1.a. For the positive labels: (-2,-1), (0,1)
For the negative labels: (0,-1), (1,0)

b. The equation of the hyperplane is $-x_1 + x_2 = 0$.
Thus, $w = [-1, 1]$ and $w_0 = 0$.

Given $d^t = \frac{w^T x^t + w_0}{\|w\|}$ we set with $g(x) = w^T x^t + w_0$

$$d_{(0,1)} = \frac{|g(x)|}{\|w\|} = \frac{|0+1|}{\sqrt{(-1)^2+(1)^2}} = \frac{1}{\sqrt{2}}$$

$$d_{(-2,-1)} = \frac{|g(x)|}{\|w\|} = \frac{|2+^-1|}{\sqrt{(-1)^2+(1)^2}} = \frac{1}{\sqrt{2}}$$

$$d_{(-1,1)} = \frac{|g(x)|}{\|w\|} = \frac{|1+1|}{\sqrt{(-1)^2+(1)^2}} = \frac{2}{\sqrt{2}}$$

c. Removing (-1,1) will not change the decision boundary because it is not a support vector.
If we remove both (1,0) and (0,-1) then the decision boundary will change because (1,0) and (0,-1) are support vectors.

d. Yes, the decision boundary will change as this positive sample is outside the positive sample's margin. By eyeballing the points it looks like it is not possible to perfectly separate the classes linearly. I would then consider using soft SVM. This would allow misclassification of this point but keep the rest of the points separated into their correct classes.

e. When C has a large value errors are weighted more. Thus, errors are not tolerated as well within the decision boundary. When c has a small value errors are weighted less and tolerated more within the decision boundary. Therefore, the smaller C is, the more general the model. i.e When C is large, the slack variable is more heavily weighted leading to a smaller margin and vice versa.

f. Hard Margin vs Soft Margin SVM
Hard Margin SVM should be used when there is no noise between classes so they are linearly seperable. Soft Margin, on the other hand, provides a principled way to handle noise and outliers.

Linear versus Kernel
If the data is linearly seperable then we would lean towards using linear SVM. If the data is not linearly seperable then a kernal SVM should be used to transform the data into a higher dimensional space to make the classes seperable.

2.a) Calculations for 1^{st} Split

Interest

$E_{high} = -4/13 \left(1/4 \log_2(1/3) + 3/4 \log_2(3/4) \right) \approx 0.2496$

$E_{med} = -4/13 \left(2/4 \log_2(2/4) + 2/4 \log_2(2/4) \right) \approx 0.3077$

$E_{low} = -5/13 \left(2/5 \log_2(2/5) + 3/5 \log_2(2/5) \right) \approx 0.3734$

$E_{int} = E_{high} + E_{med} + E_{low} \approx 0.9307$

Time of Day

$E_{eve} = -5/13 \left(1/5 \log_2(1/5) + 4/5 \log_2(4/5) \right) \approx 0.2777$

$E_{aft} = -4/13 \left(2/4 \log_2(2/4) + 2/4 \log_2(2/4) \right) \approx 0.3077$

$E_{mor} = -4/13 \left(2/4 \log_2(2/4) + 2/4 \log_2(2/4) \right) \approx 0.3077$

$E_{time} = E_{eve} + E_{aft} + E_{mor} \approx 0.8930$

Past Attendance

$E_2 = -3/13 \left(3/3 \log_2(3/3) + 0 \log_2(0) \right) = 0.0$

$E_1 = -4/13 \left(2/4 \log_2(2/4) + 2/4 \log_2(2/4) \right) = 0.3077$

$E_0 = -6/13 \left(0 \log_2(0) + 0 \log_2(0) \right) = 0.0$

$E_{past} = E_2 + E_1 + E_0 = 0.3077$

Since $E_{past} \leq E_{time} \leq E_{int}$, We split on Past Attendance

2 a) cont'd Calculations for 2^{nd} Split

Past Attendance

≥ 2: Split is pure, no need to calculate. Attend = Yes.

1: 4 Nodes

Interest

$E_{high} = -1/4\ (0 \log_2(0) + 1/1 \log_2(1/1)) = 0$ Attend = No.

$E_{med} = -1/4\ (1/1 \log_2(1/1) + 0 \log_2(0)) = 0$ Attend = Yes.

$E_{low} = -2/4\ (1/2 \log_2(1/2) + 1/2 \log_2(1/2)) = 0.5$

$E_{int} = E_{high} + E_{med} + E_{low} = 0.5$

Time of Day

$E_{eve} = -2/4\ (1/2 \log_2(1/2) + 1/2 \log_2(1/2)) = 0.5$

$E_{aft} = -2/4\ (1/2 \log_2(1/2) + 1/2 \log_2(1/2)) = 0.5$

$E_{mor} = -0/4(\qquad) = 0.0$ No Records

$E_{time} = E_{eve} + E_{aft} + E_{mor} = 1.0$

Since $E_{int} \leq E_{time}$, we split on Interest

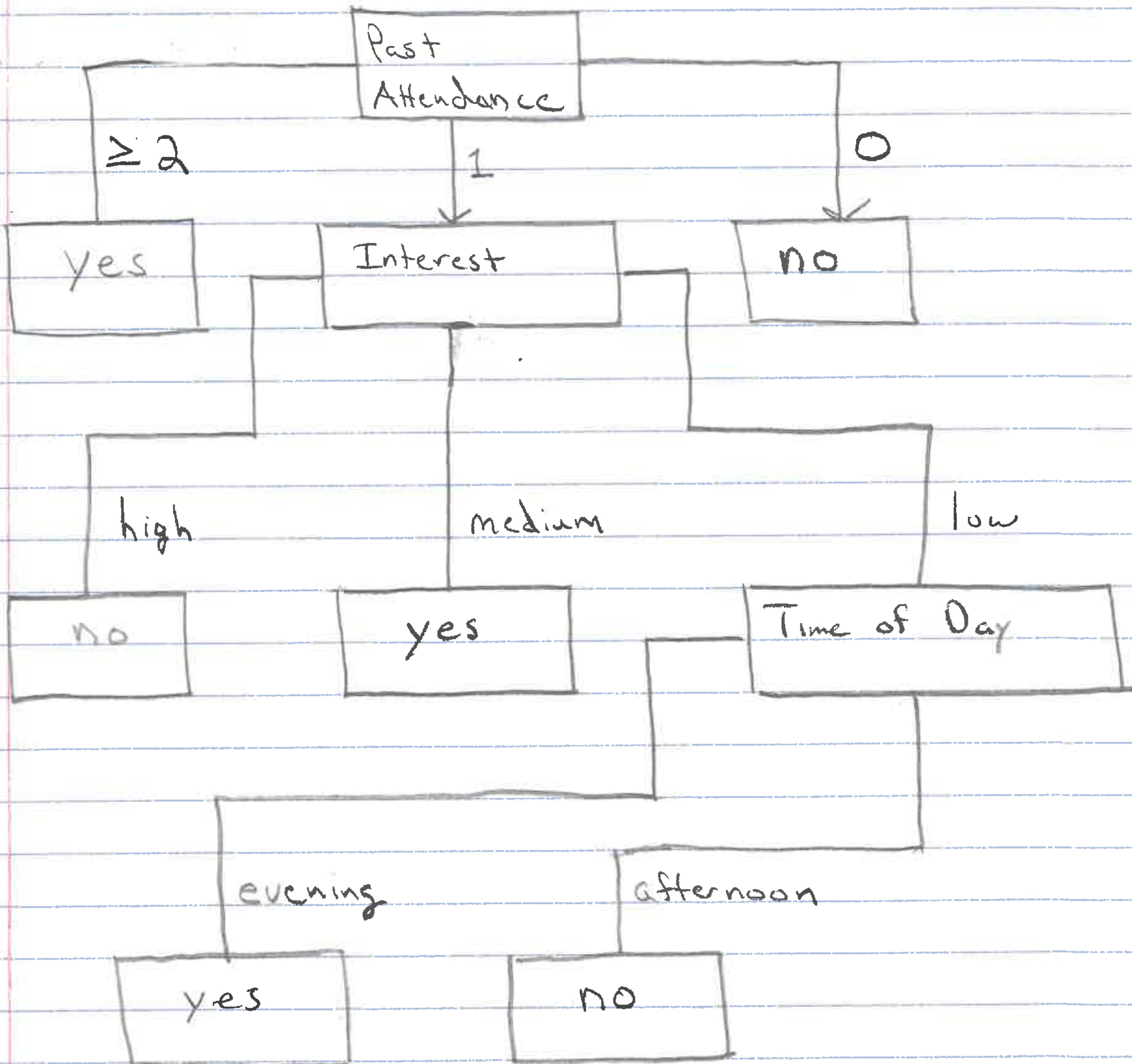
0: Split is pure, no need to calculate. Attend = No.

For 3^{rd} Split we choose the feature we haven't used yet which is Time of Day

When past attendance is 1
interest is low
and time of day is evening only record has Attend = Yes.

When past attendance is 1
interest is low
and time of day is afternoon only record has Attend = No.

2.a) cont'd



2.b) We start with the fact that Past Attendence ≥ 2. This tells us right away, according to the decision tree, that the person will attend the workshop.

3.a) The training and validation by different Θ is printed out on the next page.

The model performs best when $\Theta = 0.20$.
The accuracy on the test set when $\Theta = 0.20$ is 0.872.

It appears that when the minimum node entropy is really low the model is complex / overtrained. For example, when minimum node entropy = 0.01 the training accuracy is 1.0 and the validation accuracy is 0.863. Conversely, when the minimum node entropy is high the model appears too simple / undertrained. For example, when minimum node entropy = 4.0 the training accuracy is 0.100 and the validation accuracy is 0.117.

The "sweet spot" for model complexity / accuracy will be somewhere between the values 0.01 and 4.0 for minimum node entropy.

3.a) cont'd

Training/validation accuracy for minimum node entropy 0.010000 is 1.000 / 0.863
Training/validation accuracy for minimum node entropy 0.050000 is 0.999 / 0.863
Training/validation accuracy for minimum node entropy 0.100000 is 0.997 / 0.865
Training/validation accuracy for minimum node entropy 0.200000 is 0.990 / 0.867
Training/validation accuracy for minimum node entropy 0.500000 is 0.963 / 0.863
Training/validation accuracy for minimum node entropy 1.000000 is 0.871 / 0.840
Training/validation accuracy for minimum node entropy 2.000000 is 0.596 / 0.600
Training/validation accuracy for minimum node entropy 4.000000 is 0.100 / 0.117
Test accuracy with minimum node entropy 0.200000 is 0.872

Training/validation accuracy for minimum node gini_index 0.010000 is 0.999 / 0.847
Training/validation accuracy for minimum node gini_index 0.050000 is 0.990 / 0.852
Training/validation accuracy for minimum node gini_index 0.100000 is 0.978 / 0.851
Training/validation accuracy for minimum node gini_index 0.200000 is 0.948 / 0.845
Training/validation accuracy for minimum node gini_index 0.500000 is 0.800 / 0.767
Training/validation accuracy for minimum node gini_index 1.000000 is 0.100 / 0.117
Training/validation accuracy for minimum node gini_index 2.000000 is 0.100 / 0.117
Training/validation accuracy for minimum node gini_index 4.000000 is 0.100 / 0.117
Test accuracy with minimum node gini_index 0.050000 is 0.867

3.b. According to the results the curves appear to function slightly differently in that the gini index falls off into under training really quickly as the minimum node value climbs. For example, when the minimum node value is at 1.0 the training/validation of entropy is 0.871/0.840 while the training/validation of the gini index is 0.100/0.17.

However, if we compare the "sweet spot" of the two models, where we get the best results, we find that the two metrics are fairly similar, entropy test accuracy is 0.872 versus gini index of 0.867, with entropy being <u>very</u> slightly better.