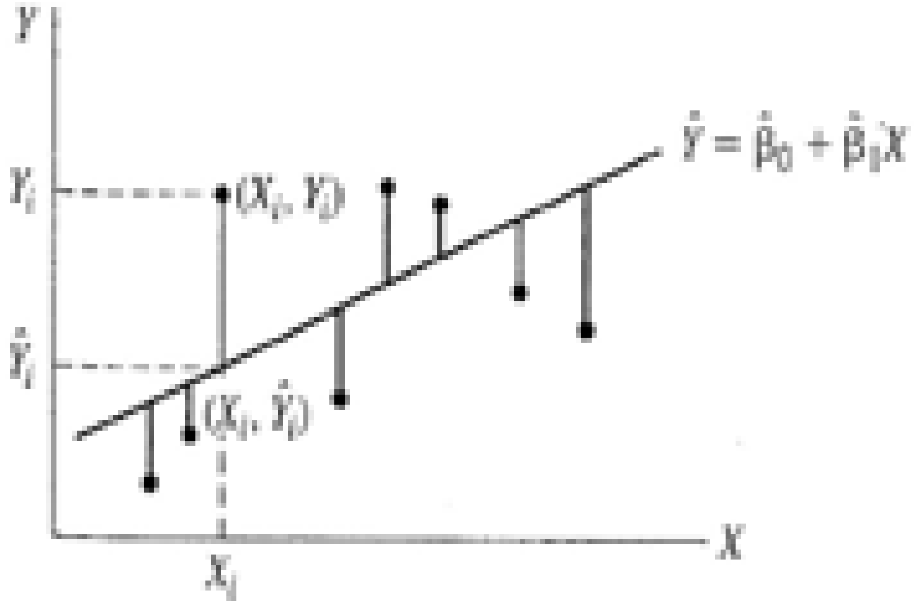
- Find "best" line
- Criterion for "best": estimate β_0 and β_1 to minimize:

$$\sum_{i=1}^{n}\left(Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_i\right)^2$$
$$= \sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 = \sum_{i=1}^{n} \hat{\varepsilon}_i^2$$

- This is the residual sum of squares, sum of squares due to error, or sum of squares about regression line
- Least Squares estimator

Rationale for LS Estimates

- $\hat{\varepsilon}^2$ measures the “deviance” of Y_i from the estimated model
- The “best” model is the one from which the data deviate the least



Least Squares Estimators

- Taking derivatives with respect to β, we obtain

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n}\left(X_i - \bar{X}\right)\left(Y_i - \bar{Y}\right)}{\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2} = \frac{S_{xy}}{S_x^2}$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X}$$

Example: SBP/age data

$$\bar{X} = 45.13 \quad \bar{Y} = 142.53$$

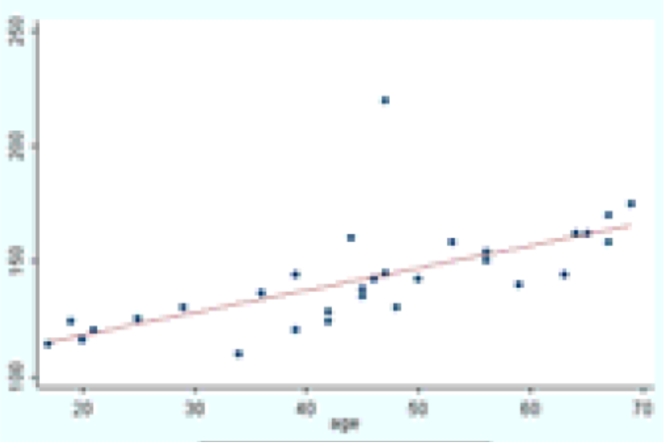
$$S_x^2 = 233.91 \quad S_y^2 = 509.91 \quad S_{xy} = 227.10$$

$$\hat{\beta}_1 = \frac{s_{xy}}{s_x^2} = \frac{227.10}{233.91} = 0.97$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X} = 142.53 - 0.97(45.13) = 98.71$$

$$\hat{y} = 98.71 + .97x$$

.97mm Hg $\uparrow$ for every 1 yr $\uparrow$ in age



Using the Model

- Using the parameter estimates, our best guess for any Y given X is

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$$

- Hence at $\bar{X}$

$$\hat{\beta}_0 + \hat{\beta}_1 \bar{X} = \left(\bar{Y} - \hat{\beta}_1 \bar{X}\right) + \hat{\beta}_1 \bar{X} = \bar{Y}$$

- Also $\hat{Y} = \bar{Y} + \hat{\beta}_1 \left(X - \bar{X}\right)$

Regression and Correlation Coefficient

Regression Coefficient

Correlation Coefficient

$$\hat{\beta}_1 = \frac{S_{xy}}{S_x^2} \text{ and } r_{xy} = \frac{S_{xy}}{S_x S_y}$$

$$\Rightarrow r_{xy} = \hat{\beta}_1 \cdot \frac{S_x}{S_y}$$

Example

Suppose we have:

x	y
1	2
2	4
-1	0
3	4
4	6

$$n = 5$$

$$\bar{x} = \frac{1+2-1+3+4}{5} = 1.8$$

$$\bar{y} = \frac{2+4+0+4+6}{5} = 3.2$$

Calculate r:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

$x_i - \bar{x}$	$y_i - \bar{y}$	$(x_i - \bar{x})(y_i - \bar{y})$
1-1.8 = -0.8	2-3.2 = -1.2	(-0.8)(-1.2) = 0.96
2-1.8 = 0.2	4-3.2 = 0.8	(0.2)(0.8) = 0.16
-1-1.8 = -2.8	0-3.2 = -3.2	(-2.8)(-3.2) = 8.96
3-1.8 = 1.2	4-3.2 = 0.8	(1.2)(0.8) = 0.96
4-1.8 = 2.2	6-3.2 = 2.8	(2.2)(2.8) = 6.16
0	0	17.2

Example

$$\sum\left(x_i-\bar{x}\right)^2=\left(-0.8\right)^2+\left(0.2\right)^2+\left(-2.8\right)^2+\left(1.2\right)^2+\left(2.2\right)^2=14.8$$

$$\sum\left(y_i-\bar{y}\right)^2=\left(-1.2\right)^2+\left(0.8\right)^2+\left(-3.2\right)^2+\left(0.8\right)^2+\left(2.8\right)^2=20.8$$

$$S_x=\sqrt{\frac{14.8}{4}}=1.92$$

$$s_y=\sqrt{\frac{20.8}{4}}=2.28$$

$$r=\frac{17.2}{\sqrt{(14.8)(20.8)}}=0.98$$

$$\hat{\beta}_1=r\cdot\frac{s_y}{S_x}=(0.98)\left(\frac{2.28}{1.92}\right)=1.16$$

$$\hat{\beta}_0=\bar{y}-\hat{\beta}_1\bar{x}=3.2-(1.16)(1.8)=1.11$$

Estimated regression line is $\hat{y}_i=1.11+1.16x_i$

Example

Fitted values are the $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$

$\hat{y}_1 = 1.11 + 1.16(1) = 2.27$ $\qquad$ $\hat{y}_4 = 1.11 + 1.16(3) = 4.59$

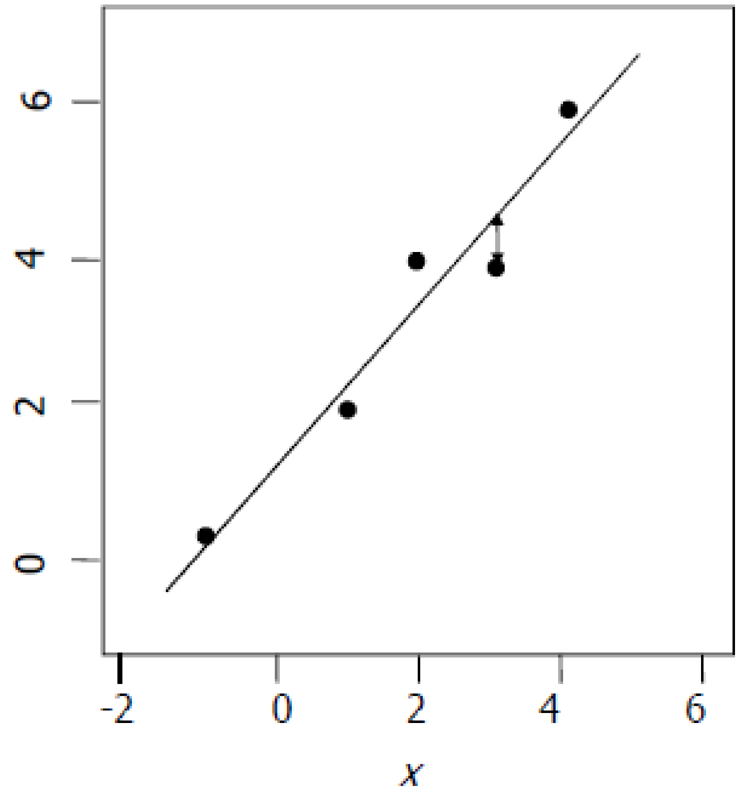
$\hat{y}_2 = 1.11 + 1.16(2) = 3.43$ $\qquad$ $\hat{y}_5 = 1.11 + 1.16(4) = 5.75$

$\hat{y}_3 = 1.11 + 1.16(-1) = -0.05$

Residuals =

$y_i - \hat{y}_i = \hat{\varepsilon}_i$	$(y_i - \hat{y}_i)^2$
2-2.27 = -0.27 4-3.43 = 0.57 0-(-0.05) = 0.05 4-4.59 = -0.59 6-5.75 = 0.25	$(-0.27)^2$ = 0.073 $(0.57)^2$ = 0.345 $(0.05)^2$ = 0.0025 $(-0.59)^2$ = 0.348 $(0.25)^2$ =0.0625
$\sum \hat{\varepsilon}_i = \sum (y_i - \hat{y}_i) = 0$	$\sum (y_i - \hat{y}_i)^2 = 0.811$

Example



$$\hat{y} = 1.11 + 1.16x$$

At $x = 3$:

$$y_i - \hat{y}_i = 4 - 4.59 = -.59$$

Logistic Regression

Logistic Regression

- Learn P(Y|X) directly
- Consider learning f: X → Y, where
  - X is a vector of real-valued features, < X_1 … X_n >
  - Y is boolean
  - assume all X_i are conditionally independent given Y
  - model $P(X_i \mid Y = y_k)$ as Gaussian $N(\mu_{ik},\sigma_i^2)$
  - model P(Y) as Bernoulli (π)
    - Y is 1 with probability π

Derivation of P(Y|X)

$$P(Y=1|X) = \frac{P(Y=1)P(X|Y=1)}{P(Y=1)P(X|Y=1) + P(Y=0)P(X|Y=0)}$$

$$= \frac{1}{1 + \frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)}}$$

$$= \frac{1}{1 + \exp(\ln \frac{P(Y=0)P(X|Y=0)}{P(Y=1)P(X|Y=1)})}$$

$$= \frac{1}{1 + \exp(\ (\ln \frac{1-\pi}{\pi}) + \boxed{\sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)})}}$$

$$P(x \mid y_k) = \frac{1}{\sigma_{ik}\sqrt{2\pi}} \quad e^{\frac{-(x-\mu_{ik})^2}{2\sigma_{ik}^2}}$$

$$\boxed{\sum_i \left(\frac{\mu_{i0} - \mu_{i1}}{\sigma_i^2} X_i + \frac{\mu_{i1}^2 - \mu_{i0}^2)}{2\sigma_i^2} \right)}$$

$$P(Y=1|X) = \frac{1}{1 + \exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

Very Convenient

$$P(Y=1|X=<X_1,...X_n>) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

implies

$$P(Y=0|X=<X_1,...X_n>) = \frac{exp(w_0+\sum_i w_i X_i)}{1+exp(w_0+\sum_i w_i X_i)}$$

implies

$$\frac{P(Y=0|X)}{P(Y=1|X)} = exp(w_0+\sum_i w_i X_i)$$

linear
classification
rule!

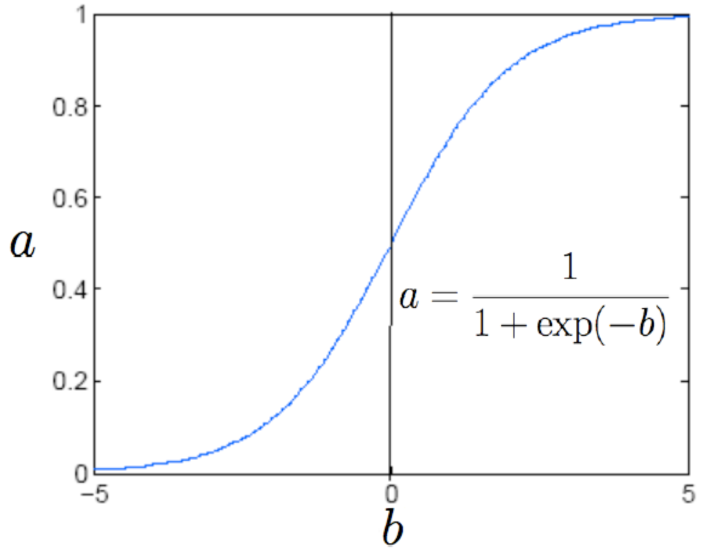
implies

$$\ln \frac{P(Y=0|X)}{P(Y=1|X)} = w_0+\sum_i w_i X_i$$

Very Convenient

- Posteriors sum to 1 and remain in [0, 1]
- Logit: $l = logit(p) = \log\left(\frac{p}{1-p}\right)$=α+βx
  - l is linear in x
- Probability: $p = \frac{e^l}{1+e^l}$

Logistic Function



$$P(Y=1|X) = \frac{1}{1+\exp(w_0 + \sum_{i=1}^{n} w_i X_i)}$$

- $p = 0, \quad l = \log\left(\frac{p}{1-p}\right) = -\infty$
- $p = \frac{1}{2}, \quad l = \log\left(\frac{p}{1-p}\right) = 0$
- $p = 1, \quad l = \log\left(\frac{p}{1-p}\right) = +\infty$

Logistic Regression More Generally

- Logistic regression when Y is not boolean (but still discrete)
  - $y \in \{y_1 \ldots y_R\}$: learn R-1 sets of weights
  - for k<R $$P(Y = y_k|X) = \frac{\exp(w_{k0} + \sum_{i=1}^{n} w_{ki} X_i)}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}$$
  - for k=R $$P(Y = y_R|X) = \frac{1}{1 + \sum_{j=1}^{R-1} \exp(w_{j0} + \sum_{i=1}^{n} w_{ji} X_i)}$$

Training Logistic Regression: MCLE

- We have L training examples: $\{\langle X^1, Y^1\rangle, \ldots \langle X^L, Y^L\rangle\}$

- Maximum likelihood estimate for parameters W

$$W_{MLE} = \arg\max_W P(< X^1, Y^1 > \ldots < X^L, Y^L > | W)$$
$$= \arg\max_W \prod_l P(< X^l, Y^l > | W)$$

- Maximum Conditional Likelihood Estimate (MCLE)

Training Logistic Regression: MCLE

- Choose parameters <w_0 … w_n> to maximize conditional likelihood of training data

$$P(Y=0|X,W) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

$$P(Y=1|X,W) = \frac{exp(w_0+\sum_i w_i X_i)}{1+exp(w_0+\sum_i w_i X_i)}$$

- Training data D= $\{\langle X^1, Y^1\rangle, \ldots \langle X^L, Y^L\rangle\}$
- Data likelihood = $\prod_l P(X^l, Y^l|W)$
- Data conditional likelihood = $\prod_l P(Y^l|X^l, W)$

$$W_{MCLE} = \arg\max_W \prod_l P(Y^l|W, X^l)$$

Conditional Log Likelihood

$$l(W) \equiv \ln \prod_l P(Y^l | X^l, W) = \sum_l \ln P(Y^l | X^l, W)$$

$$P(Y = 0 | X, W) = \frac{1}{1 + exp(w_0 + \sum_i w_i X_i)}$$

$$P(Y = 1 | X, W) = \frac{exp(w_0 + \sum_i w_i X_i)}{1 + exp(w_0 + \sum_i w_i X_i)}$$

$$\begin{aligned} l(W) &= \sum_l Y^l \ln P(Y^l = 1 | X^l, W) + (1 - Y^l) \ln P(Y^l = 0 | X^l, W) \\ &= \sum_l Y^l \ln \frac{P(Y^l = 1 | X^l, W)}{P(Y^l = 0 | X^l, W)} + \ln P(Y^l = 0 | X^l, W) \\ &= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l)) \end{aligned}$$

Maximizing Conditional Log Likelihood

$$P(Y=0|X,W) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

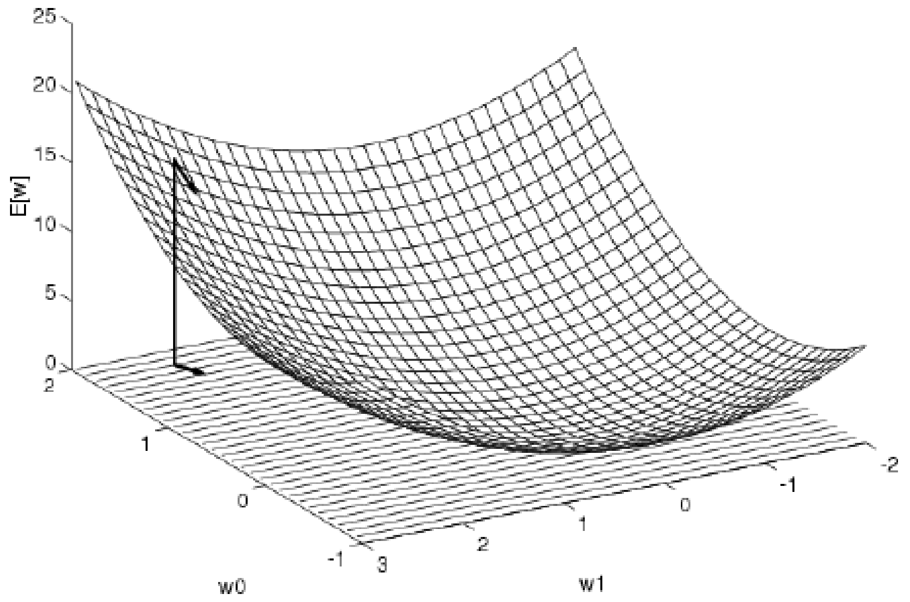
$$P(Y=1|X,W) = \frac{exp(w_0+\sum_i w_i X_i)}{1+exp(w_0+\sum_i w_i X_i)}$$

$$\begin{aligned} l(W) &\equiv \ln \prod_l P(Y^l|X^l,W) \\ &= \sum_l Y^l(w_0+\sum_i^n w_i X_i^l) - \ln(1+exp(w_0+\sum_i^n w_i X_i^l)) \end{aligned}$$

Good news: l(W) is concave function of W

Bad news: no closed-form solution to maximize l(W)

Gradient Descent



Gradient

$$\nabla E[\vec{w}] \equiv \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \cdots \frac{\partial E}{\partial w_n}\right]$$

Training rule:

$$\Delta \vec{w} = -\eta \nabla E[\vec{w}]$$

i.e.,

$$\Delta w_i = -\eta \frac{\partial E}{\partial w_i}$$

Maximize Conditional Log Likelihood: Gradient Ascent

$$
\begin{aligned}
l(W) &\equiv \ln \prod_l P(Y^l | X^l, W) \\
&= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l))
\end{aligned}
$$

$$
\frac{\partial l(W)}{\partial w_i} = \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))
$$

Maximize Conditional Log Likelihood: Gradient Ascent

$$
\begin{aligned}
l(W) &\equiv \ln \prod_l P(Y^l | X^l, W) \\
&= \sum_l Y^l (w_0 + \sum_i^n w_i X_i^l) - \ln(1 + exp(w_0 + \sum_i^n w_i X_i^l))
\end{aligned}
$$

$$
\frac{\partial l(W)}{\partial w_i} = \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))
$$

Gradient ascent algorithm: iterate until change < ε

For all i, repeat

$$
w_i \leftarrow w_i + \eta \sum_l X_i^l (Y^l - \hat{P}(Y^l = 1 | X^l, W))
$$

Logistic Regression: Summary

- Consider learning f: X → Y, where
  - X is a vector of real-valued features, $< X_1 \ldots X_n >$
  - Y is boolean
  - assume all X_i are conditionally independent given Y
  - model $P(X_i \mid Y = y_k)$ as Gaussian $N(\mu_{ik},\sigma_i^2)$
  - model P(Y) as Bernoulli (π)
- Then P(Y|X) is of this form and we can directly estimate W

$$P(Y=1|X=<X_1,...X_n>) = \frac{1}{1+exp(w_0+\sum_i w_i X_i)}$$

Linear Discriminant Functions

Augmented Feature Vector

- Linear discriminant function: $\mathbf{g(x) = w^t\, x + w_0}$
- Can rewrite it:

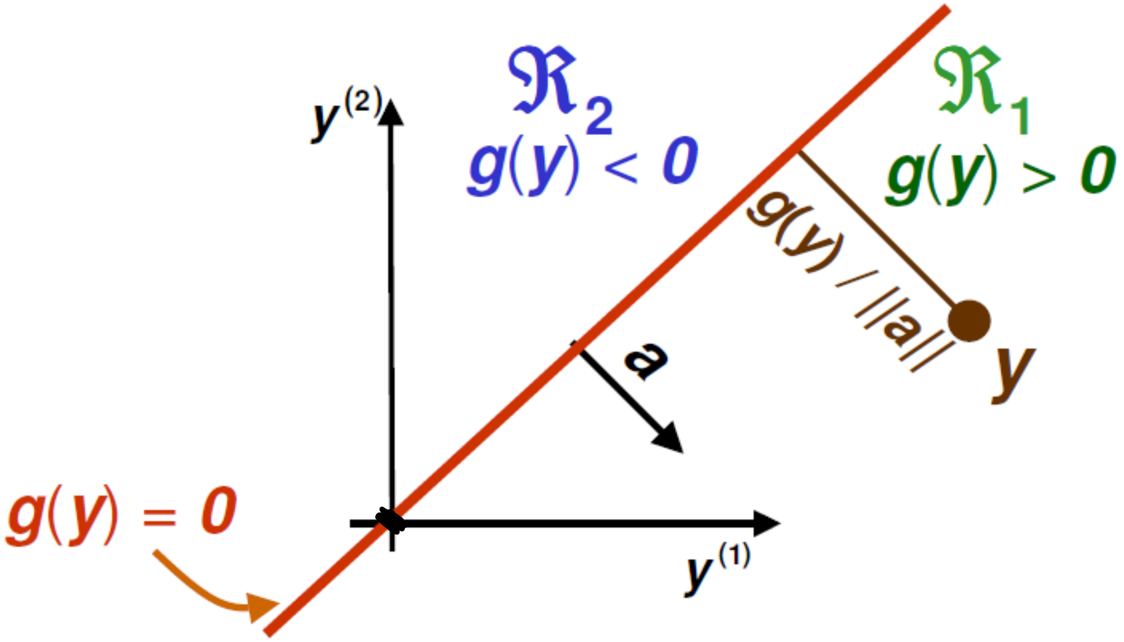
$$g(x) = \underbrace{\begin{bmatrix} w_0 & w^t \end{bmatrix}}_{\text{new weight vector } a} \underbrace{\begin{bmatrix} 1 \\ x \end{bmatrix}}_{\text{new feature vector } y} = a^t y = g(y)$$

- $\boldsymbol{y}$ is called the augmented feature vector
- Added a dummy dimension to get a completely equivalent new homogeneous problem

old problem	new problem
$g(x) = w^t x + w_0$	$g(y) = a^t y$
$\begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}$	$\begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_d \end{bmatrix}$

- Feature augmentation is done for simpler notation
- From now on, always assume that we have augmented feature vectors
  - Given samples $\mathbf{x}_1, \ldots, \mathbf{x}_n$ convert them to augmented samples $\mathbf{y}_1, \ldots, \mathbf{y}_n$ by adding a new dimension of value 1

$$\boldsymbol{y}_i = \begin{bmatrix} 1 \\ \boldsymbol{x}_i \end{bmatrix}$$



Training Error

- For the rest of this part, assume we have 2 classes
  - Samples: $\mathbf{y_1, \ldots, y_n}$ **,** some in class 1, some in class 2
- Use samples to determine weights **a** in the discriminant function $\mathbf{g(y) = a^t y}$
- What should the criterion for determining **a** be?
- For now, suppose we want to minimize the training error (the number of misclassified samples $\mathbf{y_1, \ldots, y_n}$)

- Recall that:

$$g(y_i)>0 \Rightarrow y_i \text{ classified as } c_1$$
$$g(y_i)<0 \Rightarrow y_i \text{ classified as } c_2$$

- Thus training error is **0** if

$$\begin{cases} g(y_i) > 0 & \forall y_i \in c_1 \\ g(y_i) < 0 & \forall y_i \in c_2 \end{cases}$$

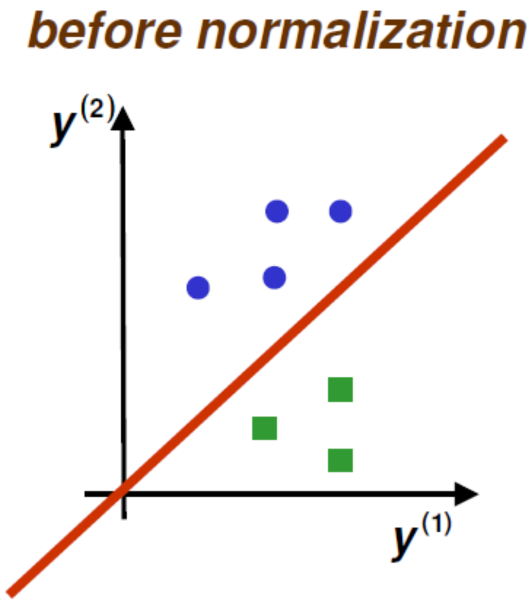
“Normalization”

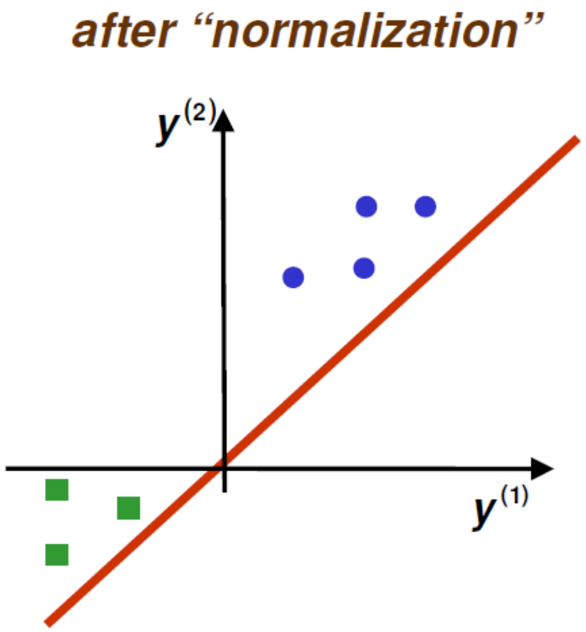
- Thus training error is 0 if: $\begin{cases} a^t y_i > 0 & \forall y_i \in c_1 \\ a^t y_i < 0 & \forall y_i \in c_2 \end{cases}$

- Equivalently, training error is 0 if: $\begin{cases} a^t y_i > 0 & \forall y_i \in c_1 \\ a^t(-y_i) > 0 & \forall y_i \in c_2 \end{cases}$

- This suggests “normalization” (a.k.a. reflection):
  1. Replace all examples from class 2 by:
  $$y_i \to -y_i \qquad \forall y_i \in c_2$$
  2. Seek weight vector ***a*** such that
  $$a^t y_i > 0 \qquad \forall y_i$$
  - If such **a** exists, it is called a separating or solution vector
  - Original samples $\mathbf{x_1,\dots, x_n}$ can indeed be separated by a line

Normalization





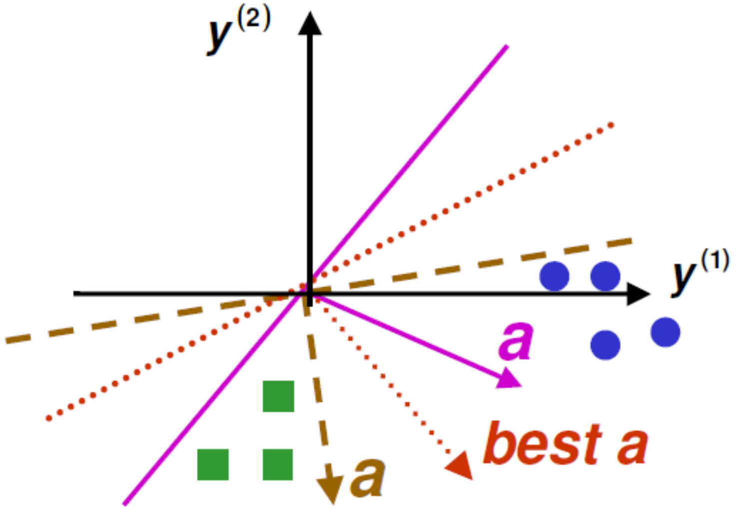
- Seek a hyperplane that separates patterns from different categories

- Seek hyperplane that puts *normalized* patterns on the same (positive) side

Solution Region

- Find weight vector ***a*** such that for all samples:

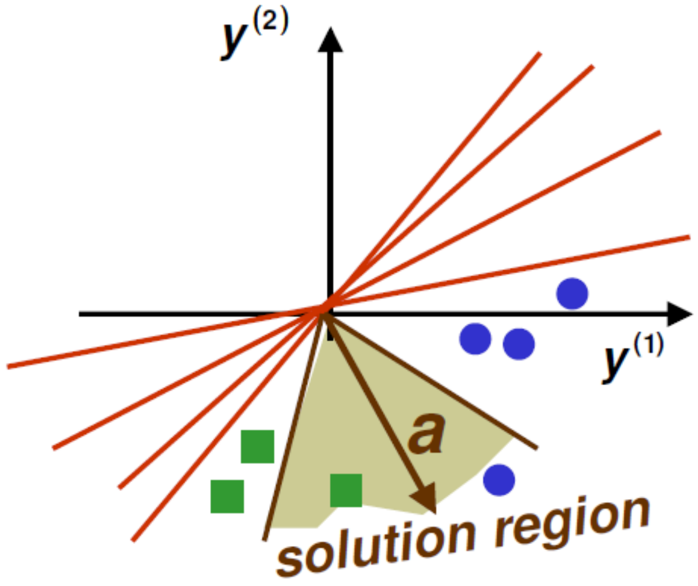
$$\mathbf{a}^t \mathbf{y}_i = \sum_{k=0}^{d} a_k y_i^{(k)} > 0$$



- In general, there can be many solutions

Solution Region

- Solution region for **a**: set of all possible solutions defined in terms of normal a to the separating hyperplane



Optimization

- Need to minimize a function of many variables

$$\mathbf{J(x) = J(x_{1},\ldots, x_{d})}$$

- We know how to minimize **J(x)**
  - Take partial derivatives and set them to zero

gradient

$$\begin{bmatrix} \frac{\partial}{\partial x_1} J(x) \\ \vdots \\ \frac{\partial}{\partial x_d} J(x) \end{bmatrix} = \nabla J(x) = 0$$

Optimization

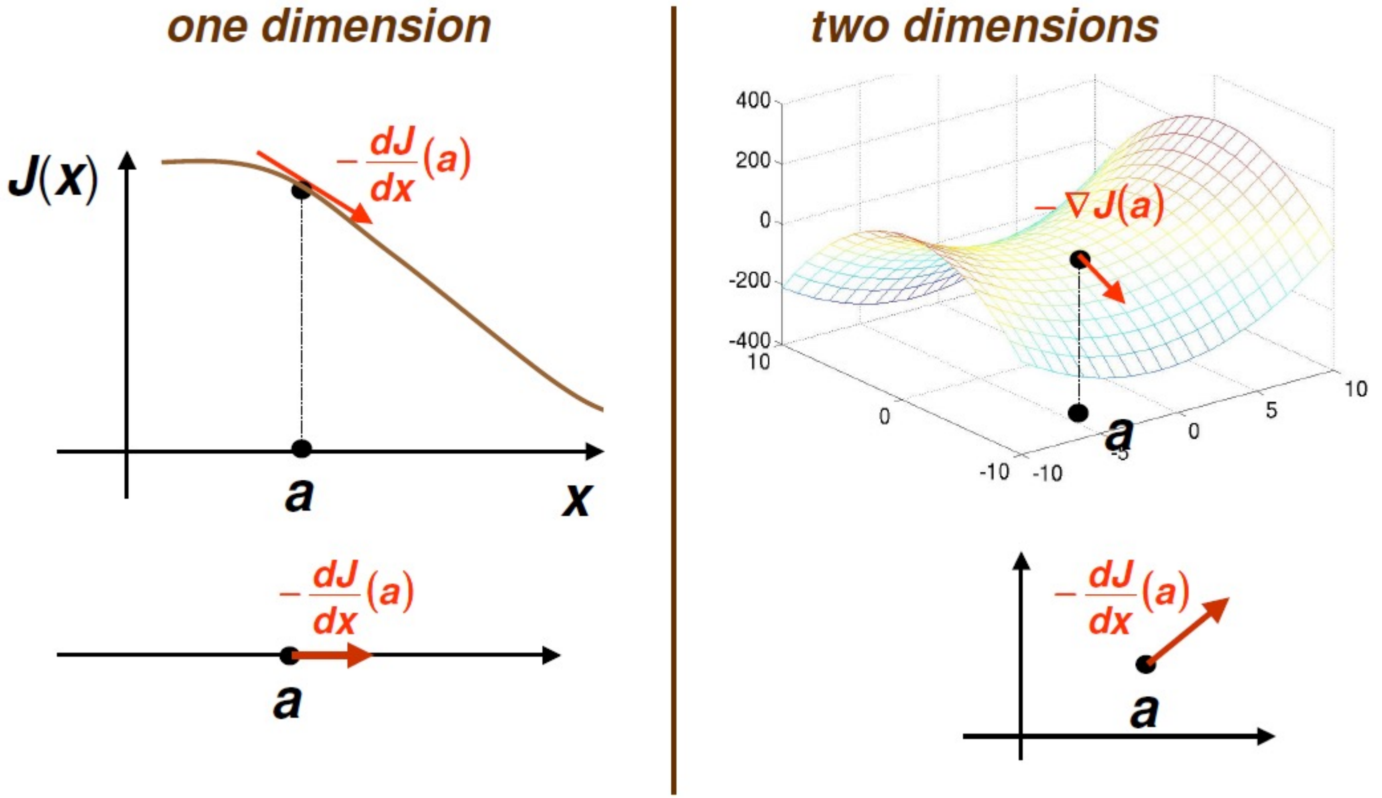
- However solving analytically is not always easy
  - For example:

$$\begin{cases} \sin\left(x_1^2 + x_2^3\right) + e^{x_4^2} = 0 \\ \cos\left(x_1^2 + x_2^3\right) + \log\left(x_5^3\right)^{x_4^2} = 0 \end{cases}$$

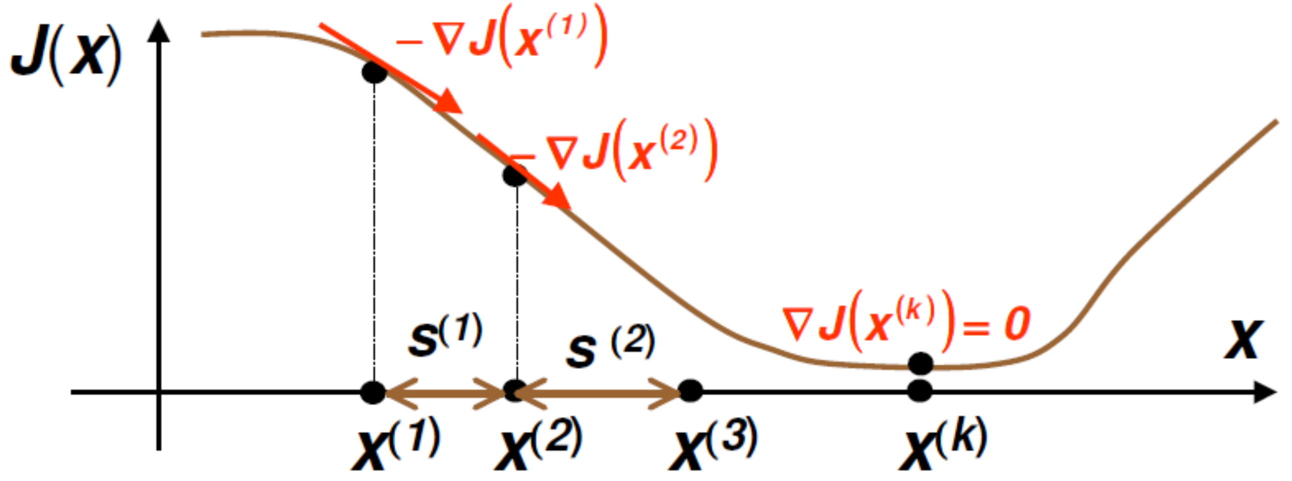
- Sometimes it is not even possible to write down an analytical expression for the derivative (example later today)

Gradient Descent

- Gradient $\nabla J(x)$ points in direction of steepest increase of **J(x),** and $-\nabla J(x)$ in direction of steepest decrease



Gradient Descent



Gradient Descent for minimizing any function ***J(x)***

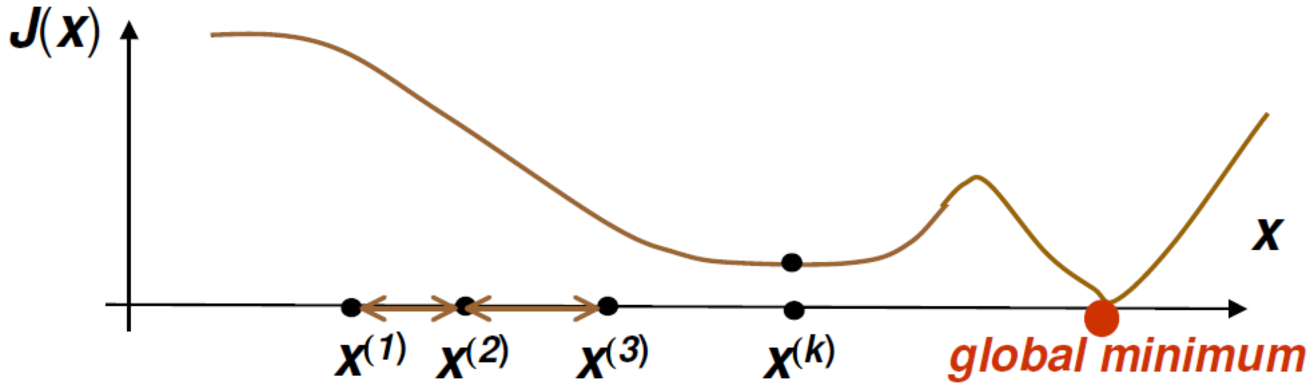
- Set k = 1 and $x^{(1)}$ to some initial guess for the weight vector
- While $\eta^{(k)} \left| \nabla J(x^{(k)}) \right| > \varepsilon$
  - Choose learning rate $\boldsymbol{\eta^{(k)}}$

$$x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla J(x)$$
$$k = k + 1$$

(update rule)

Gradient Descent

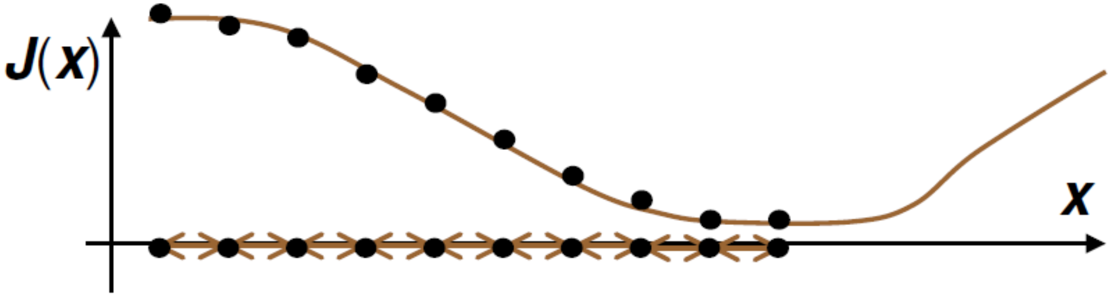
- Gradient decent is guaranteed to only find local minima



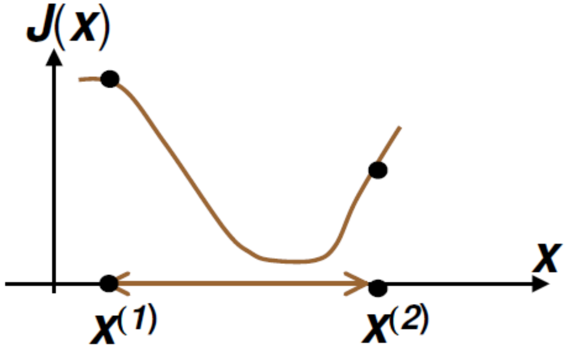
- Nevertheless gradient descent is very popular because it is simple and applicable to any function

Gradient Descent

- Main issue: how to set parameter **η** (learning rate)
  - If **η** is too small, too many iterations



  - If **η** is too large may overshoot the minimum and possibly never find it



LDF Criterion Function

- Find weight vector **a** such that for all samples $\mathbf{y_1,\dots, y_n}$

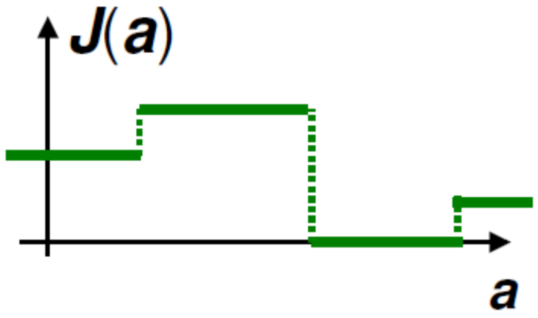
$$a^t y_i = \sum_{k=0}^{d} a_k y_i^{(k)} > 0$$

- Need criterion function **J(a)** which is minimized when **a** is a solution vector
- Let $\mathbf{Y_M}$ be the set of examples misclassified by **a**

$$\mathbf{Y_M(a) = \{ y_i \text{ s.t. } a^t y_i < 0 \}}$$

- First natural choice: number of misclassified examples

$$\mathbf{J(a) = |Y_M(a)|}$$



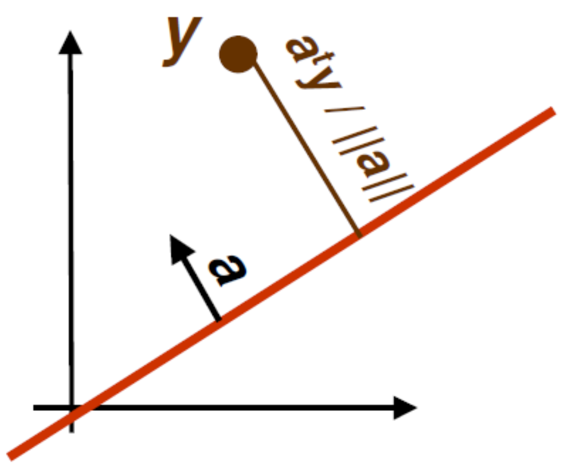
- Piecewise constant, gradient descent is useless

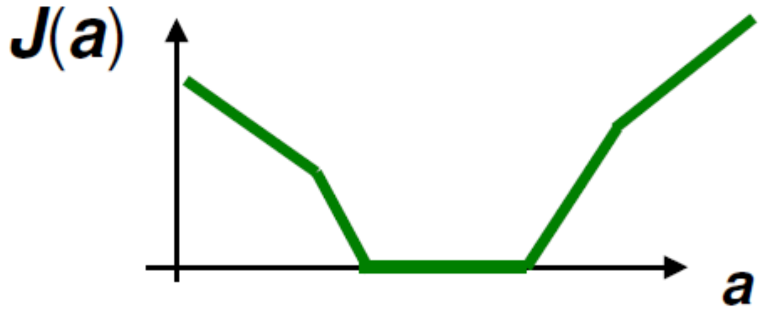
Perceptron

Perceptron Criterion Function

$$J_p(a) = \sum_{y \in Y_M} \left(-a^t y\right)$$

- If **y** is misclassified, $\mathbf{a^t y < 0}$
- Thus $\mathbf{J_p(a) > 0}$
- $\mathbf{J_p(a)}$ is **||a||** times the sum of distances of misclassified examples to decision boundary
- $\mathbf{J_p(a)}$ is piecewise linear and thus suitable for gradient descent





Perceptron Batch Rule

$$J_p(\boldsymbol{a}) = \sum_{\boldsymbol{y} \in Y_M} \left(-\boldsymbol{a}^t \boldsymbol{y}\right)$$

- Gradient of $\mathbf{J_p(a)}$ is: $\nabla J_p(\boldsymbol{a}) = \sum_{\boldsymbol{y} \in Y_M} (-\boldsymbol{y})$
  - $\mathbf{Y_M}$ are samples misclassified by $\mathbf{a^{(k)}}$
  - It is not possible to solve $\nabla J_p(\boldsymbol{a}) = \boldsymbol{0}$ analytically because of $\mathbf{Y_M}$
- Update rule for gradient descent: $\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \eta^{(k)} \nabla J(\boldsymbol{x})$
- Thus the *gradient decent batch update rule for* $\mathbf{J_p(a)}$ is:

$$\boldsymbol{a}^{(k+1)} = \boldsymbol{a}^{(k)} + \eta^{(k)} \sum_{\boldsymbol{y} \in Y_M} \boldsymbol{y}$$

- It is called batch rule because it is based on all misclassified examples

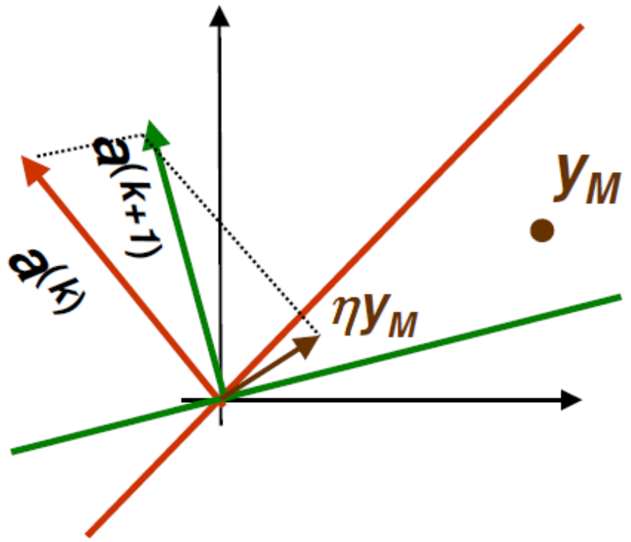
Perceptron Single Sample Rule

- The *gradient decent single sample rule* for $\mathbf{J_p(a)}$ is:

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \eta^{(k)} \mathbf{y}_M$$

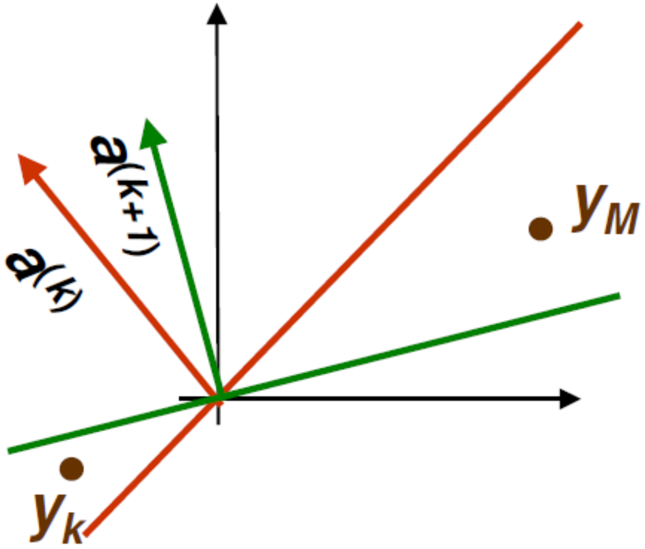
  - Note that $\mathbf{y_M}$ is one sample misclassified by **a(k)**
  - Must have a consistent way of visiting samples

- Geometric Interpretation:
  - $\mathbf{y_M}$ misclassified by $\mathbf{a^{(k)}}$ $\left(\mathbf{a}^{(k)}\right)^t \mathbf{y}_M \leq 0$
  - $\mathbf{y_M}$ is on the wrong side of decision hyperplane
  - Adding $\boldsymbol{\eta}\mathbf{y_M}$ to **a** moves the new decision hyperplane in the right direction with respect to $\boldsymbol{y}_\mathbf{M}$

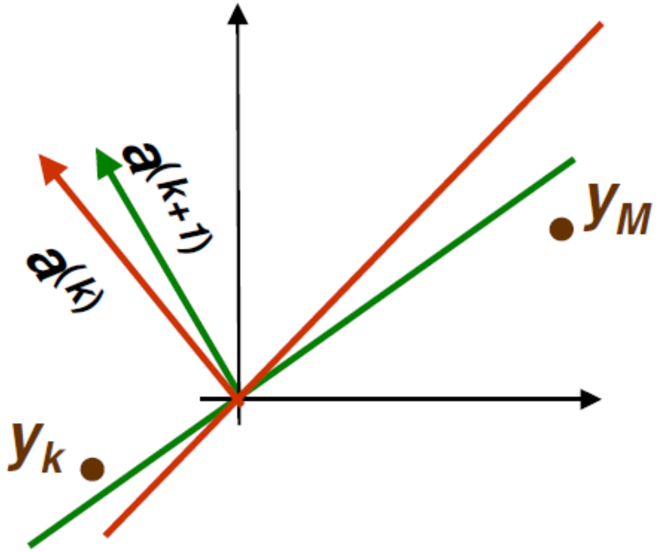


Perceptron Single Sample Rule

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \eta^{(k)} \mathbf{y}_M$$



η is too large, previously correctly classified sample $\mathbf{y}_k$ is now misclassified



η is too small, $\mathbf{y}_M$ is still misclassified

Perceptron Example

	features				grade
name	***good attendance?***	***tall?***	***sleeps in class?***	***chews gum?***	
Jane	*yes (1)*	*yes (1)*	*no (-1)*	*no (-1)*	*A*
Steve	*yes (1)*	*yes (1)*	*yes (1)*	*yes (1)*	*F*
Mary	*no (-1)*	*no (-1)*	*no (-1)*	*yes (1)*	*F*
Peter	*yes (1)*	*no (-1)*	*no (-1)*	*yes (1)*	*A*

- Class 1: students who get A
- Class 2: students who get F

	features					grade
name	***extra***	***good attendance?***	***tall?***	***sleeps in class?***	***chews gum?***	
Jane	*1*	*yes (1)*	*yes (1)*	*no (-1)*	*no (-1)*	*A*
Steve	*1*	*yes (1)*	*yes (1)*	*yes (1)*	*yes (1)*	*F*
Mary	*1*	*no (-1)*	*no (-1)*	*no (-1)*	*yes (1)*	*F*
Peter	*1*	*yes (1)*	*no (-1)*	*no (-1)*	*yes (1)*	*A*

- Augment samples by adding an extra feature (dimension) equal to 1

	features					grade
name	***extra***	***good attendance?***	***tall?***	***sleeps in class?***	***chews gum?***	
Jane	*1*	*yes (1)*	*yes (1)*	*no (-1)*	*no (-1)*	*A*
Steve	*-1*	*yes (-1)*	*yes (-1)*	*yes (-1)*	*yes (-1)*	*F*
Mary	*-1*	*no (1)*	*no (1)*	*no (1)*	*yes (-1)*	*F*
Peter	*1*	*yes (1)*	*no (-1)*	*no (-1)*	*yes (1)*	*A*

- Normalize:
  - Replace all examples from class 2 by their negative values
  - Seek ***a*** such that:

$$\mathbf{y}_i \rightarrow -\mathbf{y}_i \qquad \forall \mathbf{y}_i \in \mathbf{c}_2$$

$$\mathbf{a}^t \mathbf{y}_i > 0 \qquad \forall \mathbf{y}_i$$

	features					grade
name	***extra***	***good attendance?***	***tall?***	***sleeps in class?***	***chews gum?***	
Jane	*1*	*yes (1)*	*yes (1)*	*no (-1)*	*no (-1)*	*A*
Steve	*-1*	*yes (-1)*	*yes (-1)*	*yes (-1)*	*yes (-1)*	*F*
Mary	*-1*	*no (1)*	*no (1)*	*no (1)*	*yes (-1)*	*F*
Peter	*1*	*yes (1)*	*no (-1)*	*no (-1)*	*yes (1)*	*A*

- Single Sample Rule
  - Sample is misclassified if $\mathbf{a}^t\mathbf{y}_i = \sum_{k=0}^{4} \mathbf{a}_k \mathbf{y}_i^{(k)} < 0$
  - Gradient descent single sample rule: $\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \eta^{(k)} \sum_{\mathbf{y} \in Y_M} \mathbf{y}$
  - Set **η** fixed learning rate to $\eta^{(k)} = 1$: $\boxed{\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \mathbf{y}_M}$

- Set equal initial weights
  $\mathbf{a^{(1)} = [0.25, 0.25, 0.25, 0.25, 0.25]}$
- Visit all samples sequentially, modifying the weights after each misclassified example

name	$a^t y$	*misclassified?*
Jane	*0.25*1+0.25*1+0.25*1+0.25*(-1)+0.25*(-1) >0*	*no*
Steve	*0.25*(-1)+0.25*(-1)+0.25*(-1)+0.25*(-1)+0.25*(-1)<0*	*yes*

- New weights

$$\begin{aligned} \mathbf{a}^{(2)} = \mathbf{a}^{(1)} + \mathbf{y}_M &= [\mathbf{0.25\ \ 0.25\ \ 0.25\ \ 0.25\ \ 0.25}] + \\ &+ [-\mathbf{1}\ \ -\mathbf{1}\ \ -\mathbf{1}\ \ -\mathbf{1}\ \ -\mathbf{1}] = \\ &= [-\mathbf{0.75}\ \ -\mathbf{0.75}\ \ -\mathbf{0.75}\ \ -\mathbf{0.75}\ \ -\mathbf{0.75}] \end{aligned}$$

$$a^{(2)} = [-0.75 \ \ -0.75 \ \ -0.75 \ \ -0.75 \ \ -0.75]$$

name	$a^t y$	misclassified?
Mary	*-0.75*(-1)-0.75*1 -0.75 *1 -0.75 *1 -0.75*(-1) <0*	*yes*

- New weights

$$a^{(3)} = a^{(2)} + y_M = [-0.75 \ \ -0.75 \ \ -0.75 \ \ -0.75 \ \ -0.75] +$$
$$+ [-1 \ \ 1 \ \ 1 \ \ 1 \ \ -1] =$$
$$= [-1.75 \ \ 0.25 \ \ 0.25 \ \ 0.25 \ \ -1.75]$$

$$a^{(3)} = [-1.75 \quad 0.25 \quad 0.25 \quad 0.25 \quad -1.75]$$

name	$a^t y$	*misclassified?*
Peter	*-1.75 *1 +0.25* 1+0.25* (-1) +0.25 *(-1)-1.75*1 <0*	*yes*

- New weights

$$\begin{aligned} a^{(4)} = a^{(3)} + y_M &= [-1.75 \quad 0.25 \quad 0.25 \quad 0.25 \quad -1.75] + \\ &+ [1 \quad 1 \quad -1 \quad -1 \quad 1] = \\ &= [-0.75 \quad 1.25 \quad -0.75 \quad -0.75 \quad -0.75] \end{aligned}$$

$$a^{(4)} = [-0.75 \;\; 1.25 \;\; -0.75 \;\; -0.75 \;\; -0.75]$$

name	$a^t y$	***misclassified?***
Jane	*-0.75 *1 +1.25*1 -0.75*1 -0.75 *(-1) -0.75 *(-1)+0*	*no*
Steve	*-0.75*(-1)+1.25*(-1) -0.75*(-1) -0.75*(-1)-0.75*(-1)>0*	*no*
Mary	*-0.75 *(-1)+1.25*1-0.75*1 -0.75 *1 –0.75*(-1) >0*	*no*
Peter	*-0.75 *1+ 1.25*1-0.75* (-1)-0.75* (-1) -0.75 *1 >0*	*no*

- Thus the discriminant function is:

$$g(y) = -0.75 * y^{(0)} + 1.25 * y^{(1)} - 0.75 * y^{(2)} - 0.75 * y^{(3)} - 0.75 * y^{(4)}$$

- Converting back to the original features ***x:***

$$g(x) = 1.25 * x^{(1)} - 0.75 * x^{(2)} - 0.75 * x^{(3)} - 0.75 * x^{(4)} - 0.75$$

- Converting back to the original features ***x:***

$$1.25 * x^{(1)} - 0.75 * x^{(2)} - 0.75 * x^{(3)} - 0.75 * x^{(4)} > 0.75 \Rightarrow \textit{grade A}$$

$$1.25 * x^{(1)} - 0.75 * x^{(2)} - 0.75 * x^{(3)} - 0.75 * x^{(4)} < 0.75 \Rightarrow \textit{grade F}$$

good attendance | tall | sleeps in class | chews gum

- This is just one possible solution vector
- If we started with weights

$a^{(1)}$=[0,0.5, 0.5, 0, 0],

- The solution would be [-1,1.5, -0.5, -1, -1]

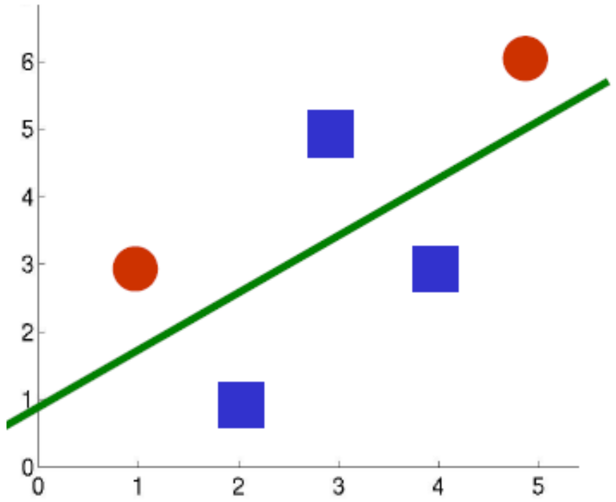
$$1.5 * x^{(1)} - 0.5 * x^{(2)} - x^{(3)} - x^{(4)} > 1 \Rightarrow \textit{grade A}$$

$$1.5 * x^{(1)} - 0.5 * x^{(2)} - x^{(3)} - x^{(4)} < 1 \Rightarrow \textit{grade F}$$

- In this solution, being tall is the least important feature

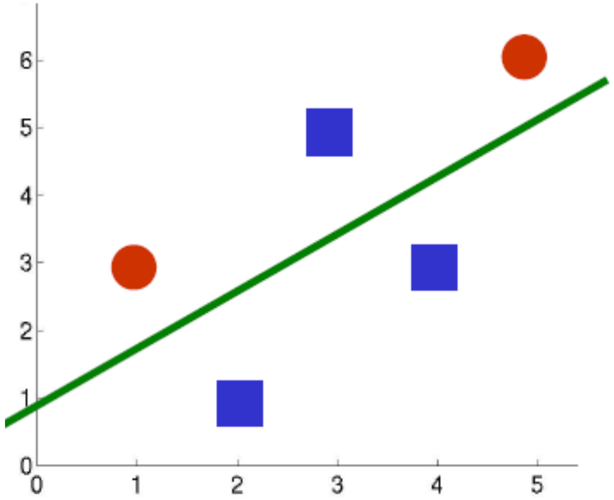
LDF: Non-separable Example

- Suppose we have 2 features and the samples are:
  - Class 1: [2,1], [4,3], [3,5]
  - Class 2: [1,3] and [5,6]
- These samples are not separable by a line
- Still would like to get approximate separation by a line
  - A good choice is shown in green
  - Some samples may be “noisy”, and we could accept them being misclassified



LDF: Non-separable Example

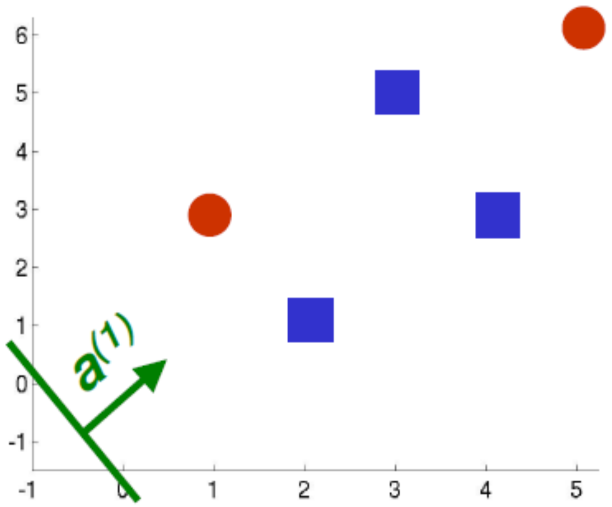
- Obtain $\boldsymbol{y_1, y_2, y_3, y_4}$ by adding extra feature and "normalizing"



$$y_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} \quad y_3 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ -1 \\ -3 \end{bmatrix} \quad y_5 = \begin{bmatrix} -1 \\ -5 \\ -6 \end{bmatrix}$$

LDF: Non-separable Example

- Apply Perceptron single sample algorithm
- Initial equal weights
  $a^{(1)}$ = [1 1 1]
  - Line equation $x^{(1)}+x^{(2)}+1=0$
- Fixed learning rate ***η = 1***



$$a^{(k+1)} = a^{(k)} + y_M$$

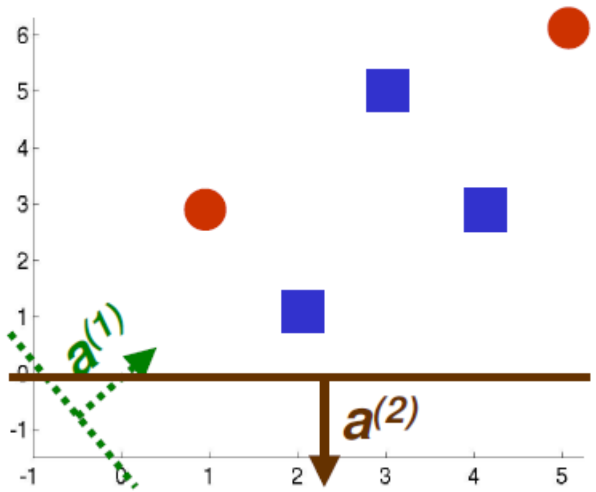
$$y_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} \quad y_3 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ -1 \\ -3 \end{bmatrix} \quad y_5 = \begin{bmatrix} -1 \\ -5 \\ -6 \end{bmatrix}$$

- $y^t_1 a^{(1)} = [1\ 1\ 1]*[1\ 2\ 1]^t > 0$ ✔
- $y^t_2 a^{(1)} = [1\ 1\ 1]*[1\ 4\ 3]^t > 0$ ✔
- $y^t_3 a^{(1)} = [1\ 1\ 1]*[1\ 3\ 5]^t > 0$ ✔

LDF: Non-separable Example

$$a^{(1)} = [1 \;\; 1 \;\; 1] \qquad a^{(k+1)} = a^{(k)} + y_M$$

$$y_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} \quad y_3 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ -1 \\ -3 \end{bmatrix} \quad y_5 = \begin{bmatrix} -1 \\ -5 \\ -6 \end{bmatrix}$$



- $y^t{}_4 a^{(1)}$=[1 1 1]*[-1 -1 -3]^t = -5< 0

$$a^{(2)} = a^{(1)} + y_M = [1 \;\; 1 \;\; 1] + [-1 \;\; -1 \;\; -3] = [0 \;\; 0 \;\; -2]$$

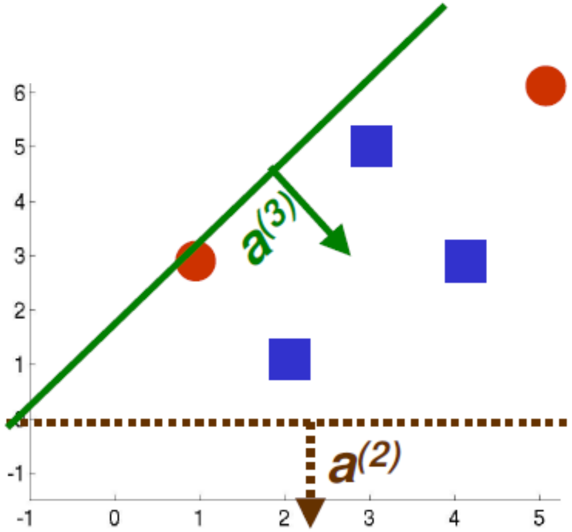
- $y^t{}_5\, a^{(2)}$=[0 0 -2]*[-1 -5 -6]^t = 12 > 0 ✔
- $y^t{}_1\, a^{(2)}$=[0 0 -2]*[1 2 1]^t < 0

$$a^{(3)} = a^{(2)} + y_M = [0 \;\; 0 \;\; -2] + [1 \;\; 2 \;\; 1] = [1 \;\; 2 \;\; -1]$$

LDF: Non-separable Example

$$a^{(3)} = [1\ \ 2\ \ -1] \qquad a^{(k+1)} = a^{(k)} + y_M$$

$$y_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} \quad y_3 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ -1 \\ -3 \end{bmatrix} \quad y_5 = \begin{bmatrix} -1 \\ -5 \\ -6 \end{bmatrix}$$



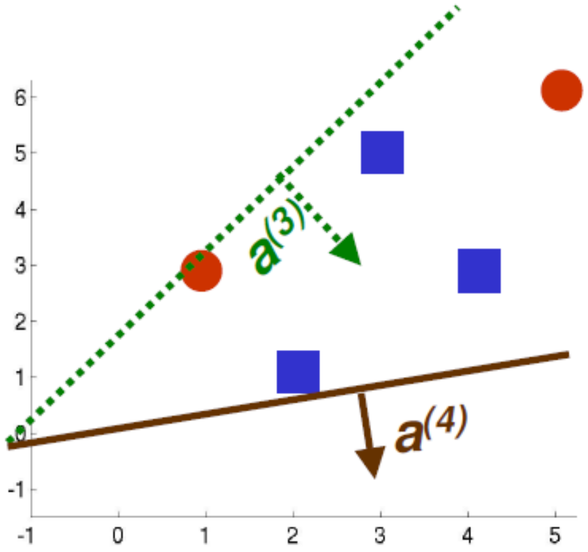
- $y^t_2\, a^{(3)}$=[1 4 3]*[1 2 -1]^t=6 > 0 ✓
- $y^t_3\, a^{(3)}$=[1 3 5]*[1 2 -1]^t > 0 ✓
- $y^t_4\, a^{(3)}$=[-1 -1 -3]*[1 2 -1]^t = 0

$$a^{(4)} = a^{(3)} + y_M = [1\ \ 2\ \ -1] + [-1\ \ -1\ \ -3] = [0\ \ 1\ \ -4]$$

LDF: Non-separable Example

$$a^{(4)} = [0 \;\; 1 \; -4] \qquad a^{(k+1)} = a^{(k)} + y_M$$

$$y_1 = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} \quad y_2 = \begin{bmatrix} 1 \\ 4 \\ 3 \end{bmatrix} \quad y_3 = \begin{bmatrix} 1 \\ 3 \\ 5 \end{bmatrix} \quad y_4 = \begin{bmatrix} -1 \\ -1 \\ -3 \end{bmatrix} \quad y_5 = \begin{bmatrix} -1 \\ -5 \\ -6 \end{bmatrix}$$



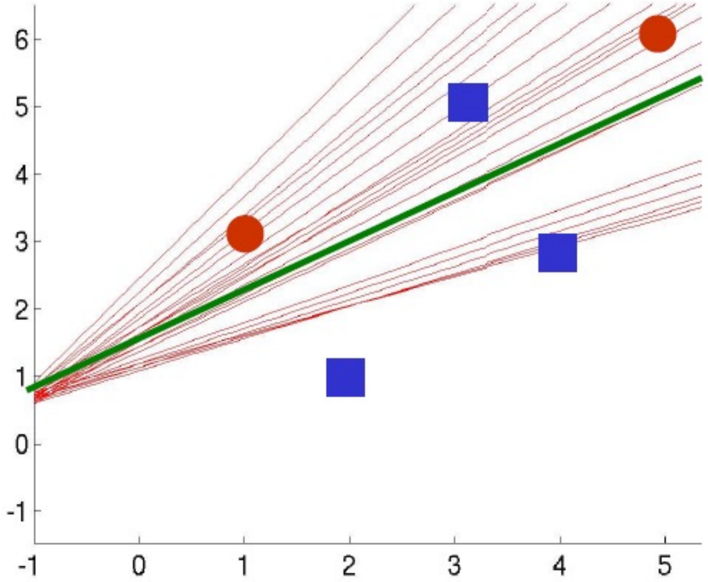
- $y_5^t a^{(4)}$=[-1 -5 -6]*[0 1 -4]=19>0
- $y_1^t a^{(4)}$=[1 2 1]*[0 1 -4]=-2<0
- ….

LDF: Non-separable Example

- We can continue this forever
- There is no solution vector ***a*** satisfying for all ***i***

$$a^t y_i = \sum_{k=0}^{5} a_k y_i^{(k)} > 0$$

- Need to stop but at a good point
- Solutions at iterations 900 through 915
  - Some are good and some are not
- How do we stop at a good solution?



Convergence of Perceptron Rules

- If classes are linearly separable and we use fixed learning rate, that is for $\boldsymbol{\eta^{(k)}}$ **=const**
- Then, *both the single sample and batch perceptron rules converge to a correct solution* *(could be any* ***a*** in the solution space)
- If classes are not linearly separable:
  - The algorithm does not stop, it keeps looking for a solution which does not exist

Convergence of Perceptron Rules

- If classes are not linearly separable:
  - By choosing appropriate learning rate, we can always ensure convergence: $\eta^{(k)} \to 0 \;\; as \;\; k \to \infty$
  - For example inverse linear learning rate: $\eta^{(k)} = \frac{\eta^{(1)}}{k}$
  - For inverse linear learning rate, convergence in the linearly separable case can also be proven
  - No guarantee that we stopped at a good point, but there are good reasons to choose inverse linear learning rate