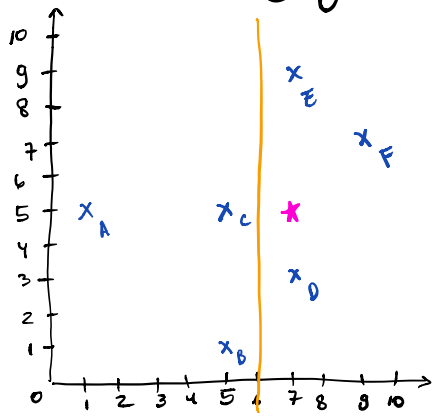
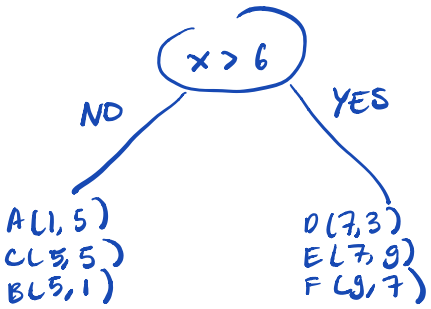
IA ML Past Paper 2013/2014

1.a. Decision Boundaries

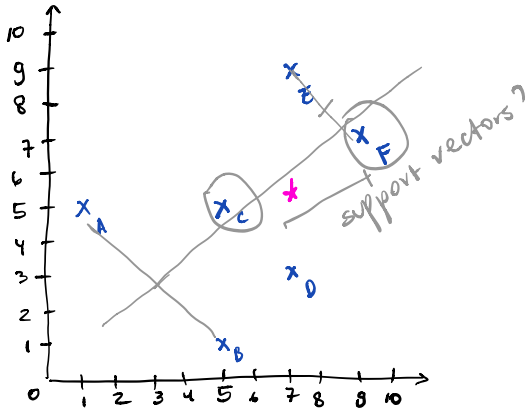
positive: A(1, 5) C(5, 5) E(7, 9)
negative: B(5, 1) D(7, 3) F(9, 7)

i. Plot and see DB given by DT.

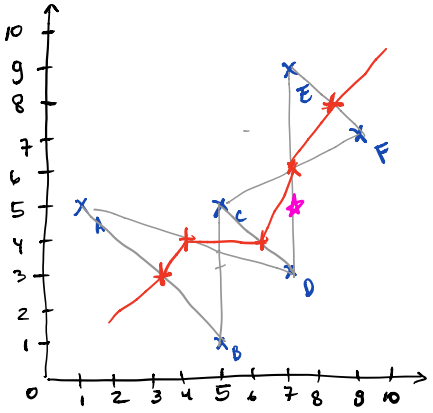




ii. DB given by an SVM with linear kernel function



iii. K-nearest neighbors, k=1



iv. new data point
G: (7, 5)
positive!

for all.

b. Optimisation

i.- Gradient Descent Algorithm

initialise $\underline{w}$
while $E(\underline{w})$ is unacceptably high
calculate $g \leftarrow \partial E / \partial w$
$w \leftarrow w - \eta g$
end while
return $\underline{w}$

$\eta \rightarrow$ learning rate / step-size

- Effect of learning rate η
  - η too small → slow
  - η too big → skip over the minima
- Local minima problem
  Is this a problem if we use log. regr and $E(w)$ = negative log-LL
  No because the error function is CONVEX
  ↓
  second derivative always positive
  i.e. any local minimum is global!

ii. Purpose of regularisation
- add complexity param. to a learning algorithm
- capacity control
- prevent overfitting / underfitting

IAML 2013-2014

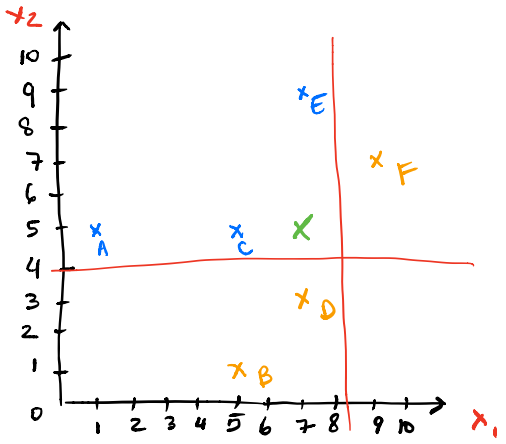
1.a. Decision Boundaries.

Dataset with $[x_1, x_2]$

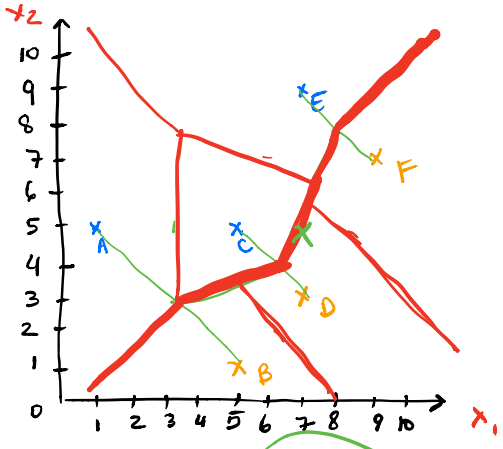
positive → A(1,5) C(5,5) E(7,9)

negative → B(5,1) D(7,3) F(9,7)

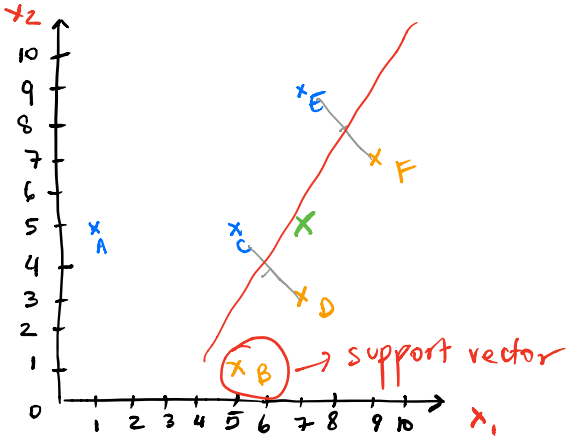
i. Plot DB from DT.



ii. KNN with K = 1



ii. Plot DB from SVM



iv. New point [7,5]

DT (+)

SVM (−)

KNN (−) or (+)

b. Optimisation

i. Gradient descent algorithm

initialise w
while $E(\underline{w})$ unacceptably high
$g \leftarrow \partial E / \partial w$
$w \leftarrow w - \eta g$
end while
return w

- Learning rate
  - too small → slow
  - too high → skip actual minima
- Logistic regression
  - convex
    - any minima is global minima
    - because the second derivative of the error fn. is always positive
    - always converges

ii. Regularisation

add complexity parameter to a learning algorithm
example method: squared error w/ quadratic regularisation

$$E(w) = |y - \phi w|^2 + \lambda |w|^2$$
$$\hat{w} = (\phi^T \phi + \lambda I)^{-1} \phi^T y$$

optimisation approach: calculus

2. Nearest Neighbor

$[x_1, x_2, x_3]$.

a. How to predict x_3 from x_1 and x_2

calculate euclidean distance of (x_1, x_2) to all of the x_1 and x_2 s in the dataset.

i.e. $D(\underline{x}, \underline{y}) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$

Pick the closest datasets and average x_1, x_2 over the points.

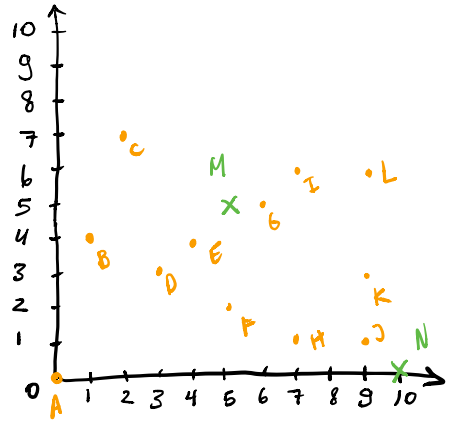
e.g. $[1, 2, 3]$ and $[4, 5, 6]$ are closest

$$x_3 = \frac{3+6}{2} = 4.5$$

Parameters: distance function, no. of nearest neighbors

ANSWER: Do nearest neighbor on $[x_1, x_2, x_3]$.
Since we know x_1, x_2, decide x_3 by taking mean of the points in its neighbors.

b. New points M: [5, 5] M: [10, 0]



M nearest neighbors = E, G

$X_{3M} = \frac{9+9}{2} = 9$

N nearest neighbors = J, K, H

$X_{3N} = \frac{1+1+1}{3} = 1$

c. Interpolation or extrapolate

extrapolate: estimation of a value based on extending a known sequence of values

interpolation: estimation of a value within two known values in a sequence of values.

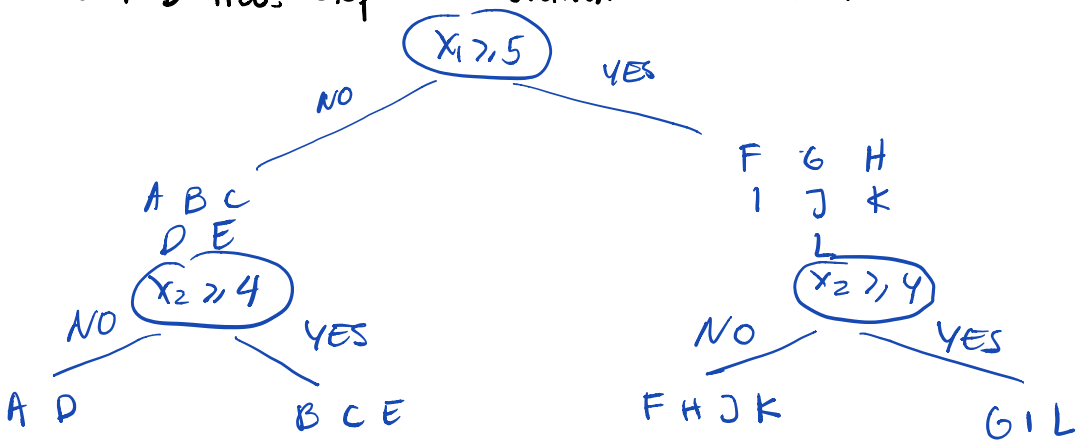
More confident in M → interpolate.
N → extrapolate, out of range

d. k-D tree

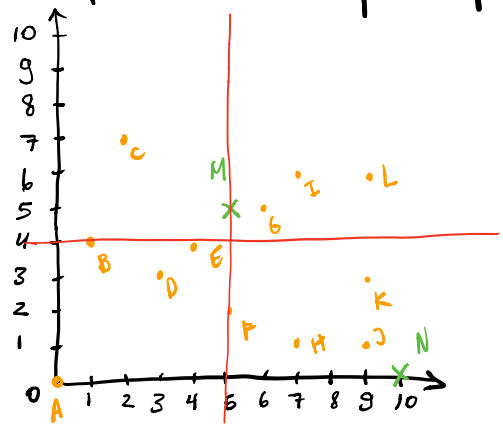
low dimensional real valued data

HOW: pick random dimension, find median, split data, repeat

e. Do k-D tree, stop when branch has 3 or fewer



f. use KD tree to compute predictions



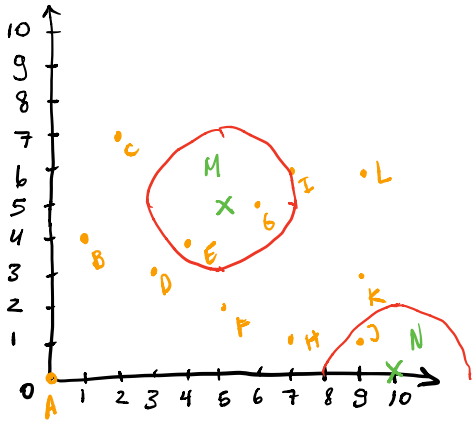
M nearest neighbors = G, I, L

$$x_{3_M} = \frac{9 + 5 + 1}{3} = 5$$

N nearest neighbors = F, H, J, K

$$x_{3_N} = \frac{2 + 1 + 1 + 1}{4} = 1.25$$

g. Parzen window. Radius = 3



M nearest neighbors = E, G

$$x_{3_M} = \frac{9 + 9}{2} = 9$$

N nearest neighbor = J

$$x_{3_N} = 1$$

3. Linear Methods

b. Logistic Regression

$$P(Y=1|X) = \sigma(w_0 + w_1 X)$$

$$\sigma(z) = \frac{1}{1+\exp(-z)}$$

* $\sigma'(z) = \sigma(z) \cdot (1-\sigma(z))$

$l(\underline{w})$ → log likelihood for logistic regression

i. Give equations for $\partial l / \partial w_0$ and $\partial l / \partial w_1$

To maximise, you have to maximise the likelihood of the ENTIRE DATASET.

Therefore, $l(\underline{w}) = y_i \log \sigma(w_0 + w_1 x) + (1-y_i) \log (1 - \sigma(w_0 + w_1 x))$

$$L(\underline{w}) = \sum_{i=1}^{n} y_i \log \sigma(w_0 + w_1 x) + (1-y_i) \log (1 - \sigma(w_0 + w_1 x))$$

$$l(\underline{w}) = \sum_{i=1}^{n} y_i \log \sigma(\underline{w}^T \underline{x}) + (1-y_i) \log (1 - \sigma(\underline{w}^T \underline{x}))$$

$$\partial l / \partial w_j = \sum_{i=1}^{n} y_i \cdot \frac{1}{\sigma(\underline{w}^T \underline{x})} \cdot \sigma(\underline{w}^T \underline{x}) \cdot (1 - \sigma(\underline{w}^T \underline{x})) \cdot x_{ij} + (1-y_i) \cdot \frac{1}{1 - \sigma(\underline{w}^T \underline{x})} \cdot (-\sigma(\underline{w}^T \underline{x})(1 - \sigma(\underline{w}^T \underline{x}))) x_{ij}$$

$$= \sum_{i=1}^{n} y_i (1 - \sigma(\underline{w}^T \underline{x})) x_{ij} + (1-y_i)(-\sigma(\underline{w}^T \underline{x}))$$

$$= \sum_{i=1}^{n} x_{ij} (y_i - y_i \sigma(\underline{w}^T \underline{x}) - \sigma(\underline{w}^T \underline{x}) + y_i \cdot \sigma(\underline{w}^T \underline{x}))$$

$$= \sum_{i=1}^{n} x_{ij} [y_i - \sigma(\underline{w}^T \underline{x})]$$

$$\partial l / \partial (w_0) = \sum_{i=1}^{n} 1 \cdot [y_i - \sigma(z_i)] = \sum_{i=1}^{n} [y_i - \sigma(w_0 + w_1 x)]$$

(× always 1 for w_0 → bias)

$$\partial l / \partial (w_1) = \sum_{i=1}^{n} x_i [y_i - \sigma(z_i)] = \sum_{i=1}^{n} x_i [y_i - \sigma(w_0 + w_1 x)]$$

ii. Dataset collected

Instance	x	y
0	0.0	1
1	−0.5	0

$w = (-1, 1)$ $\quad \eta = 0.2$ $\quad$ gradient ascent

Compute new weight vector $w' = w + \eta g$

Instance	x	y	$p(y=1\|x) = \sigma(w^T x)$	$y_i - \sigma(w^T x)$
0	0.0	1	0.269	0.731
1	−0.5	0	0.182	−0.182

add bias $[1, 0]$, $[1, -0.5]$

$0 \rightarrow [-1 \quad 1]\begin{bmatrix}1\\0\end{bmatrix} = -1 \rightarrow \sigma(-1) = \frac{1}{1+\exp(1)} = 0.269$

$1 \rightarrow [-1 \quad 1]\begin{bmatrix}1\\-0.5\end{bmatrix} = -1 - 0.5 \rightarrow \sigma(-1.5) = \frac{1}{1+\exp(1.5)} = 0.182$

$\partial L/\partial w_0 = (1 \times 0.731) + (1 \times -0.182) = 0.549$

$\partial L/\partial w_1 = (0 \times 0.731) + (-0.5 \times -0.182) = 0.091$

$w' = \begin{bmatrix}-1\\1\end{bmatrix} + 0.2\begin{bmatrix}0.549\\0.091\end{bmatrix} = \begin{bmatrix}-0.8902\\1.0182\end{bmatrix}$

b. Support Vector Machines

i. Maximum Margin Hyperplane

Margin → distance between the decision boundary and the closest training point.

Maximum Margin Hyperplane
→ the decision boundary that has the maximum distance towards the closest training point.

i.e. the hyperplane $w^T x + w_0$ such that

$$\min_i |w^T x_i + w_0| = 1$$

ii. Solution for weight vector of MMH, define SV, how to make a class prediction.

margin → $\min_i \cdot \frac{1}{\|w\|} |w^T x_i + w_0|$

Support vectors are the training instances closest to the maximum margin hyperplane. They have direct bearing on the optimum location of the MMH.

Predict 1 if sign $(w^T x + w_0)$ positive, negative otherwise.

iii. Verify that

$$\phi(x) = \begin{bmatrix} 1 \\ 2x_1 \\ 2x_2 \\ 2x_1^2 \\ 2\sqrt{2}x_1x_2 \\ 2x_2^2 \end{bmatrix}$$

corresponds to the kernel

$K(x,y) = (1 + 2(x \cdot y))^2$ for $x, y \in \mathbb{R}^2$

$K(x_i, x_j) = x_i^T x_2 = \phi(x)^T \phi(y)$

$\phi(x)^T \phi(y) = 1 + 4x_1y_1 + 4x_2y_2 + 4x_1^2y_1^2 + 8x_1x_2y_1y_2 + 4x_2^2y_2^2$

$= 1 + 4(x_1y_1 + x_2y_2) + 4(x_1y_1 + 2x_1x_2y_1y_2 + x_2y_2^2)$

$= 1 + 4(x \cdot y) + 4(x \cdot y)^2$

$= [1 + 2(x \cdot y)]^2$

hence proved

iv. Compare log. regr with SVM linear kernel, i.e. $K(x,y) = x \cdot y$

dense, all points affect result/prediction

sparse, only support vectors affect result.