HW4

Nicole Sullivan

1

> Suppose we use a linear SVM classifier for a binary classification problem with a set of data points shown in Figure 1, where the samples closest to the boundary are illustrated: samples with positive labels are (0, 1), (-2, -1), (1, 3), (-1, 1.5), (-1.5, 0.5), and samples with negative labels are (0.5, -0.5), (-0.5, -2), (1.5, -1), (3, 0.5), (-1, -2.5).

1a

> List the support vectors.

Support vectors for:

- the (+) margin: (0, 1) and (-2, -1)
- the (-) margin: (0.5, -0.5)

1b

> Pick three samples and calculate their distances to the hyperplane $x_1 - x_2 = 0$.

The equation for the distance of a sample to the hyperplane is: $d^i = \frac{w^T x^i + w_0}{\|w\|}$. We can use this equation to calculate the distance for 3 different points. In our case, $w = [1, -1]$ and $w_0 = 0$.

Point 1: (0, 1)

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

Point 2: (0.5, -0.5)

$d_{(0.5,-0.5)} = \frac{|0.5-0.5|}{\sqrt{(1)^2+(-1)^2}} = \frac{1}{\sqrt{2}}$

Intuitively, this makes sense that both the support vector are the same distance from the hyperplane since $\rho = \frac{1}{\|w\|}$. Let's try a point that's not a support vector to make sure it's distance is **greater** than that of the support vectors.

Point 3: (1, 3)

$d_{(1,3)} = \frac{|1-3|}{\sqrt{(1)^2+(-1)^2}} = \frac{2}{\sqrt{2}}$

1c

> If the sample (0, 1) is removed, will the decision boundary change? What if we remove the sample (-1, -2.5) instead?

If (0, 1) is removed the decision boundary **will** change because (0, 1) helps define the margin for the positive class. However, removing (-1, -2.5) will have no impact on the decision boundary because it isn't a support vector.

1d

> If a new sample (2, 0.5) comes as a positive sample, will the decision boundary change? If so, what method will you use in this case?

Yes, the decision boundary will change because the new positive sample makes it impossible to **perfectly** linearly separate the classes, let alone with a margin. In this case, I'd use soft margin classification and add a slack variable to allow misclassification of this point while preserving correct classification for the other points.

1e

> In the soft margin SVM method, C is a hyperparameter (see Eqs. 13.10 or 13.11 in Chapter 13.3 of the textbook). What would happen when you use a very large value of C? How about using a very small one?

A large value of C puts more weight on errors - therefore, if errors are extremely costly, we'd want to use a very large value of C. When C is small, larger amounts of error are tolerated in the optimal decision boundary and more weight is placed on the **margin** between the classes. If we care more about generalizability, then we want to use a smaller value of C.

1f

> In real-world applications, how would you decide which SVM methods to use (hard margin vs. soft margin, linear vs. kernel)?

Hard margin: this is best used when the data are linearly separable and there's little-to-no noise amongst the classes (rare in real-world applications)

Soft margin: I'd use this when there's more noise amongst the classes, but they're linearly separable with a small amount of error

Kernel: I would use a kernel SVM when data are highly non-linear and a transformation/mapping to a higher dimensional space is required to make the classes linearly separable

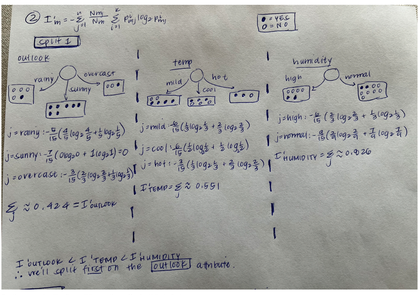
2

> Table 2 shows data collected on a person's decision to go for a run or not go for a run depending on the weather conditions that day. Answer the following questions.

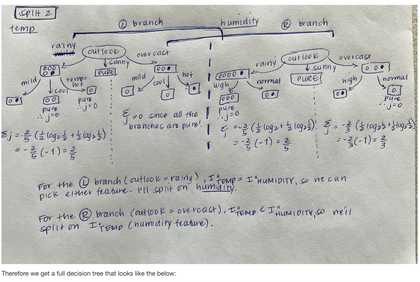
2a

> We wish to build a decision tree to help decide if the runner will go for a run tomorrow. Draw the decision tree that fits this data and show how to calculate each node split using entropy as the impurity measure. Note: If the entropy is the same for two or more features, you can select any of the features to split.

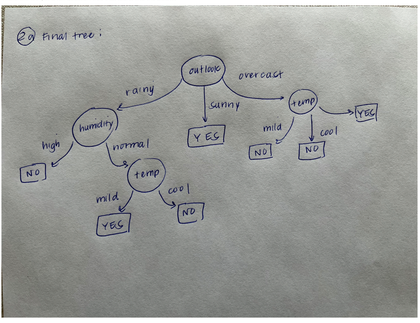
For the first split:



For the second split, after splitting on the **outlook** attribute first:



Therefore we get a full decision tree that looks like the below:



2b

> Based on the decision tree, will the runner go for a run if tomorrow is Overcast, the temperature is Cool, and the Humidity level is Normal?

No, according to the decision tree, the runner will **not** go for a run in these conditions.

3

> In this programming exercise you will implement a Decision Tree for optical-digit classification. You will train your Decision Tree using the optdigits train.txt data, tune the minimum node entropy using optdigits valid.txt data, and test the prediction performance using the optdigits test.txt data. For each file, the first 64 columns correspond to the binary features for different samples while the last one stores the labels. The minimum node entropy is used as a threshold to stop splitting the tree, and set the node as a leaf.

3a

> Implement a Decision Tree with the minimum node entropy θ=0.01, 0.05, 0.1, 0.2, 0.5, 0.8, 1.0 and 2.0. The minimum node entropy is used to determine the leaf nodes, i.e., a node is a leaf node if its node entropy is lower than the selected minimum node entropy. Report the training and validation accuracy by different θ. What θ should you use? Report the accuracy on the test set using this θ.

- $\theta = 0.01$
  - training accuracy: 1.000
  - validation accuracy: 0.863
- $\theta = 0.05$
  - training accuracy: 0.999
  - validation accuracy: 0.863
- $\theta = 0.1$
  - training accuracy: 0.997
  - validation accuracy: 0.865
- $\theta = 0.2$
  - training accuracy: 0.990
  - validation accuracy: 0.867
- $\theta = 0.5$
  - training accuracy: 0.963
  - validation accuracy: 0.863
- $\theta = 0.8$
  - training accuracy: 0.919
  - validation accuracy: 0.856
- $\theta = 1.0$
  - training accuracy: 0.871
  - validation accuracy: 0.840
- $\theta = 2.0$
  - training accuracy: 0.596
  - validation accuracy: 0.600

The best θ of the 8 θs looked at was **0.2**. While it had lower training accuracies than the smaller θs, it had the highest validation accuracy amongst all 8 θ*values*, which means it will generalize better than others that are more overfit to the training data. The test accuracy for $\theta = 0.2$ was **0.872**.

3b

> What can you say about the model complexity of the Decision Tree, given the training and validation accuracy? Briefly explain.

The complexity of the decision tree in the $\theta = 0.2$ is lower than that of those with smaller θs (0.01, 0.05, 0.1), but greater than that of those with larger θs (0.5, 0.8, 1.0, 2.0). We know this because high training accuracy with high prediction error indicates **high** model complexity, while low training accuracy and high prediction error indicates **low** model complexity. However, the $\theta = 0.2$ decision tree was neither: between the two of these extremes, it lands in the "sweet spot" of the bias-variance trade-off, generating the highest testing accuracy possible, while still retaining training accuracy. Models that both minimize prediction error and maximize generalizability are of middling/medium complexity; hence, we know that are $\theta = 0.2$ decision tree was of medium complexity.