CS/ENGR M148 L9:
Decision Trees and Random Forests

Sandra Batista

Administrative News

Quiz 2 Thursday, 10/31/24 on PS2!

Only 15 minutes, T/F, multiple choice/select

Please bring laptop and hard copy of notes.

Midterm next Tuesday!

100 minutes. Covering lectures 1-10

Please bring laptop/device to scan exam, and hard copy of notes.

For CAE accommodations:

Please schedule your testing at CAE testing center for quizzes, midterm (100 minutes regular time), and final by 10/29/24.

Administrative News

Please contact TA first for homework submission help. I can help if TA cannot resolve.

This week in discussion section:

Lab on KNN classification

Project Data Check-in: Already posted on BruinLearn

New for Project Check-ins Early:

11am-11:50am Fridays in Boelter 5436 with our wonderful TA Yihe

LAs will be helping with project check-ins!

Join our slido for the week…

https://app.sli.do/event/vb9RXFWoKnxhYMBAnTwdgA

Today's Learning Objectives

Students will be able to:

- Review: Evaluate classification problems with quantitative metrics
- Review: Apply KNN algorithm by hand to a small sample data set
- Decision Trees and Random Forests

Classification

A binary response is often referred to as the **class label** of the observation.

Classification problems: Prediction problems with binary responses that involve *classifying* each observation as belonging to one of the two classes.

(It is possible to have more than 2 classes…)

Evaluating Classification

Prediction accuracy

$$\text{prediction accuracy} = \frac{(\text{number of correct predictions})}{n}$$

Prediction error

$$\text{prediction error} = \frac{(\text{number of incorrect predictions})}{n}$$

Sensitivity and Specificity

The **true positive rate or** “**sensitivity**” or “**recall**”

$$\begin{aligned}\text{true positive rate} &= \frac{(\text{number of correctly predicted positive class obs})}{(\text{number of positive class observations})}\\ &= \frac{(\text{number of true positives})}{(\text{number of positive class observations})}\end{aligned}$$

The **true negative rate** (often called “**specificity**”) is the proportion of negative class observations whose class is correctly predicted

False positive rate = 1-specificity

Tradeoff between sensitivity and specificity

ROC Curves

Receiver Operating Characteristics (ROC) curve plot true positive rate against true negative rate for various thresholds to compare models and algorithms.

A**rea under the curve (AUC)** quantifies predictive potential of algorithm by computing the literal area under the ROC curve.

How to create ROC Curves

1. For each possible threshold, run the classification algorithm on the training (or validation) data set (or even cross-validation data sets)
2. Calculate the true positive rate and true negative rate
3. Plot the **1-true negative rate** against the **true positive rate** for each threshold
4. Calculate the area under the ROC curve for AUC (perfect classification is 1, but .8 or better is good)
5. Often use CV on validation set to calculate ROC and compare across algorithms

Your turn

Can you calculate the ROC Curve and AUC for the following confusion matrices for different classification probability thresholds for a logistic regression?

Threshold = .6		
	predicted positive	predicted negative
observed positive	2	8
observed negative	1	9
Threshold = .5		
	predicted positive	predicted negative
observed positive	6	4
observed negative	3	7
Threshold = .4		
	predicted positive	predicted negative
observed positive	8	2
observed negative	4	6
Threshold = .3		
	predicted positive	predicted negative
observed positive	10	0
observed negative	5	5

Your turn

Can you calculate the ROC Curve and AUC for the following confusion matrices for different classification probability thresholds for a logistic regression?

Today's Learning Objectives

Students will be able to:

- ✓ Review: Evaluate classification problems with quantitative metrics
- ✕ Review: Apply KNN algorithm by hand to a small sample data set
- ✕ Decision Trees and Random Forests

Similarity-Based Learning

Similarity-based learning *is classifying new data based on previous observations.*

One of the simplest and best known machine learning algorithms for this type of reasoning is called the nearest neighbor algorithm.

Motivational example: An alien sees an animal that we know is a platypus, but the alien has not seen before.

The alien has seen and hear ducks, frogs, and lions. How will this alien classify this new animal?

[Kelleher et al 2015]

Similarity Metrics

- *A similarity metric measures the similarity between two instances according to a feature space*
- *Mathematically, a metric must conform to the following four criteria:*
  1. *Non-negativity*: $metric(\mathbf{a}, \mathbf{b}) \geq 0$
  2. *Identity*: $metric(\mathbf{a}, \mathbf{b}) = 0 \Leftrightarrow \mathbf{a} = \mathbf{b}$
  3. *Symmetry*: $metric(\mathbf{a}, \mathbf{b}) = metric(\mathbf{b}, \mathbf{a})$
  4. *Triangular Inequality*:
     $metric(\mathbf{a}, \mathbf{b}) \leq metric(\mathbf{a}, \mathbf{c}) + metric(\mathbf{b}, \mathbf{c})$

  where $metric(\mathbf{a}, \mathbf{b})$ *is a function that returns the distance between two instances* **a** *and* **b**.

[Kelleher et al 2015]

Minkowski distance

*The Minkowski distance between two instances **a** and **b** in a feature space with m descriptive features is:*

$$Minkowski(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{m} abs(\mathbf{a}[i] - \mathbf{b}[i])^p \right)^{\frac{1}{p}}$$

where different values of the parameter p result in different distance metrics

- *The Minkowski distance with p = 1 is the Manhattan distance and with p = 2 is the Euclidean distance.*
- *The larger the value of p the more emphasis is placed on the features with large differences in values*

[Kelleher et al 2015]

kNN (k nearest neighbor)

1. As in the general problem of classification, we have a set of data points for which we know the correct class labels.
2. When we get a new data point, we compare it to each of our existing data points and find similarity.
3. Take the most similar *k* data points *(k* nearest neighbors).
4. From these *k* data points, take the majority vote of their labels. The winning label is the label/class of the new datapoint.

Choice of k will affect classification and is hyperparameter.

[Shah 2020]

Applying KNN algoritm

1. Calculate distance to all other points

Table: *The distances (Dist.) between the query instance with SPEED = 6.75 and AGILITY = 3.00 and each instance in Table 2 [25].*

ID	SPEED	AGILITY	DRAFT	Dist.	ID	SPEED	AGILITY	DRAFT	Dist.
18	*7.00*	*4.25*	*yes*	*1.27*	*11*	*2.00*	*2.00*	*no*	*4.85*
12	5.00	2.50	no	1.82	19	7.50	8.00	yes	5.06
10	4.25	3.75	no	2.61	3	2.25	5.50	no	5.15
20	7.25	5.75	yes	2.80	1	2.50	6.00	no	5.20
9	4.00	4.00	no	2.93	13	8.25	8.50	no	5.70
6	4.50	5.00	no	3.01	2	3.75	8.00	no	5.83
8	3.00	3.25	no	3.76	14	5.75	8.75	yes	5.84
15	4.75	6.25	yes	3.82	5	2.75	7.50	no	6.02
7	3.50	5.25	no	3.95	4	3.25	8.25	no	6.31
16	*5.50*	*6.75*	*yes*	*3.95*	*17*	*5.25*	*9.50*	*yes*	*6.67*

[Kelleher et al 2015]

Applying KNN algorithm

2. Take the k nearest neighbors

3. Take majority vote for new label

Table: *The distances (Dist.) between the query instance with SPEED = 6.75 and AGILITY = 3.00 and each instance in Table 2* [25]*.*

ID	SPEED	AGILITY	DRAFT	Dist.	ID	SPEED	AGILITY	DRAFT	Dist.
18	*7.00*	*4.25*	*yes*	*1.27*	*11*	*2.00*	*2.00*	*no*	*4.85*
12	5.00	2.50	no	1.82	19	7.50	8.00	yes	5.06
10	4.25	3.75	no	2.61	3	2.25	5.50	no	5.15
20	7.25	5.75	yes	2.80	1	2.50	6.00	no	5.20
9	4.00	4.00	no	2.93	13	8.25	8.50	no	5.70
6	4.50	5.00	no	3.01	2	3.75	8.00	no	5.83
8	3.00	3.25	no	3.76	14	5.75	8.75	yes	5.84
15	4.75	6.25	yes	3.82	5	2.75	7.50	no	6.02
7	3.50	5.25	no	3.95	4	3.25	8.25	no	6.31
16	*5.50*	*6.75*	*yes*	*3.95*	*17*	*5.25*	*9.50*	*yes*	*6.67*

What happens for new athlete for k=1? k = 3? For k = 5?

[Kelleher et al 2015]

Decision Boundaries

Decision boundaries *are the surfaces separating classes in classification.*

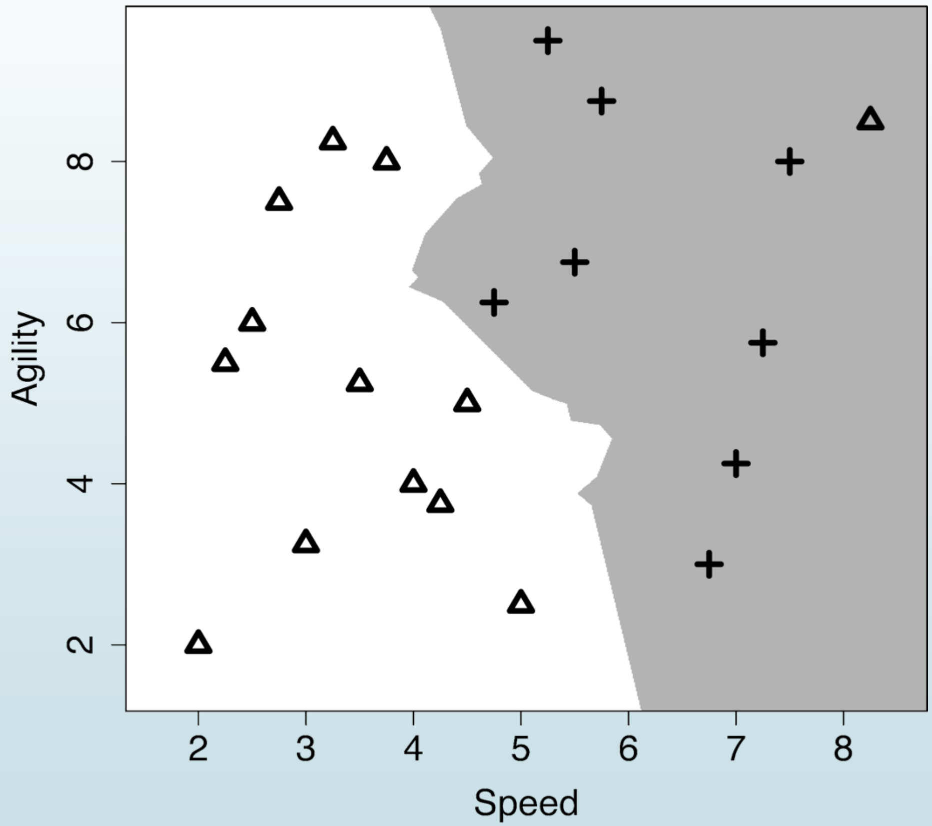


Figure: *The decision boundary using majority classification of the nearest 3 neighbors.*

[Kelleher et al 2015]

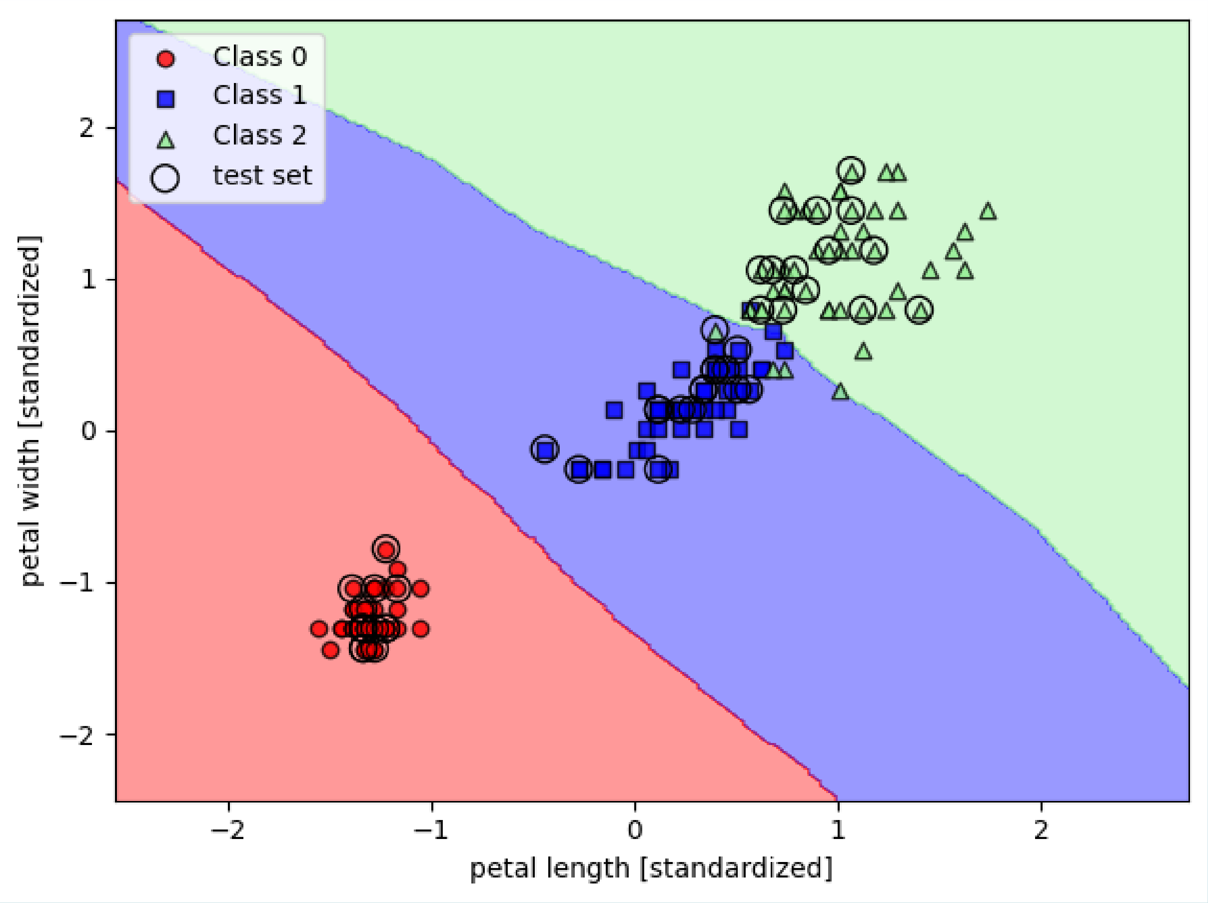
Sklearn K Neighbors

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors=10, p=2, metric='minkowski')
knn.fit(X_train_std, y_train)
plot_decision_regions(X=X_combined_std, y=y_combined, classifier=knn,
                      test_idx=range(105, 150))
plt.xlabel('petal length [standardized]')
plt.ylabel('petal width [standardized]')
plt.legend(loc='upper left')
plt.tight_layout()
plt.show()
```

Using the sklearn iris data set to classify irises into 3 classes based on petal length and width

[Raschka et al 2022]

Decision Boundaries



Using the sklearn iris data set to classify irises into 3 classes based on petal length and width

[Raschka et al 2022]

Today's Learning Objectives

Students will be able to:

- ✓ Review: Evaluate classification problems with quantitative metrics
- ✓ Review: Apply KNN algorithm by hand to a small sample data set
- ✗ Decision Trees and Random Forests

Decision Trees

Motivation: How can we decide a question to ask about this data set to decide whether an email is spam or not?

We want the the **most informative descriptive feature** that will help us split the data into subsets of spam or not based on the answer

Table: *An email spam prediction dataset.*

ID	SUSPICIOUS WORDS	UNKNOWN SENDER	CONTAINS IMAGES	CLASS
376	True	False	True	Spam
489	true	true	false	spam
541	true	true	false	spam
693	false	true	true	ham
782	false	false	false	ham
976	false	false	false	ham

[Kelleher et al 2015]

Decision Trees

*A **decision tree** consists of:*

*a **root node** (or starting node),*

interior nodes

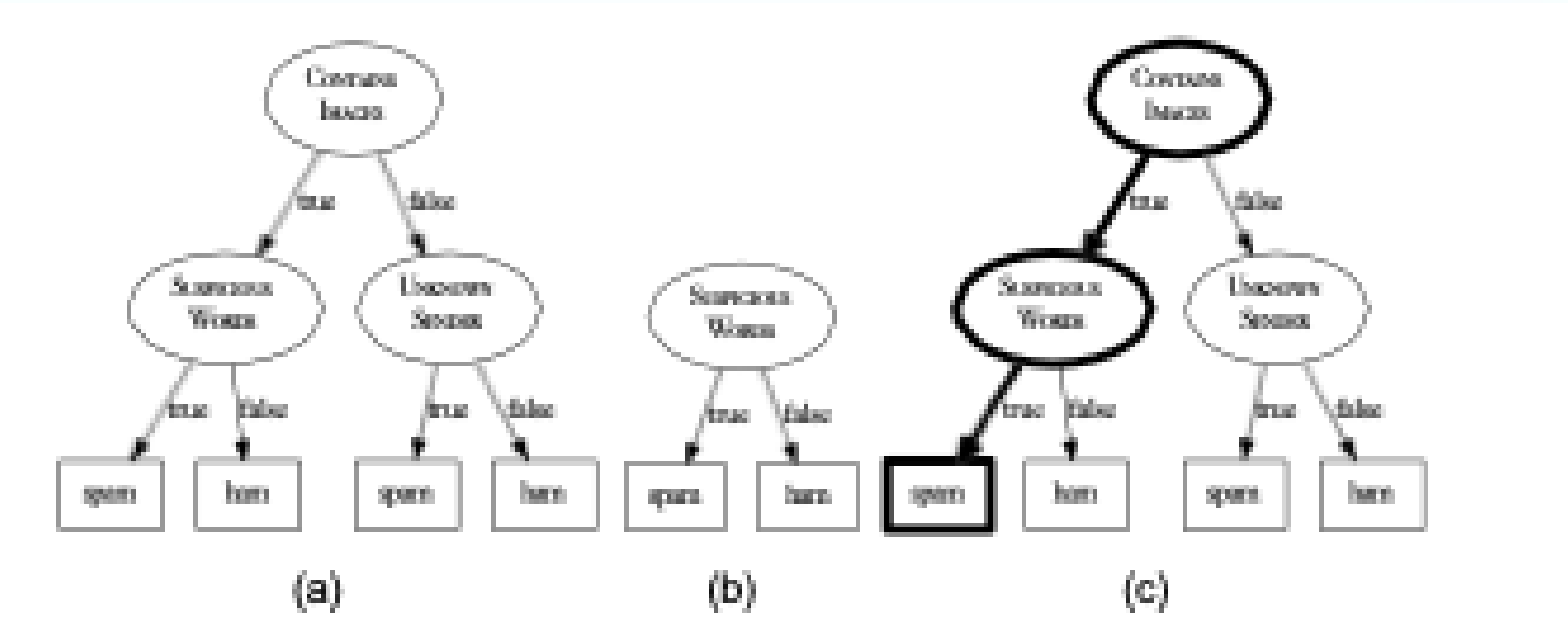
*and **leaf nodes** (or terminating nodes).*

Each of the non-leaf nodes (root and interior) in the tree specifies a test to be carried out on one of the query's descriptive features.

Each of the leaf nodes specifies a predicted classification for the query.

[Kelleher et al 2015]

Spam Decision Tree



[Kelleher et al 2015]

How a decision tree divides data

*A binary **decision tree** will divide data based on question:*

1) *For continuous variable, can ask above or below threshold*
2) *For categorical variable can ask if equal to specific level*

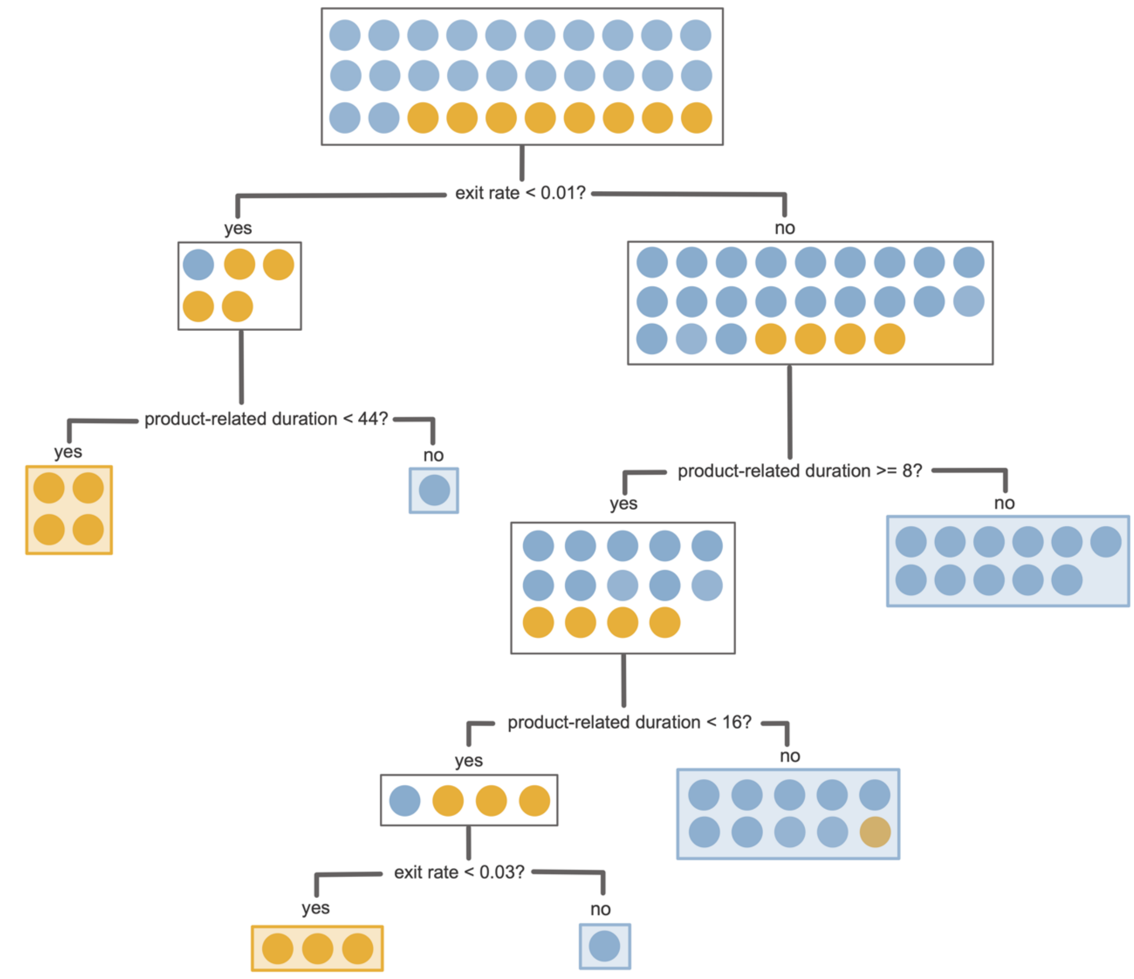
*Each **node** contains a subset of data starting with all data at **root***

***Child nodes** have splits of data*

***Terminal or leaf nodes** contain splits of data where predictions made*

[Yu, Barter 2024]

UCI Shopping Decision Tree Example



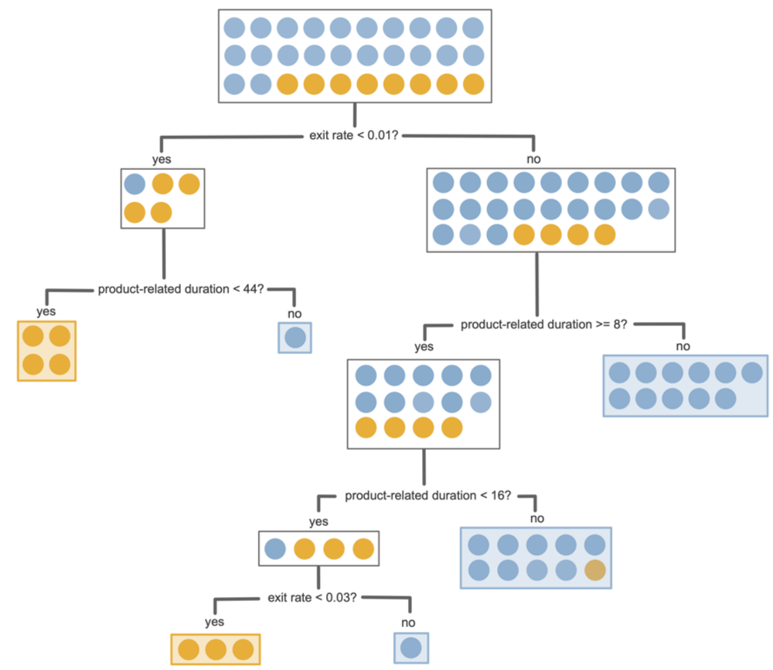
[Yu, Barter 2024]

How to predict response?

Given a new sample, traverse decision tree answering questions for new sample.

Once arrive at leaf node

1) *For continuous variable, predict average of samples in node*
2) *For categorical variable, calculate positive class probability in the node and use threshold (.5) to predict positive class or not.*

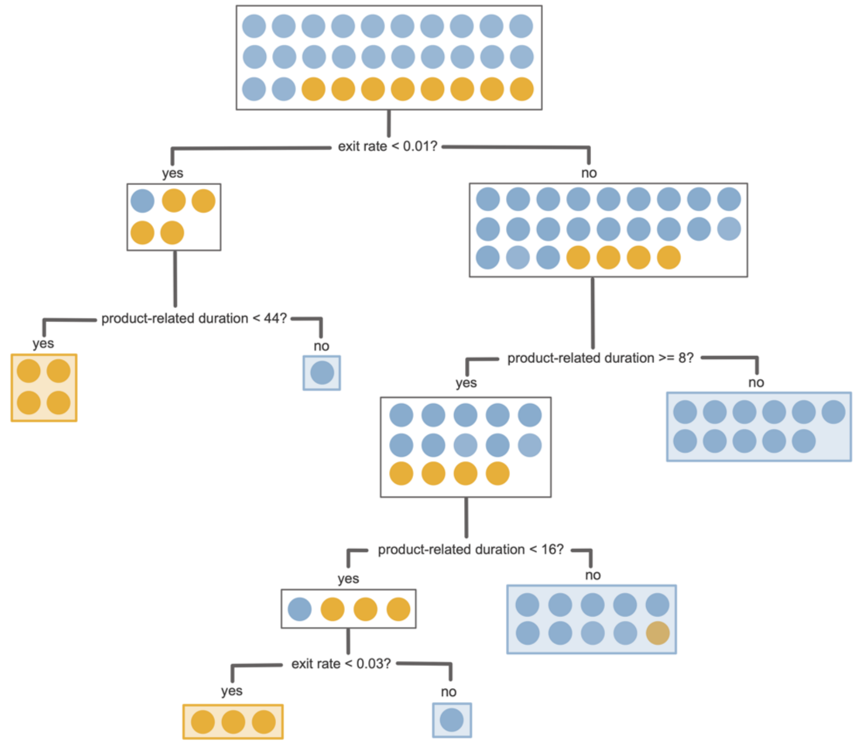


[Yu, Barter 2024]

What's your prediction?

A new user spent 20 minutes on product-related pages and had an average exit rate of 0.042.

Do you predict a purchase?



[Yu, Barter 2024]

Classification and Regression Algorithm (CART)

The ***Classification and Regression Algorithm (CART)*** *aims to split data to minimize the variance in child nodes.*

'Variance' is different for continuous and binary variables.

Important observation: Nodes can be a mix of different types of classes of observations, but want to minimize this.

[Yu, Barter 2024]

Variance Split for Continuous Response

*The **variance for split** is a weighted variance of the left and right child nodes.*

$$\text{Variance measure for split} = \frac{n_{\text{left}}}{n_{\text{parent}}}\text{Var}_{\text{left}} + \frac{n_{\text{right}}}{n_{\text{parent}}}\text{Var}_{\text{right}}$$

Variance decreases in child nodes.

[Yu, Barter 2024]

Variance Split for Continuous Response

Choose the split the minimizes this weighted variance for the response variable.

$$\text{Variance measure for split} = \frac{n_{\text{left}}}{n_{\text{parent}}}\text{Var}_{\text{left}} + \frac{n_{\text{right}}}{n_{\text{parent}}}\text{Var}_{\text{right}}$$

Split	Variance measure
living area < 1,625	1,343,422,176
living area < 1,428	808,915,591
living area < 905	1,255,484,629

(Example using Ames housing data, so response variable is price)

[Yu, Barter 2024]

Gini Impurity Split for Binary Response

*A **pure node** contains all observations in the same class.*

*An **impure node** contains observations in different classes.*

***Gini impurity** measures impurity of node and calculates probability two randomly chosen observations with replacement are from different classes.*

Maximal impurity would be .5, equally likely to get sample from either class.

[Yu, Barter 2024]

Gini Impurity Split for Binary Response

$$\text{Gini impurity} = P(\text{two random obs in different classes}).$$

$$\text{Gini impurity} = 1 - P(\text{two random obs in the same class}).$$

$$\text{Gini impurity} = 1 - \Big(P(\text{both pos class}) + P(\text{both neg class})\Big)$$

[Yu, Barter 2024]

Gini Impurity Split for Binary Response

Let p1 be probability of success (positive class)
And p0 probability of failure or (negative class)

$$\text{Gini impurity} = 1 - p_1^2 - p_0^2 = 2p_0p_1,$$

What is Gini impurity when p1 =.5 and p1 = .25?

[Yu, Barter 2024]

How is Gini impurity related to Bernoulli Variable Variance?

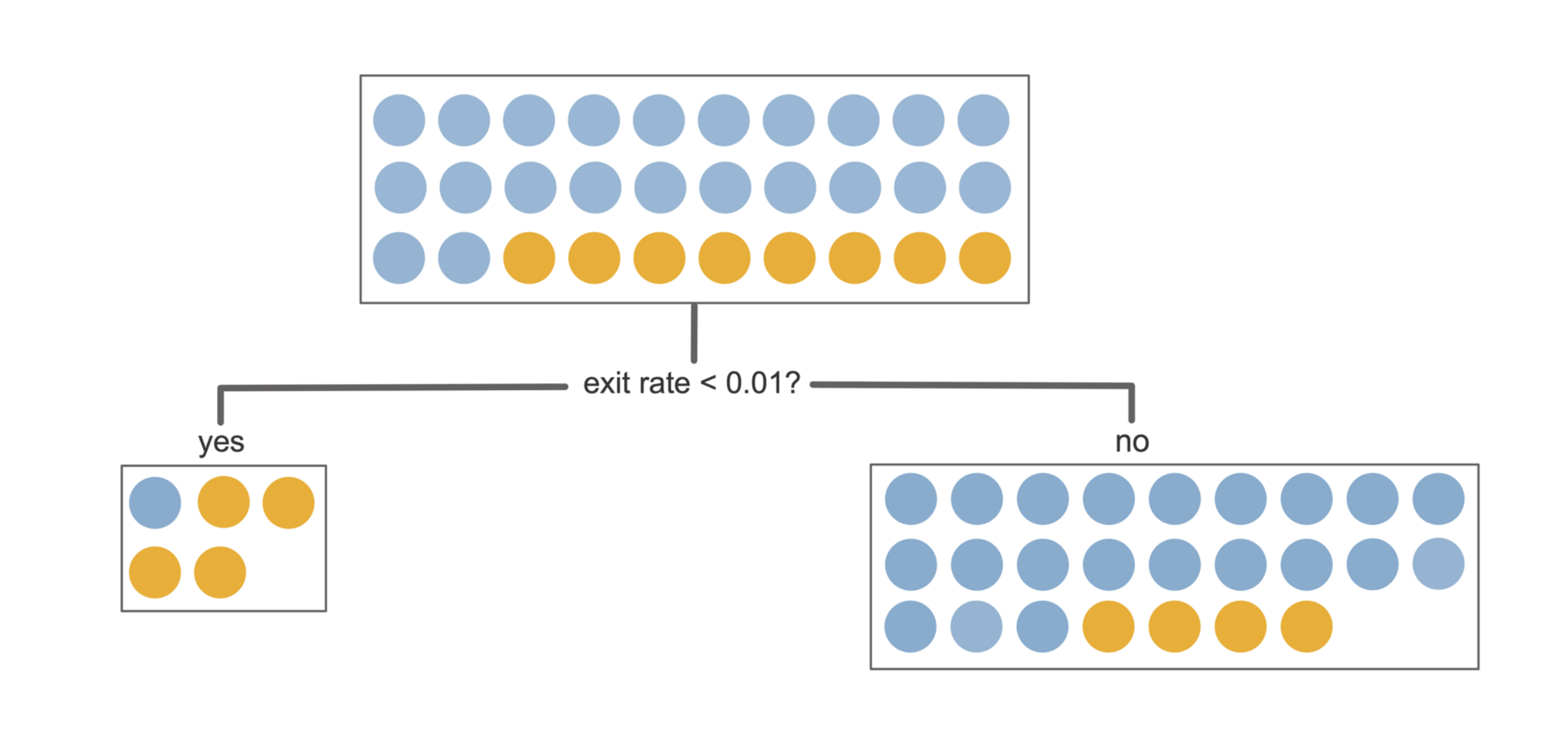
Let p1 be probability of success (positive class)
And p0 probability of failure or (negative class)

$$\text{Gini impurity} = 1 - p_1^2 - p_0^2 = 2p_0p_1,$$

[Yu, Barter 2024]

Your turn?

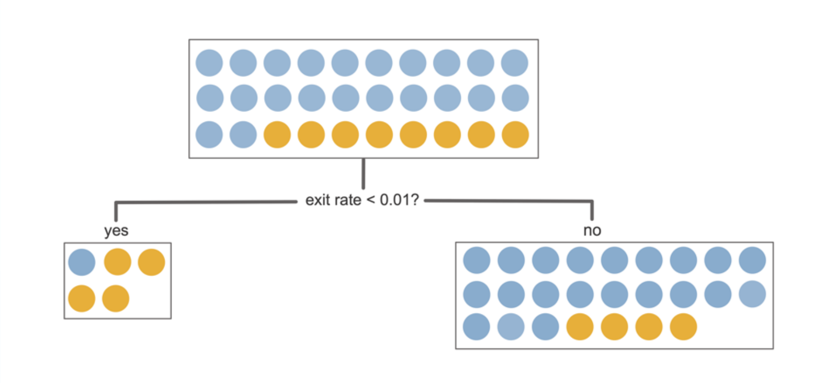
Calculate the Gini impurity of the left and right nodes



[Yu, Barter 2024]

Your turn?

Calculate the Gini impurity of the left and right nodes



[Yu, Barter 2024]

Gini impurity for split

Gini impurity for split is weighted average of Gini impurity for left and right child nodes.

$$\text{Gini impurity for split} = \frac{n_{\text{left}}}{n_{\text{parent}}}\text{Gini}_{\text{left}} + \frac{n_{\text{right}}}{n_{\text{parent}}}\text{Gini}_{\text{right}}$$

$$\text{Gini impurity for split} = \frac{5}{30} \times 0.32 + \frac{25}{30} \times 0.27 = 0.28.$$

Gini impurity decreases for child nodes

[Yu, Barter 2024]

Gini impurity for split

Choose the split to minimize the **Gini impurity split,**

Split	Gini
exit_rates < 0.01	0.28
exit_rates < 0.025	0.37
exit_rates < 0.031	0.36

[Yu, Barter 2024]

Regularization for CART

To avoid decision trees that are too deep, **regularize CART** by using **stopping criteria on hyperparameters:**

1) **Maximum tree depth** (maximum number of splits)
2) **Minimum node size**: The minimum number of observations required to split a parent node.

[Yu, Barter 2024]

To make predictions using CART

•**Positive class probability prediction**: The predicted positive class probability is the proportion of the *training observations* in the leaf node that are in the positive class.

•**Binary class prediction**: A binary prediction can be computed using the "majority" class (the class that makes up at least 50 percent of the observations) of the *training observations* in the leaf node or threshold on probability.

***Continuous response* predictions** can be computed using the average response of the *training observations* in the leaf node.

[Yu, Barter 2024]

CART summary

Start with a top-level parent node that contains all the training data observations. Then:

1. Conduct a **greedy search** for the “best” split of the parent node by identifying splits, calculating variance metric, and select split to minimize variance metric.

2. Implement the best split identified in the previous step to create two child nodes.

3. For each resulting child node, repeat steps 1 and 2 until stopping criteria. A child node that is not split further is called a **“leaf node”** or **“terminal node”**.

4. Continue until all child nodes are **leaf nodes**.

5. Generate predictions using average continuous response or positive class proportion.

[Yu, Barter 2024]

Random Forests

Random Forest (RF) algorithm is an **ensemble** algorithm that works by aggregating the predictions computed across many different decision trees.

This can handle wider ranges of continuous response variables and address overfitting.

[Yu, Barter 2024]

Random Forests

Random Forest (RF) algorithm is an **ensemble** algorithm that combines many decision trees:

•Training each tree using a different random bootstrap sample of the training data.

•Considering a different random subset of the predictor variables for each node split

[Yu, Barter 2024]

Random Forests Predictions

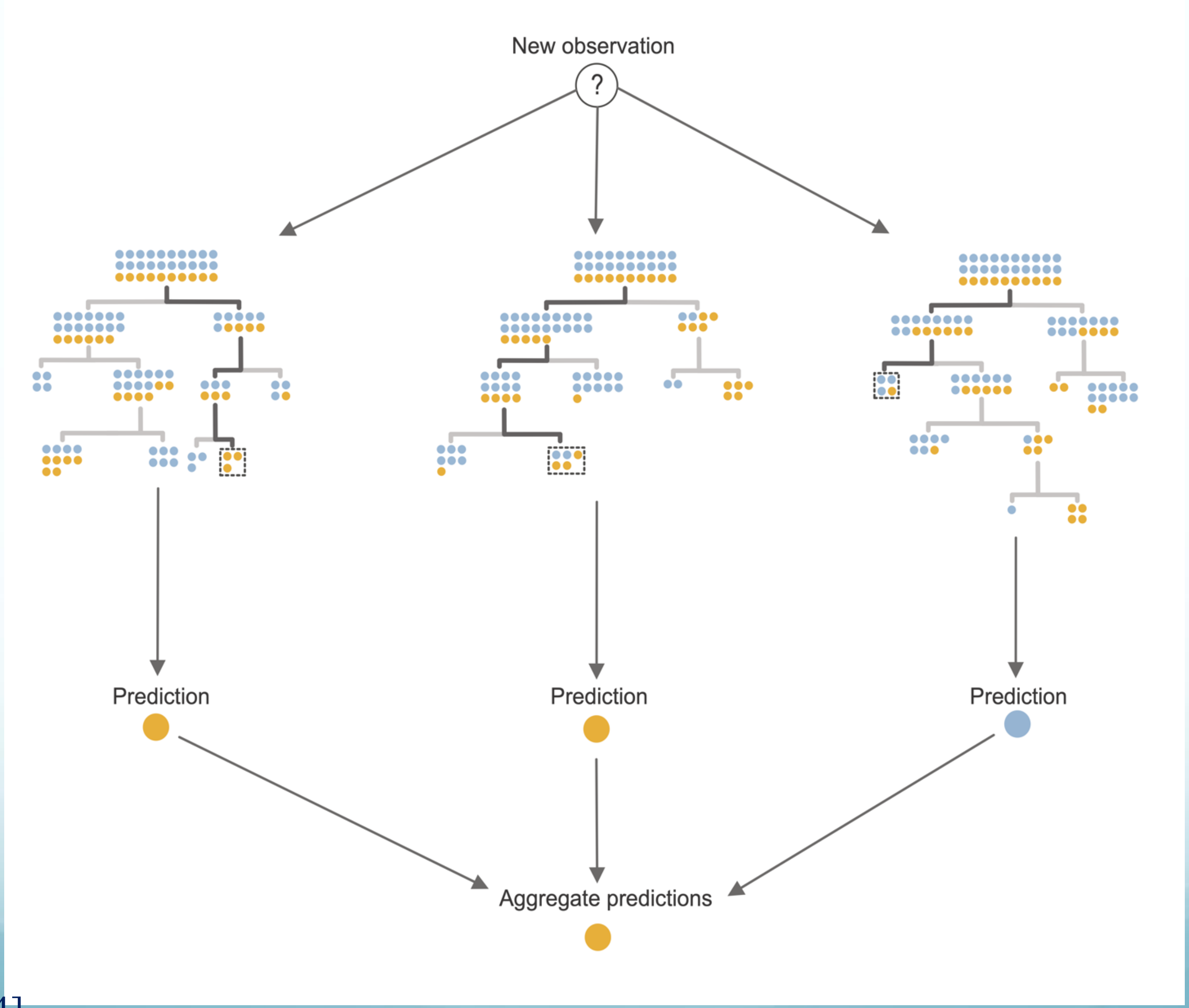
For binary response problems,
a **class probability prediction** can be computed based on the average probability prediction from the individual trees.

Binary class response predictions are computed either using a majority vote of the individual tree binary predictions or by applying a threshold to the class probability predictions.

A **continuous response prediction** can be computed by averaging the predicted responses across the individual trees.

[Yu, Barter 2024]

Random Forests Example



[Yu, Barter 2024]

Random Forests Hyperparameters

•**Number of variables to try**: The number of variables randomly selected for consideration at each split
•**Maximum depth**: The maximum depth of the tree
Minimum node size: The minimum number of observations in a node after which no more splits will occur.
•**Number of trees**: The number of trees in the forest

[Yu, Barter 2024]

A word on feature importance

Sometimes we want to learn about features in a model not just the predictions.

Regression and Logistic regression had **parameters** that we were estimating.

Those **parameters** were the coefficients of the predictors in the models.

To use coefficients to compare features, we must **standardize the features** before fitting the model or **standardize the coefficients**.

A word on feature importance

Notice that with decision trees and random forests, we are not learning any **parameters** for the models: **non-parametric supervised learning**

Very important: Decision Trees and Random Forest are **invariant to monotonic transformations** of the underlying variables, (i.e. logarithmic, square-root transformations or standardization do not affect results)

However, we may still want to learn what features are helping us in predictions.

[Yu, Barter 2024]

A word on feature importance

For Random Forests:

permutation feature importance score: measures how much the mean squared error (MSE; for continuous responses) or prediction accuracy (for binary responses) decreases when retraining the RF algorithm using a permuted version of x with the original unpermuted other predictive features.

This relates to hypothesis testing. (We'll come back to this a little later in course.)

[Yu, Barter 2024]

A word on feature importance

For Random Forests:

Gini impurity feature importance score or ***mean decrease in impurity (MDI)*** for feature x can be computed by aggregating the decreases in Gini impurity (binary) or variance (continuous) across each split involving feature x across all the trees in the forest.

[Yu, Barter 2024]

Your turn:
Decision Trees and Random Forests

Please get the Jupyter notebook for decision tress and random forests on shopping data:

Go to:
https://colab.research.google.com/drive/18D0e6yNsKb_osQVpyZebXuZY29HJubVO?usp=sharing

Save a copy to your Google Drive and keep notes there...

[Yu, Barter 2024]

Today's Learning Objectives

Students will be able to:

- ✓ Review: Evaluate classification problems with quantitative metrics
- ✓ Review: Apply KNN algorithm by hand to a small sample data set
- ✓ Decision Trees and Random Forests

Citations:

Yu, B., & Barter, R. L. (2024). Veridical data science: The practice of responsible data analysis and decision making. The MIT Press.

Shah. C. (2020) A hands-on introduction to data science. Cambridge University Press.

Kelleher, J. D., MacNamee, B.,, D'Arcy, A. (2015). *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. Cambridge, MA: MIT Press. ISBN: 978-0-262-02944-5

Sebastian **Raschka**, Yuxi (Hayden) **Liu**, and Vahid Mirjalili. **Machine Learning** with PyTorch and Scikit-Learn. Packt Publishing, **2022**