Lecture 2: Terminology, Baselines, Decision Trees

Firas Moosvi (Slides adapted from Varada Kolhatkar)

Announcements

- Things due this week
  - Homework 1 (hw1): Due May 16 17:59
- You can find the tentative due dates for all deliverables here.
- Please monitor Ed Discussion (especially pinned posts and instructor posts) for announcements.
- I'll assume that you've watched the pre-lecture videos.

Surveys

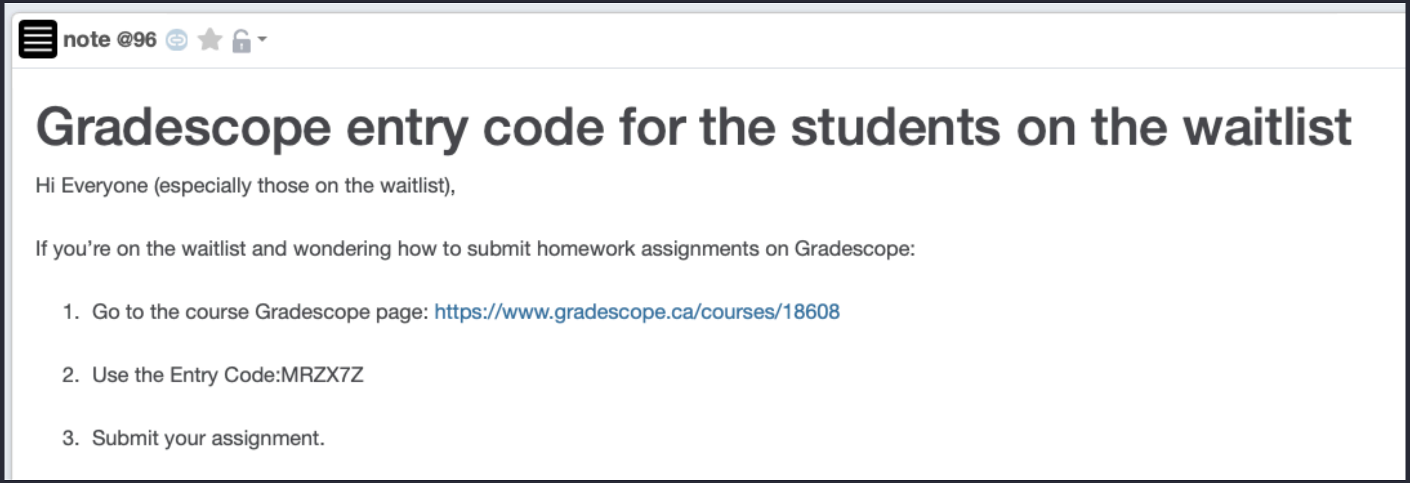
- Please complete the anonymous restaurant survey on Qualtrics here.
- We will try to analyze this data set in the coming weeks.

Gradescope

Make sure you can submit your assignment before the hw1 due date!

It is **required** for you to work in a GitHub repository, please maintain your GitHub repo up-to-date.

For students on the waitlist: Gradescope Entry code is **9KK5ZR**.



note @96

Gradescope entry code for the students on the waitlist

Hi Everyone (especially those on the waitlist),

If you're on the waitlist and wondering how to submit homework assignments on Gradescope:

1. Go to the course Gradescope page: https://www.gradescope.ca/courses/18608
2. Use the Entry Code:MRZX7Z
3. Submit your assignment.

Demo: Submit hw1 on Gradescope

We are going to practice submitting HW1 on Gradescope so you all do it at least once!

Checklist for you in the first week

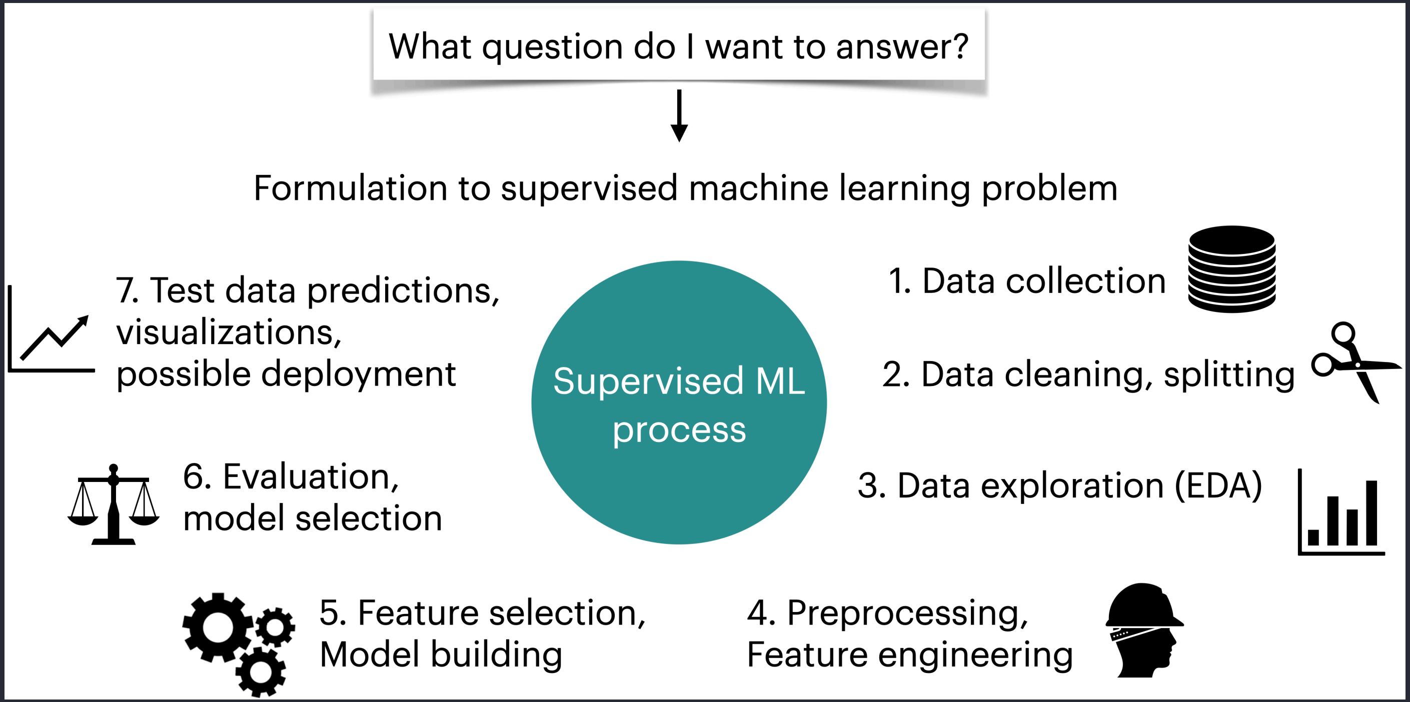
- ☐ Are you able to access course Canvas shell?
- ☐ Are you able to access course forum: Ed Discussion?
- ☐ Are you able to access Gradescope? (If not, refer to the Gradescope Student Guide.)
- ☐ Are you able to access iClicker Cloud for this course?
- ☐ Did you follow the setup instructions here to create a course conda environment on your computer?
- ☐ Did you complete the "Getting to know you" survey on Canvas?
- ☐ Did you complete the anonymous restaurant survey on Qualtrics?
- ☐ Are you almost finished or at least started with homework 1?

Suggested Workflow for working with Jupyter Notebooks

- Create a folder on your computer that will have all the CPSC 330 repos:
  - `~/School/Year3/CPSC330/` <– Consider this your CPSC parent folder
- Create subfolders for: `hw`, `class`, `practice`
- In the `hw` folder, you will then clone `hw1`, `hw2`, `hw3`, etc…
- In the `class` folder, you will clone the `cpsc330-2025S1` repo which contains all the class jupyter notebooks
  - Do **not** make any changes to files in this directory/repo, you will have trouble when you pull stuff during each class.
  - If you did make changes, you can reset to the last commit and DESTROY any changes you made (be careful with this command) using: `git reset --hard`
- In the `practice` folder, you can **copy** any notebooks (`.ipynb`) and files (like data/*.csv) you want to try running locally and experiment

Recap: Machine learning workflow

Supervised machine learning is quite flexible; it can be used on a variety of problems and different kinds of data. Here is a typical workflow of a supervised machine learning systems.


What question do I want to answer?
Formulation to supervised machine learning problem
Supervised ML process
1. Data collection
2. Data cleaning, splitting
3. Data exploration (EDA)
4. Preprocessing, Feature engineering
5. Feature selection, Model building
6. Evaluation, model selection
7. Test data predictions, visualizations, possible deployment

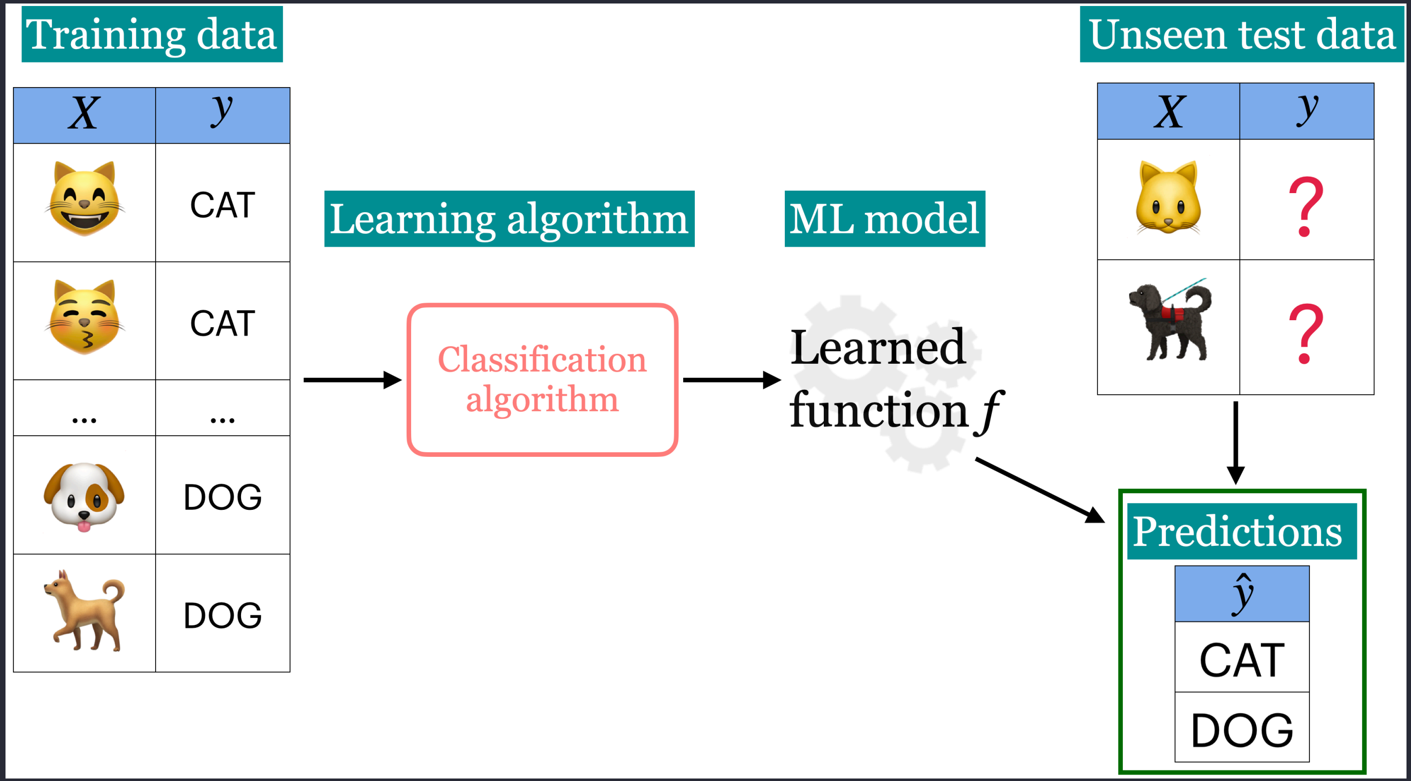
We will build machine learning pipelines in this course, focusing on some of the steps above.

Recap: What is ML?

- ML uses data to build models that find patterns, make predictions, or generate content.
- It helps computers learn from data to make decisions.
- No one model works for every situation.

Recap: Supervised learning

- We wish to find a model function f that relates X to y.
- We use the model function to predict targets of new examples.



In the first part of this course, we'll focus on supervised machine learning.

iClicker 2.1: Terminology

iClicker cloud join link: https://join.iclicker.com/YJHS

Select all of the following statements which are True (iClicker)

a. Predicting spam is an example of machine learning.

b. Predicting housing prices is not an example of machine learning.

c. For problems such as spelling correction, translation, face recognition, spam identification, if you are a domain expert, it's usually faster and scalable to come up with a robust set of rules manually rather than building a machine learning model.

d. If you are asked to write a program to find all prime numbers up to a limit, it is better to implement one of the algorithms for doing so rather than using machine learning.

e. Google News is likely be using machine learning to organize news.

iClicker 2.2: Supervised vs unsupervised

Clicker cloud join link: https://join.iclicker.com/YJHS

Select all of the following statements which are examples of supervised machine learning

iClicker 2.3: Classification vs. Regression

Clicker cloud join link: https://join.iclicker.com/YJHS

Select all of the following statements which are examples of regression problems

- a. Predicting the price of a house based on features such as number of bedrooms and the year built.
- b. Predicting if a house will sell or not based on features like the price of the house, number of rooms, etc.
- c. Predicting percentage grade in CPSC 330 based on past grades.
- d. Predicting whether you should bicycle tomorrow or not based on the weather forecast.
- e. Predicting appropriate thermostat temperature based on the wind speed and the number of people in a room.

ML Framework in CPSC 330

- There are many frameworks to do do machine learning.
- We'll mainly be using `scikit-learn` framework.

Running example

Imagine you're in the fortunate situation where, after graduating, you have a few job offers and need to decide which one to choose. You want to pick the job that will likely make you the happiest. To help with your decision, you collect data from like-minded people. Here are the first few rows of this toy dataset.

```python
toy_happiness_df = pd.read_csv(DATA_DIR + 'toy_job_happiness.csv')
toy_happiness_df
```

	supportive_colleagues	salary	free_coffee	boss_vegan	happy?
0	0	70000	0	1	Unhappy
1	1	60000	0	0	Unhappy
2	1	80000	1	0	Happy
3	1	110000	0	1	Happy
4	1	120000	1	0	Happy
5	1	150000	1	1	Happy
6	0	150000	1	0	Unhappy

Terminology

Features, target, example

- What are the **features** X?
  - features = inputs = predictors = explanatory variables = regressors = independent variables = covariates
- What's the target y?
  - target = output = outcome = response variable = dependent variable = labels
- Can you think of other relevant features for this problem?
- What is an example?

Classification vs. Regression

- Is this a **classification** problem or a **regression** problem?

	supportive_colleagues	salary	free_coffee	boss_vegan	happy?
0	0	70000	0	1	Unhappy
1	1	60000	0	0	Unhappy
2	1	80000	1	0	Happy
3	1	110000	0	1	Happy
4	1	120000	1	0	Happy
5	1	150000	1	1	Happy
6	0	150000	1	0	Unhappy

Prediction vs. Inference

- **Inference** is using the model to understand the relationship between the features and the target
  - Why certain factors influence happiness?
- **Prediction** is using the model to predict the target value for new examples based on learned patterns.
- Of course these goals are related, and in many situations we need both.

Training

- In supervised ML, the goal is to learn a function that maps input features (X) to a target (y).
- The relationship between X and y is often complex, making it difficult to define mathematically.
- We use algorithms to approximate this complex relationship between X and y.
- **Training** is the process of applying an algorithm to learn the best function (or model) that maps X to y.
- In this course, I'll help you develop an intuition for how these models work and demonstrate how to use them in a machine learning pipeline.

Separating X and y

- In order to train a model we need to separate X and y from the dataframe.

```python
X = toy_happiness_df.drop(columns=["happy?"]) # Extract the feature set by removing the target column "happy
y = toy_happiness_df["happy?"] # Extract the target variable "happy?"
```

Baseline

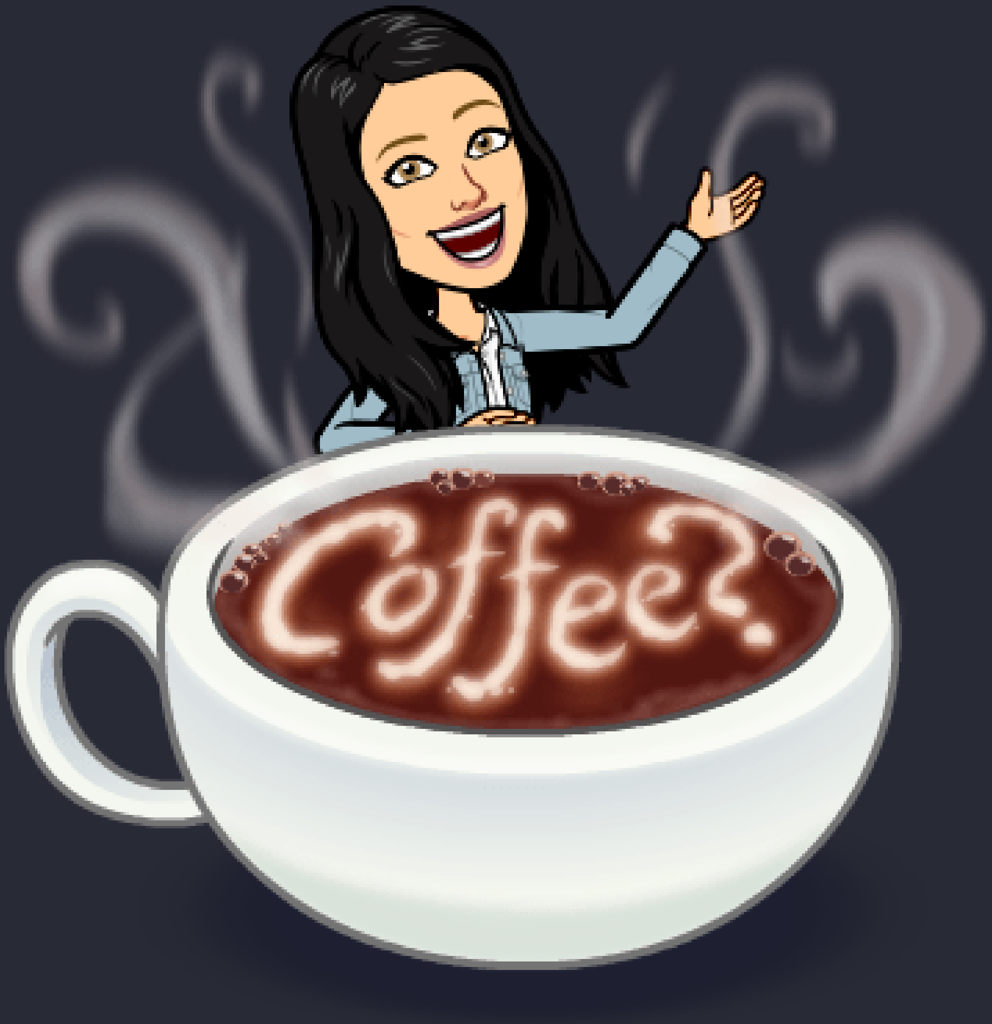
- Let's try a simplest algorithm of predicting the most popular target!

```python
from sklearn.dummy import DummyClassifier
# Initialize the DummyClassifier to always predict the most frequent class
model = DummyClassifier(strategy="most_frequent")
# Train the model on the feature set X and target variable y
model.fit(X, y)
# Add the predicted values as a new column in the dataframe
toy_happiness_df['dummy_predictions'] = model.predict(X)
# Show the dataframe
toy_happiness_df
```

	supportive_colleagues	salary	free_coffee	boss_vegan	happy?	du
0	0	70000	0	1	Unhappy	Ha
1	1	60000	0	0	Unhappy	Ha
2	1	80000	1	0	Happy	Ha
3	1	110000	0	1	Happy	Ha
4	1	120000	1	0	Happy	Ha
5	1	150000	1	1	Happy	Ha
6	0	150000	1	0	Unhappy	Ha

Break

Let's take a break!



Decision trees

Activity: 20 Questions

Let’s play 20 questions! You can ask me up to 20 Yes/No questions to figure out the answer.

I’m thinking of a person - who is it ?

Intuition

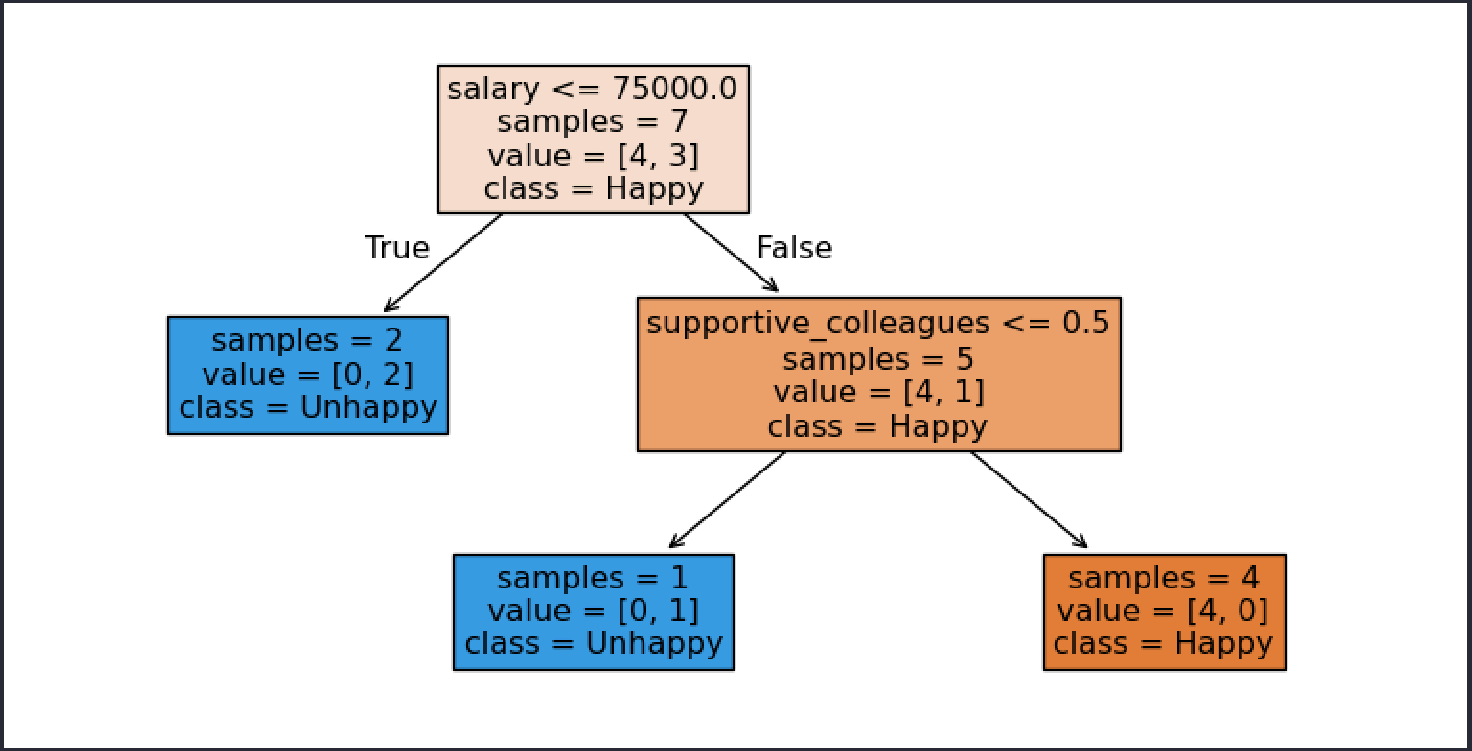
- Decision trees find the “best” way to split data to make predictions.
- Each split is based on a question, like ‘Are the colleagues supportive?’
- The goal is to group data by similar outcomes at each step.
- Now, let’s see a decision tree using sklearn.

Decision tree with `sklearn`

Let's train a simple decision tree on our toy dataset.

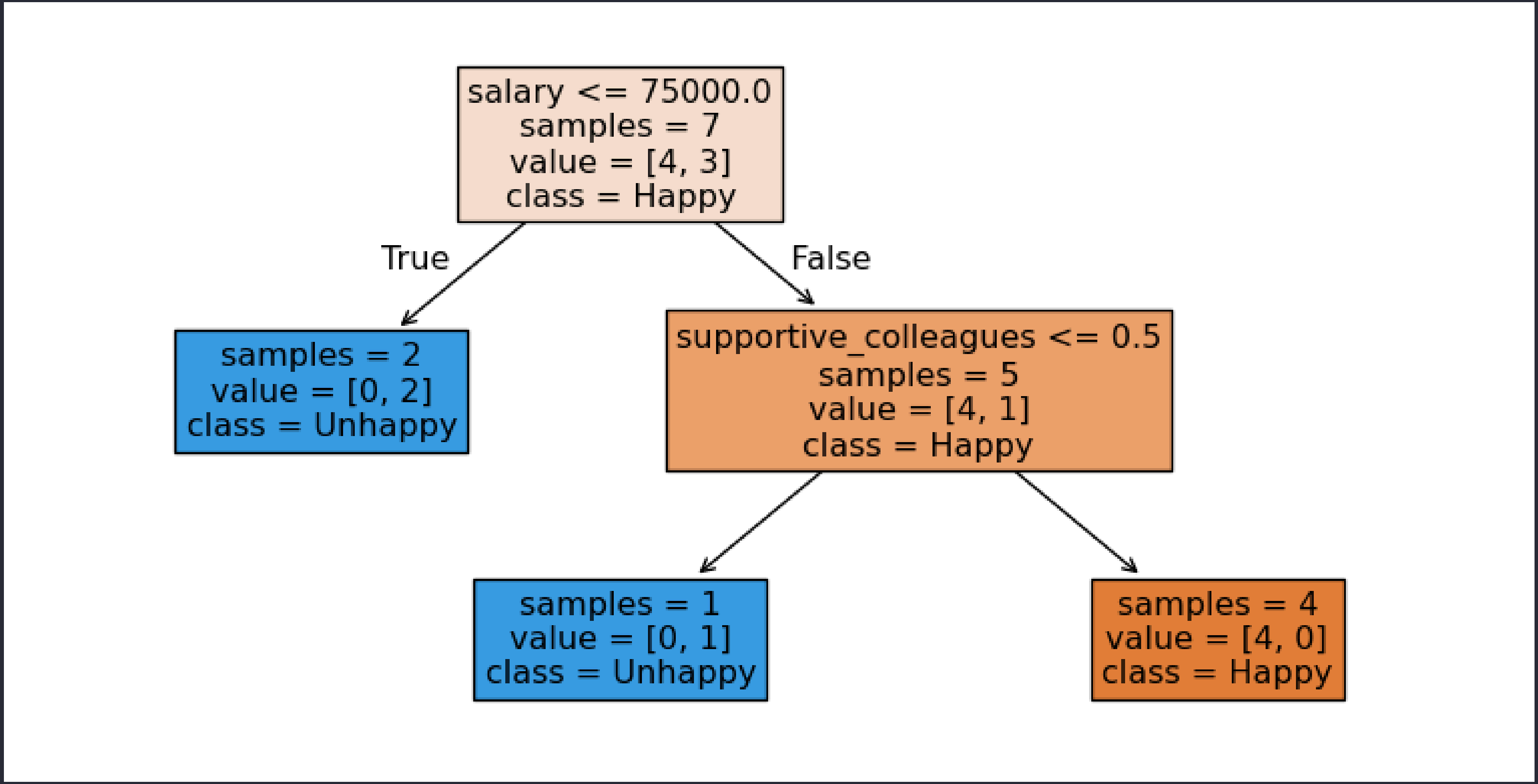
```python
from sklearn.tree import DecisionTreeClassifier # import the classifier
from sklearn.tree import plot_tree

model = DecisionTreeClassifier(max_depth=2, random_state=1) # Create a class object
model.fit(X, y)
plot_tree(model, filled=True, feature_names = X.columns, class_names=["Happy", "Unhappy"],
          impurity = False, fontsize=12);
```



Prediction

- Given a new example, how does a decision tree predict the class of this example?
- What would be the prediction for the example below using the tree above?
  - supportive_colleagues = 1, salary = 60000, coffee_machine = 0, vegan_boss = 1,

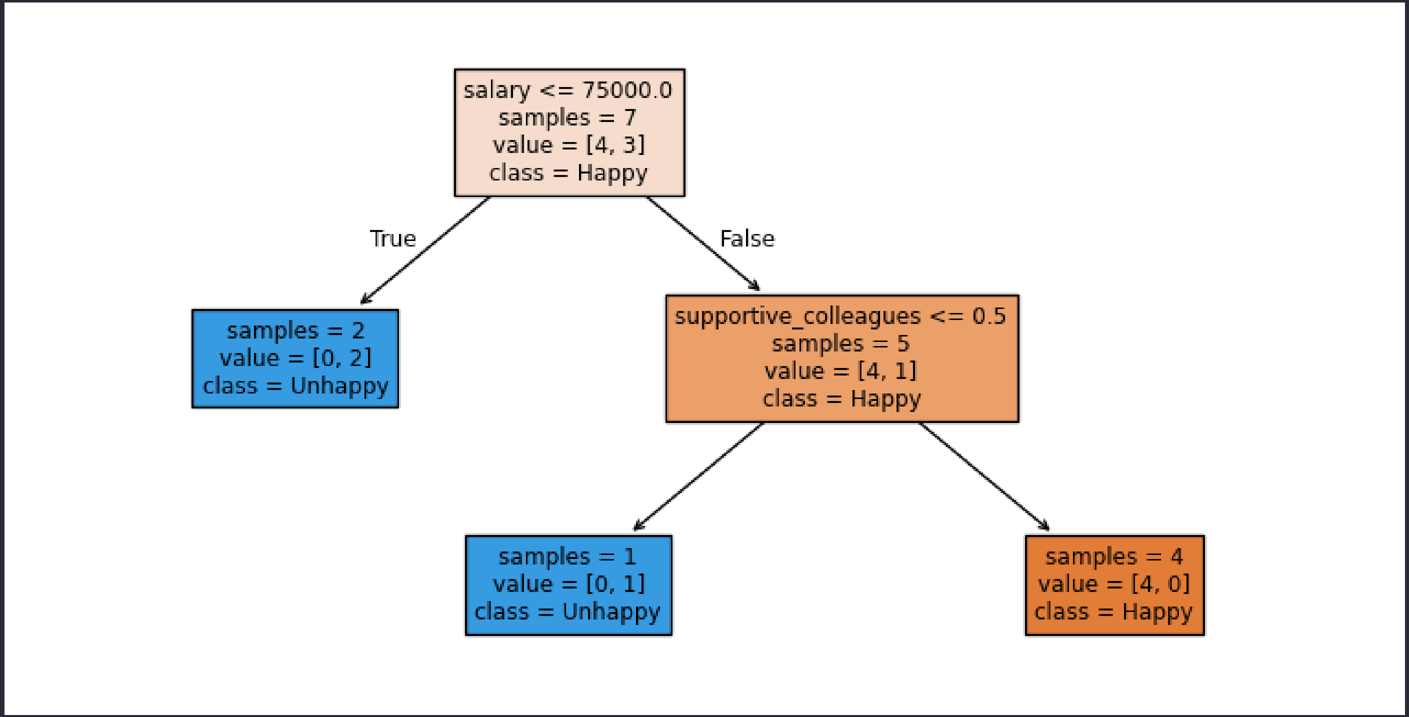


Prediction with `sklearn`

- What would be the prediction for the example below using the tree above?
  - supportive_colleagues = 1, salary = 60000, coffee_machine = 0, vegan_boss = 1,

```python
test_example = [[1, 60000, 0, 1]]
print("Model prediction: ", model.predict(test_example))
plot_tree(model, filled=True, feature_names = X.columns, class_names = ["Happy", "Unhappy"], impurity = Fals
```

```
Model prediction:  ['Unhappy']
```



Training (high level)

- How many possible questions could we ask in this context?

	supportive_colleagues	salary	free_coffee	boss_vegan
0	0	70000	0	1
1	1	60000	0	0
2	1	80000	1	0
3	1	110000	0	1
4	1	120000	1	0
5	1	150000	1	1
6	0	150000	1	0

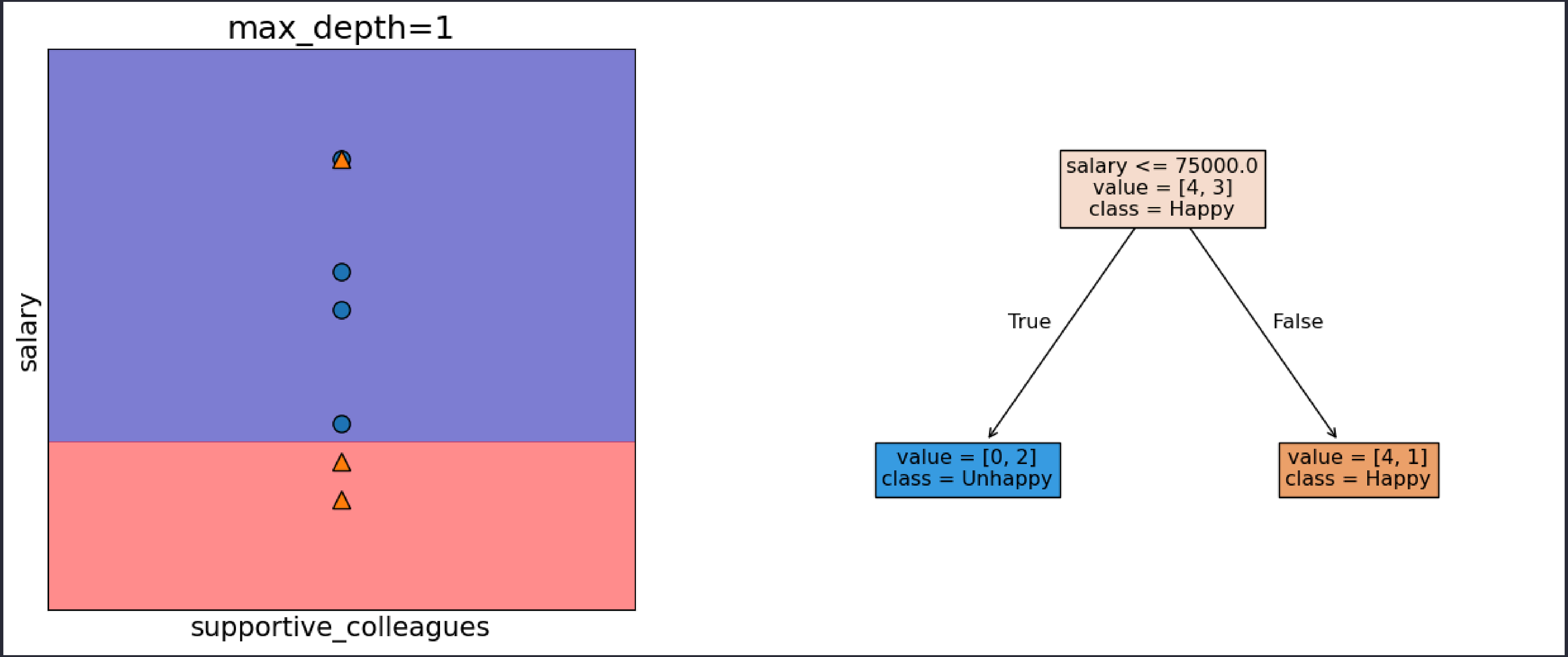
Training (high level)

- Decision tree learning is a search process to find the “best” tree among many possible ones.
- We evaluate questions using measures like **information gain** or the **Gini index** to find the most effective split.
- At each step, we aim to split the data into groups with more certainty in their outcomes.

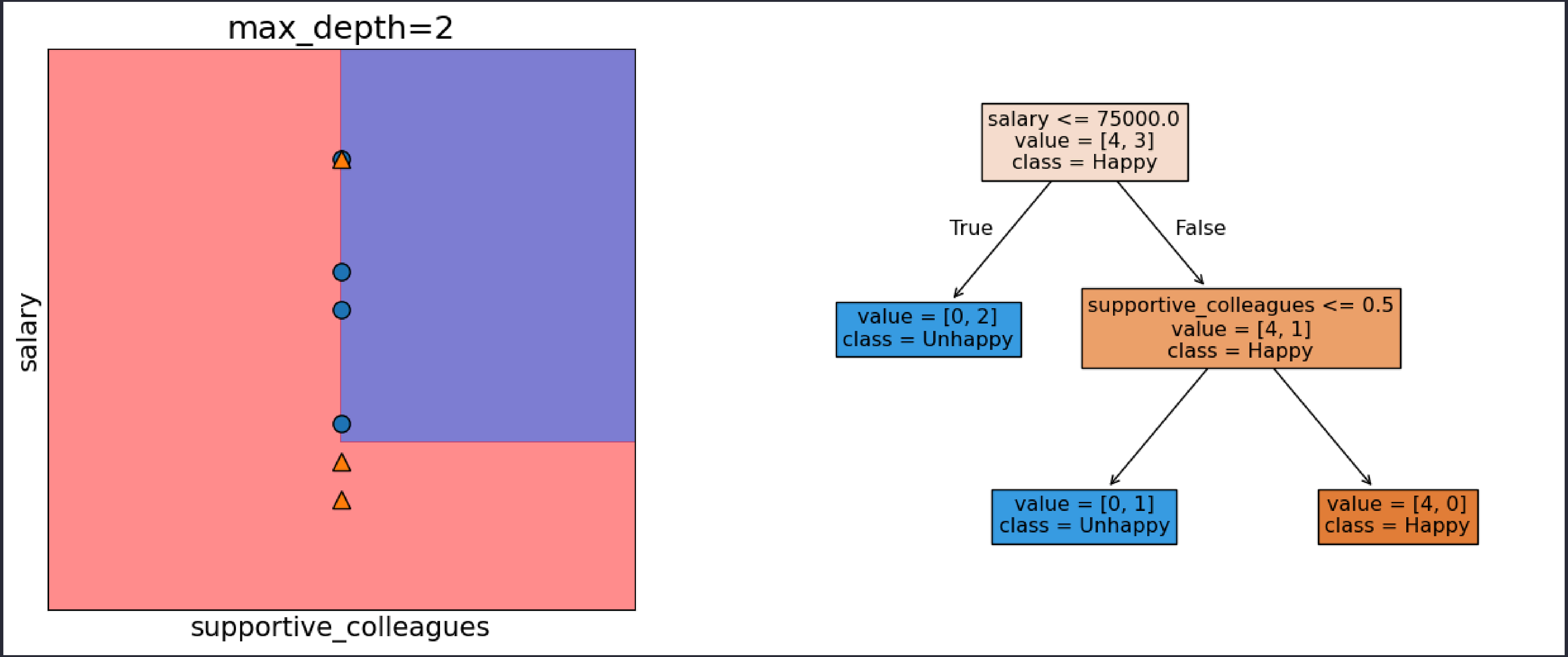
Parameters vs. Hyperparameters

- Parameters
  - The questions (features and thresholds) used to split the data at each node.
  - Example: salary <= 75000 at the root node
- Hyperparameters
  - Settings that control tree growth, like `max_depth`, which limits how deep the tree can go.

Decision boundary with `max_depth=1`



Decision boundary with `max_depth=2`



iClicker 2.5: Baselines and Decision trees

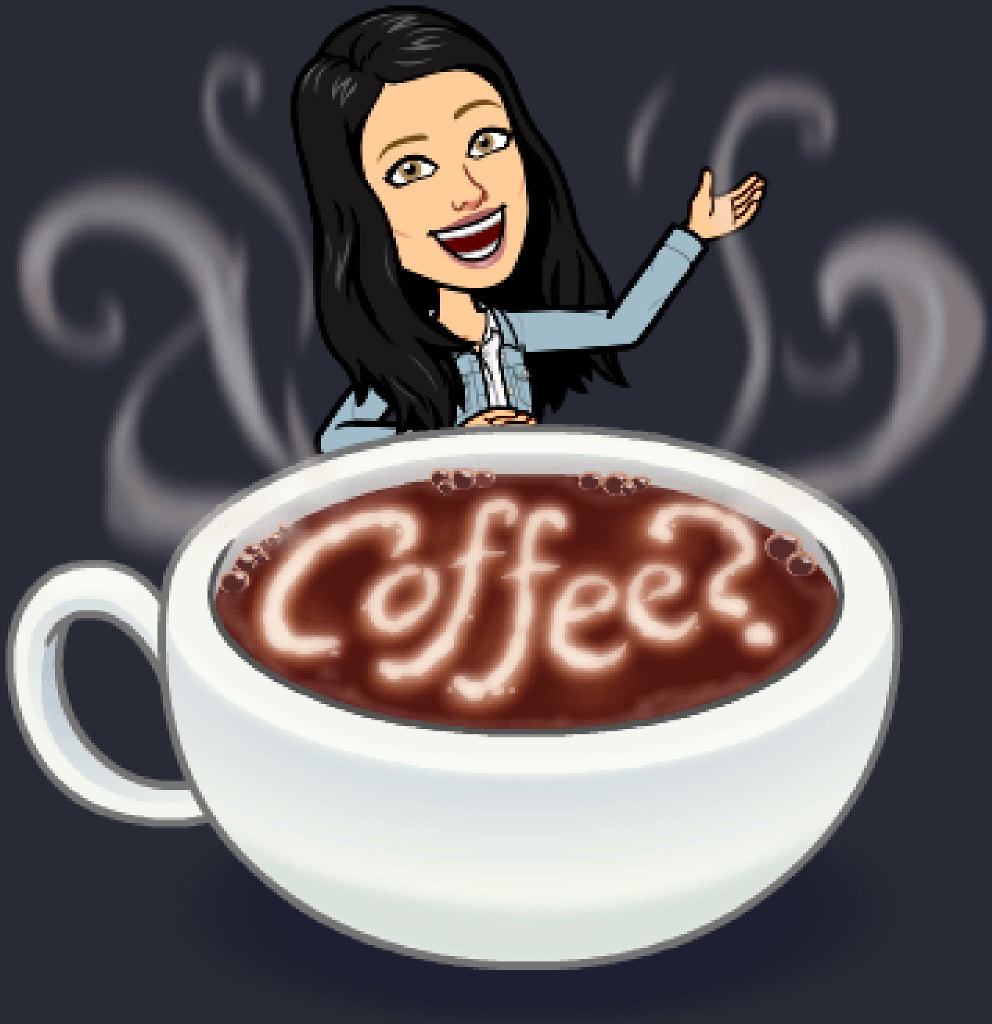
iClicker cloud join link: https://join.iclicker.com/YJHS

Select all of the following statements which are TRUE.

- a. Change in features (i.e., binarizing features above) would change DummyClassifier predictions.
- b. predict takes only X as argument whereas fit and score take both X and y as arguments.
- c. For the decision tree algorithm to work, the feature values must be binary.
- d. The prediction in a decision tree works by routing the example from the root to the leaf.

Break

Let’s take a break!



Group Work: Class Demo & Live Coding

In some of the classes, we will do a bit of live coding to get your used to practical machine learning. You are **highly encouraged** to follow along - we won't usually finish *everything* in the demo, but it should be a significant portion that you can finish off after class.

For this demo, each student should click this link to create a new repo in their accounts, then clone that repo locally to follow along with the demo from today.