Introduction

In the last lesson, we learned about decision trees. And saw that decision trees essentially provide us with a series of tests to determine if our data falls into one category or another.

In this lesson, we'll see how to start with a dataset and, from there, construct a decision tree. Our technique will be the following.

To predict whether our data will fall into one group or another, we find a characteristic that most separates our data into those different groups. Then, we find the characteristic that most separates our remaining data. When we are done separating all of our data, we have a mechanism to predict what will occur with future data. Let's see this by way of example.

Our Data

Once again, we have the following data about our customer leads.

Attended College	Under Thirty	Borough	Income	Customer
?	Yes	Manhattan	< 55	0
Yes	Yes	Brooklyn	< 55	0

?	No	Brooklyn	< 55	1
No	No	Queens	> 55	1
?	No	Queens	< 55	1
Yes	No	Queens	>55	0
Yes	No	Queens	>55	0
Yes	Yes	Manhattan	>55	0

Once again, we use the data to predict whether or not a lead will be a customer. Just like always, we have the "answers" for this data, which tell ius if our past leads became a customer. That information is in the `customer` column. The number 1 means that the lead became a customer, and 0 means they did not.

The rest of our columns are the features of the data we'll use to predict whether someone becomes a customer.

Our Procedure

We'll predict whether someone will be come a customer by looking at the characteristics that most distinguished customers from non-customers. So we'll use a series of tests. For the above observations we can ask questions like:

- Does the lead have a graduate level of education?
- Is the lead under thirty?
- Is the lead from Manhattan? Or from Brooklyn?
- Is the lead's income over 55k?

The responses to these tests can help us predict whether or not someone will become a customer. If we see that that everyone under thirty is a customer, then we might be able to use this to predict how future leads will turn out. Once we choose one test that best separates our data, we'll then look at the rest of the data that was not separated, and find the test that best separates this remaining data.

Let's summarize our procedure.

```
Given a subset of data
For each feature in our dataset
  ○ Split the data according to the feature
  ○ Compute consistency of data in each split
Choose the feature with the highest consistency to
split the data
Repeat for remaining subset
```

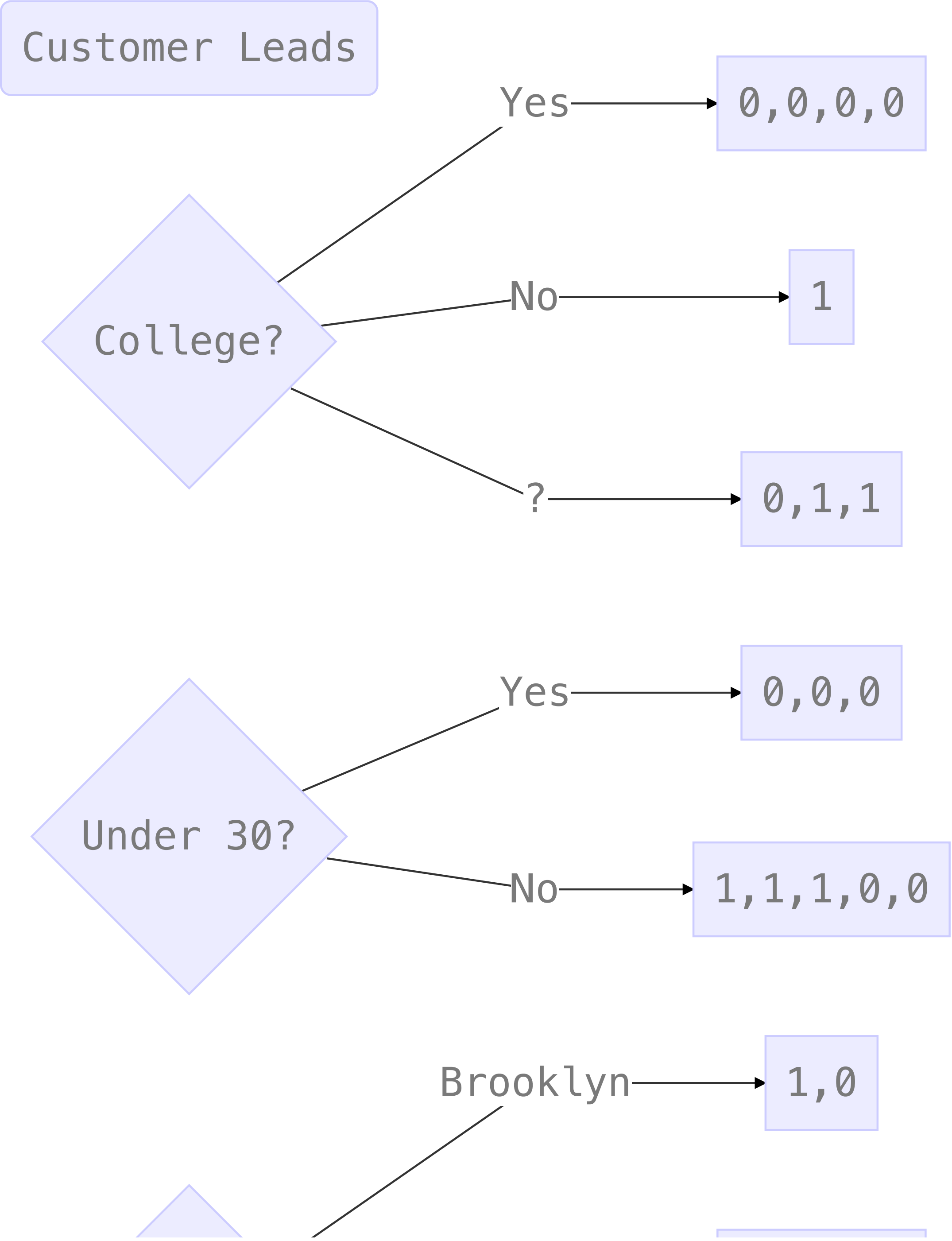
Let's try it out on the data to see how it works. Let's go.

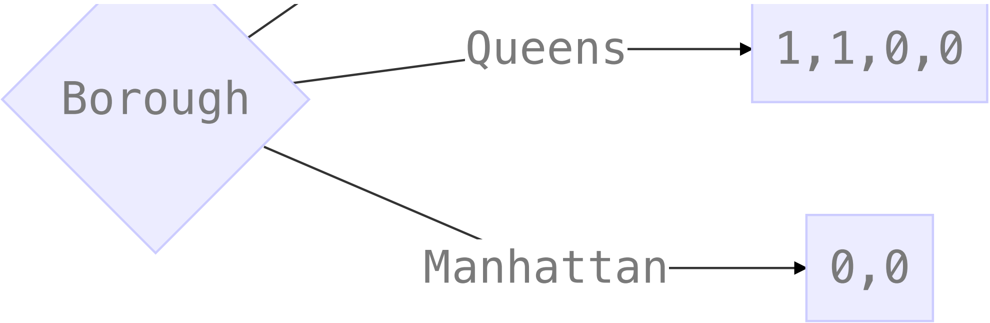
1. Split the data according to each feature

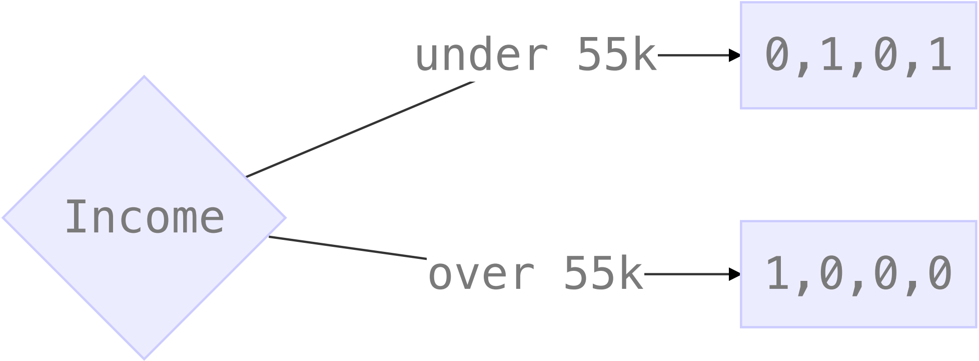
Let's take another look at our first steps.

```
Given a subset of data
For each feature in our dataset
  ○ Split the data according to the feature
  ○ Compute consistency of data in each split
```

When beginning our process we don't use a subset of data, but our entire dataset, and then split the data for along each feature. We have the splits below, as well as how it splits the data into customers and non-customers. (The 1 represents a customer, and 0 represents a non-customer.)







Looking at the splits above, we see some splits do a fairly good job of segmenting our leads into customers and non-customers. Others, not so much.

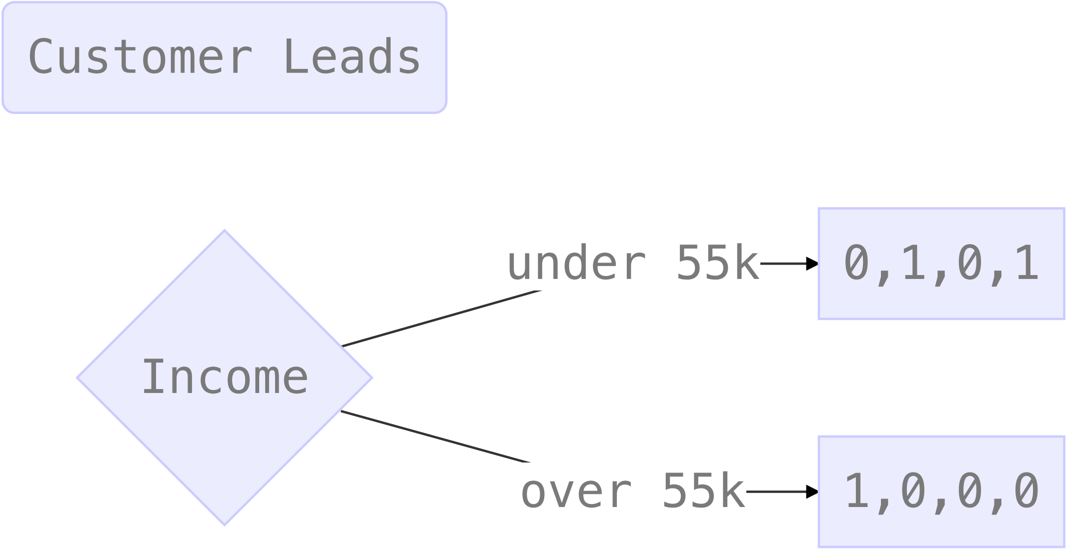
2. Compute consistency of data in each split

So the next step, is to compute the consistency in each split. That is, we want to choose the test that best separates our leads into customers and non-customers. After all, if we test whether leads earn less than 55k, and we find the same proportion of customers and non-customers as we started with, this test didn't help us predict the future outcome.

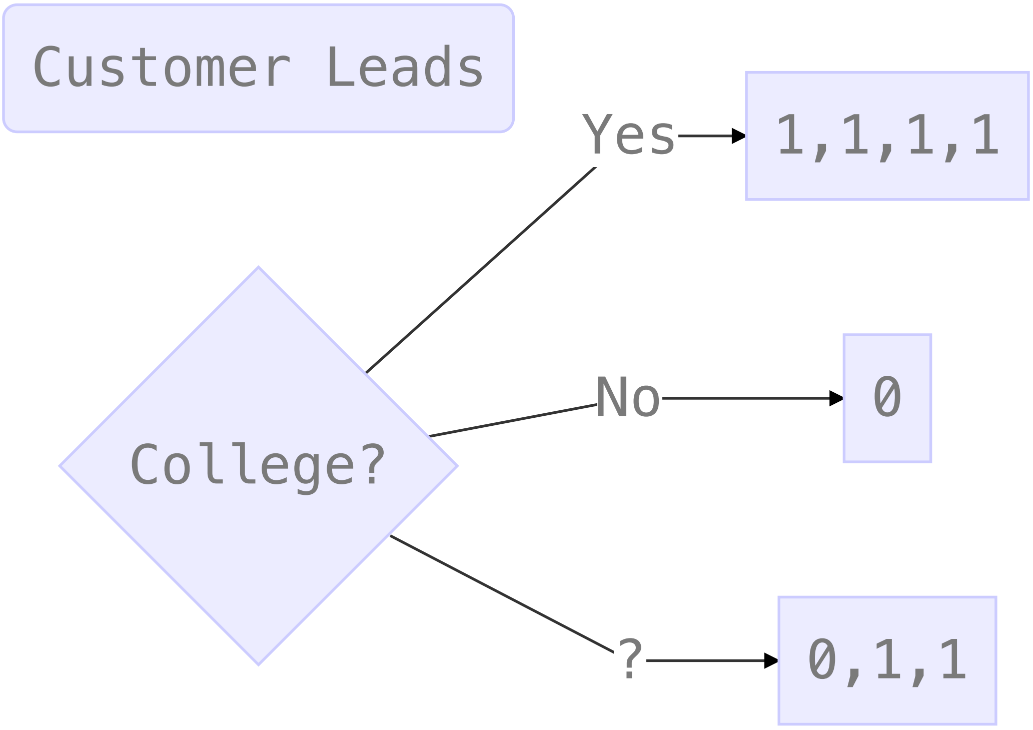
For now, we'll rank each test by counting the number of leads that the test places into a homogeneous group (that is, a group where all are either a customer, or all or either not.).

Take a look at the diagrams below.

> *The income test scores a zero as there are no leads in a homogeneous group.*



> *The college test does the best across all of our features with a score of a five.*



3. Choose the feature with the highest consistency

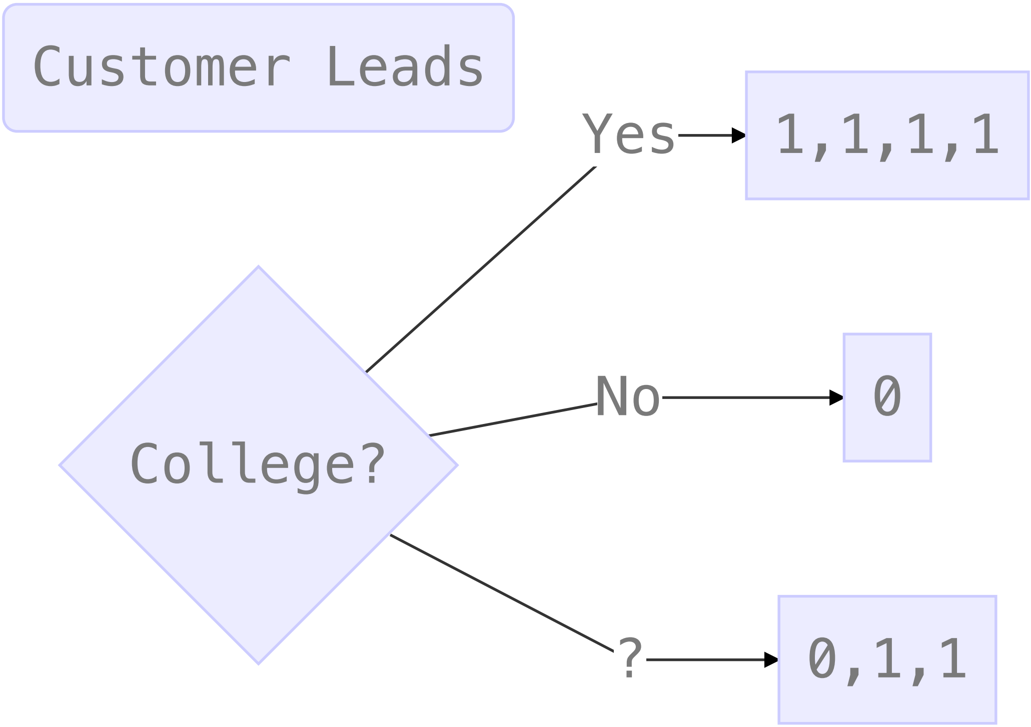
Because the college test scores the highest across all of our features, we start with using college to predict whether or not someone will become a customer.

- If a lead attended college, we predict they will become a customer.
- If they did not attend, we predict that they will not become a customer.

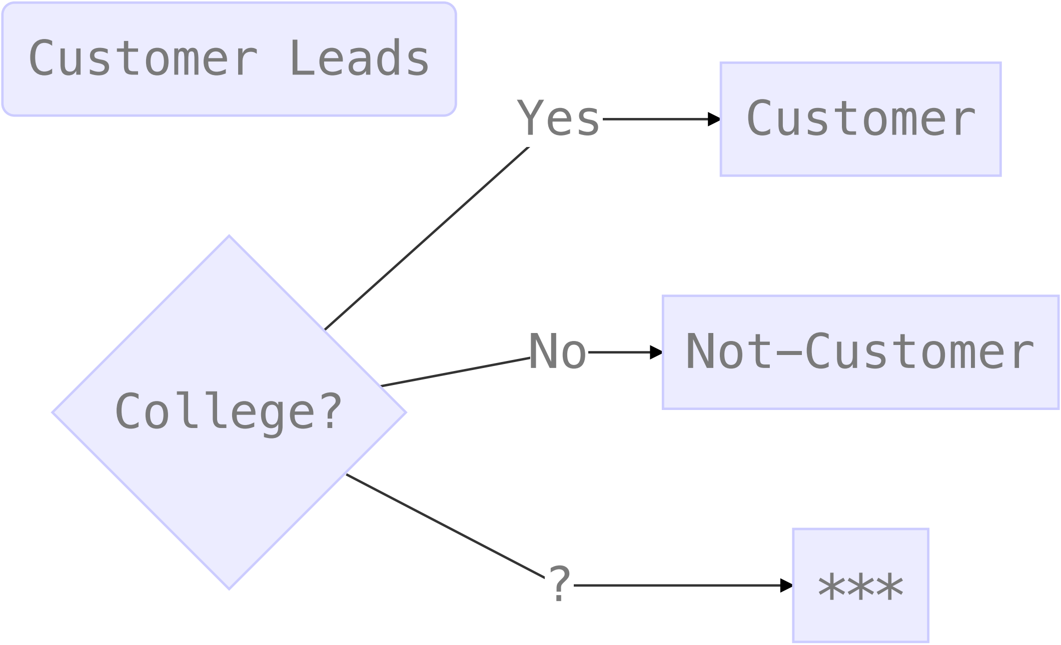
- And when we see a question mark for college, we do not yet make a prediction.

  So we just turned our above data into a hypothesis as to our future outcomes.

In other words we started with this.



And turned it into this.



The diagram is the beginning of a decision tree. It isn't yet finished, as it does not make a prediction about leads who do not provide us with college information. So to make predictions about these leads, we again look at our data to continue training our decision tree.

4. Repeat for remaining subset

Looking at our training data, we see that we have three observations with question marks next to attended college, and these are the observations that we cannot yet make a prediction for.

Attended College	Under Thirty	Borough	Income	Customer	Customer Predictions

?	Yes	Manhattan	< 55	0	?
Yes	Yes	Brooklyn	< 55	0	1
?	No	Brooklyn	< 55	1	?
No	No	Queens	> 55	1	0
?	No	Queens	< 55	1	?
Yes	No	Queens	>55	0	1
Yes	No	Queens	>55	0	1
Yes	Yes	Manhattan	>55	0	1

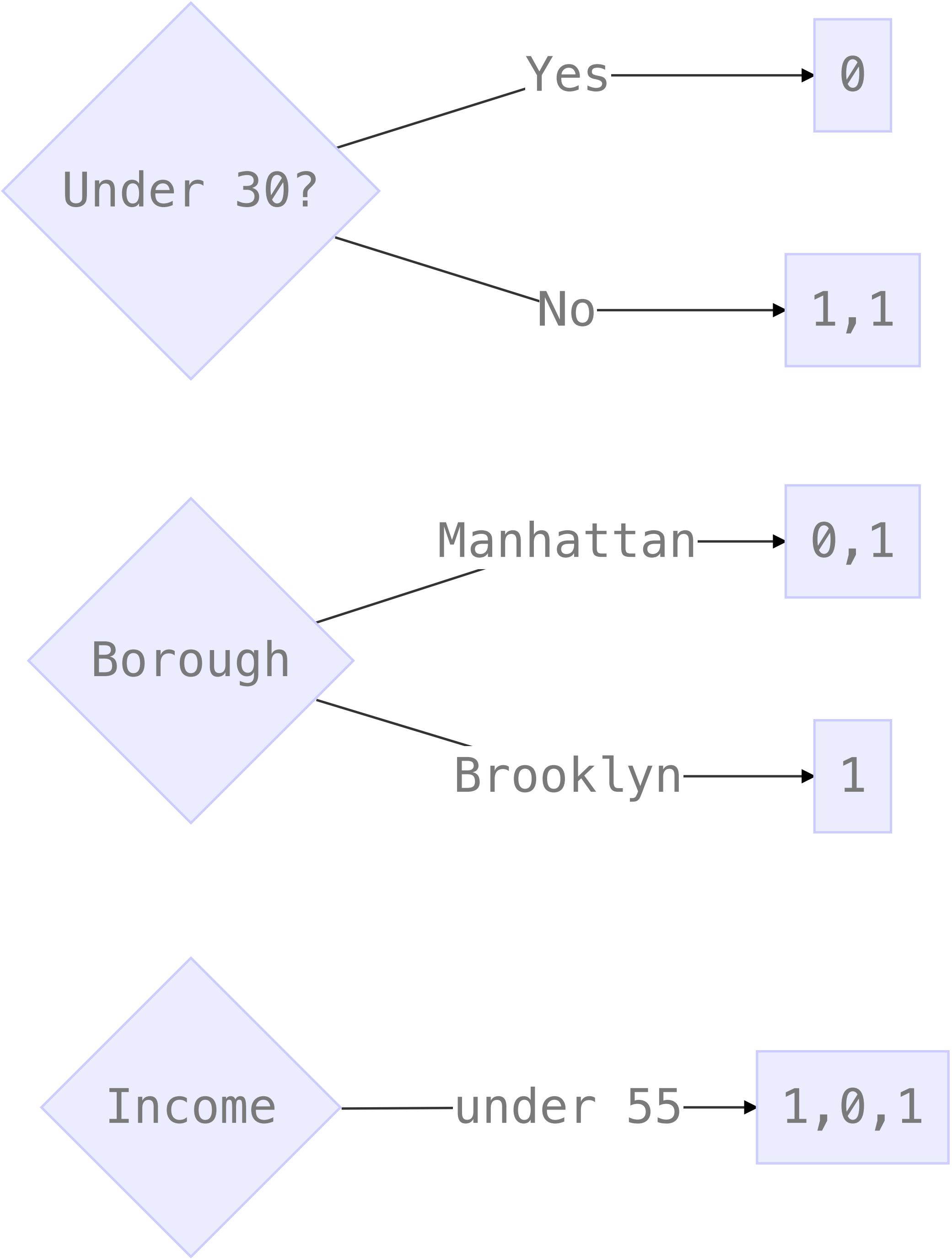
So the next thing we do focus on the remaining training data we did not yet assign a prediction to, and perform the same procedure over again.

Attended College	Under Thirty	Borough	Income	Customer	Customer Predictions
?	Yes	Manhattan	< 55	0	?
?	No	Brooklyn	< 55	1	?
?	No	Manhattan	< 55	1	?

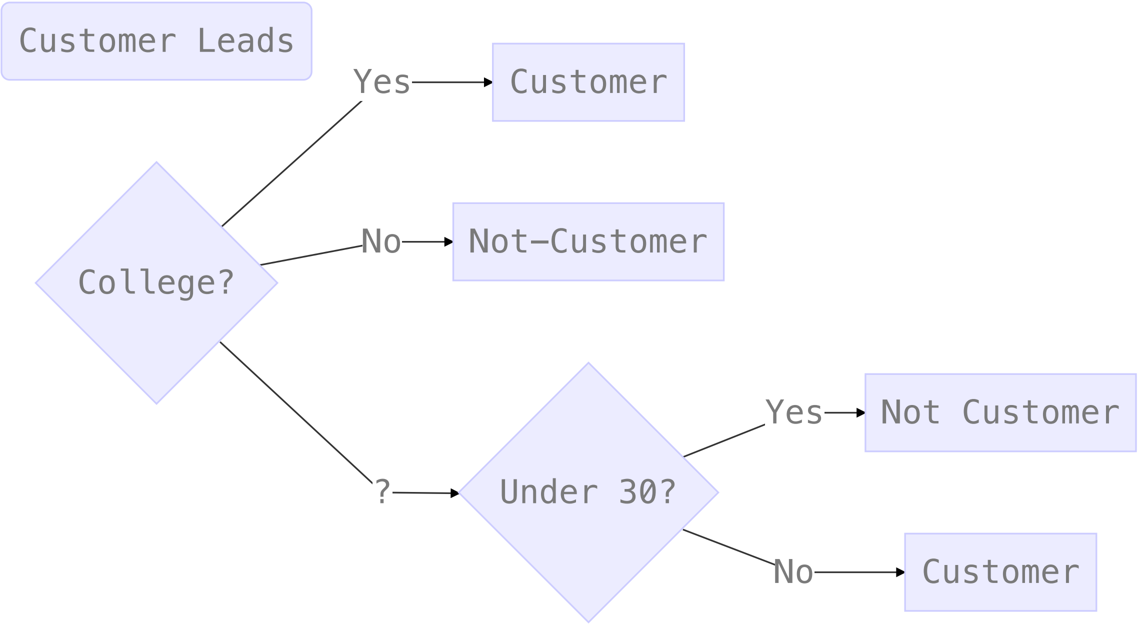
Remember our procedure.

```
Given a subset of data
For each feature in our dataset
  ○ Split the data according to the feature
  ○ Compute consistency of data in each split
Choose the feature with the highest consistency to split the data
Repeat for remaining subset
```

So once again, we split our data and assign a score to each of our tests for the remaining training data.


Under 30?
Yes
0
No
1,1
Borough
Manhattan
0,1
Brooklyn
1
Income
under 55
1,0,1

This time, we see that our `under 30` test does the best with a score of 3. In fact, it separates all of our remaining data into homogeneous groups. We add this to our decision tree, so that our decision tree becomes the following.



So this is our hypothesis function for a decision tree. And we can use it to predict whether or not a future lead will become a customer. So if we see the below data about a lead, we just pass it through our trained decision tree to see what it will predict.

Attended College	Under Thirty	Borough	Income
?	No	Manhattan	< 55

Our decision tree tells us to first look at college, and because college has a value of ?, we then move to under thirty. Because the lead is not under thirty, we predict the lead will become a customer.

Working with larger datasets

In the above example, we were able to use our features to perfectly segment our leads into customers and non-customers. However, when we have larger datasets, we may not be able to accomplish this. So this causes a couple of issues:

1. How can we count only the data in homogenous groups if no group is homogenous

Because features may not split our data into homogenous groups, we use a different metric. Essentially, we penalize each feature for the amount of "disorder" in the group. So if the split results in ten customers and two non-customers, this split receives a lower penalty than a split that results in half customers and half non-customers. So the principle stays the same, we choose splits that best separates our data, we just don't require perfect separation.

2. How do we make predictions with data that is not perfectly homogenous

We may find that our leaf nodes also have a mix of target data. So there may be some customers and some non-customers when training our decision tree. How do we assign a label? Well we just use majority rule. If most of the training data that falls into the leaf node ends up being a customer, then that is the value we assign.

Summary

In this lesson, we learned about training a decision trees. We saw that we can train our decision tree by creating various tests, and then prioritizing tests that best separate our data into homogenous groups. Then we saw that the end result is a hypothesis function as a decision tree which we can use to predict our data.

We summarized our procedure for training a tree as the following.

```
Given a subset of data
For each feature in our dataset
  ○ Split the data according to the feature
  ○ Compute consistency of data in each split
Choose the feature with the highest consistency to
split the data
Repeat for remaining subset
```