(1) a) Support Vectors: Positive Samples: $(0,1)$ $(-2,-1)$ $\rightarrow x_1 - x_2 = -1$

Negative Samples: $(0.5,-0.5)$ $\rightarrow x_1 - x_2 = 1$

b) $d = \frac{|g(x)|}{\|w\|}$ hyperplane: $x_1 - x_2 = 0$ $\quad g(x) = w^T x$ $\quad w = \begin{bmatrix} 1 \\ -1 \end{bmatrix} \Rightarrow \|w\| = \sqrt{(1)^2 + (-1)^2}$

$(1,3) \rightarrow d = \frac{|1-(-3)|}{\sqrt{2}} = \frac{4}{\sqrt{2}} = 2\sqrt{2}$

$(1.5,-1) \rightarrow d = \frac{|1.5-(1)|}{\sqrt{2}} = \frac{0.5}{\sqrt{2}} = \frac{\sqrt{2}}{4}$

$(3,0.5) \rightarrow d = \frac{|3-(-0.5)|}{\sqrt{2}} = \frac{3.5}{\sqrt{2}} = \frac{7\sqrt{2}}{4}$

c) Removing $(0.5,-0.5)$ will change the boundry decision since currently it is the closest sample and our support vector.

Removing $(-1,-2.5)$ from current sample data will NOT change the decision boundry since it is not the closest sample.

But if we assume that $(0.5,-0.5)$ is removed then new closest samples will be $(-1,-2.5)$ and $(-0.5,-2)$ which will be our new support vector (they will be on same line) so now if I try to remove $(-1,-2.5)$ again my decision boundry won't change because I still have $(-0.5,-2)$ sample.

d) if new negative sample $(0,2)$ comes out, the classes become linearly inseparable and will definitely affect our decision boundry.

Solution would be considering soft margin classification by using slack variables.

Another option is projecting datapoints into higher dimensional space where they are linearly separable and then use Kernel SVMs to find the decision boundry

① e) (13.10) $L_p = \frac{1}{2}\|w\|^2 + C\sum_t \xi_t$

hyperparameter C is a regularization parameter which defines tradeoff between Margin maximization and loss/error minimization.

when C is very large, focus is more on avoiding misclassification and not maximizing the margin or in other words we define a high penalty for unseparable points and we may store many support vectors and we end up overfitting.

But if C is very small, the focus will be on maximizing the margin and misclassification mistakes are given less importance and it might result in finding a very easy solution and end up underfitting.

f) Soft Margin vs. Hard Margin

- if my data is linearly separable, then I'll just use Hard Margin
- but if my data is noisy or with outliers, Soft Margin provides a principled way to handle this situation

Linear vs Kernel

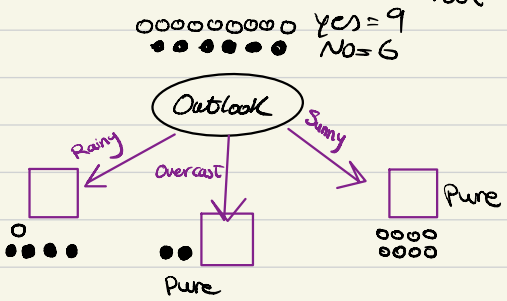
Kernel is a very general method that can be applied to other methods. But if data again is not linearly separable, we can transform the data into higher dimensional space where we can linearly separate classes, in this case Kernel SVMs are very useful.

(2) Impurity at Node, Node Entropy $= -\sum_i p^i \log_2 p^i$

Impurity after split $= -\sum_j \frac{N_j}{N} \sum_i p_j^i \log_2 p_j^i$

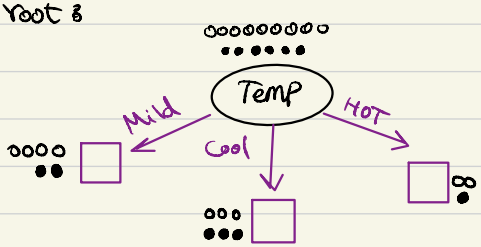
Considering Outlook as root:

$$I_{root} = -\left(\frac{9}{15}\log_2\frac{9}{15} + \frac{6}{15}\log_2\frac{6}{15}\right) = 0.966$$



$$I_{split} = -\left[\frac{5}{15}\left(\frac{1}{5}\log_2\frac{1}{5} + \frac{4}{5}\log_2\frac{4}{5}\right) + \frac{2}{15}\left(0\log_2 0 + 1\log_2 1\right) + \frac{8}{15}\left(1\log_2 1 + 0\log_2 0\right)\right] = 0.24$$

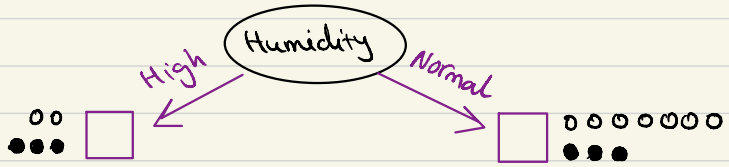
If Temperature is root:



$$I_{Split} = -\left[\frac{6}{15}\left(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}\right) + \frac{6}{15}\left(\frac{3}{6}\log_2\frac{3}{6} + \frac{3}{6}\log_2\frac{3}{6}\right) + \frac{3}{15}\left(\frac{2}{3}\log_2\frac{2}{3} + \frac{1}{3}\log_2\frac{1}{3}\right)\right]$$

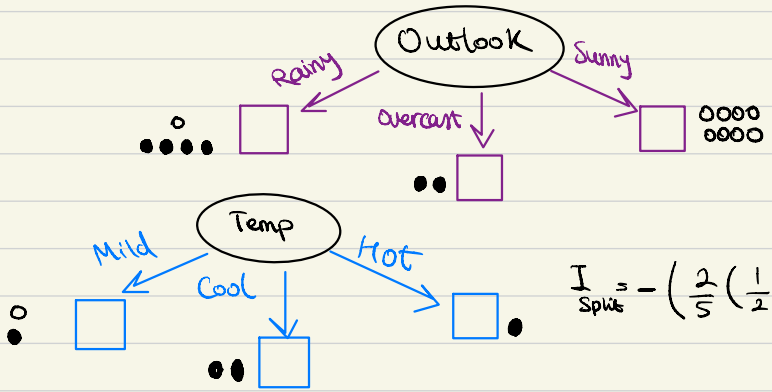
$$= 0.94$$

(2) a) if Humidity is root:

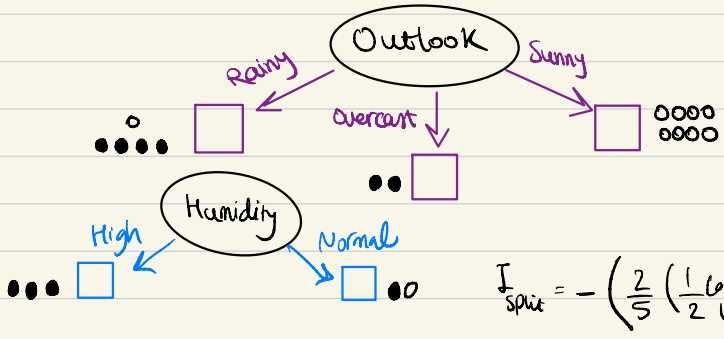


$$I_{Split} = -\left[\frac{5}{15}\left(\frac{2}{5}\log_2\frac{2}{5} + \frac{3}{5}\log_2\frac{3}{5}\right) + \frac{10}{15}\left(\frac{7}{10}\log_2\frac{7}{10} + \frac{3}{10}\log_2\frac{3}{10}\right)\right] = 0.906$$

$\Rightarrow$ Outlook has the lowest impurity, so then we better start root with outlook



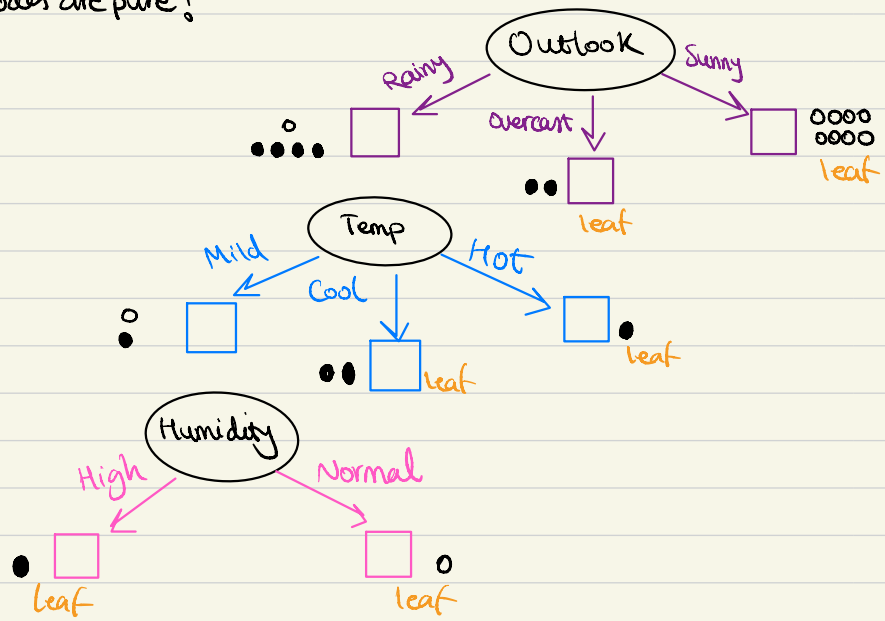
$$I_{Split} = -\left(\frac{2}{5}\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right)\right) = 0.4$$



$$I_{Split} = -\left(\frac{2}{5}\left(\frac{1}{2}\log_2\frac{1}{2} + \frac{1}{2}\log_2\frac{1}{2}\right)\right) = 0.4$$

So after root, both temperate & humidity have same impurity, so it doesn't really matter which comes first.

*All leaf nodes are pure!



b) Overcast, Cool temp, high humidity → will NOT run

Rainy, Hot temp, Normal humidity → will NOT run

(3) a)

Training / Validation accuracy for min node entropy 0.01 is 1.000 / 0.863
" " " 0.05 " 0.999 / 0.863
" " " 0.1 " 0.997 / 0.865
" " " 0.2 " 0.990 / 0.867 ←
" " " 0.4 " 0.979 / 0.861
" " " 0.8 " 0.919 / 0.856
" " " 1 " 0.871 / 0.840
" " " 2 " 0.596 / 0.600

Test accuracy with min node entropy 0.2 is 0.872

b) We can understand from the accuracy results that with minimum node entropy we get the best test accuracy but average validation accuracy which means that we are overfitting our model.

So there should be a tradeoff between node entropy and loss computation and we can realize that by higher node entropy (0.2) we got the maximum accuracy for validation test and also test accuracy with same entropy turned out to be high also, almost as high as validation accuracy which tells us that we are not overfitting our model