Introduction to Machine Learning

What is Machine Learning?

There are many possible definitions.

One definition: **Machine learning** is when a machine (that is, a computer) **improves** on a **task** with respect to some **performance measure**.

Examples:

- Self-driving cars
- Facial recognition
- Targeted Ads

How does Machine Learning work?

Illustrative Example

Suppose a car company wants to decide which cars to buy ad time for in the next Super Bowl. They have a dataset of cars promoted in past Super Bowl ads, and information about the **miles per gallon** (`mpg`) and **horsepower** (`hp`) of each car. In addition, they know which ad campaigns were deemed **"Successful"** or **"Unsuccessful."** Which cars should the company buy ad time for?

Illustrative Example

Suppose a car company wants to decide which cars to buy ad time for in the next Super Bowl. They have a dataset of cars promoted in past Super Bowl ads, and information about the **miles per gallon** (`mpg`) and **horsepower** (`hp`) of each car. In addition, they know which ad campaigns were deemed **"Successful"** or **"Unsuccessful."** Which cars should the company buy ad time for?

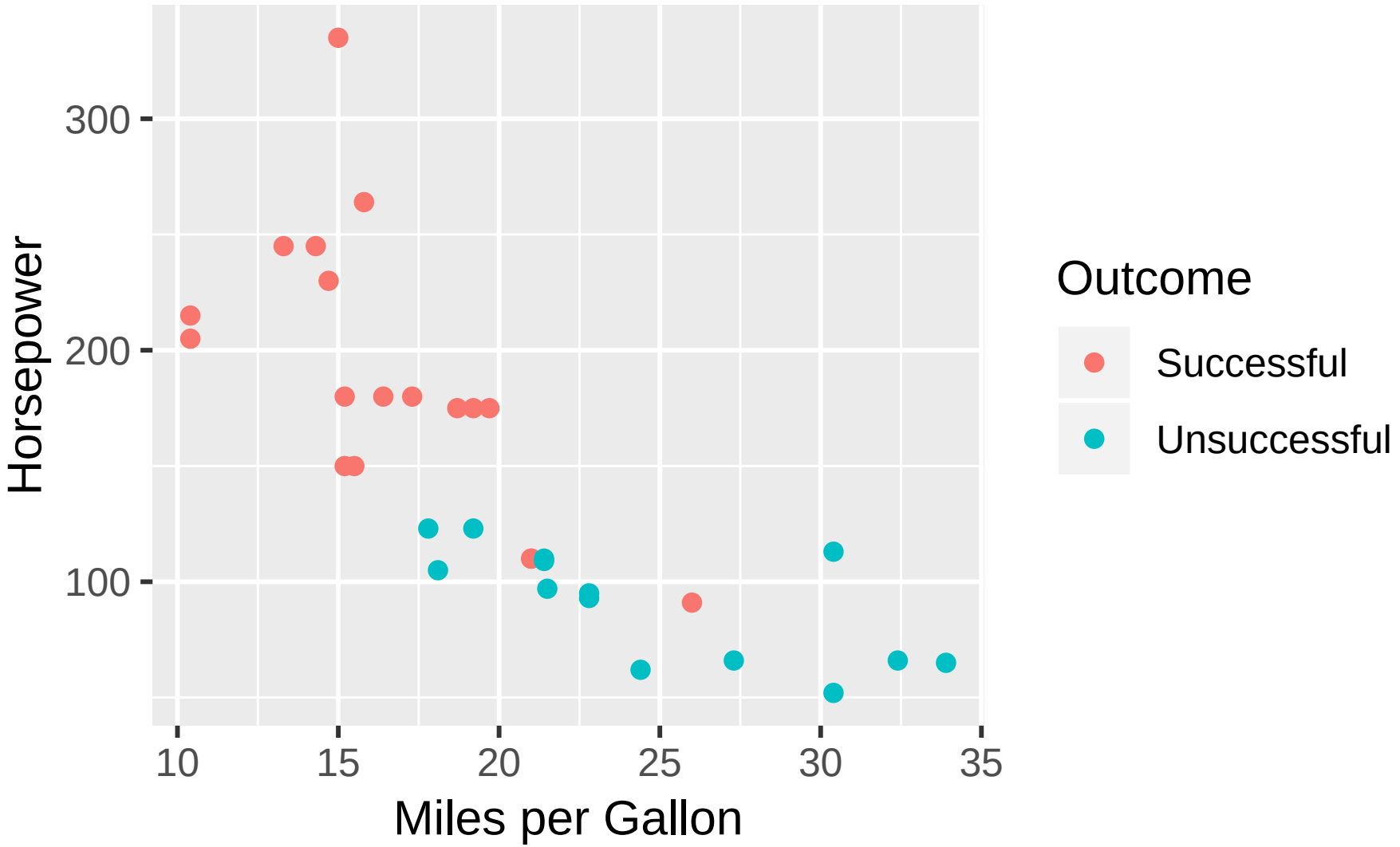
At its heart, this is a **prediction** problem. In order to **decide which cars to buy ad time for**, we want to use **data from previous ad campaigns** to make a **prediction** on which cars will have successful ad campaigns.

Machine Learning in Context

Machine learning is when a machine (that is, a computer) **improves** on a **task** with respect to some **performance measure**.

In the case of **supervised learning** for Super Bowl car advertisements, we want to develop a way to **improve** how well we **predict future successful car advertisements**.

Visualizing the Data



Prediction

To do this, we'll use **machine learning**. More specifically, this is a subset of machine learning called **supervised learning**.

Two branches of machine learning:

- **Unsupervised Learning** is when you have no **outcome variable** and are trying to create groupings based on the data.
- **Supervised Learning** is when you have an **outcome variable**, and you try to make a prediction about future data points.

General Supervised Learning Framework

In supervised learning, we are focused on finding the relationship between a **label** y and **features** x.

$$y = f(x)$$

"Learning" is finding a function f that minimizes **future error** in recovering y.

In our case, we will use **decision trees** to find that function f to minimize future error when trying to determine **success of car advertisement campaigns** (y) based on **MPG and HP** (x)

Linear Regression as Machine Learning?

Think about finding a **least squares regression line** to show the relationship between **two numerical variables** and **predict** values of the **outcome variable**.

$$y = f(x)$$

The regression line fits this definition of **machine learning**. The "f(x)" in this case takes the form of a linear regression line.

The Model Building Process

- **Explore** the data. (Descriptive statistics, visualizations)
- **Prepare** the data. (Data cleaning, generate features and labels)
- **Split** the data into training and test sets. (cross-validation, holdout samples)
- **Train** the model on the training set. (Logistic Regression, Decision Tree, K-Nearest Neighbors algorithm)
- **Predict** on the test set and **evaluate** based on certain metrics. (precision, recall)
- **Repeat and Conclusions**

Machine Learning Applications

Machine Learning Applications

Predictive Modeling for Public Health: Preventing Childhood Lead Poisoning. *Potash et al. 2015*

- Lead Poisoning is a big issue for children living in old homes with lead paint.
- Blood tests can identify elevated lead levels ... but by then, it's too late.
- Use Machine Learning to predict risk of a child being poisoned to provide intervention.

Machine Learning Applications

Automated Injury Coding. *Measure 2014*

- Written descriptions of injuries in the workplace.
- Goal: Code injuries from narratives.
- About 85% of primary codes are assigned using ML.

Machine Learning Applications

A Longitudinal Framework for Predicting Nonresponse in Panel Surveys. *Kern et al. 2019*

- Goal: Predict nonresponse in panel surveys to proactively provide incentives.
- Used data from previous responses to predict nonresponse.

Other Machine Learning Applications

Machine Learning can also be used to improve data in other ways.

- Record Linkage (e.g., Christen 2008)
- Imputation (e.g., Batista and Monard 2002)
- Unsupervised Learning for Text Analysis (e.g., Latent Dirichlet Allocation)

Why Machine Learning?

Goal: Build adaptable and scalable systems that are effective and easy to maintain.

Adaptable: Relationships may change over time.

Easy to Maintain: Rule-based systems may become cumbersome and difficult to update.

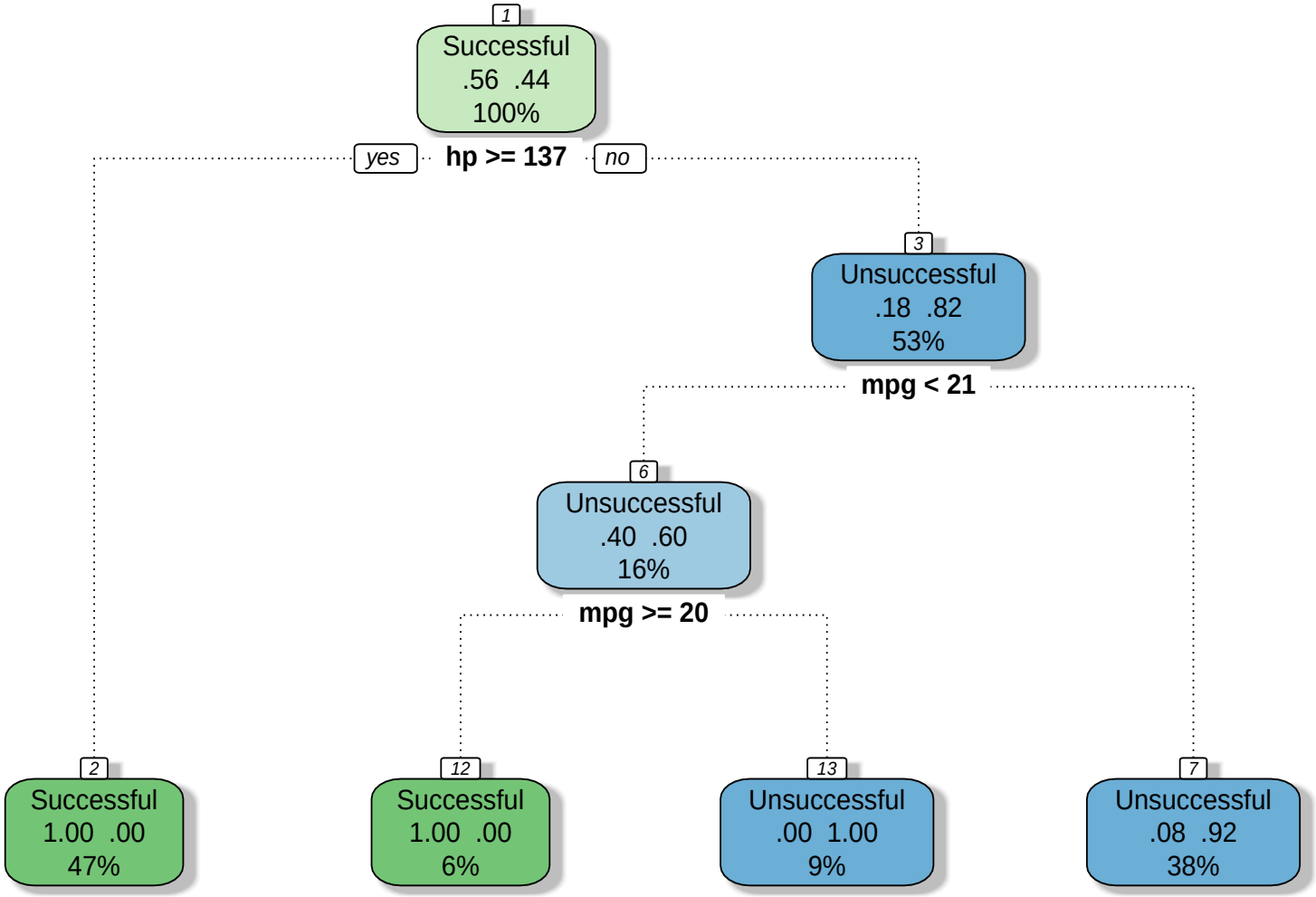
Example Using Decision Trees

How it Works

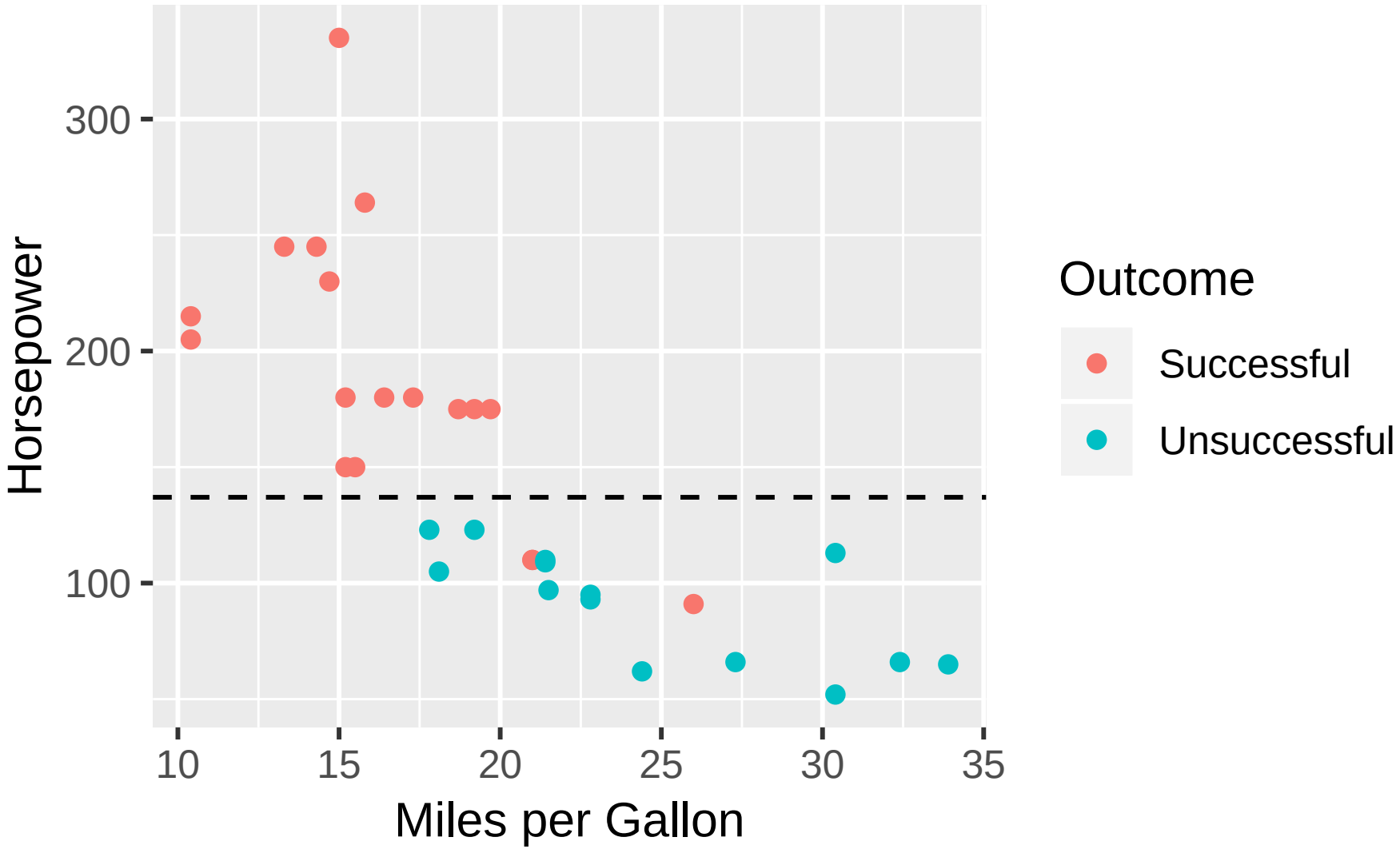
Rough outline of the tree building algorithm

- Consider each feature (in this case, horsepower and MPG).
- Order the values in each feature.
- Compute a measure of how well a partition would split the data for each possible break between values.
- Make the split that maximizes this value.
- Repeat until you reach a stopping criteria.

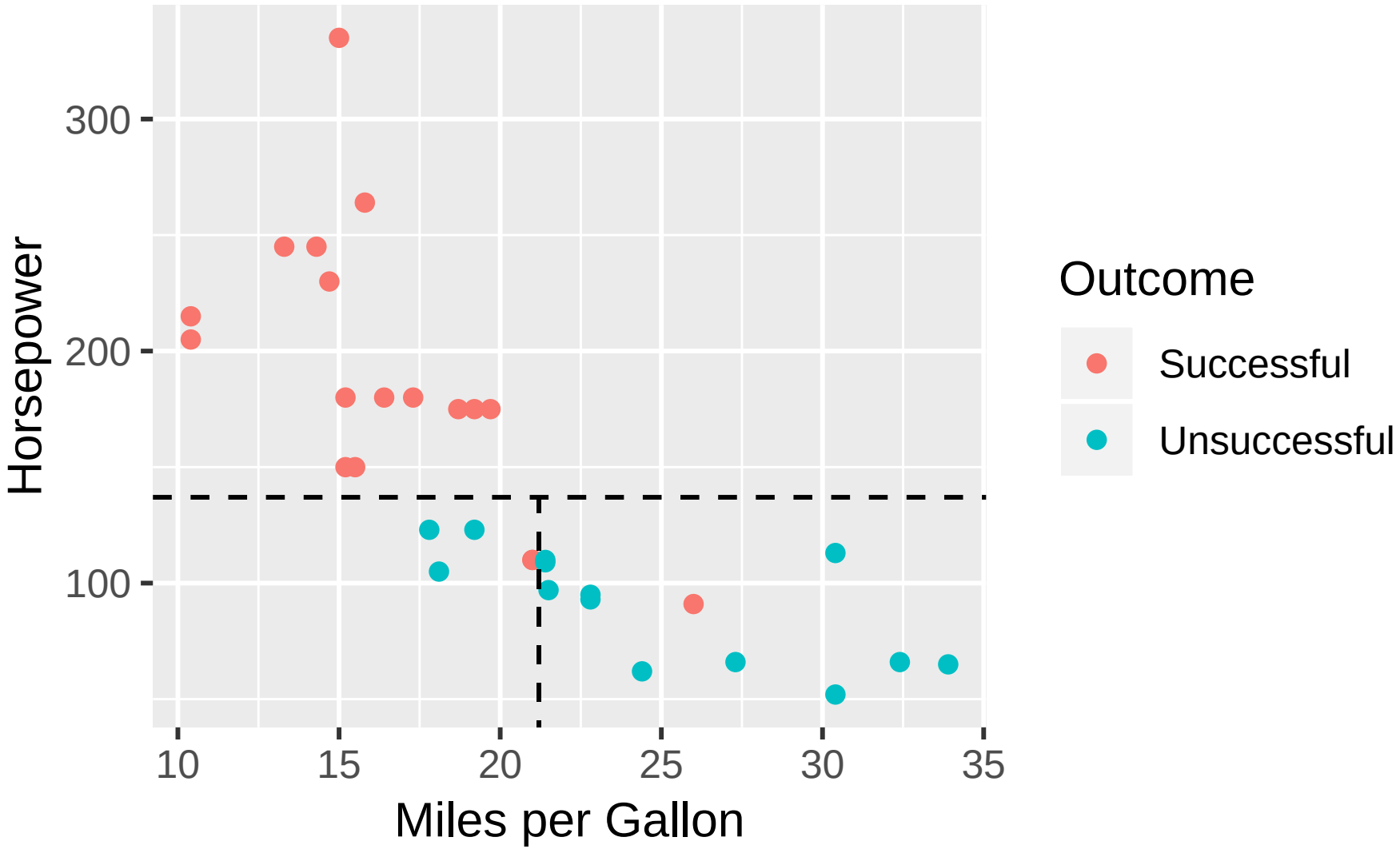
Tree Diagram



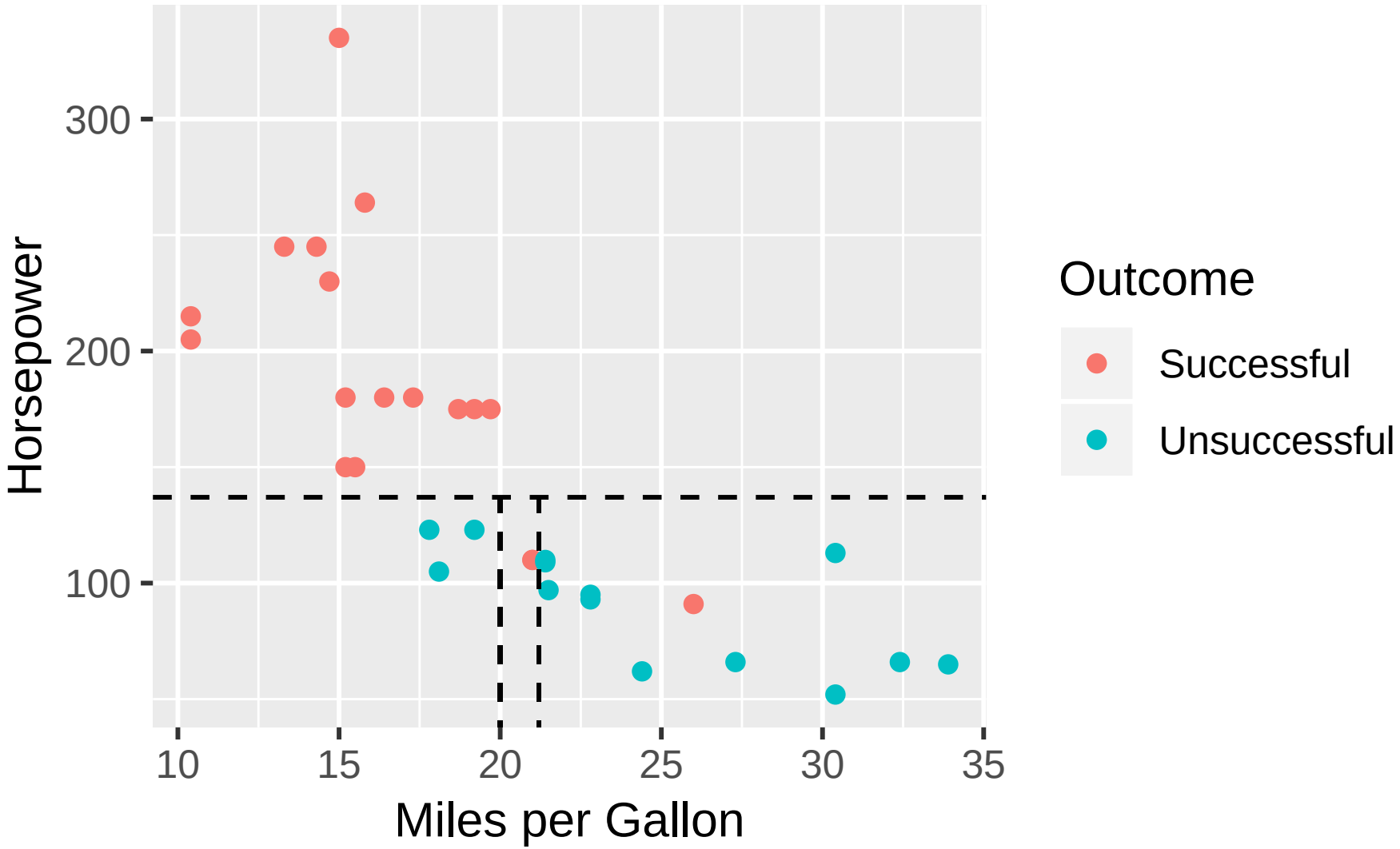
Partitioning



Partitioning



Partitioning



Tuning a Tree

Note that there are multiple places where we need to make decisions on parameters for building a tree.

- Measure for optimal splitting (Gini impurity, information gain).
- Minimum node size for stopping splits.
- Max depth

How should we determine each of these parameters?

By using a **training** set and a **testing** set.

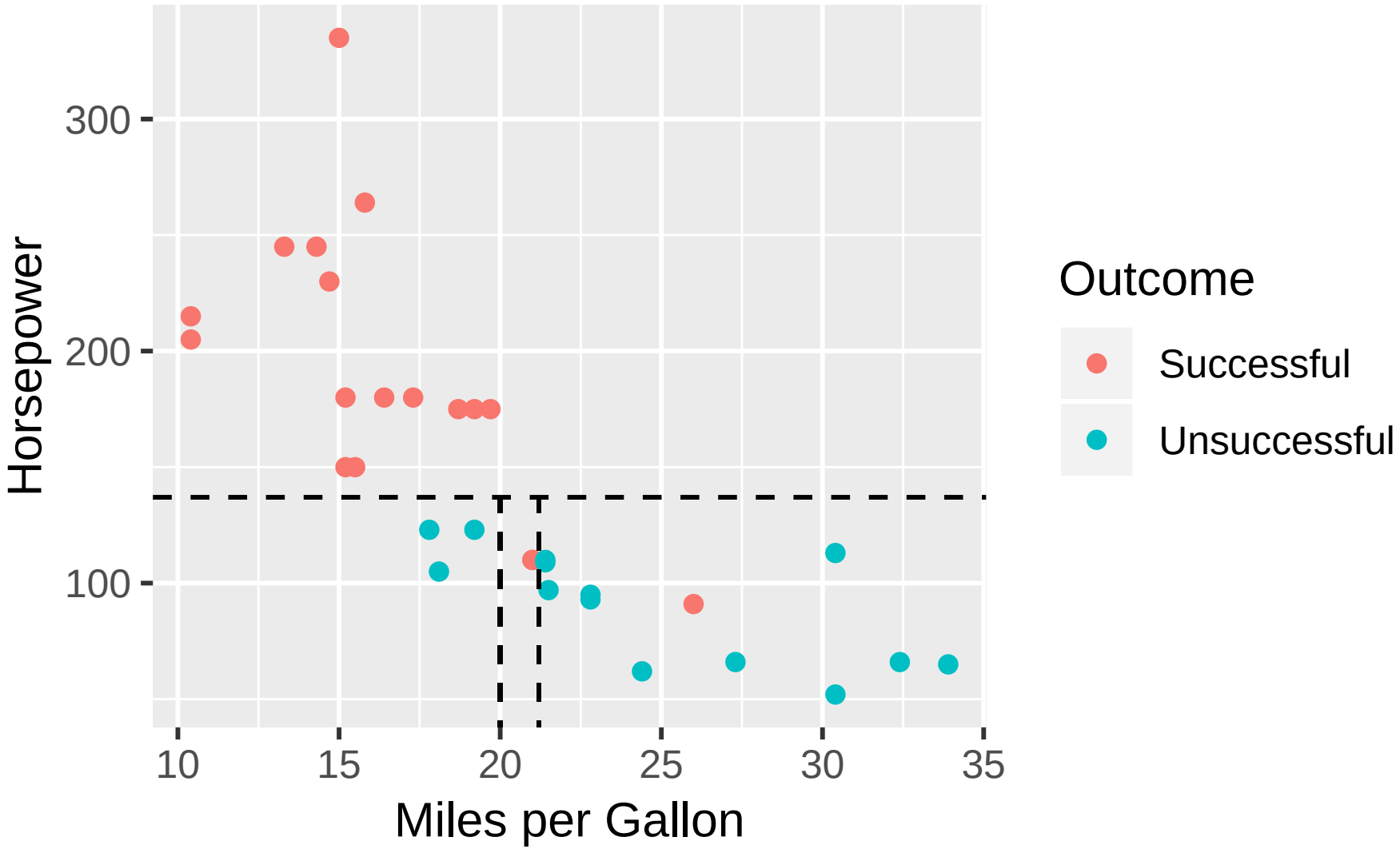
Splitting into Train/Test Sets

Motivation

Suppose we have a machine learning model (e.g. decision trees). How do we know it can **generalize** to the overall population?

That is, we know that the model fits well on the data we used ... precisely because we fit it on that data. How do we know it will predict **future** data points well?

Example


Horsepower
300
200
100
10
15
20
25
30
35
Miles per Gallon
Outcome
Successful
Unsuccessful

Overfitting

The main issue is that our model isn't **generalizable** to the overall population because it is fit to be **too specific to our data**. This is called **overfitting**.

Validation

In order to make sure our results are generalizable, we typical **hold out** a portion of our data to **test** the model on. That is, we split our data into two subsets:

- The **training set** is the subset that we use to build the model on.
- The **test set** is the subset that we use to evaluate how well our model is doing.

Train/Test Split Intuition

Another way to think about it: we want to be sure that our model works well if we apply it to new data.

So, we are pretending we only have a portion of the data, and treating a smaller subset of it as the "new data".

Holdout Sample

Probably the simplest method is to do a **holdout sample**. This consists of splitting the dataset into two groups — the **train set** and the **test set** — randomly. The train set is typically smaller than the train set, since we want to use as much of the data as possible on fitting the model.

Example

Suppose we have a dataset of 10,000 observations about homes in Chicago, and we want to be able to predict which homes will have the highest risk of containing lead poisoning hazards.

How would we approach this?

Example

Suppose we have a dataset of 10,000 observations about homes in Chicago, and we want to be able to predict which homes will have the highest risk of containing lead poisoning hazards.

How would we approach this?

Our population is Chicago homes. We want to predict homes in Chicago that have the highest risk of lead poisoning. Since we want to be able to predict this *in advance*, we build a model that can generalize to all homes in Chicago. We take a sample of 8,000 homes from our dataset, build the models on those 8,000 homes, then test to evaluate how well our models perform on the remaining 2,000

Selecting the Train/Test Split

The holdout sample gives us a random split of the data, so that a portion of it is reserved for testing.

But what if the test set that we choose just happens to be unusual in some way due to random chance?

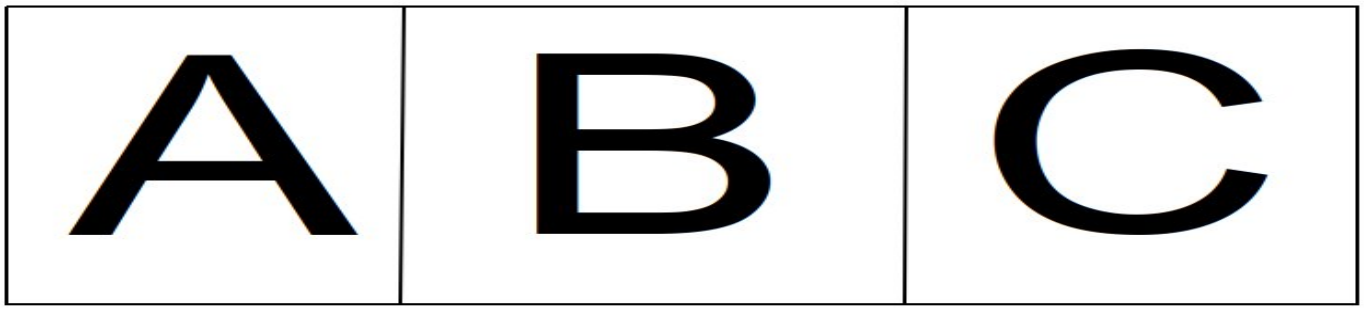
Cross-Validation

A popular method is **k-fold cross-validation**. To do k-fold cross validation, you

1. **Split the data** into k equal partitions.
2. Designate one partition as the **test set** and the rest as the **training test**.
3. **Fit the model** using the training set and evaluate on the test set.
4. **Choose another** of the remaining partitions that you haven't set aside as a test set yet.
5. **Repeat** until every single partition has been a test set once.

Example: Three-fold Cross-Validation

To do a **three-fold cross-validation**, you first split the data into three equal parts. Let's call them **A, B, and C**.



First, you fit the model on **B** and **C**, then test on **A**.

Then, you fit the model on **A** and **C**, then test on **B**.

Finally, you fit the model on **A** and **B**, then test on **C**.

Other Methods

There are many possible ways to validate your results.

- Leave one out cross validation
- Random subsampling
- Temporal holdouts

Machine Learning Evaluation

Motivation

Suppose we have a machine learning model (e.g. decision trees) that we fit on a **training set**, and we need to now evaluate it on the **test set**. How should we do this?

Example

Suppose the following cars were in the test set.

Horsepower	Miles Per Gallon	Outcome
200	15	Success
100	15	Failure
150	25	Success
122	31	Failure
110	12	Failure
304	11	Failure
283	15	Failure

Example

Using the training set, we built a model, and produced the following predictions

Horsepower	Miles Per Gallon	Outcome	Prediction
200	15	Success	Success
100	15	Failure	Failure
123	25	Success	Failure
122	31	Failure	Failure
110	12	Failure	Failure
304	11	Failure	Success
283	15	Failure	Success

Confusion Matrix

We can express the **predicted** and **actual** outcomes in a two-by-two table instead.

	Success	Failure
Predicted Success	1	2
Predicted Failure	1	3

Accuracy

Accuracy is given

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{All Predictions}}$$

Accuracy

Accuracy is given

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{All Predictions}}$$

Example:

	Success	Failure
Predicted Success	1	2
Predicted Failure	1	3

Accuracy

Accuracy is given

$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{All Predictions}}$$

Example:

	Success	Failure
Predicted Success	1	2
Predicted Failure	1	3

$$\text{Accuracy} = \frac{1+3}{1+2+1+3} = \frac{4}{7}$$

Accuracy Can Be Misleading!

Generally, **accuracy** is actually **NOT** a good measure.

Accuracy Can Be Misleading!

Generally, **accuracy** is actually **NOT** a good measure.

Example: Suppose you want to detect credit card fraud to decide which accounts to shut down. It is estimated that 99.9% of accounts are not fraudulent.

Accuracy Can Be Misleading!

Generally, **accuracy** is actually **NOT** a good measure.

Example: Suppose you want to detect credit card fraud to decide which accounts to shut down. It is estimated that 99.9% of accounts are not fraudulent.

Predicting every single account to be not fraudulent gives you a 99.9% accuracy ... but this isn't helpful at all.

Precision

Precision is given by

$$\text{Precision} = \frac{\text{True Positive}}{\text{All Positive Predictions}}.$$

In words, **precision** is how correct your positive predictions are.

Precision

Precision is given by

$$\text{Precision} = \frac{\text{True Positive}}{\text{All Positive Predictions}}.$$

In words, **precision** is how correct your positive predictions are.

Example:

	Success	Failure
Predicted Success	1	2
Predicted Failure	1	3

Precision

Precision is given by

$$\text{Precision} = \frac{\text{True Positive}}{\text{All Positive Predictions}}.$$

In words, **precision** is how correct your positive predictions are.

Example:

	Success	Failure
Predicted Success	1	2
Predicted Failure	1	3

$$\text{Precision} = \frac{1}{1+2} = 0.333$$

Recall

Recall is given by

$$\text{Recall} = \frac{\text{True Positive}}{\text{All Actual Positives}}.$$

In words, **recall** is how many **positives** you were able to recover.

Recall

Recall is given by

$$\text{Recall} = \frac{\text{True Positive}}{\text{All Actual Positives}}.$$

In words, **recall** is how many **positives** you were able to recover.

Example:

	Success	Failure
Predicted Success	1	1
Predicted Failure	1	3

Recall

Recall is given by

$$\text{Recall} = \frac{\text{True Positive}}{\text{All Actual Positives}}.$$

In words, **recall** is how many **positives** you were able to recover.

Example:

	Success	Failure
Predicted Success	1	1
Predicted Failure	1	3

$$\text{Recall} = \frac{1}{1+1} = 0.5$$

What's Good?

In general, there is no standard for what constitutes a "good" **precision** or **recall**. It all depends on your frame of reference.

Example: Suppose you want to detect credit card fraud to decide which accounts to shut down. It is estimated that 99.9% of accounts are not fraudulent.

What's Good?

In general, there is no standard for what constitutes a "good" **precision** or **recall**. It all depends on your frame of reference.

Example: Suppose you want to detect credit card fraud to decide which accounts to shut down. It is estimated that 99.9% of accounts are not fraudulent.

If we chose fraudulent accounts **at random**, we'd have a **precision** of about 0.1%. So, a **model with 10% precision** would be excellent, because that means the model gives you a **100x lift** over random chance.

What's Good?

Example: Suppose you want to detect credit card fraud to decide which accounts to shut down. It is estimated that 99.9% of accounts are not fraudulent. Would a model with **10% recall** be considered good?

What's Good?

Example: Suppose you want to detect credit card fraud to decide which accounts to shut down. It is estimated that 99.9% of accounts are not fraudulent. Would a model with **10% recall** be considered good?

If we chose 1% of all accounts **randomly** to flag as fradulent, then our **recall** would be 1%. If our model flags 1% of all counts as fraudulent and our recall is 10%, it is giving us a **10x lift** over random chance.

What to Use?

Which metric matters more depends on what you care about.

Recall: Suppose a car company wants to decide which cars to buy ad time for in the next Super Bowl. They have a dataset of cars promoted in past Super Bowl ads, and information about the **miles per gallon** (`mpg`) and **horsepower** (`hp`) of each car. In addition, they know which ad campaigns were deemed **"Successful"** or **"Unsuccessful."** Which cars should the company buy ad time for?

What to Use?

Which metric matters more depends on what you care about.

Recall: Suppose a car company wants to decide which cars to buy ad time for in the next Super Bowl. They have a dataset of cars promoted in past Super Bowl ads, and information about the **miles per gallon** (`mpg`) and **horsepower** (`hp`) of each car. In addition, they know which ad campaigns were deemed **"Successful"** or **"Unsuccessful."** Which cars should the company buy ad time for?

In this case, we might want to use **precision** because Super Bowl ads are expensive.

Machine Learning Overview

The Model Building Process Revisited

- **Explore** the data. (Descriptive statistics, visualizations)
- **Prepare** the data. (Data cleaning, generate features and labels)
- **Split** the data into training and test sets. (cross-validation, holdout samples)
- **Train** the model on the training set. (Logistic Regression, Decision Tree, K-Nearest Neighbors algorithm)
- **Predict** on the test set and **evaluate** based on certain metrics. (precision, recall)
- **Repeat and Conclusions**

The Model Building Process

- **Explore** the data. (Descriptive statistics, visualizations)
- **Prepare** the data. (Data cleaning, generate features and labels)
- **Split** the data into training and test sets. (cross-validation, holdout samples)
- **Train** the model on the training set. (decision tree algorithm)
- **Predict** on the test set and **evaluate** based on certain metrics. (precision, recall)
- **Repeat**

Machine Learning

This general structure applies to more than just decision trees — you can use it with any machine learning algorithm, or just anything that outputs a prediction.

Other models/algorithms to try are:

- Logistic Regression
- Support Vector Machines
- Random Forests
- Boosting
- ... and more!

K-Nearest Neighbors

Example: Chronic Kidney Disease

Doctors want to know whether their patients are at risk for developing chronic kidney disease (CKD). They have a dataset of many patients with measurements from a blood test, as well as whether the patients developed CKD. We want to develop a model to predict whether a patient will get CKD using just the information from the blood test.

Intuition

Intuition behind K-Nearest Neighbors: When predicting future data, we might want to consider how other similar observations might have performed in the past.

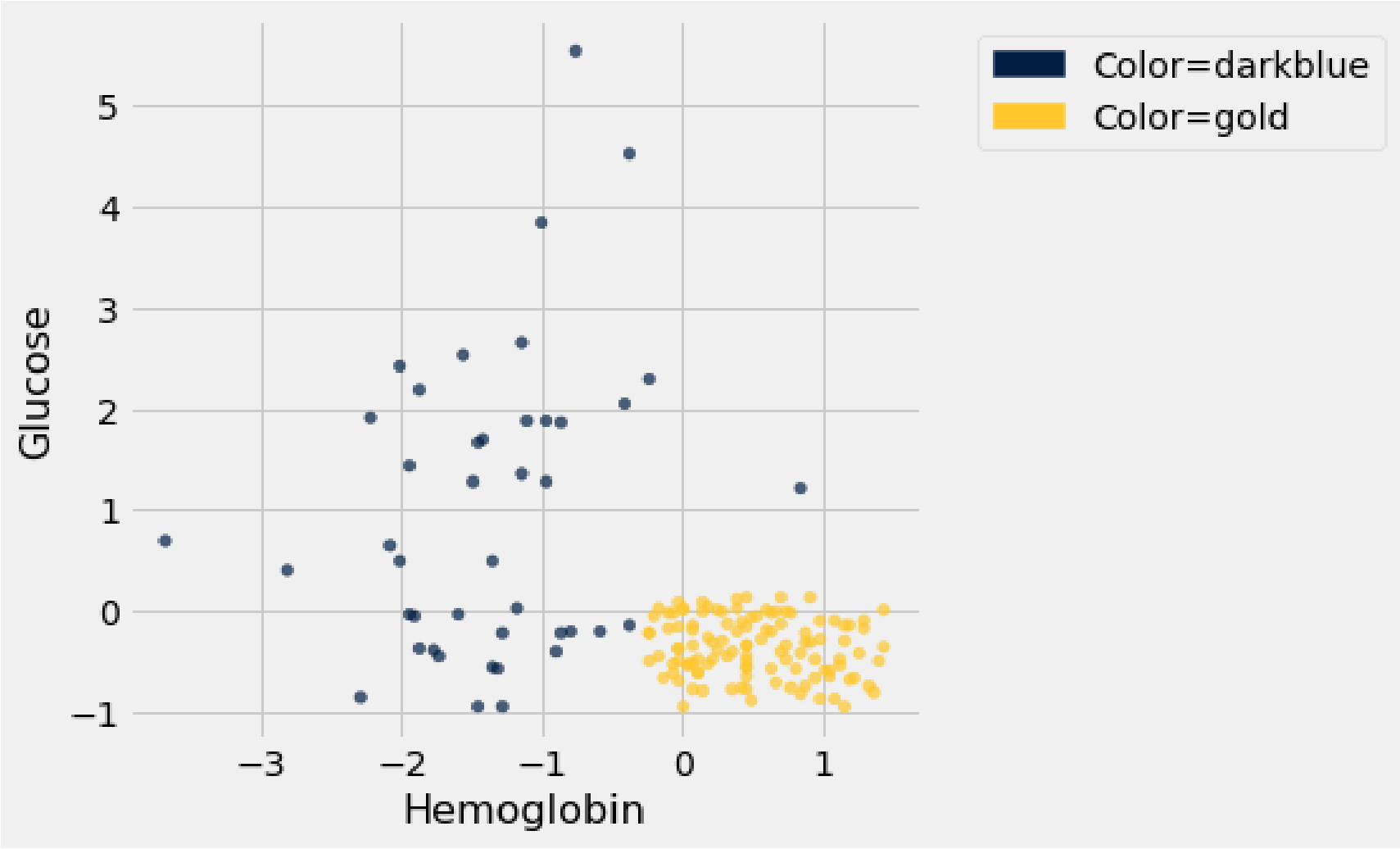
Example: When trying to predict whether a patient will develop chronic kidney disease, we might want to think about whether patients with similar blood test results got CKD.

K-Nearest Neighbors Algorithm

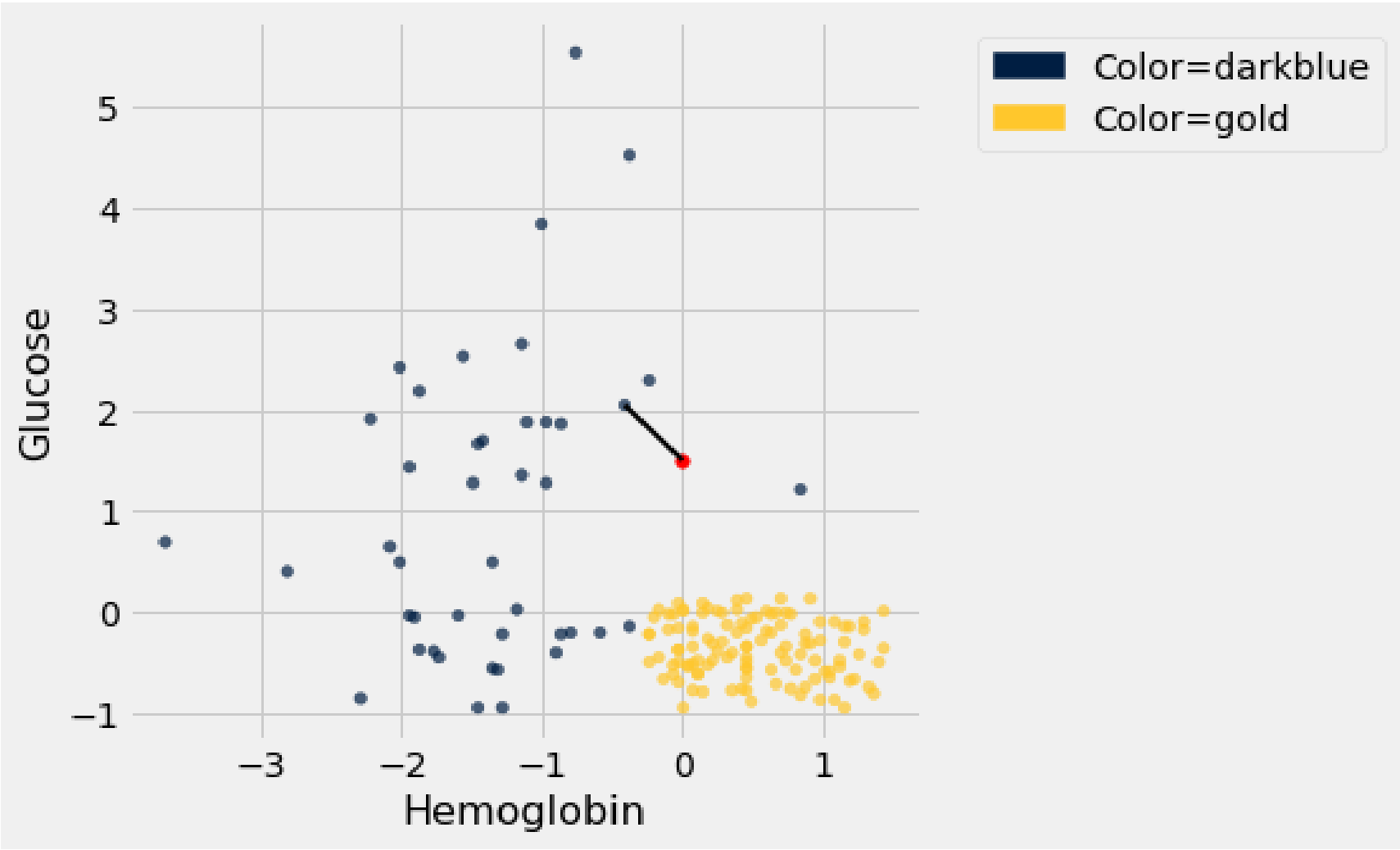
Find the data point that is closest to the one you want to predict.

The predicted class is the class of the nearest data point.

Data


Color=darkblue
Color=gold
Glucose
Hemoglobin
5
4
3
2
1
0
−1
−3
−2
−1
0
1

Prediction using K-Nearest Neighbors

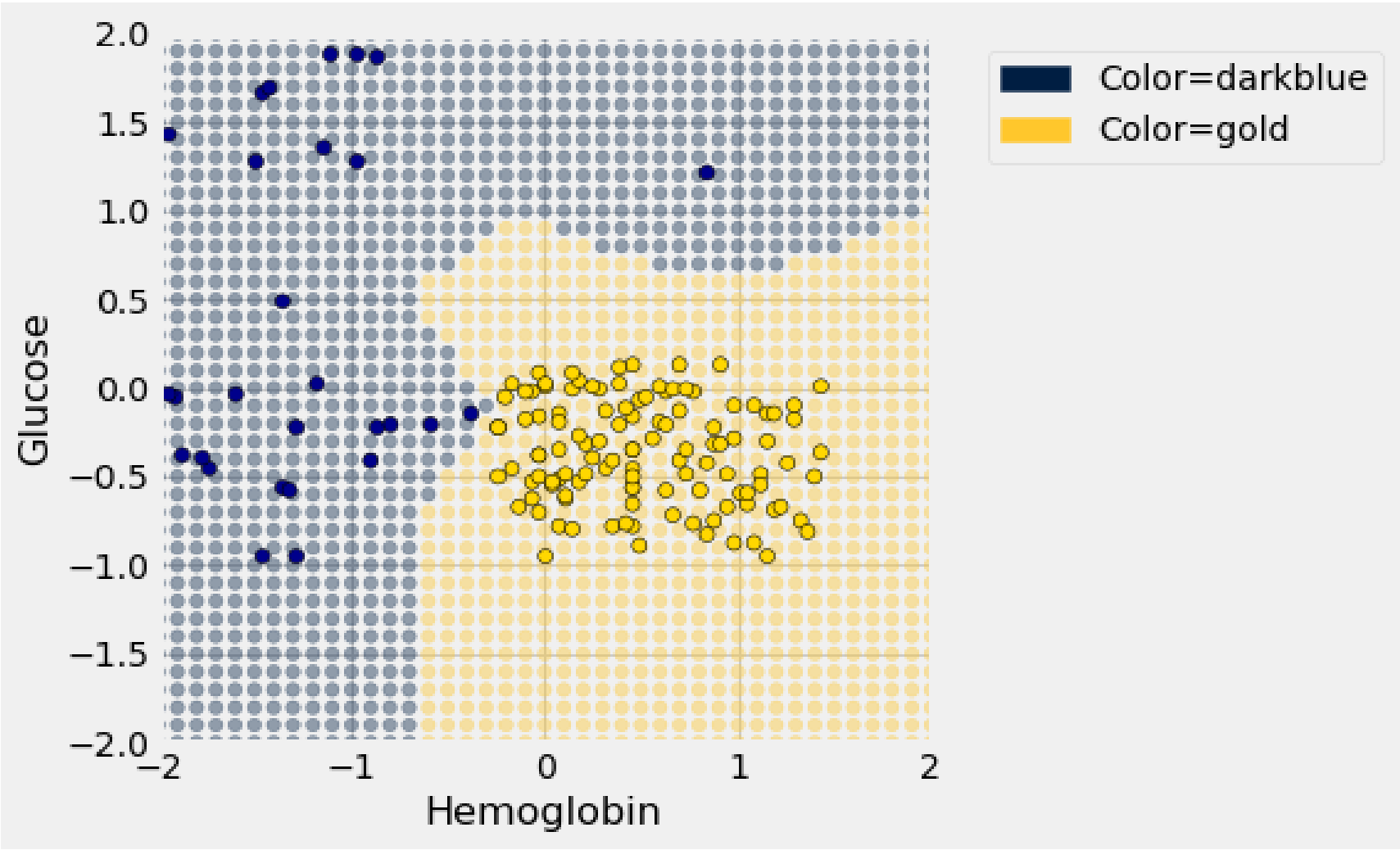


K-Nearest Neighbors

This shows how we might predict any new data point. We might be more interested in the **overall model**. That is, given any input, what would be our predicted output?

We can describe this model using a **decision boundary**.

The K-Nearest Neighbors Model

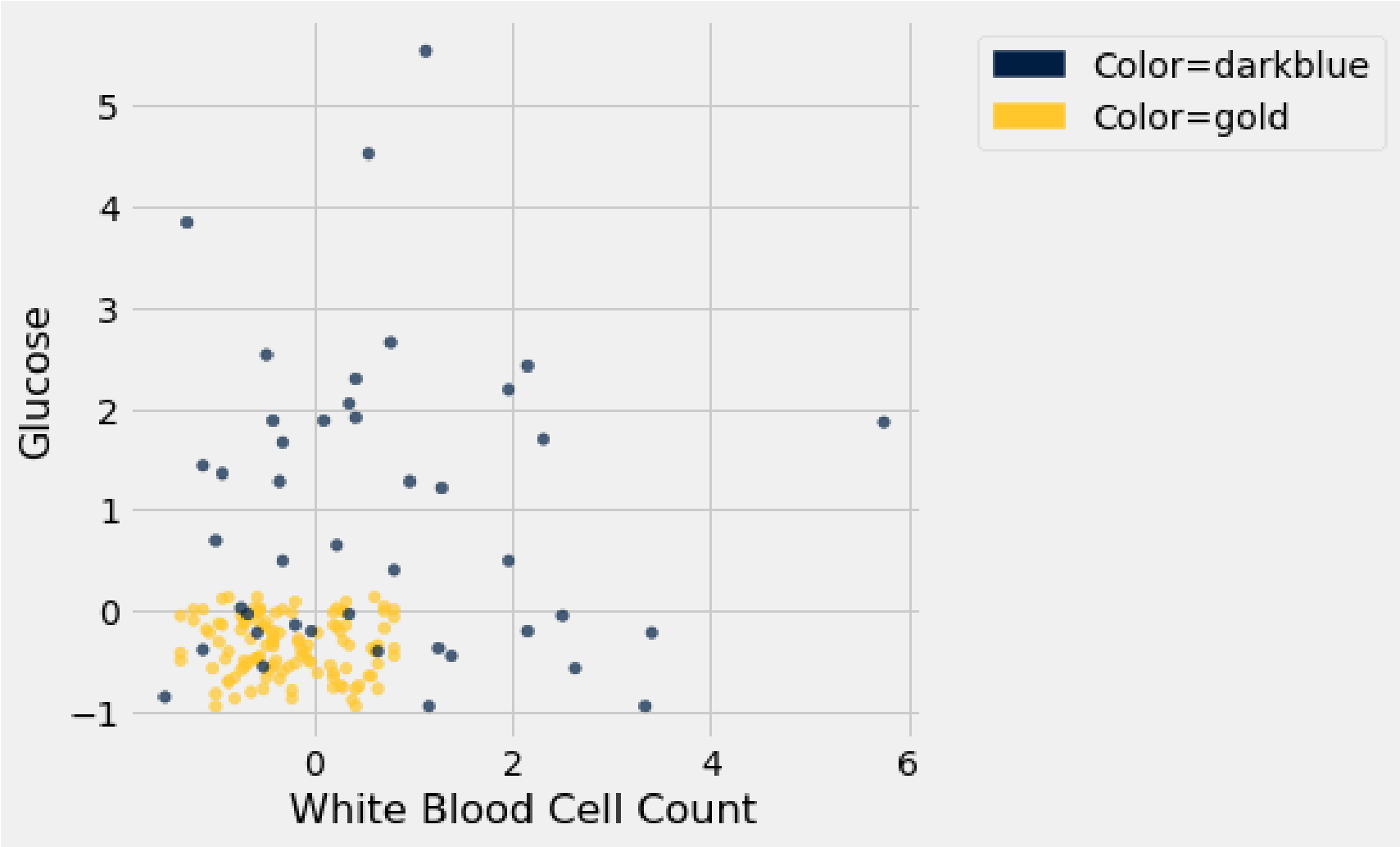


K-Nearest Neighbors

What we've described is actually 1-Nearest Neighbors. Usually, we choose a number of closest points and use those together to determine predicted class.

For example, we might set k = 5 and choose five closest points and predict using a "vote" of those classes.

Another Example



Considerations

Choosing K

- We want to choose K to minimize **overfitting**.
- K is arbitrary! You may want to try multiple values to see which performs best.
- Use an odd number (to avoid ties)

Bias and Fairness in Machine Learning

Bias and Fairness in ML

Remember: Your machine learning models will only be as good as the data you give it!

This is particularly important remember in social science research, where you need be very careful about the possibility of bias and think about issues of fairness when implementing policy/action based on your ML models.

Some Myths

Myth: Since Machine Learning models are designed to be objective and are based on data — which itself is always true — they will always be fair.

Myth: If I don't use race in my analysis, then my analysis can't be racist. (Also applies sex and sexist, etc.)

Myth: If I use race in my analysis, then my analysis is always racist. (Also applies sex and sexist, etc.)

Bias in ML

The **process** might not necessarily be biased, but the data you provide and the approach to the research question **most definitely can be.**

As researchers, we need to be careful about this. "Unfair" models can come up even when we aren't trying to build them in an unfair way.

Example: Chronic Kidney Disease

Doctors want to know whether their patients are at risk for developing chronic kidney disease (CKD). They have a dataset of many patients with measurements from a blood test, as well as whether the patients developed CKD. We want to develop a model to predict whether a patient will get CKD using just the information from the blood test.

Considerations

One extreme example: Suppose we found out that the study had only enrolled white males. We wouldn't be able to generalize to the overall population, and the study results could only be applied to white males.

But even if we make sure to include a mix of people from different backgrounds, we can be in trouble of building a biased model.

CKD Example

Doctors want to know whether their patients are at risk for developing chronic kidney disease (CKD). They enroll patients from a variety of backgrounds randomly chosen to **represent the overall population** and choose the **model with the best precision** (precision = 0.92).

What could go wrong with this scenario?

CKD Example

Doctors want to know whether their patients are at risk for developing chronic kidney disease (CKD). They enroll patients from a variety of backgrounds randomly chosen to **represent the overall population** and choose the **model with the best precision** (precision = 0.92).

What could go wrong with this scenario?

What if precision for **white patients** was 0.99 and precision for **non-white patients** was 0.8?

Interventions

Why is this an issue?

If a **beneficial intervention** is provided to only the **most at-risk people**, then we want to make sure that intervention is **distributed fairly** amongst all people.

Myths

Myth: Since Machine Learning models are designed to be objective and are based on data — which itself is always true — they will always be fair.

A common saying in data science: **"Garbage in, garbage out."**

If your data are biased in some way, then your results will be biased, no matter how "fair" the model-building process is.

Myths

Myth: If I don't use race in my analysis, then my analysis can't be racist. (Also applies sex and sexist, etc.)

Even if you don't use certain variables, such as race, your model might end up performing very differently by race.

Myths

Myth: If I use race in my analysis, then my analysis is always racist. (Also applies sex and sexist, etc.)

You can account for difference in race, and still have a fair model. The key part is that you are fair in how you provide the intervention.

What can be done?

One way to combat it: `http://www.datasciencepublicpolicy.org/projects/aequitas/`

What can be done?

Questions to ask:

- Where is the data coming from?
- How are the algorithms set?
- Do decision-makers understand what the analysis is doing?
- How is the trade-off between false positives and false negatives set?