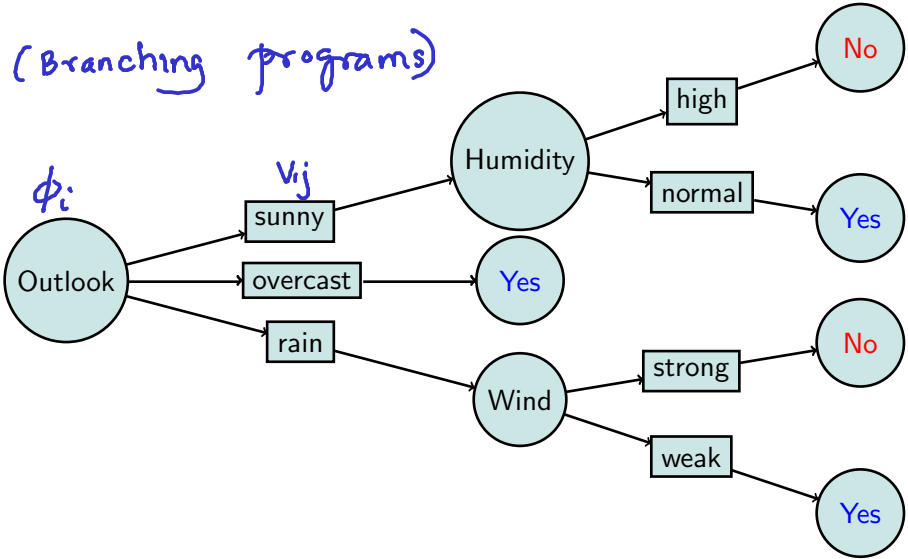
Lecture 26: Other (Non-linear) Classifiers: Boosting in Decision Tree Learning and Support Vector Classification

Instructor: Prof. Ganesh Ramakrishnan

Recap: The Canonical Playtennis Dataset

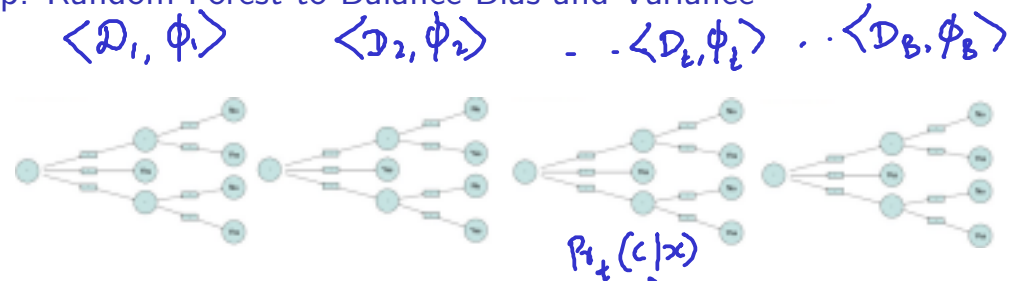
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Recap: Decision Trees: Cascade of step functions



Recap: **Bagging** (**B**ootstrap **AG**gregation) Illustrated through Decision Trees

Recap: Random Forest to Balance Bias and Variance



- Decision for a new test point $\mathbf{x}$: $\Pr(c \mid \mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T} \Pr_t(c \mid \mathbf{x})$
- Each single decision tree, viewed as an estimator of the *ideal* tree has high variance, with very less bias (assumptions)
- But since the decision trees T_i and T_j are uncorrelated, when decision is averaged out across them, it tends to be very accurate.

Bias and Variance

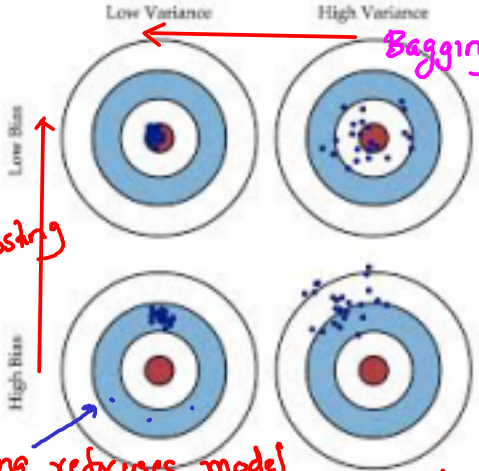
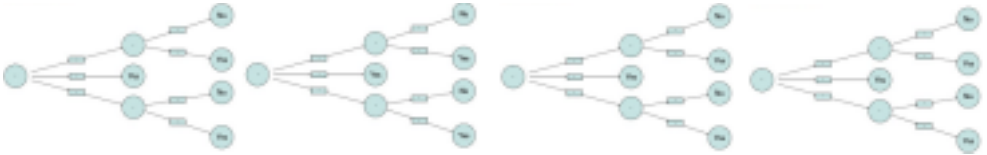


Figure 1: The distance of the cluster from the eye represents bias and the spread of the cluster represents variance.

From **Bagging** to **Boosting**

Weak Models: From **Bagging** to **Boosting**



Bagging: Ensemble of **Independently Weakly Learnt** Models (Eg: Trees $\{T_s\}_1^B$):

$\Pr(c \mid \mathbf{x}) = \frac{1}{|B|} \sum_{t=1}^{B} \Pr_t(c \mid \mathbf{x})$

Weak ⇒ Performs better than random

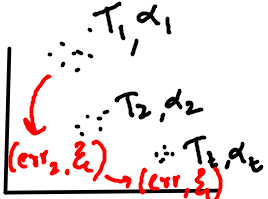
Trained on subset of data using subset of features

Boosting: Sequence of iteratively learnt weak models

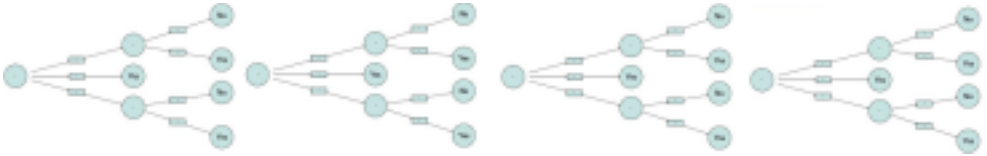
$\Pr(c|x) = \frac{1}{|B|} \sum_{t=1}^{B} \alpha_t \Pr_t(c|x)$

α_t → wt of t^{th} model

ξ_i = wt of i^{th} example: BOOSTING

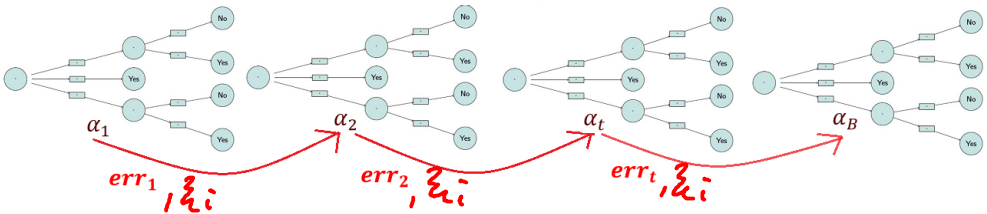


Weak Models: From **Bagging** to **Boosting**



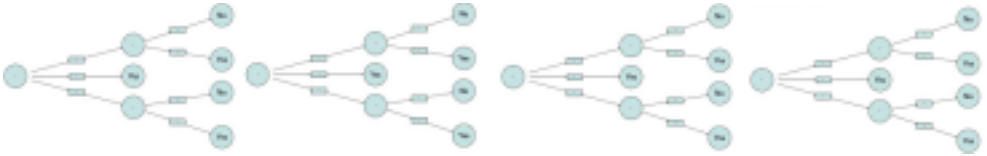
Bagging: Ensemble of **Independently Weakly Learnt** Models (Eg: Trees $\{T_s\}_1^B$):
$\Pr(c \mid \mathbf{x}) = \frac{1}{|B|} \sum_{t=1}^{B} \Pr_t(c \mid \mathbf{x})$

$T_t = f(\xi_i),\ \alpha_t = f'(err_t),\ \xi_i = f''(\alpha_t, T_t)$

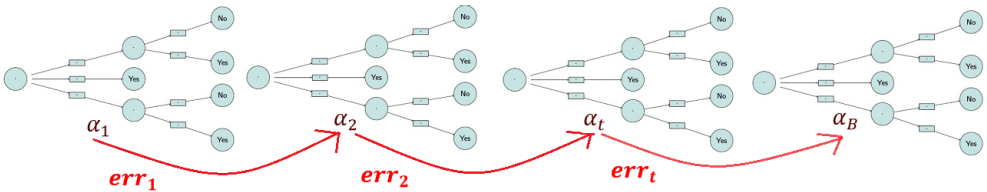


Boosting: Wtd combinations of **Iteratively Weakly Learnt** Models (Eg: Trees $\{\alpha_t, T_t\}_1^B$):

Weak Models: From **Bagging** to **Boosting**

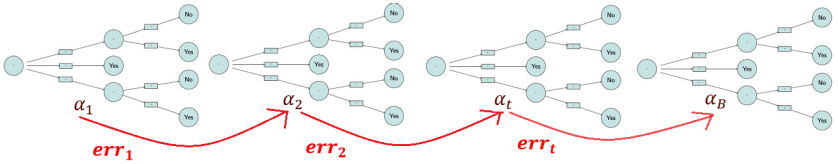


Bagging: Ensemble of **Independently Weakly Learnt** Models (Eg: Trees $\{T_s\}_1^B$):
$\Pr(c \mid \mathbf{x}) = \frac{1}{|B|} \sum_{t=1}^{B} \Pr_t(c \mid \mathbf{x})$



Boosting: Wtd combinations of **Iteratively Weakly Learnt** Models (Eg: Trees $\{\alpha_t, T_t\}_1^B$):
$\Pr(c \mid \mathbf{x}) = \frac{1}{|B|} \sum_{t=1}^{B} \alpha_t \Pr_t(c \mid \mathbf{x})$ where $\alpha_t = (1/2) \ln((1 - err_t)/err_t)$ $err_t \uparrow \downarrow \; \alpha_t \downarrow \uparrow$

Adaptive Boosting of Iteratively Learnt Weak Models



Error driven weighted linear combinations of models: $\alpha_t = (1/2)\ln((1 - err_t)/err_t)$

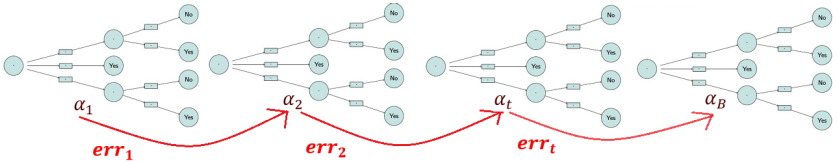
ξ_i = If confident T_t (α_t high) makes mistake on $x^{(i)}$, ξ_i should be high

$\sim$

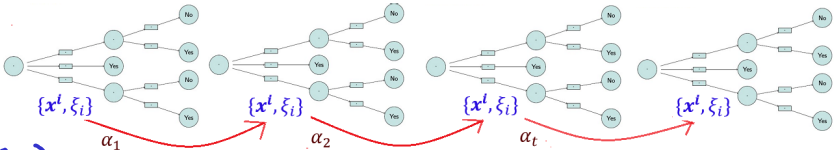
= If any model T_t (ind of α_t) is correct on x^i, ξ_i should be low

= If not-so-confident T_t (α_t low) makes mistake on $x^{(i)}$ ξ_i should be small

Adaptive Boosting of Iteratively Learnt Weak Models



Error driven weighted linear combinations of models: $\alpha_t = (1/2)\ln\left((1 - err_t)/err_t\right)$



if $y^{(i)} = T_t(x^i)$, $\xi_i = \xi_i$ (unchanged)

Reweighting of each data instance $\mathbf{x}^{(i)}$ before learning the next model T_t:

$\xi_i = \xi_i \exp\left(\alpha_t \delta\left(y^{(i)} \neq T_t\left(\mathbf{x}^{(i)}\right)\right)\right)$. Note that $err_t = \frac{\sum_{i=1}^{m} \xi_i \delta\left(y^{(i)} \neq T_t\left(\mathbf{x}^{(i)}\right)\right)}{\sum_{i=1}^{m} \xi_i}$

Adaboost Algorithm

(Adaptive boosting)

Sampling uniformly at random

Initialize each instance weight $\xi_i = \frac{1}{m}$. For $t = 1$ to B do:

1. Learn the t^{th} model T_t by weighing example $\mathbf{x}^{(i)}$ by ξ_i
2. Compute the corresponding error on the training set $err_t = \frac{\sum_{i=1}^{m} \xi_i \delta\left(y^{(i)} \neq T_t\left(\mathbf{x}^{(i)}\right)\right)}{\sum_{i=1}^{m} \xi_i}$
3. Compute the error driven weighted linear factor for T_t: $\alpha_t = (1/2) \ln\left((1 - err_t)/err_t\right)$
4. Reweigh each data instance $\mathbf{x}^{(i)}$ before learning the next model: $\xi_i = \xi_i \exp\left(\alpha_t \delta\left(y^{(i)} \neq T_t\left(\mathbf{x}^{(i)}\right)\right)\right)$.

Importance sampling

Adaboost Algorithm: Motivation (Tutorial 9)

- Freund & Schapire, 1995: Converting a "weak" PAC[1] learning algorithm that performs just slightly better than random guessing into one with arbitrarily high accuracy.
- Let $C_t(\mathbf{x}) = \sum_{j=1}^{t} \alpha_j T_j(\mathbf{x})$ be the boosted linear combination of classifiers until t^{th} iteration.
- Let the error to be minimized over α_t be the sum of its exponential loss on each data point,

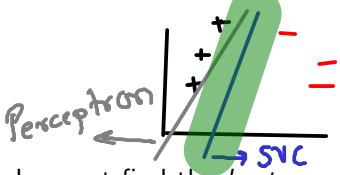
misclassification error — Adaboost minimizes.

$$\mathbf{E}_t = \sum_{i=1}^{m} \delta\left(y^{(i)} \neq sign\left(C_t\left(\mathbf{x}^{(i)}\right)\right)\right) \leq \sum_{i=1}^{m} \exp\left(-y^{(i)} C_t\left(\mathbf{x}^{(i)}\right)\right)$$

- Claim1: The error that is the sum of exponential loss on each data point is an upper bound on the simple sum of training errors on each data point
- Claim2: $\alpha_t = (1/2)\ln\left((1 - err_t)/err_t\right)$ actually minimizes this upper bound.
- Claim3: If each classifier is slightly better than random, that is if $err_t < 1/K$, Adaboost achieves zero training error exponentially fast

[1] http://web.cs.iastate.edu/~honavar/pac.pdf

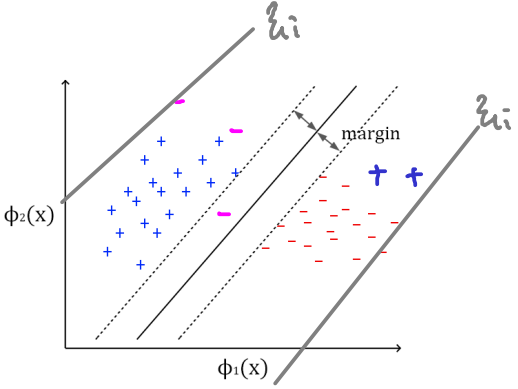
Support Vector Classification



- Perceptron does not find the *best* seperating hyperplane, it finds *any* seperating hyperplane.
- In case the initial $\mathbf{w}$ does not classify all the examples, the seperating hyperplane corresponding to the final $\mathbf{w}^*$ will often pass through an example.
- The seperating hyperplane does not provide enough breathing space – this is what SVMs address and we already saw that for regression!

- Perceptron does not find the *best* seperating hyperplane, it finds *any* seperating hyperplane.
- In case the initial $\mathbf{w}$ does not classify all the examples, the seperating hyperplane corresponding to the final $\mathbf{w}^*$ will often pass through an example.
- The seperating hyperplane does not provide enough breathing space – this is what SVMs address and we already saw that for regression!
  - **We now quickly do the same for classification**

Support Vector Classification: Separable Case



Q: What if you did not have perfect seperability

$w^T \phi(x) + b \geq 1 - \xi_i$

$\mathbf{w}^\top \phi(\mathbf{x}) + b \geq +1$ *for* $y = +1$

$\mathbf{w}^\top \phi(\mathbf{x}) + b \leq -1$ *for* $y = -1$

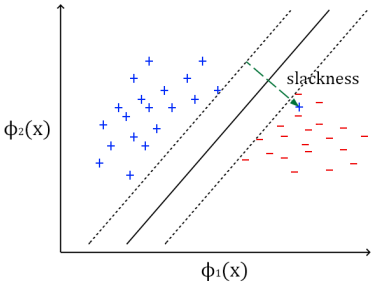
$\mathbf{w}, \phi \in \mathbb{R}^m$

$w^T \phi(x) + b \leq -1 + \xi_i$

There is large margin to seperate the +ve and -ve examples

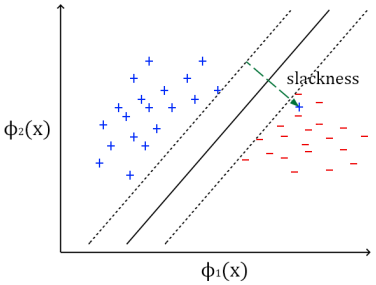
Margin expression?

Support Vector Classification: Non-separable Case



When the examples are not linearly seperable, we need to consider the slackness ξ_i (always $+$ve) of each example $\mathbf{x}^{(i)}$ (how far a misclassified point is from the seperating hyperplane):

Support Vector Classification: Non-separable Case



When the examples are not linearly seperable, we need to consider the slackness ξ_i (always +ve) of each example $\mathbf{x}^{(i)}$ (how far a misclassified point is from the seperating hyperplane):

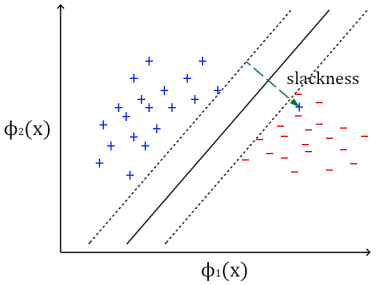
$\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b \geq +1 - \xi_i$ (*for* $y^{(i)} = +1$)

$\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b \leq -1 + \xi_i$ (*for* $y^{(i)} = -1$)

Multiplying $y^{(i)}$ on both sides, we get:

$$y^{(i)}(\mathbf{w}^\top \phi(x^{(i)}) + b) \geq (1 - \xi_i)$$

Support Vector Classification: Non-separable Case



When the examples are not linearly seperable, we need to consider the slackness ξ_i (always +ve) of each example $\mathbf{x}^{(i)}$ (how far a misclassified point is from the seperating hyperplane):

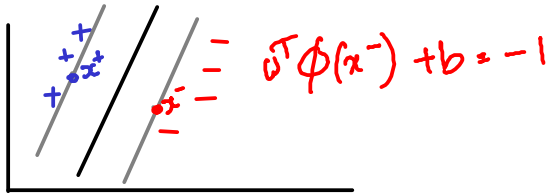
$\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b \geq +1 - \xi_i$ (*for* $y^{(i)} = +1$)

$\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b \leq -1 + \xi_i$ (*for* $y^{(i)} = -1$)

Multiplying $y^{(i)}$ on both sides, we get:

$$y^{(i)}(\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i, \ \forall i = 1, \ldots, n$$

Maximize the margin



- We maximize the margin $(\phi(\mathbf{x}^+) - \phi(\mathbf{x}^-))^\top [\frac{\mathbf{w}}{\|\mathbf{w}\|}]$
- Here, $\mathbf{x}^+$ and $\mathbf{x}^-$ lie on boundaries of the margin.
- Recall that $\mathbf{w}$ is perpendicular to the separating surface
- We project the vectors $\phi(\mathbf{x}^+)$ and $\phi(\mathbf{x}^-)$ on $\mathbf{w}$, and normalize by $\mathbf{w}$ as we are only concerned with the direction of $\mathbf{w}$ and not its magnitude

Simplifying the margin expression

- Maximize the margin $(\phi(\mathbf{x}^+) - \phi(\mathbf{x}^-))^\top [\frac{\mathbf{w}}{\|\mathbf{w}\|}]$
- At $\mathbf{x}^+$: $y^+ = 1$, $\xi^+ = 0$ hence, $(\mathbf{w}^\top \phi(\mathbf{x}^+) + b) = 1$ — (1)
  At $\mathbf{x}^-$: $y^- = -1$, $\xi^- = 0$ hence, $-(\mathbf{w}^\top \phi(\mathbf{x}^-) + b) = 1$ — (2)

$$\left(\phi(x^+) - \phi(x^-)\right)^\top \frac{w}{\|w\|} = \frac{2}{\|w\|}$$

Max margin $\Rightarrow$ $\max \frac{2}{\|w\|} \equiv \min \frac{1}{2}\|w\|^2$

→ Regularization & max margin classification coincide!!

Simplifying the margin expression

- Maximize the margin $(\phi(\mathbf{x}^+) - \phi(\mathbf{x}^-))^\top [\frac{\mathbf{w}}{\|\mathbf{w}\|}]$
- At $\mathbf{x}^+$: $y^+ = 1$, $\xi^+ = 0$ hence, $(\mathbf{w}^\top \phi(\mathbf{x}^+) + b) = 1$ —(1)
  At $\mathbf{x}^-$: $y^- = -1$, $\xi^- = 0$ hence, $-(\mathbf{w}^\top \phi(\mathbf{x}^-) + b) = 1$ —(2)
- Adding (2) to (1),
  $\mathbf{w}^\top(\phi(\mathbf{x}^+) - \phi(\mathbf{x}^-)) = 2$
- *Thus, the margin expression to maximize is:* $\frac{2}{\|\mathbf{w}\|}$

Formulating the objective

- Problem at hand: Find $\mathbf{w}^*, b^*$ that maximize the margin.
- $(\mathbf{w}^*, b^*) = \arg\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$ OR $\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2$
  s.t. $y^{(i)}(\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i$ and $\xi_i \geq 0, \forall i = 1, \ldots, n$
- However, as $\xi_i \to \infty$, $1 - \xi_i \to -\infty$

$\Rightarrow$ All constraints trivially satisfied by letting $\|\mathbf{w}\| \to 0$ & $b \to 0$

Formulating the objective

- Problem at hand: Find $\mathbf{w}^*, b^*$ that maximize the margin.
- $(\mathbf{w}^*, b^*) = \arg\max_{\mathbf{w},b} \frac{2}{\|\mathbf{w}\|}$
  s.t. $y^{(i)}(\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i$ and
  $\xi_i \geq 0$, $\forall i = 1, \ldots, n$
- However, as $\xi_i \to \infty$, $1 - \xi_i \to -\infty$
- Thus, with arbitrarily large values of ξ_i, the constraints become easily satisfiable for any $\mathbf{w}$, which defeats the purpose.
- *Hence, we also want to minimize the ξ_i's. E.g.*, minimize $\sum \xi_i$

$$(w^*, b^*) = \underset{w,b}{\operatorname{argmin}} \ \frac{1}{2}\|w\|^2 + C\sum \xi_i$$
$$\text{s.t. } \forall i \quad y^i(w^T \phi(x^i) + b) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

Optionally: $C\sum \xi_i^2$ instead of $C\sum \xi_i$

Objective

- $(\mathbf{w}^*, b^*, \xi_i^*) = \underset{\mathbf{w},b,\xi_i}{\arg\min} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$
  s.t. $y^{(i)}(\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i$ and
  $\xi_i \geq 0$, $\forall i = 1, \ldots, n$
- Instead of maximizing $\frac{2}{\|\mathbf{w}\|}$, minimize $\frac{1}{2} \|\mathbf{w}\|^2$
  ($\frac{1}{2} \|\mathbf{w}\|^2$ *is monotonically decreasing with respect to* $\frac{2}{\|\mathbf{w}\|}$)
- C determines the trade-off between the error $\sum \xi_i$ and the margin $\frac{2}{\|\mathbf{w}\|}$

Support Vector Machines
Dual Objective

2 Approaches to Showing Kernelized Form for Dual

Monkey work (Jump)

1. **Approach 1:** The Reproducing Kernel Hilbert Space and Representer theorem (Generalized from derivation of Kernel Logistic Regression, Tutorial 7, Problem 3)
   See `http://qwone.com/~jason/writing/kernel.pdf` for list of kernelized objectives
2. **Approach 2:** Derive using First principles (provided for completeness in Tutorial 9)

Donkey work !

Approach 1: Recall Representer Theorem & RKHS

1. (Optional) The solution $f^* \in \mathcal{H}$ (Hilbert space) to the following problem

$$f^* = \arg\min_{f \in \mathcal{H}} \sum_{i=1}^{m} \mathbf{E}\left(f\left(\mathbf{x}^{(i)}\right), y^{(i)}\right) + \Omega(\|f\|_K)$$

   can be always written as $f^*(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i K(\mathbf{x}, \mathbf{x}^{(i)})$, provided $\Omega(\|f\|_K)$ is a ....

2. More specifically, if $f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$ and $K(\mathbf{x}', \mathbf{x}) = \phi^T(\mathbf{x})\phi(\mathbf{x}')$ then the solution $\mathbf{w}^* \in \Re^n$ to the following problem

Note: I need NOT be differentiable!

$$(\mathbf{w}^*, b^*) = \arg\min_{\mathbf{w}, b} \sum_{i=1}^{m} \mathbf{E}\left(f\left(\mathbf{x}^{(i)}\right), y^{(i)}\right) + \Omega(\|\mathbf{w}\|_2)$$

   can be always written as $\phi^T(\mathbf{x})\mathbf{w}^* + b = \sum_{i=1}^{m} \alpha_i K(\mathbf{x}, \mathbf{x}^{(i)})$, provided $\Omega(\|\mathbf{w}\|_2)$ is a monotonically increasing function of $\|\mathbf{w}\|_2$. $\Re^n$ is the Hilbert space and $K(., \mathbf{x}) : \mathcal{X} \to \Re$ is the **Reproducing (RKHS) Kernel**

The Representer Theorem and SVC

1. The SVC Objective

$$(\mathbf{w}^*, b^*, \xi_i^*) = \arg\min_{\mathbf{w},b,\xi_i} C\sum_{i=1}^{m}\xi_i + \frac{1}{2}\|\mathbf{w}\|_2^2$$

s.t. $y^{(i)}(\mathbf{w}^\top\phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i$ and
$\xi_i \geq 0,\ \forall i = 1, \ldots, m$

2. Can be rewritten as

$$(\mathbf{w}^*, b^*, \xi_i^*) = \arg\min_{\mathbf{w},b,\xi_i} C\sum_{i=1}^{m}\xi_i + \frac{1}{2}\|\mathbf{w}\|_2^2$$

s.t. $\xi_i \geq 0,\ \xi_i \geq 1 - y^i(w^T\phi(x^i) + b)$

$\Leftrightarrow \xi_i \geq \max(0, 1 - y^i(w^T\phi(x^i) + b)$

$\Leftrightarrow \xi_i = \max(0, 1 - y^i(w^T\phi(x^i) + b)$ since each ξ_i should be as small as possible

The Representer Theorem and SVC

1. The SVC Objective

$$(\mathbf{w}^*, b^*, \xi_i^*) = \arg\min_{\mathbf{w},b,\xi_i} C\sum_{i=1}^{m}\xi_i + \frac{1}{2}\|\mathbf{w}\|_2^2$$

s.t. $y^{(i)}(\mathbf{w}^\top\phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i$ and
$\xi_i \geq 0,\ \forall i = 1, \ldots, m$

2. Can be rewritten as

$$(\mathbf{w}^*, b^*, \xi_i^*) = \arg\min_{\mathbf{w},b,\xi_i} C\sum_{i=1}^{m}\xi_i + \frac{1}{2}\|\mathbf{w}\|_2^2$$

s.t. $\max\left(1 - y^{(i)}(\mathbf{w}^\top\phi(\mathbf{x}^{(i)}) + b), 0\right) = \xi_i$

Hinge loss

3. That is,

$$(\mathbf{w}^*, b^*, \xi_i^*) = \arg\min_{\mathbf{w},b,\xi_i} C\sum_{i=1}^{m}\max\left(1 - y^{(i)}(\mathbf{w}^\top\phi(\mathbf{x}^{(i)}) + b), 0\right) + \frac{1}{2}\|\mathbf{w}\|_2^2$$

The Representer Theorem and SVC (contd.)

1. If $f(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$ and $K(\mathbf{x}', \mathbf{x}) = \phi^T(\mathbf{x})\phi(\mathbf{x}')$ and given the SVC objective

$$(\mathbf{w}^*, b^*, \xi_i^*) = \underset{\mathbf{w},b,\xi_i}{\arg\min}\, C\sum_{i=1}^{m} \max\left(1 - y^{(i)}(\mathbf{w}^\top\phi(\mathbf{x}^{(i)}) + b), 0\right) + \frac{1}{2}\|\mathbf{w}\|_2^2$$

2. Setting $\mathbf{E}\left(f\left(\mathbf{x}^{(i)}\right), y^{(i)}\right) = C\max\left(1 - y^{(i)}(\mathbf{w}^\top\phi(\mathbf{x}^{(i)}) + b), 0\right)$ and $\Omega(\|\mathbf{w}\|_2) = \frac{1}{2}\|\mathbf{w}\|_2^2$, we can apply the Representer theorem to SVC, so that $\phi^T(\mathbf{x})\mathbf{w}^* + b = \sum_{i=1}^{m}\alpha_i K(\mathbf{x}, \mathbf{x}^{(i)})$

Substitute

Approach 2: Derivation using First principles

Derivation similar to that for Support Vector Regression, and provided for completeness in extra slide deck as well as in Tutorial 9

- The dual optimization problem becomes:

$$\max_{\alpha} -\frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y^{(i)} y^{(j)} K\left(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}\right) + \sum_i \alpha_i$$

s.t.
$\alpha_i \in [0, C]$, $\forall i$ and
$\sum_i \alpha_i y^{(i)} = 0$

Derivation of SVC Dual using First Principles (Solution to Tutorial 9, Problem 1)

Dual Objective

Dual function

- Let $L^*(\alpha, \mu) = \min_{\mathbf{w},b,\xi} L(\mathbf{w}, b, \xi, \alpha, \mu)$
- By weak duality theorem, we have:
  $L^*(\alpha, \mu) \leq \min_{\mathbf{w},b,\xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$
  s.t. $y^{(i)}(\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b) \geq 1 - \xi_i$, and
  $\xi_i \geq 0$, $\forall i = 1, \ldots, n$
- The above is true for any $\alpha_i \geq 0$ and $\mu_i \geq 0$
- Thus,

$$\max_{\alpha,\mu} L^*(\alpha, \mu) \leq \min_{\mathbf{w},b,\xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i$$

Dual objective

- In case of SVM, we have a strictly convex objective and linear constraints – therefore, strong duality holds:

$$\max_{\alpha,\mu} L^*(\alpha,\mu) = \min_{w,b,\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

- This value is precisely obtained at the $(\mathbf{w}^*, b^*, \xi^*, \alpha^*, \mu^*)$ that satisfies the necessary (and sufficient) optimality conditions
- Assuming that the necessary and sufficient conditions (KKT or Karush–Kuhn–Tucker conditions) hold, our objective becomes:

$$\max_{\alpha,\mu} L^*(\alpha,\mu)$$

- $L(w, b, \xi, \alpha, \mu) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{n} \xi_i + \sum_{i=1}^{n} \alpha_i (1 - \xi_i - y^{(i)}(\mathbf{w}^\top \phi(\mathbf{x}^{(i)}) + b)) - \sum_{i=1}^{n} \mu_i \xi_i$
- We obtain $\mathbf{w}$, b, ξ in terms of α and μ by setting $\nabla_{w,b,\xi} L = 0$:
  - **w.r.t. w:** $\mathbf{w} = \sum_{i=1}^{n} \alpha_i y^{(i)} \phi(\mathbf{x}^{(i)})$
  - **w.r.t. b:** $-b \sum_{i=1}^{n} \alpha_i y^{(i)} = 0$
  - **w.r.t. ξ_i:** $\alpha_i + \mu_i = C$
- Thus, we get:
  $L(w, b, \xi, \alpha, \mu)$
  $= \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y^{(i)} y^{(j)} \phi^\top(\mathbf{x}^{(i)}) \phi(\mathbf{x}^{(j)}) + C \sum_i \xi_i + \sum_i \alpha_i - \sum_i \alpha_i \xi_i -$
  $\sum_i \alpha_i y^{(i)} \sum_j \alpha_j y^{(j)} \phi^\top(\mathbf{x}^{(j)}) \phi(\mathbf{x}^{(i)}) - b \sum_i \alpha_i y^{(i)} - \sum_i \mu_i \xi_i$
  $= -\frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y^{(i)} y^{(j)} \phi^\top(\mathbf{x}^{(i)}) \phi(\mathbf{x}^{(j)}) + \sum_i \alpha_i$

- The dual optimization problem becomes:

$$\max_{\alpha} -\frac{1}{2}\sum_{i}\sum_{j}\alpha_i\alpha_j y^{(i)}y^{(j)}\phi^{\top}(\mathbf{x}^{(i)})\phi(\mathbf{x}^{(j)}) + \sum_{i}\alpha_i$$

s.t.
$\alpha_i \in [0, C]$, $\forall i$ and
$\sum_i \alpha_i y^{(i)} = 0$

- Deriving this did not require the complementary slackness conditions
- Conveniently, we also end up getting rid of μ

Lecture 26b: Unsupervised Learning: Dimensionality Reduction, Embeddings, PCA etc

Instructor: Prof. Ganesh Ramakrishnan

Recall: Supervised Feature Selection based on Gain

- S is a sample of training examples, p_{C_i} is proportion of examples with class C_i in S
- Entropy measures impurity of S: $H(S) \equiv \sum_{i=1}^{K} -p_{C_i} \log_2 p_{C_i}$
- Selecting R best attributes: Let $\mathcal{R} = \emptyset$
- $Gain(S, \phi_i)$ = expected **Gain** due to choice of ϕ_i Eg: Gain based on **entropy** - $Gain(S, \phi_i) \equiv H(S) - \sum_{v \in Values(\phi_i)} \frac{|S_v|}{|S|} H(S_v)$

  Do:

  1. $\phi^* = \underset{\phi_i \setminus \mathcal{V}}{\text{argmax}}\, Gain(S, \phi_i)$
  2. $\mathcal{R} = \mathcal{R} \cup \{\phi^*\}$

  Until $|\mathcal{R}| = R$

Q: Other measures of **Impurity** based **Gain**: **Gini Index**, **Classification Error**, *etc.*

From Supervised to Unsupervised Dimensionality Reduction

Recap: Feature Selection

- Supervised (greedy) Feature Selection based on Gain
- Optimally feature subset Selection (Eg: Lasso or Iterative Hard Thresholding)

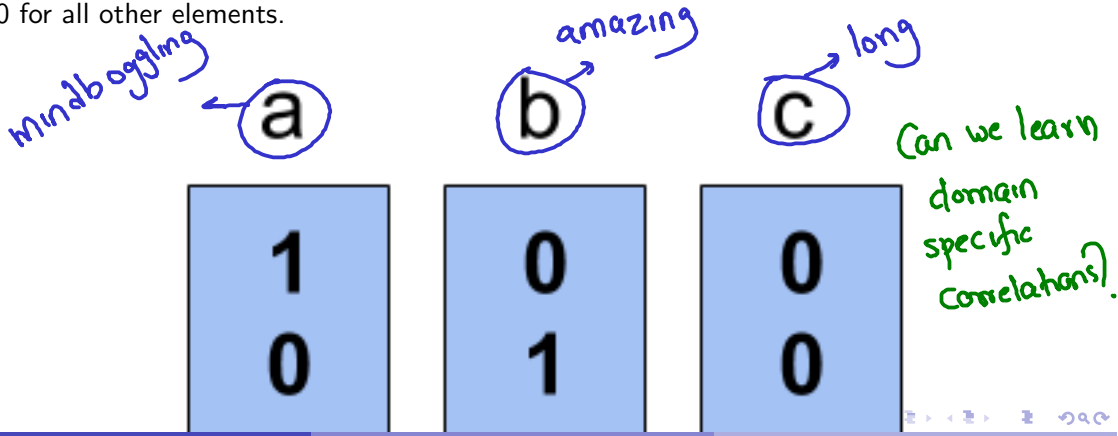
Question: What if one wants to do dimensionality reduction independent of any class-based supervision?

Recap: One Hot Encoding for Characters

- With 3 characters in vocabulary, a,b and c, what would be the best encoding to inform each character occurrence to the network?
- One Hot Encoding: Give a unique key k to each character in alpha-numeric order, and encode each character with a vector of vocabulary size, with a 1 for the k^{th} element, and 0 for all other elements.

Recap: One Hot Encoding for Characters

- With 3 characters in vocabulary, a,b and c, what would be the best encoding to inform each character occurrence to the network?
- One Hot Encoding: Give a unique key k to each character in alpha-numeric order, and encode each character with a vector of vocabulary size, with a 1 for the k^{th} element, and 0 for all other elements.



Encoding Words

How to encode the words for the task of labeling a drama reviews as "liked" or "not liked" ?

- Review 1: The drama was interesting, loved the way each scene was directed. I simply loved everything in the drama.
- Review 2: I had three boring hours. Very boring to watch.
- Review 3: I liked the role each that was assigned to each super star. Especially loved the performance of actor.
- Review 4: Though I hate all the dramas of the director, this one was an exception with lot of entertainment.

Encoding Words

How to encode the words for the task of labeling a ''*drama*' reviews as "liked" or "not liked" ?

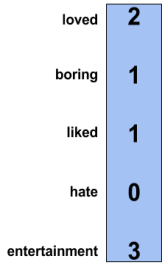
- One Hot Encoding of Words.
- Bag Of Words, similar to one hot encoding of characters
  - Use the vocabulary of highly frequent words in reviews.
  - Use the word frequency in each review instead of "1".

Encoding Words

How to encode the words for the task of labeling a ‘’*drama*’ reviews as "liked" or "not liked" ?

- One Hot Encoding of Words.
- Bag Of Words, similar to one hot encoding of characters
  - Use the vocabulary of highly frequent words in reviews.
  - Use the word frequency in each review instead of "1".

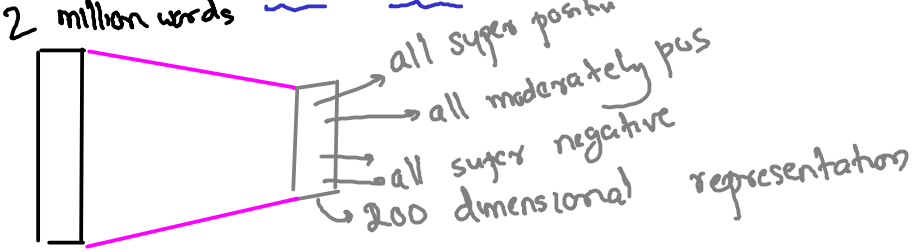
A review in Bag Of Words Form:-



(Word) Embedding: Motivation

Limitations of Bag of Words or One Hot Encoding for words

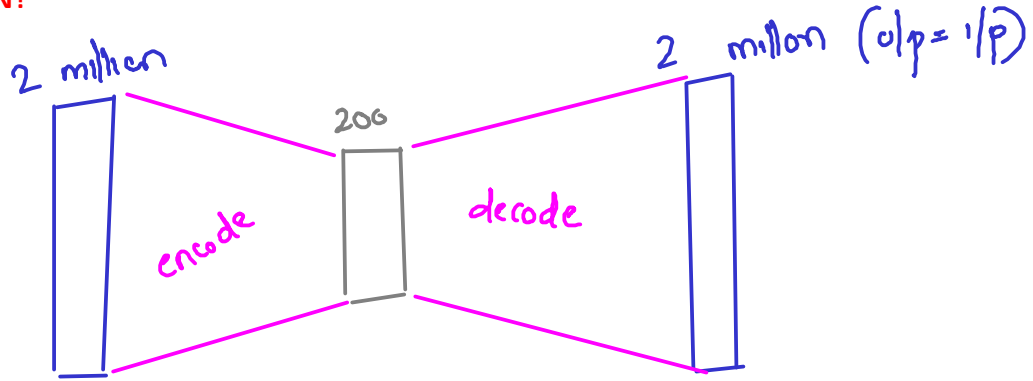
- High Dimension: In real life scenario, the vocabulary size could be huge.
- Lacks Contextual Similarity - e.g. liked and loved are contextually similar words.



(Word) Embedding: Motivation [Auto encoder]

Dimensionality Reduction techniques.

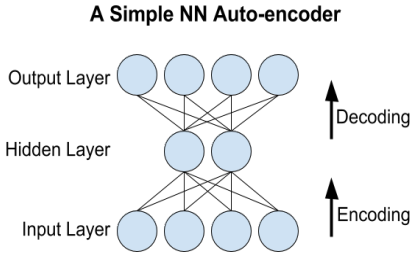
- Bag of Frequent Words: Contextual similarity is still lacking.
- **What happens if one passes a one hot encoded word as both input and output to a NN?**



(Word) Embedding: Motivation

Dimensionality Reduction techniques.

- Bag of Frequent Words: Contextual similarity is still lacking.
- **What happens if one passes a one hot encoded word as both input and output to a NN?**
- **NN Auto-encoder: Output has same form as input**. We extract the encoded vector from the hidden layer.



(Word) Embedding: Motivation

After unsupervised training with lot of online data, can a machine answer questions like:-

- King - Man + Woman = ? Queen
- If France:Paris, then Japan:? Tokyo

Can embedding reflect this?

(Word) Embedding: Motivation

A Hypothetical Word Vector Representation

	King	Queen	Woman	Princess
Royalty	0.98	0.98	0.01	0.93
Masculinity	0.98	0.04	0.02	0.02
Femininity	0.05	0.92	0.99	0.95
Age	0.7	0.6	0.5	0.2

(Word) Embedding: Motivation

A Hypothetical Word Vector Representation

	King	Queen	Woman	Princess
Royalty	0.98	0.98	0.01	0.93
Masculinity	0.98	0.04	0.02	0.02
Femininity	0.05	0.92	0.99	0.95
Age	0.7	0.6	0.5	0.2

- What would be the vector for Man?

(Word) Embedding: Motivation (Interesting papers on injecting logical rules)

A Hypothetical Word Vector Representation

	King	Queen	Woman	Princess
Royalty	0.98	0.98	0.01	0.93
Masculinity	0.98	0.04	0.02	0.02
Femininity	0.05	0.92	0.99	0.95
Age	0.7	0.6	0.5	0.2

- What would be the vector for Man? [0.1 0.9 0.1 0.7]
- King - Man + Woman = ?

(Word) Embedding: Motivation

A Hypothetical Word Vector Representation

	King	Queen	Woman	Princess
Royalty	0.98	0.98	0.01	0.93
Masculinity	0.98	0.04	0.02	0.02
Femininity	0.05	0.92	0.99	0.95
Age	0.7	0.6	0.5	0.2

- What would be the vector for Man?
- King - Man + Woman = ?
- If King:Man then Queen:?

(Word) Embedding: Motivation

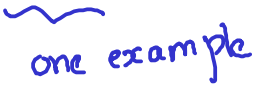
A Hypothetical Word Vector Representation

	King	Queen	Woman	Princess
Royalty	0.98	0.98	0.01	0.93
Masculinity	0.98	0.04	0.02	0.02
Femininity	0.05	0.92	0.99	0.95
Age	0.7	0.6	0.5	0.2

- What would be the vector for Man? **[0.01, 0.98, 0.05, 0.6]'**
- King - Man + Woman = ? Queen (Subtraction/addition approximates vector for Queen)
- If King:Man then Queen:? Woman (Vector differences of both pairs nearly same)

(Word) Embedding

- **(Word) Embedding:** Building a low-dimensional vector representation from corpus of text, which preserves contextual similarity.
- **In simple terms:** Need an efficient language of numbers which deep neural networks can understand as close as possible to the way we understand words.
- **Training:** Continuous Bag of Words Model.

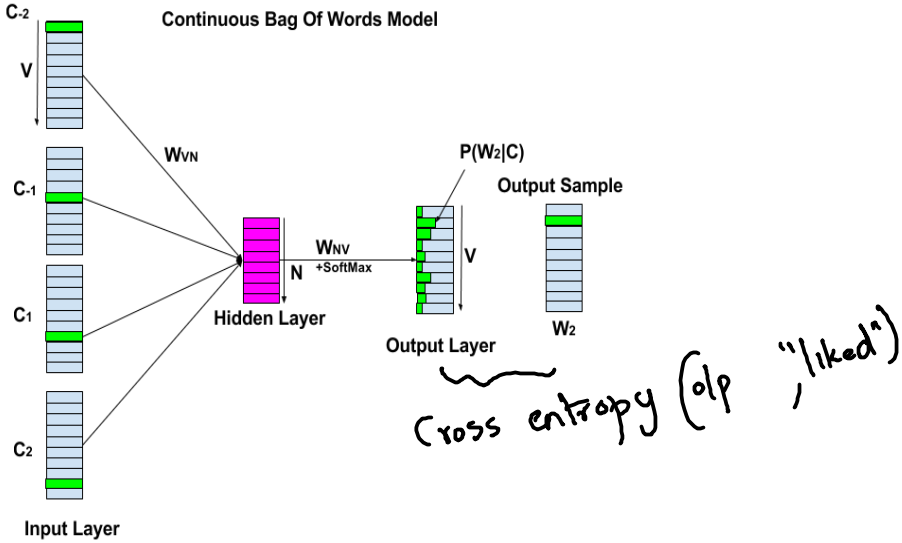


(Word) Embedding

- **(Word) Embedding:** Building a low-dimensional vector representation from corpus of text, which preserves contextual similarity.
- **In simple terms:** Need an efficient language of numbers which deep neural networks can understand as close as possible to the way we understand words.
- **Training:** Continuous Bag of Words Model.
  - Take words in one hot encoded form.
  - Consider the sentence, "... I really liked the **drama**....".
  - Take a N (say 5) word window around each such word w.
  - Train the Neural Network with (top V frequent) context words set $\mathcal{C}$ as input and the central word w as output.
  - For the example above use C = {"I", "really", "the" "drama"} as input and w = **"liked"** as output.

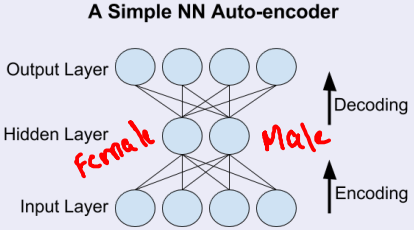
drop "the", "an" "a"

(Word) Embedding: Unsupervised Training

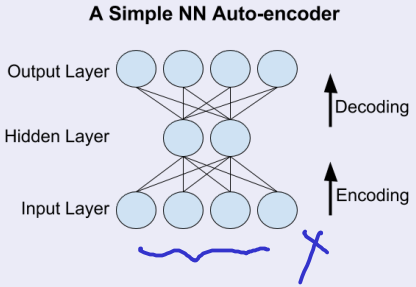


What if we want Embeddings to be Orthogonal?

(or logically consistent with rules)

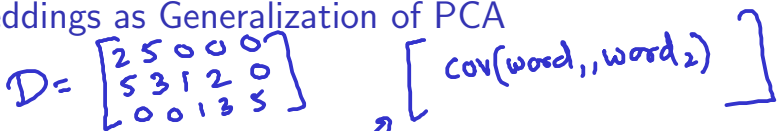


What if we want Embeddings to be Orthogonal?



- Let $\mathbf{X}$ be a random vector and Γ its covariance matrix.
- **Principal Component Analysis:** Find a rotation of the original coordinate system and express $\mathbf{X}$ in that system so that each new coordinate expresses as much as possible of the variability in $\mathbf{X}$ as can be expressed by a linear combination of the n entries of $\mathbf{X}$. This has application in data transformation, feature discovery, feature selection and so on.

Embeddings as Generalization of PCA



- Let $\mathbf{X}$ be a random vector and Γ its covariance matrix. Let $\mathbf{e}_1, \ldots, \mathbf{e}_n$ be the n (normalized) eigenvectors of Γ.
- The n principal components of $\mathbf{X}$ are said to be $\mathbf{e}_1^T\mathbf{X}$, $\mathbf{e}_2^T\mathbf{X}$, ..., $\mathbf{e}_n^T\mathbf{X}$.

Embeddings in decreasing imp → orthogonal embeddings.

$\lambda_1 > \lambda_2 > \cdots > \lambda_n$

1. Let $p(X_1) = \mathcal{N}(0,1)$ and $p(X_2) = \mathcal{N}(0,1)$ and $cov(X_1, X_2) = \theta$. Find all the principal components of the random vector $\mathbf{X} = [X_1, X_2]^T$. [Tutorial 10]
2. Now, let $\mathbf{Y} = \mathcal{N}(\mathbf{0}, \Sigma) \in \Re^p$ where $\Sigma = \lambda^2 I_{p\times p} + \alpha^2 ones(p,p)$ for any $\lambda, \alpha \in \Re$. Here, $I_{p\times p}$ is a $p \times p$ identity matrix while $ones(p,p)$ is a $p \times p$ matrix of 1's. Find atleast one principal component of $\mathbf{Y}$. [Tutorial 10]