Consider the following training data points and their corresponding α values

$$\boldsymbol{x}^1 = \begin{bmatrix} 2 \\ 0 \\ -1.5 \end{bmatrix}, y^1 = +1, \alpha^1 = 0.5 \qquad \boldsymbol{x}^2 = \begin{bmatrix} 1 \\ -0.5 \\ 0 \end{bmatrix}, y^2 = +1, \alpha^2 = 2$$

$$\boldsymbol{x}^3 = \begin{bmatrix} 5 \\ 0 \\ 2 \end{bmatrix}, y^3 = -1, \alpha^3 = 2 \qquad \boldsymbol{x}^4 = \begin{bmatrix} 13 \\ -8 \\ 7 \end{bmatrix}, y^4 = -1, \alpha^4 = 0$$

$$\boldsymbol{x}^5 = \begin{bmatrix} 7 \\ -6 \\ 0 \end{bmatrix}, y^5 = -1, \alpha^5 = 0.5 \qquad \boldsymbol{x}^6 = \begin{bmatrix} -5 \\ 0 \\ -5 \end{bmatrix}, y^6 = +1, \alpha^6 = 0$$

Which of the above are considered support vectors?

Any point with $\alpha > 0$. In other words: $\mathbf{x}^1$, $\mathbf{x}^2$, $\mathbf{x}^3$, and $\mathbf{x}^5$

Use the above data to compute the separating hyperplane **w** found by the linear SVM.

$$w = \sum_i \alpha^i y^i \boldsymbol{x}^i = 0.5 \times 1 \times \begin{bmatrix} 2 \\ 0 \\ -1.5 \end{bmatrix} + 2 \times 1 \times \begin{bmatrix} 1 \\ -0.5 \\ 0 \end{bmatrix} + 2 \times -1 \times \begin{bmatrix} 5 \\ 0 \\ 2 \end{bmatrix} + 0.5 \times -1 \times \begin{bmatrix} 7 \\ -6 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{10.5} \\ \mathbf{2} \\ \mathbf{-4.75} \end{bmatrix}$$

Consider a Bayesian classifier and a data set of containing 30 features. Assume there are 100 training data points. We report the highest Likelihood classifier given *n* features below:

#Feats	**1**	**2**	**3**	**4**	**5**	**6**	**7**	…	**21**
MaxL	$6x10^{-23}$	$1x10^{-21}$	$2x10^{-20}$	$4x10^{-20}$	$8x10^{-19}$	$2x10^{-19}$	$4x10^{-19}$		$1x10^{-9}$

#Feats	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**
MaxL	$1x10^{-8}$	$1x10^{-7}$	$6x10^{-7}$	$4x10^{-6}$	$1x10^{-5}$	$5x10^{-5}$	$9x10^{-5}$	$2x10^{-4}$	$3x10^{-4}$

What is the first feature removal step (how many features left?) when AIC no longer improves?

AIC: log(L)-k:

k	30	29	28	27	**26**	
AIC	-38.1	-37.5	-37.3	-36.9	**-37.5**	

Removing 4th feature, no longer benefits AIC.

What is the first feature selection step (how many features chosen?) when BIC no longer improves?

BIC: log(L)-0.5 x k x log(m):
log(100)=4.6

k	1	2	3	**4**		
AIC	-53.5	-53.0	-52.3	**-53.9**		

Adding 4th featyre, no longer benefits BIC

Consider the following Principal Components:

$$\boldsymbol{u}_1 = \begin{bmatrix} 0.41 \\ -0.41 \\ 0 \\ 0.82 \end{bmatrix} \qquad \boldsymbol{u}_2 = \begin{bmatrix} 0 \\ 0.53 \\ 0.80 \\ 0.27 \end{bmatrix}$$

For each data point below, calculate the weights z for each principal component.

$$\boldsymbol{x}^1 = \begin{bmatrix} -4.4 \\ 3.8 \\ -1.0 \\ -9.2 \end{bmatrix}$$

$$z_j^i = \boldsymbol{u}_j^T \boldsymbol{x}^i \qquad z_1^1 = \boldsymbol{u}_1^T \boldsymbol{x}^1 = -4.4 \times 0.41 + 3.8 \times -0.41 - 1.0 \times 0 - 9.2 \times 0.82 \approx \mathbf{-11}$$

$$z_2^1 = \boldsymbol{u}_2^T \boldsymbol{x}^1 = -4.4 \times 0 + 3.8 \times 0.53 - 1.0 \times 0.8 - 9.2 \times .27 \approx \mathbf{-1.3}$$

$$\boldsymbol{x}^2 = \begin{bmatrix} 0.1 \\ 0.7 \\ 1.2 \\ 0.7 \end{bmatrix}$$

$$z_j^i = \boldsymbol{u}_j^T \boldsymbol{x}^i \qquad z_1^2 = \boldsymbol{u}_1^T \boldsymbol{x}^2 = 0.1 \times 0.41 + 0.7 \times -0.41 + 1.2 \times 0 + 0.7 \times 0.82 \approx \mathbf{0.3}$$

$$z_2^1 = \boldsymbol{u}_2^T \boldsymbol{x}^1 = 0.1 \times 0 + 0.7 \times 0.53 + 1.2 \times 0.8 + 0.7 \times .27 \approx \mathbf{1.5}$$

$$\boldsymbol{x}^3 = \begin{bmatrix} 2.3 \\ -2.5 \\ -0.3 \\ 4.4 \end{bmatrix}$$

$z_j^i = \boldsymbol{u}_j^T \boldsymbol{x}^i$ $\quad z_1^2 = \boldsymbol{u}_1^T \boldsymbol{x}^2 = 2.3 \times 0.41 - 2.5 \times -0.3 + 1.2 \times 0 + 4.4 \times 0.82 \approx \mathbf{5.3}$

$$z_2^1 = \boldsymbol{u}_2^T \boldsymbol{x}^1 = 2.3 \times 0 - 2.5 \times 0.53 - 0.3 \times 0.8 + 4.4 \times .27 \approx \mathbf{-0.4}$$

Using only **u**₁ , compute the reconstruction error for each of the above data points.

$\tilde{x}^i = z_1^i \boldsymbol{u}_1$

$Err^i = \sum_j \left(\boldsymbol{x}_j^i - \tilde{\boldsymbol{x}}_j^i\right)^2$

$$\tilde{x}^1 = \begin{bmatrix} -4.5 \\ 4.5 \\ 0 \\ -9.02 \end{bmatrix}$$

$$\begin{aligned} Err^1 &= ((-4.4 + 4.5)^2 + (3.8 - 4.5)^2 + (-1 - 0)^2 + (-9.2 + 9.02)^2) \\ &= 0.01 + 0.49 + 1 + 0.03 = \mathbf{1.53} \end{aligned}$$

1.53

$$\tilde{x}^2 = \begin{bmatrix} 0.12 \\ -0.12 \\ 0 \\ 0.25 \end{bmatrix}$$

$$Err^2 = ((0.1 - 0.12)^2 + (0.7 + 0.12)^2 + (1.2 - 0)^2 + (0.7 - 0.25)^2)$$
$$= 0.0004 + 0.67 + 1.44 + 0.20 \approx \mathbf{2.3}$$

~~1.7~~ **2.3**

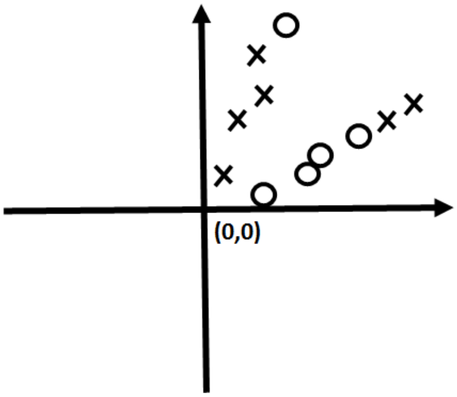
$$\tilde{x}^3 = \begin{bmatrix} 2.2 \\ -2.2 \\ 0 \\ 4.3 \end{bmatrix}$$

$$Err^3 = ((2.3 - 2.2)^2 + (-2.5 + 2.2)^2 + (-0.3 - 0)^2 + (4.4 - 4.3)^2)$$
$$= 0.01 + 0.09 + 0.09 + 0.01 \approx \mathbf{0.2}$$

0.2

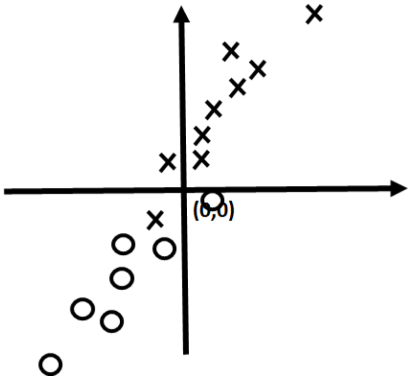
We observe the following data points in two dimensions. We wish to define new dimensions to better describe the data. Should we use PCA, ICA, or NMF?

A



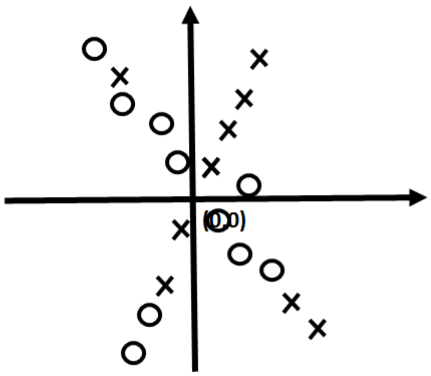
NMF – all features are positive

B



PCA – one main direction of variance

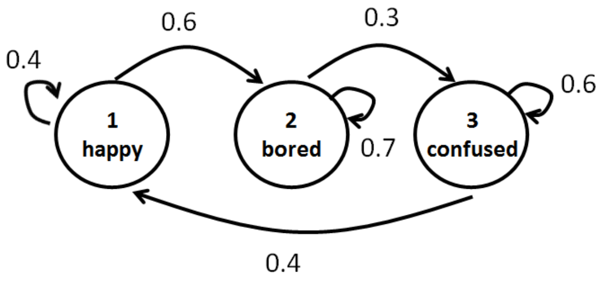
C



ICA – two non-orthogonal directions of variance, both positive and negative

Consider an HMM with 3 states:
1: happy,   2: bored,   3: confused
And 4 potential outputs at each time (each output is punctuation mark)
1: !   2: ?   3: .   4: -

The transition matrix is capture by the diagram



The initial probabilities are:

$$\pi_{happy} = 0.2$$
$$\pi_{bored} = 0.6$$
$$\pi_{confused} = 0.2$$

The emission matrix is ϕ:

State\Obs	!	?	.	-
Happy	0.8	0.1	0.1	0
Bored	0	0.2	0.7	0.1
Confused	0.1	0.7	0	0.2

Which of the following are impossible state sequences:

a) happy, confused, happy, happy

impossible

b) bored, bored, bored, confused

possible

c) happy, happy, bored, confused, happy

possible

d) bored, bored, happy, happy

impossible

Presume the sequence:
?, ?, ?, ?

What is the forward value $\alpha_1(happy)$

$\phi_{?,happy}\pi_{happy} = 0.1 \times 0.2 = \mathbf{0.02}$

What is the forward value $\alpha_2(confused)$

$\alpha_2(confused) = \phi_{?,confused} \sum_{mood} A_{confused,mood}\alpha_1(mood)$

Compute $\alpha_1(mood)$:

$\alpha_1(happy) = 0.02$ (see above) $\alpha_1(bored) = 0.2 \times 0.6 = 0.12$

$\alpha_1(confused) = 0.7 \times 0.1 = 0.07$

$\alpha_2(confused) = \phi_{?,confused} \sum_{mood} A_{confused,mood}\alpha_1(mood)$

$= 0.7 \times [0 \times 0.2 + 0.3 \times 0.12 + 0.6 \times 0.07] \approx \mathbf{0.06}$

Final answer: **0.06**

Note: Viterbi was only briefly covered in class. I include this problem for review/clarification of the process but will make Viterbi worth few points (if any) on the exam.

Using the Viterbi algorithm, what is the sequence of most probable states for each observation sequence below:

!, ?, !

First compute $\delta_t(i)$ for all t and i

$\delta_1(i) = \pi_i \phi_{!,i}$ $\quad$ $\delta_1(happy) = 0.2 \times 0.8 = 0.16$ $\quad$ $\delta_1(bored) = 0.6 \times 0 = 0$

$\delta_1(confused) = 0.2 \times 0.1 = 0.02$

$\delta_2(i) = \phi_{?,i} \max_j A_{i,j} \delta_1(j)$

$$\delta_2(happy) = 0.1 \times \max_j A_{happy,j} \delta_1(j) = 0.1 \times \max \begin{cases} 0.4 \times 0.16 \\ 0 \times 0 \\ 0.4 \times 0.02 \end{cases} = 0.0064$$

$$\delta_2(bored) = 0.2 \times \max_j A_{bored,j} \delta_1(j) = 0.2 \times \max \begin{cases} 0.6 \times 0.16 \\ 0.7 \times 0 \\ 0 \times 0.02 \end{cases} = 0.0192$$

$$\delta_2(confused) = 0.7 \times \max_j A_{confused,j} \delta_1(j) = 0.7 \times \max \begin{cases} 0 \times 0.16 \\ 0.3 \times 0 \\ 0.6 \times 0.02 \end{cases} = 0.0084$$

$\delta_3(i) = \phi_{!,i} \max_j A_{i,j} \delta_1(j)$

$$\delta_3(happy) = 0.2 \times \max_j A_{happy,j} \delta_2(j) = 0.8 \times \max \begin{cases} 0.4 \times 0.0064 \\ 0 \times 0.0192 \\ 0.4 \times 0.0084 \end{cases} = 0.00269$$

$$\delta_3(bored) = 0.6 \times \max_j A_{bored,j} \delta_2(j) = 0 \times \max \begin{cases} 0.6 \times 0.0064 \\ 0.7 \times 0.0192 \\ 0 \times 0.0084 \end{cases} = 0$$

$$\delta_3(confused) = 0.2 \times \max_j A_{confused,j} \delta_2(j) = 0.1 \times \max \begin{cases} 0 \times 0.0064 \\ 0.3 \times 0.0192 \\ 0.6 \times 0.0084 \end{cases} = 0.00058$$

Find largest $\delta_3(i)$: **q_3=happy**

What $\delta_2(i)$ maximizes $\delta_3(happy)$?, **q_2=confused**

What $\delta_1(i)$ maximizes $\delta_2(confused)$? **q_1=confused**

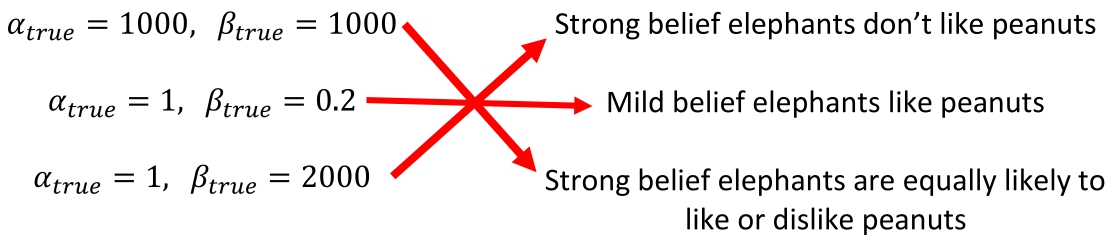
How can we compute the emission probabilities for each state if we have a set of training sequences where both the observations and the underlying states are known? (Describe in a sentence and/or write a mathematical expression.)

For each entry $\phi_{obs,state}$ count the number of times that *state* occurs and the number of times the *state* occurs with the *obs* observation. Then $\phi_{obs,state} = \frac{\#D(obs \wedge state)}{\#D(state)}$

How can we compute the emission probabilities if we have a set of training sequences where the observations are known but the underlying states are NOT known? (Describe in 1-3 sentences and/or write 1-3 mathematical expressions.)

Use expectation-maximization. First hypothesize HMM parameters ϕ, π, A and then compute the probability of state s_i at each time: $S_t(i)=P(q_t=s_i|o_1,\ldots,o_t)$ using forward and backward probabilities. Then re-compute $\phi_{obs,state} = \frac{\sum_{t|o_t=obs} S_t(state)}{\sum_i \sum_{t|o_t=j} S_t(state)}$

We seek to apply MAP to learn P(X|Y; θ), where X is a binary feature “likes peanuts?” (yes or no) and Y represents the class “is an elephant” (true or false). We use α_{true} and β_{true} to represent the probability prior belief X=yes and X=no given Y=true. **Match each $\boldsymbol{\alpha_{true}}$, $\boldsymbol{\beta_{true}}$ pairing on the left with its corresponding meaning on the right.**



$\alpha_{true} = 1000, \;\; \beta_{true} = 1000$

$\alpha_{true} = 1, \;\; \beta_{true} = 0.2$

$\alpha_{true} = 1, \;\; \beta_{true} = 2000$

Strong belief elephants don’t like peanuts

Mild belief elephants like peanuts

Strong belief elephants are equally likely to like or dislike peanuts

What are the effects of changing the following parameters in gradient ascent learning for logistic regression?

1: ε

Decrease in ε decreases the amount of change to the learned classifier parameters at each update. Increasing ε will increase the amount of change.

2: λ

Increase in λ will decrease the effect of the corresponding regularization term. Decreasing λ will increase the effect of the corresponding regularization term.

We have data points with 10 features each. Each feature is a numeric value along the real number line. We wish to learn a classifier to label each data point as one of four classes. Using logistic regression (including +b term!), how many parameters must we learn?

We have (4-1)x(10+1) = 3x11 = 33 features to learn. 11 parameters per classifier, 3 classifiers total. 11 parameters correspond to 10 features plus an offset (+b).

How many parameters must we learn if we wish to classify the data above into only **two** classes using a linear SVM?

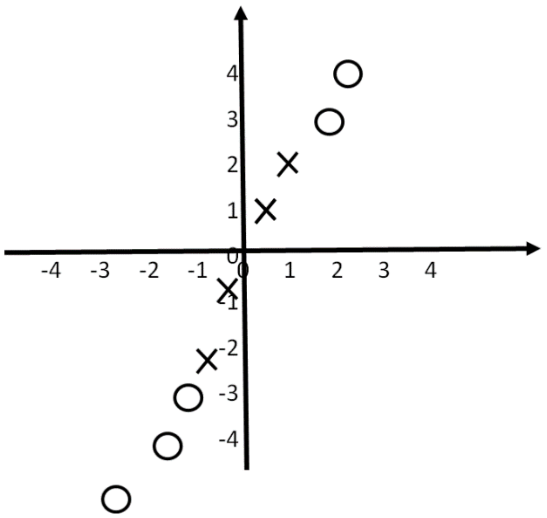
11 parameters, including **w** (10 features) and b.

How many parameters must we learn if we wish to classify the data above into only two classes using a quadratic kernel SVM?

There are two potential interpretations. Either we can say we learn 10^2+1 = 101 parameters, for all quadratic combinations of all features, or we say the number of parameters are the number of training point weights α^i. Most of these weights will be drive to zero and the remaining ones will be used to identify support vectors for classification.

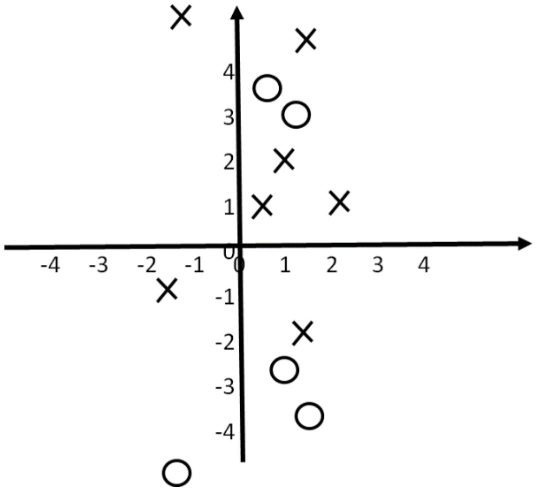
Define a mapping function that will allow a linear separator to distinguish between the two classes for each set of data points. (x_1 is horizontal axis, x_2 is vertical axis)

A



$\varphi(x_1, x_2) = (2x_1+x_2)^2$

B

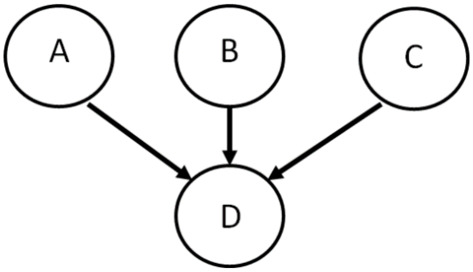


$\varphi(x_1, x_2) = (x_2, x_2^2, x_2^3)$

This will allow us to find a cubic function along x_2 that will rise above 0 and fall below 0 four times, corresponding to the four alternating regions in the y axis.

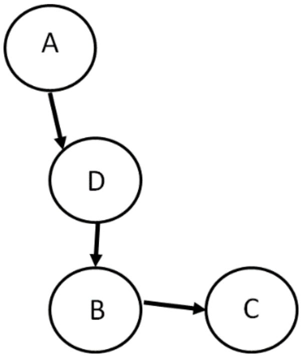
Write the formula to find the total joint probability for each of the following Bayesian Networks (to find P(A,B,C,D)).

A:



P(A)P(B)P(C)P(D|A,B,C)

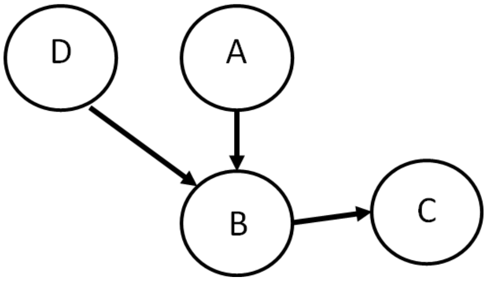
B:



P(A)P(D|A)P(B|D)P(C|B)

This is basically a Markov model!

C:



Compute P(subset of variables)

P(D)P(A)P(B|A,D)P(C|B)

For each Bayes net above, how can we find the value of P(B)

Example A:
P(B)

Example B:
$\sum_{A,D} \boldsymbol{P(A)P(D|A)P(B|D)}$

Example C:

$$\sum_{A,D} \boldsymbol{P(D)P(A)P(B|A,D)}$$

Let us presume we wish to classify whether an animal is a mammal or a bird, and we use 30 features x_1, … x_{30} as a basis for classification. (For example, x_1 can be size, x_2 can be typical speed of motion, x_3 can be blood temperature, etc.) For a given animal we observe the following feature vector: $\boldsymbol{x} = \begin{bmatrix} 10 \\ 1 \\ 55 \\ \vdots \\ 2 \\ 16 \\ 240 \end{bmatrix}$

and we are informed of the following probabilities:

$P(x_1=10|Y=bird)=0.01$ $P(x_2=1|Y=bird)=0.2$ … $P(x_{29}=16|Y=bird)=0.05$ $P(x_{30}=240|Y=bird)=0.2$

$P(x_1=10|Y=mammal)=0.06$ $P(x_2=1|Y=mammal)=0.08$ … $P(x_{29}=16|Y=mammal)=0.1$ $P(x_{30}=240|Y=mammal)=0.07$

If we use Naïve Bayes on a standard computer, we will find $P(x_1,\ldots,x_{30}|Y=bird)=0$ and $P(x_1,\ldots,x_{30}|Y=mammal)=0$, despite the fact that **none** of the terms $P(x_i|Y=mammal)=0$ nor $P(x_i|Y=bird)=0$

a) Why does this happen on a computer?

When fractional values become sufficiently low, the computer will round to 0.

b) What mathematical operation can we use to prevent this problem?

We can use logairthms to convert the product of low probabilities, like 10^{-20} to the sum of 2-3 digit negative numbers like -50 .