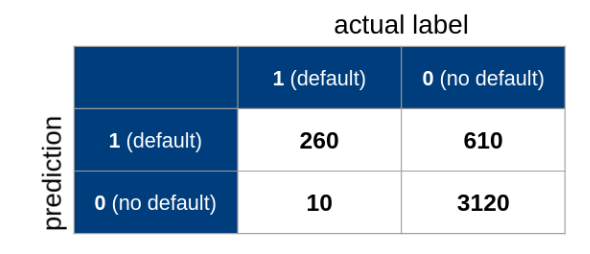
**CS5228 Tutorial 3 – Classification & Regression**

**Q1: Classification Metrics**

Assume that you have trained a binary classifier that aims to predict if a bank customer will default on his/her credit. Your test data contained 4,000 samples and your final model yields the following confusion matrix:



1. Calculate the Specificity, Sensitivity, Recall, Precision, and F1 score.

• Specificity = 0.84

• Sensitivity/Recall = 0.96

• Precision = 0.3

• F1=0.46

1. For each metric, provide a verbal interpretation of the resulting value in the context of predicting a customer’s likelihood to default on his or her credit! Discuss if we can be happy with the result, or what result might cause problems in practice!

• Specificity: If a customer will not default, the classifier will very likely predict it correctly (with 84% probability).

• Sensitivity/Recall: If a customer will default, the classifier will very likely predict it correctly (with 96% probability).

• Precision: If the classifier predicts that a customer will default, it will be only correct 30% of the time

• F1: A balanced consideration of both Recall and Precision.

The rather high values for Accuracy and Specificity can be a bit misleading since the dataset is a bit imbalanced given that most customers do not default on the credit. However, the main problem is the low Precision value. It essentially means that the classifier will predict default ”too often”, i.e., in many cases where the customer would not default. If the bank decides to approve or deny credit based on this result, many customers will unjustifiably not qualify for a credit.

1. In imbalanced datasets, a *majority* class contains most of the samples whereas a *minority* class contains only a fraction of samples (assuming a binary classification task). List example applications where you would expect a very imbalanced dataset.

• Credit card fraud

• Intrusion detection

• Earthquake detection system

1. While not covered in the lecture, what do you think can be done to correct imbalance (beyond picking the right metric)?

• Collect more minority class data, if possible and/or practical

• Generation of synthetic data, if possible and/or practical

• Undersampling of majority class

• Oversampling of minority class

**Q2: Assessing Classification Errors**

In binary classification, we can make 2 types of errors:

* **False Positives** (FP), also called Type I Error
* **False Negatives** (FN), also called Type II Error

In many cases these two types of errors are not equally problematic.

1. List 1 example application where False Positives are more problematic than False Negatives, and vice versa. Provide a brief explanation!

**Spam detector**: false positives lead to real emails being detected as spam, which can lead to users missing important emails. False negatives just lead to an additional spam message that gets through to the user.

**Flagging defective products for manual review**: false negatives can lead to defective products being shipped out, while false positives just increase the amount of manual review that needs to be done.

1. Say you trained a binary sentiment classifier that classifies social media posts (e.g., tweets) into ”negative” and ”positive”. Your datasets for training and testing were balanced and sufficiently large. Let’s assume that the F1 score of your classifier is 0.85. How would you assess if this is a good result?

• A ”random guesser” would get an F1 score of around 0.5, so the classifier is certainly much better than that

• Compare the classifier with the “next-best alternative”: e.g. the existing solution the classifier is aiming to replace

• While an F1 score 1.0 is the theoretical upper bound, in practice the goal is often below that. This is particularly true for such subjective use cases like sentiment analysis where different people might give different sentiments to the same tweet. For example, if people agreed on tweets’ sentiments in 90% of the cases, then the classifier is rather close to this number.

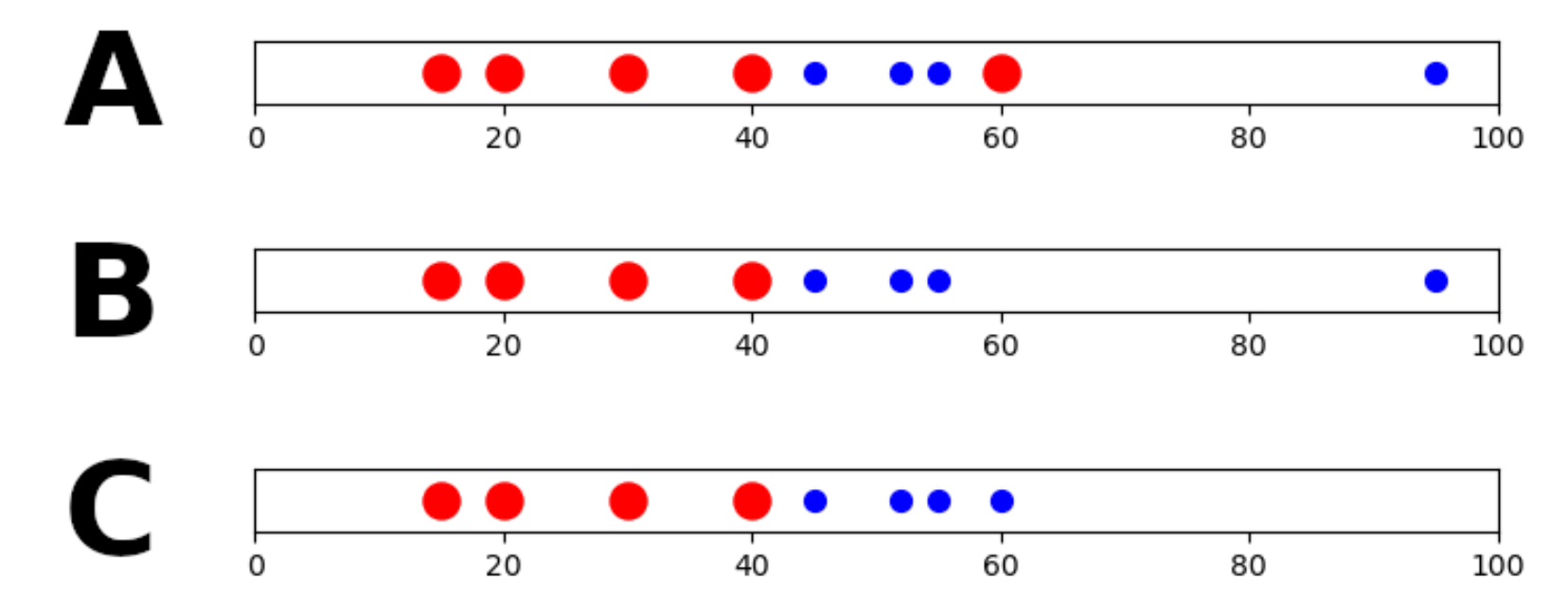
1. Say you trained a classifier that identifies whether an image contains a Car, Boat, or Plane, and the F1-score is very high, say, 0.99. Your datasets for training and testing were balanced and sufficiently large. What might be a reason why the classifier would suddenly perform poorly in practice?  
     
   • The dataset may not be representative enough of the situations faced in practice..

• Example: a ”bad” dataset for vehicle classification could contain only images with planes in the sky, boats on the open sea, and cars on roads. In this case, the classifier might identify planes because of the blue/grey background (sky), and make mistakes in practice when different backgrounds are present.

**Q3: Data Distribution and Decision Trees**

The figure below shows 3 distributions of the values for a single feature. The color and shape of the dots reflect the class label. Since we only have two colors/sizes, the example application is a binary classification task.

In A, we observe what can be called a “mislabeled point” at x=60, and an “outlier” at x=95. In B, the mislabeled point is removed, while in C, the outlier is removed.

* 

1. How many splits would the decision tree fitting algorithm perform in each of these cases?

1 split for B and C (at around x=42)

3 splits for A (at around x=42, x=58, x=80)

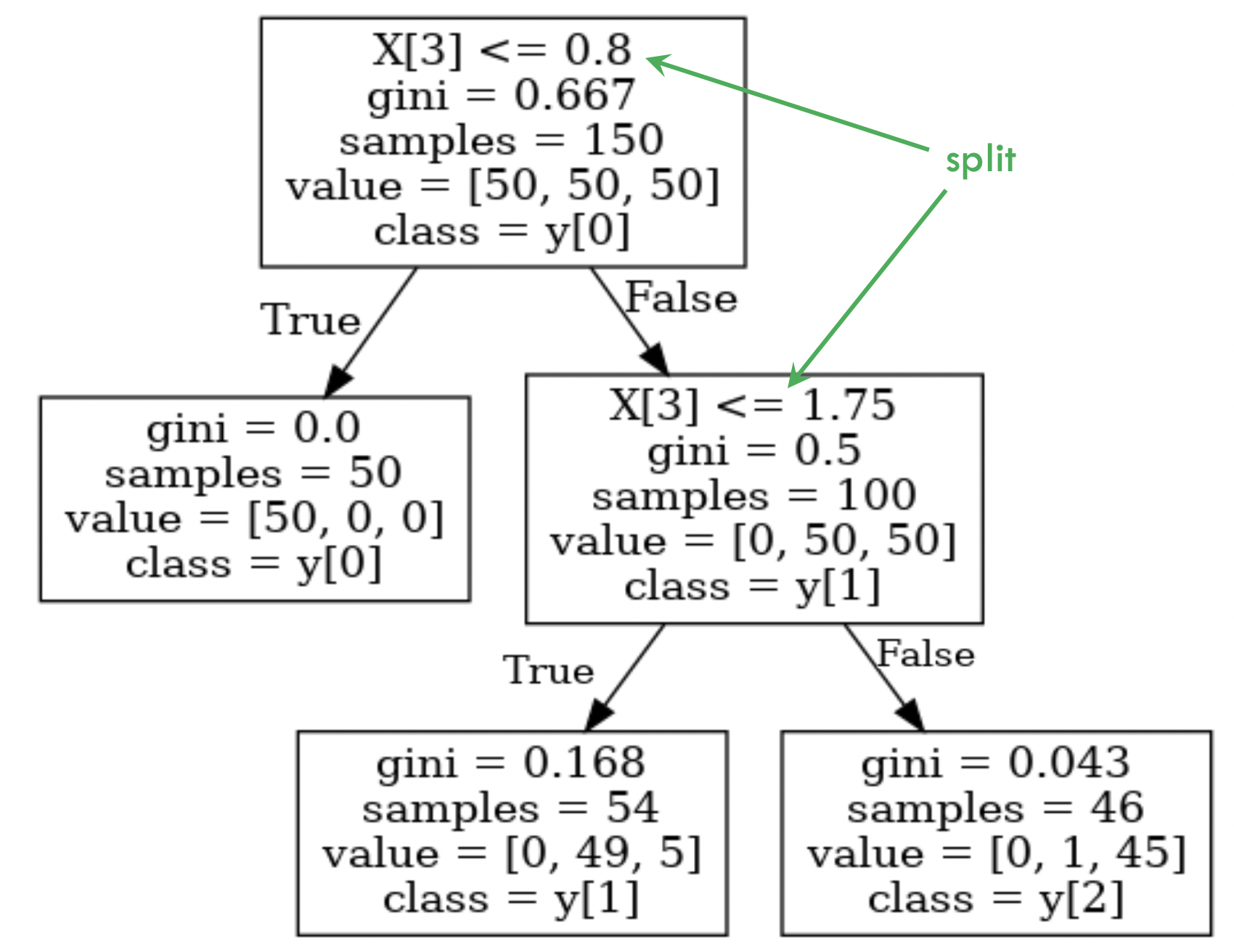
1. Given your previous answer, summarize how “mislabeled points” and “outliers” affect the learned decision tree in this case.

Mislabelled points corrupt the decision tree, forcing them to learn extra splits which makes their predictions more noisy and inaccurate.

Outliers may not always cause issues: if as in this case the outliers conform to the class label of the nearest class, they do not cause any change in the learned decision tree.

**Q4: Interpreting Decision Trees**

The Decision Tree shown below has been trained over the iris Dataset with a maximum depth of 2. Each data sample has 4 numerical features, and labeled with 1 out of 3 classes. Below, X[3] refers to the 3rd feature; samples refers to the number of samples in a node; value refers to the distribution of class labels of the samples in a node; and class refers to the predicted class of a node.



1. Based on the learned decision tree, how would you describe the characteristics of each class in the dataset?

Class 0 can be perfectly identified by X[3] being <= 0.8.

Class 1 and 2 can then be mostly separated based on whether X[3] <= 1.75, but not perfectly.

1. How would the decision tree change if we additionally standardized each feature before training the decision tree?

Only the split locations would change; but all other aspects of the decision tree (e.g. the split variables, and which samples go to which nodes) remain the same.

(This is because standardizing each variable does not change the ordering of the values of each feature (and decision trees only care about the order of the values, not the values themselves))