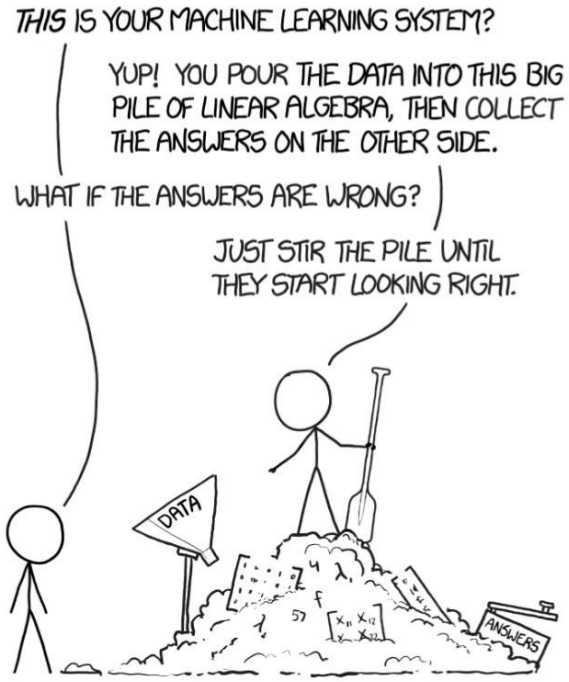
# The Crash Course on Machine Learning

Alex Mariakakis
University of Toronto




With content borrowed from
Jason Mayes (Google, link)

# Purpose

What is covered

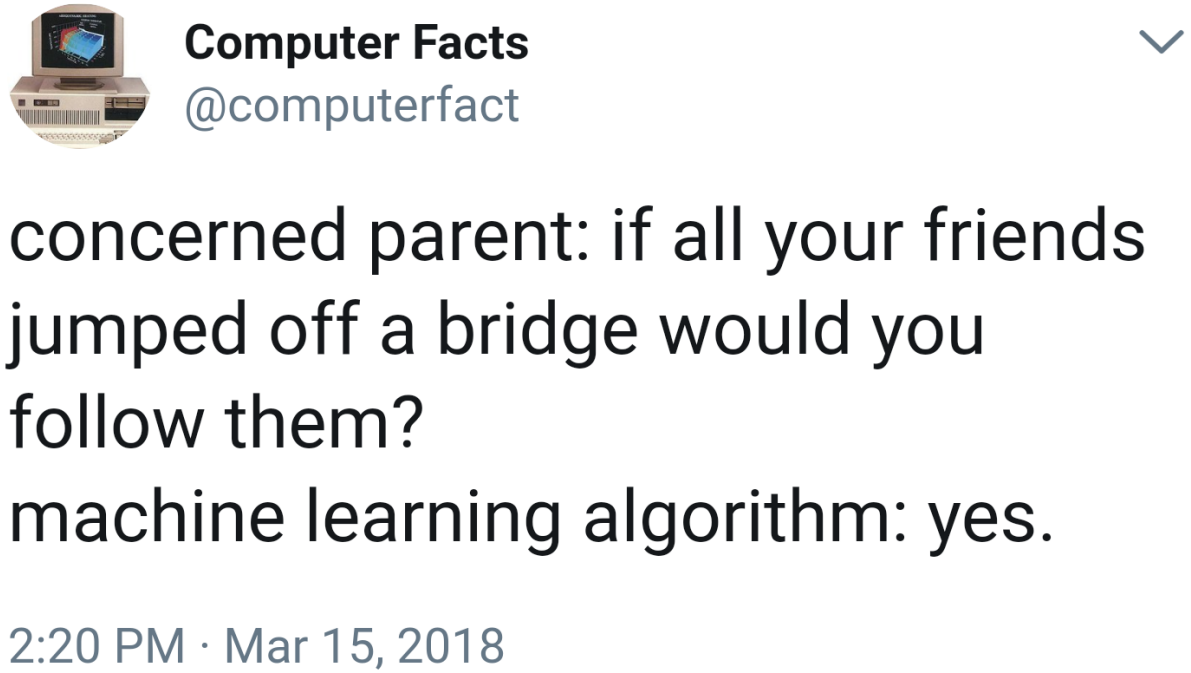
- Terms and definitions (link)
- Thought process for tackling problems

What is not covered

- Underlying math
- Code examples
  - Scikit for Python
  - Weka for Java

# What Is Machine Learning?

Machine learning gives computers the ability to *learn by example* without explicit programming



# Without Explicit Programming

```
if email contains “Viagra”:
    then mark as spam;
if email contains “limited
offer”:
    then mark as spam;
if email contains “job offer”:
    then do not mark as spam;
...
```

Traditional Programming

```
while accuracy low:
    classify some emails;
    check errors;
    change to reduce errors;
```

Machine Learning Programs

# Without Explicit Programming

Machine learning takes some data to train the system

It then learns patterns from the data so that it can predict for future data

The best part is that this “code” (not patterns) can be re-used for many different problems!

# Types of Learning

Supervised Learning: data comes with the expected outputs; you tell the system what the categories are beforehand






These are called “cats”

These are called “dogs”

# Types of Learning

Unsupervised Learning: data does not come with the expected outputs; you let the system learn what the categories are






Put these images into two different groups

# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Feature Representation

There are millions of ways to represent data!

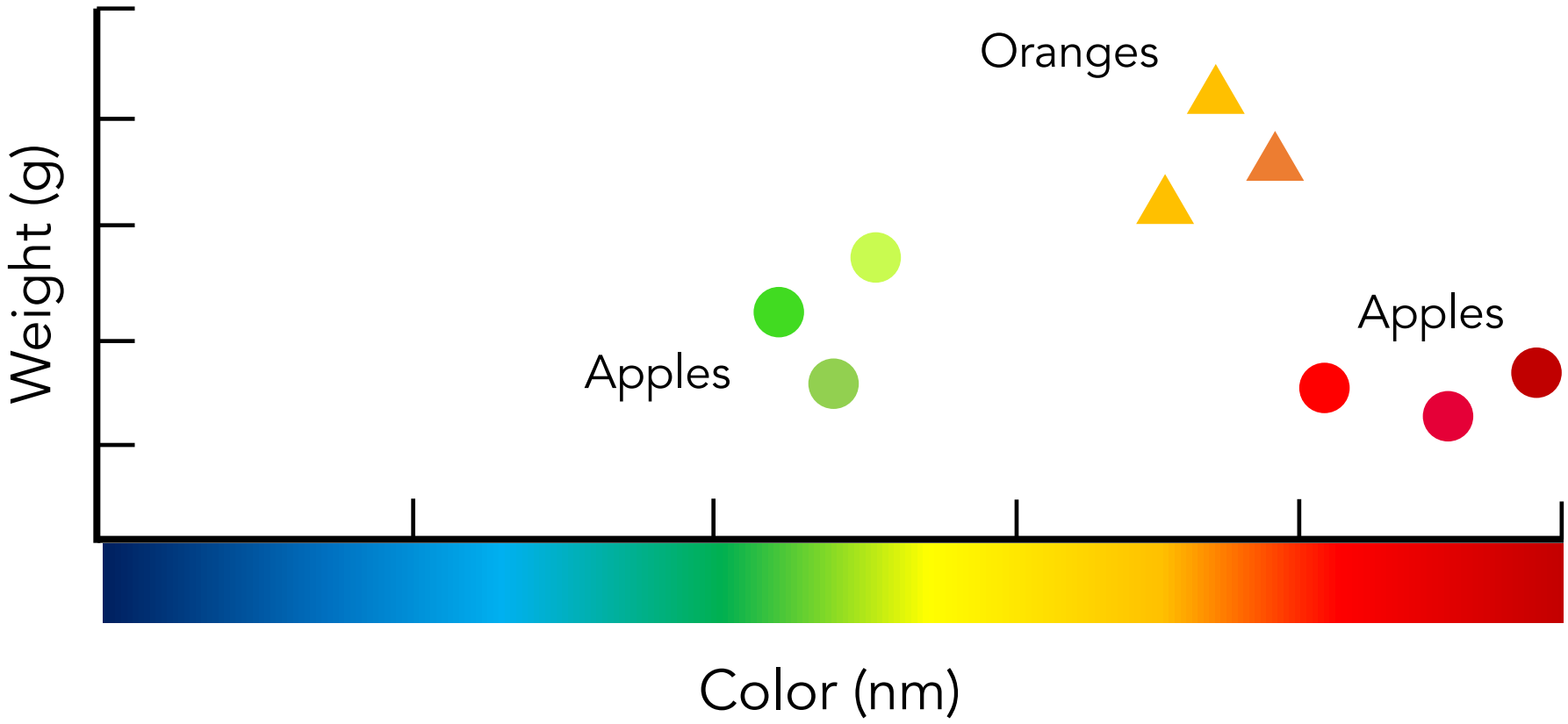
Weight = 220 g
Color = red (or green)

What's the difference between an apple and an orange?

Weight = 340 g
Color = orange

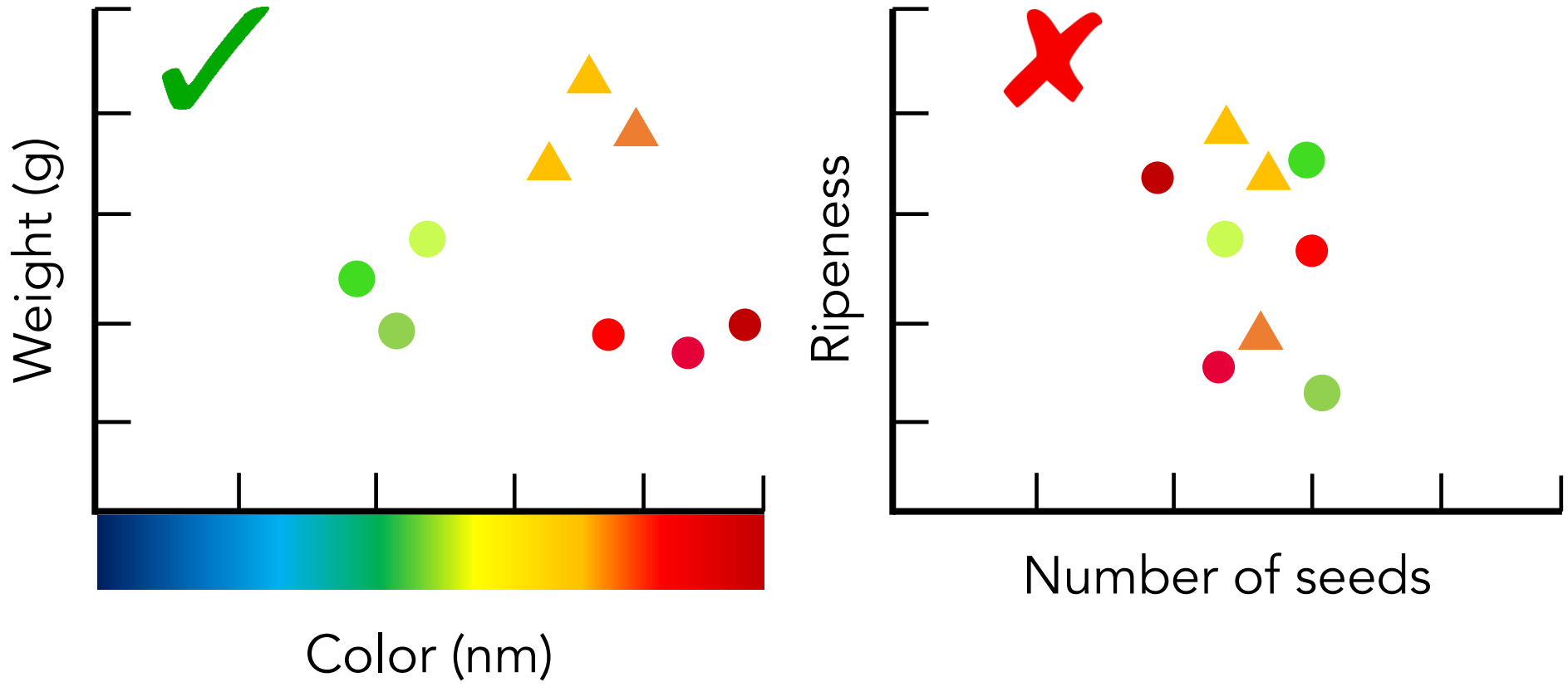
# Examining Features

Each feature can be thought of as a dimension of a new coordinate system for representing the data



# What Makes A Good Feature Set

A feature set is good when they make it easy to distinguish groups in combination with one another



# Other Kinds of Features

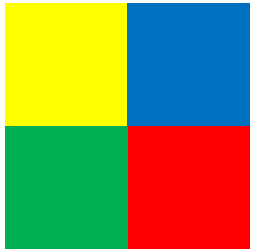
There are millions of ways to represent data!



Color

$\langle 255, 255, 0 \rangle$

RGB values



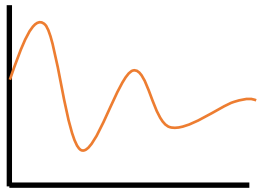
Image

$$\left\langle \begin{matrix} 255, 255, 0 \\ 0, 0, 255 \\ 0, 255, 0 \\ 255, 0, 0 \end{matrix} \right\rangle$$

Group of RGB values

# Other Kinds of Features

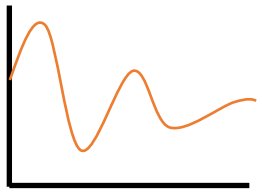
There are millions of ways to represent data!



Time-varying signal
(e.g., audio)

$\langle 10, 12, 8, 5, 3, 5, \ldots \rangle$

Series of a fixed
number of samples



Time-varying signal
(e.g., audio)

Mean = 5
Average slope = -0.1
Kurtosis = 0.1

Summary features that
describe the signal

# Label Representation

## Regression

Maps features to continuous output values

Q: “What is the price of this fruit?”

A: “457 cents”

## Classification

Maps features to categorical output values

Q: ”Is this an apple or an orange?”

A: Orange

# Label Representation

## Regression

Maps features to continuous output values

Labels would be single integers, floats, etc.

## Classification

Maps features to categorical output values

Labels would be categories represented with numbers (e.g., normal = 0, abnormal = 1) or one-hot encoded

# Data Representation

All entries in your dataset should generally have the same number of features and labels

Some possible solutions:

- Placeholder values (0, NaN, etc.) to fill-in blanks
  - Bad idea for models that cannot interpret placeholders differently from “real” data
- Cut all sequences to shortest length in dataset
- Boil down everything to summary features
  - Ask yourself if summary features are comparable to sequences of different length
- Apply dynamic time-warping to standardize length

# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Training: Train-Test Splits

Split your dataset into three groups

1. Train: fed to the model so it can learn
2. Test: never been seen by the model; used for producing an accuracy measurement
3. Validation (optional): used to determine model-specific parameters (e.g., number of decision tree levels) or when training should stop

Consider how data should be grouped

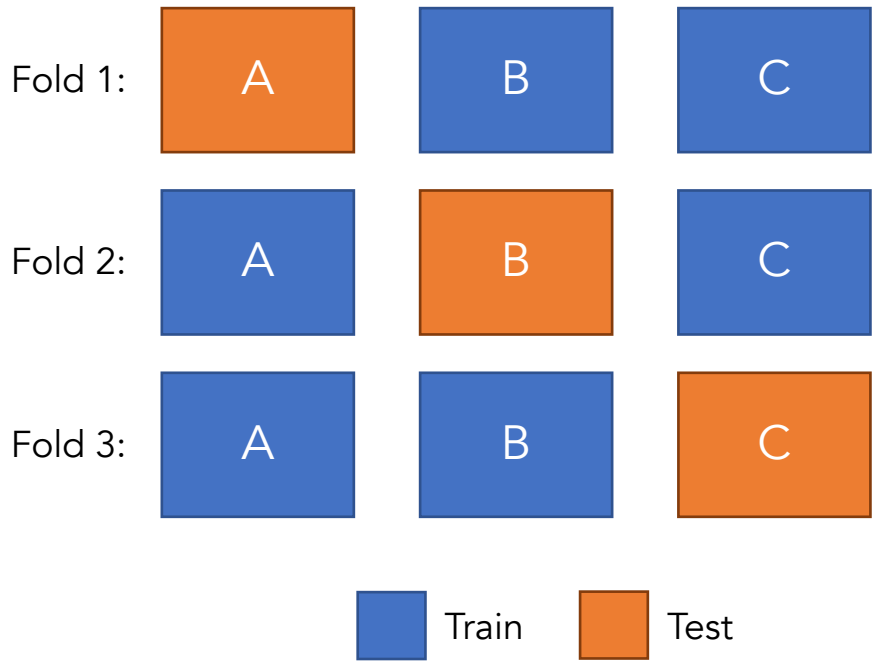
Ex: User 1's data should not be in both train and test sets

# Training: Cross-Validation

Repeated train-test splits so that you can test on all data

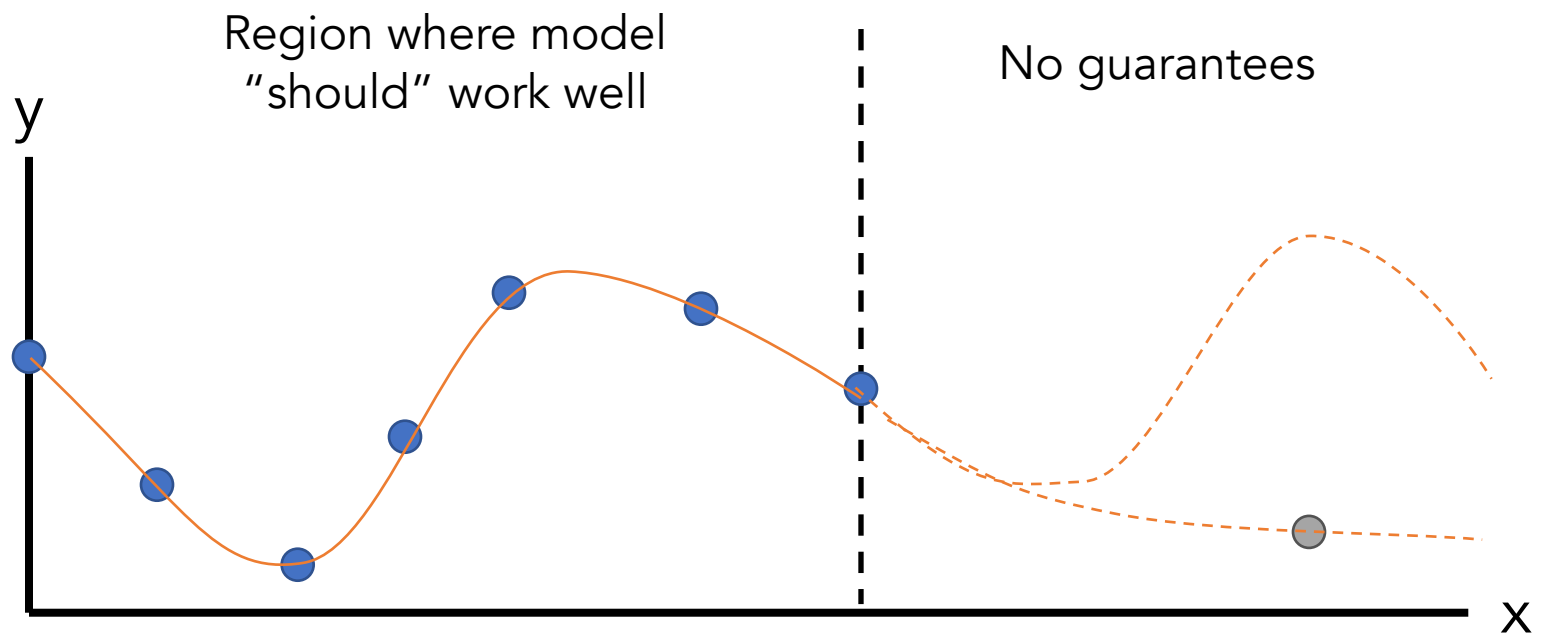
N-fold cross-validation: split dataset into N subsets

Leave-one-out cross-validation: special case of N-fold where N=number of users, samples, etc.



# Training

If your model hasn't been trained on a certain kind of data before, it probably won't predict well on it



# Training

Remember: ML involves learning by example

| Model trained to distinguish… |  |  |  |
|---|---|---|---|
| Car vs. bike vs. plane | car | bike | plane |
| Car vs. bike | car | bike | car |

This issue becomes even more apparent when your dataset is small

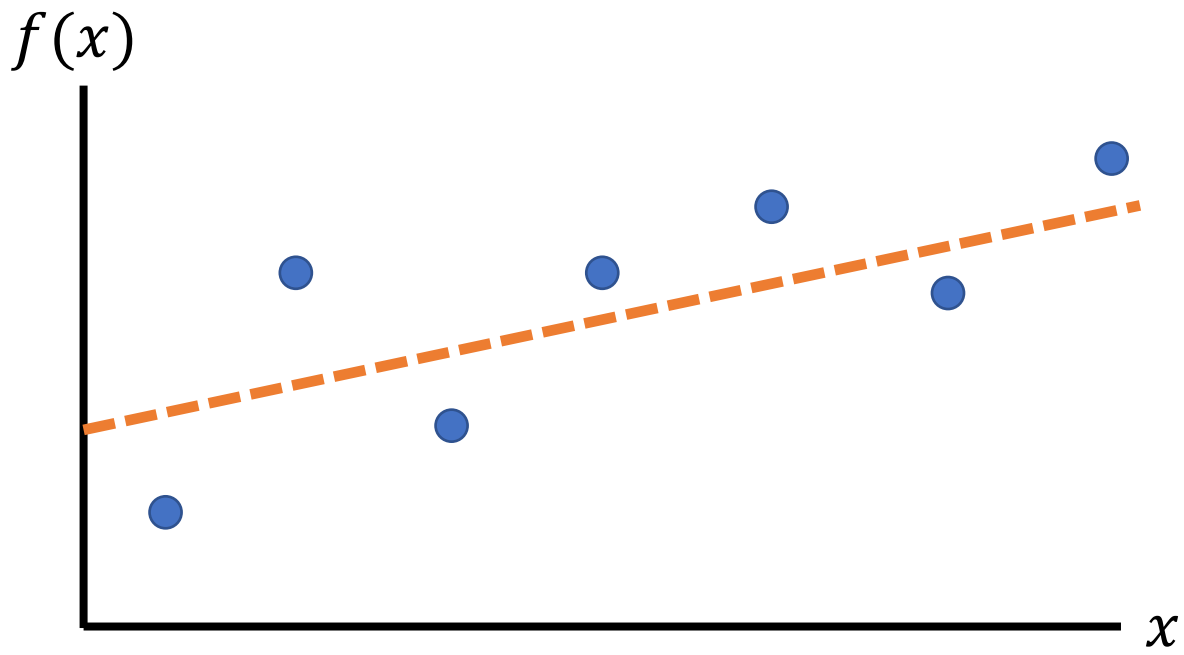
# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Regression: Linear Regression

Maps a linear combination of features to a continuous label $(-\infty, \infty)$

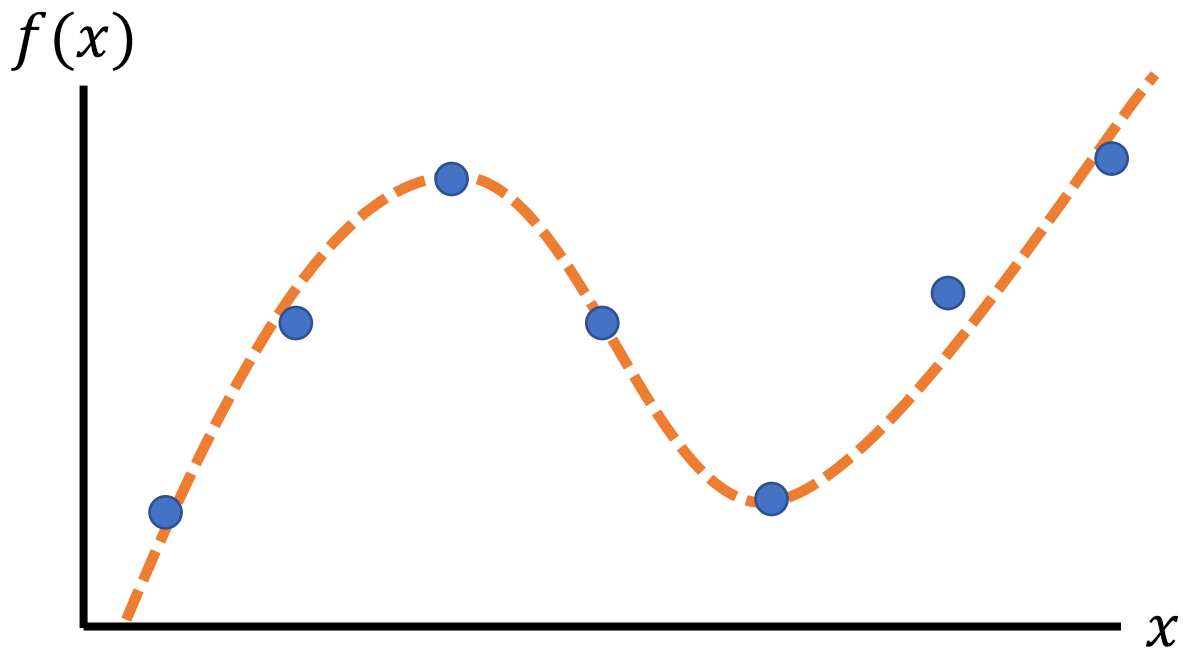
$$y = w_1 x_1 + w_2 x_2 + \cdots + b$$



# Regression: Polynomial Fit

Maps a linear combination of powers of a feature to a continuous label $(-\infty, \infty)$

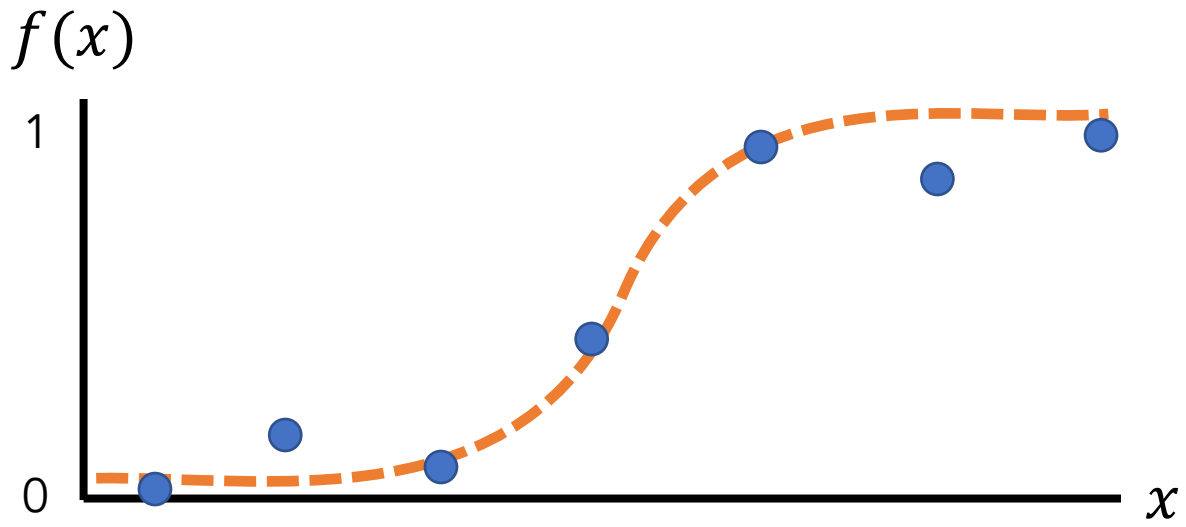
$$y = w_1 x^n + w_2 x^{n-1} + \cdots + b$$



# Regression: Logistic Regression

Maps a logit function on a combination of features to a continuous label $(0, 1)$

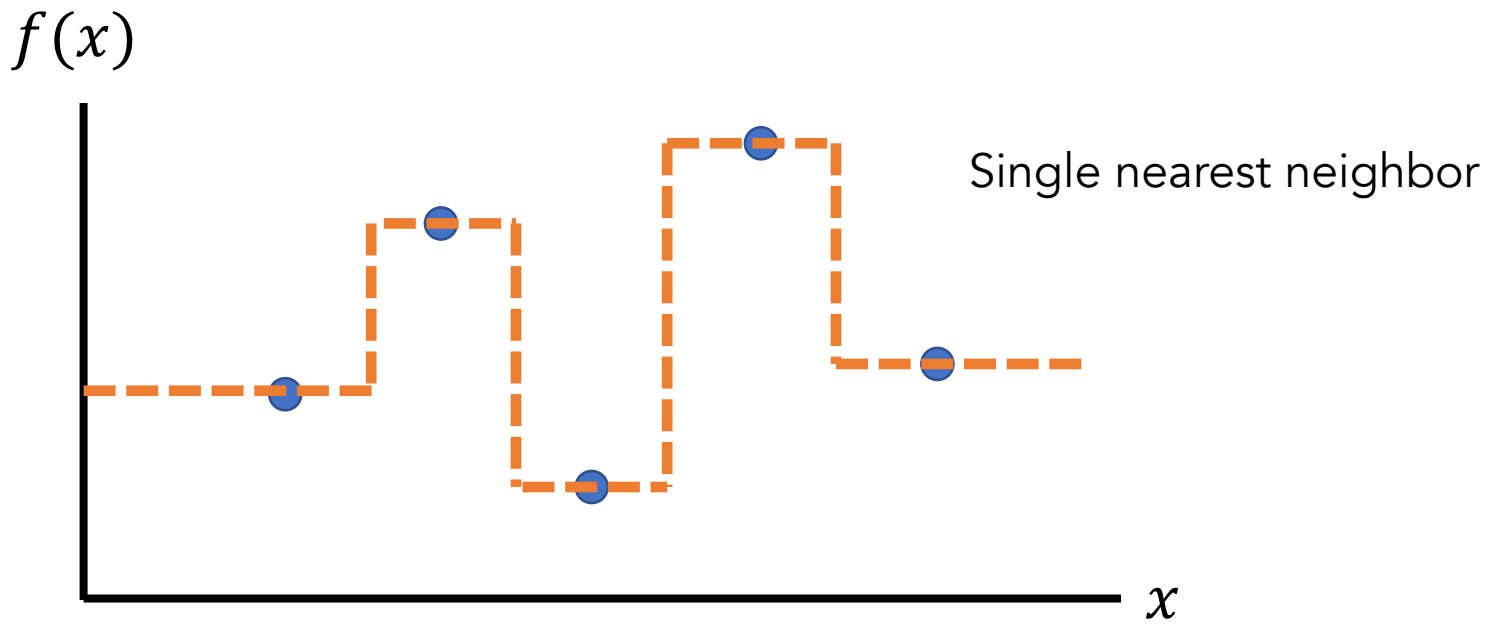
$$y = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + \cdots + b)}}$$



# Regression: Nearest Neighbor

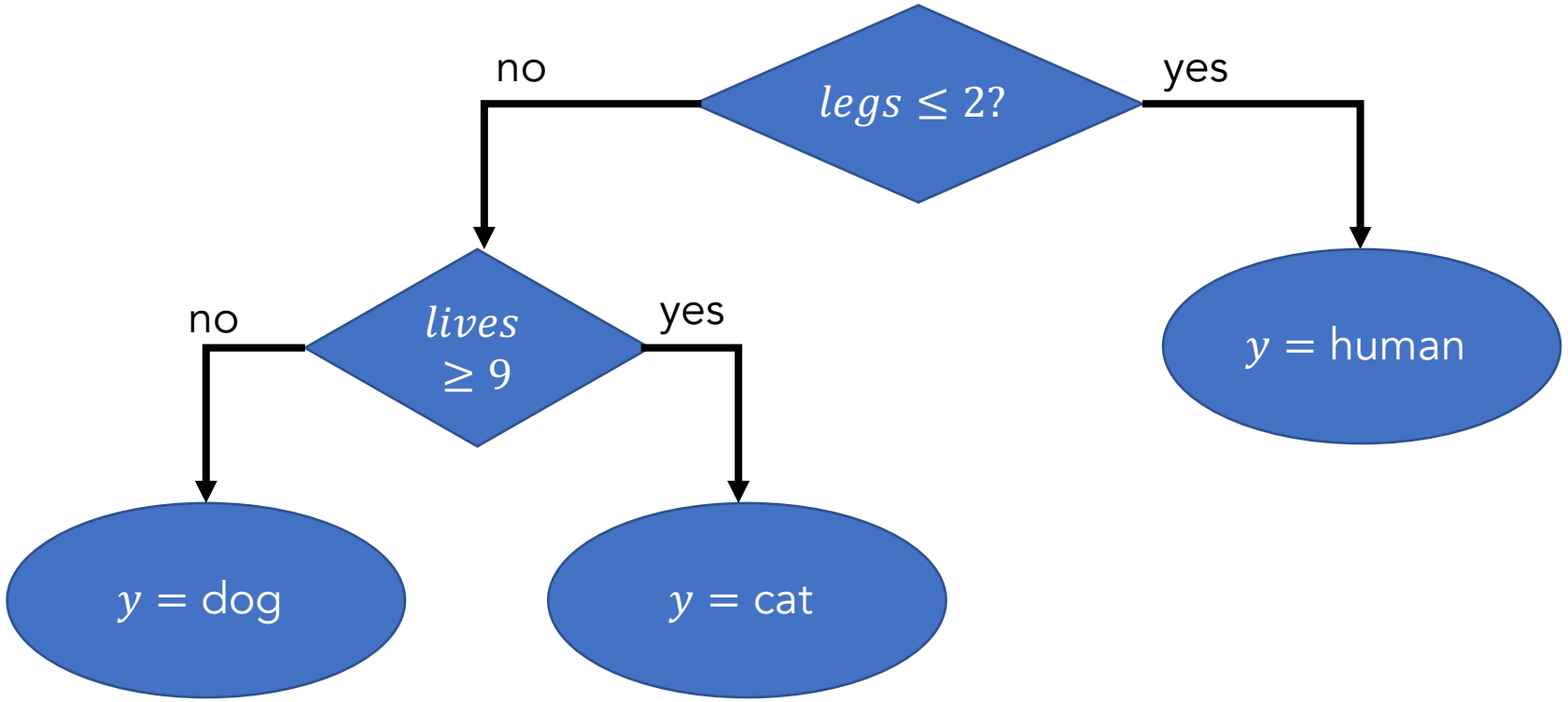
Finds the training sample(s) that is most similar and computes the new result based on their label(s)

If multiple neighbors, can use their average, median, etc.



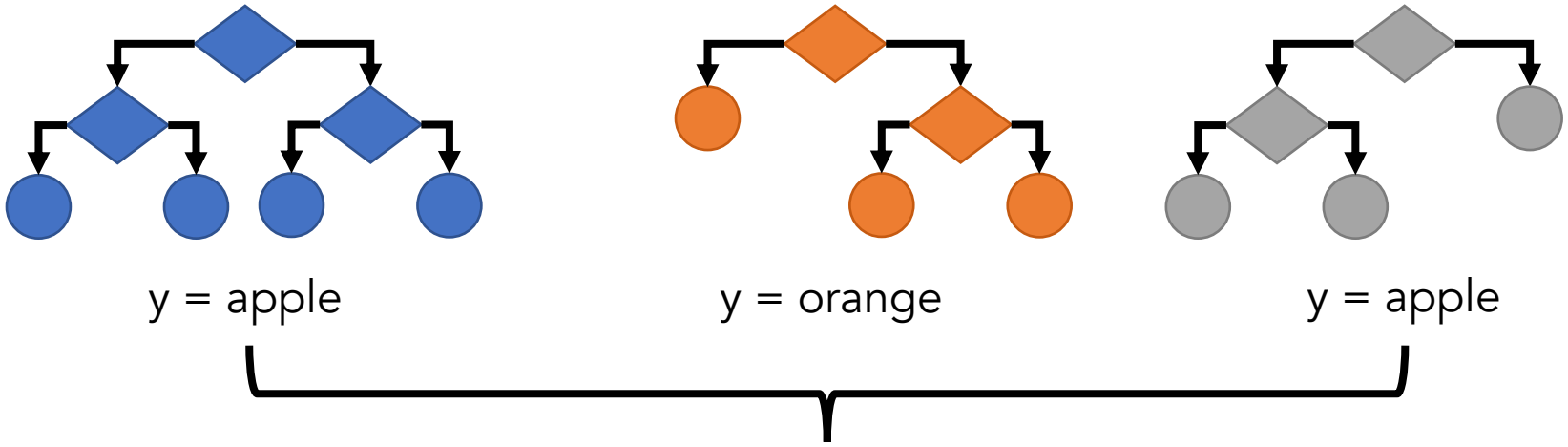
# Classification: Decision Tree

Maps a combination of features to a class label based on a series of binary decisions



# Classification: Random Forest

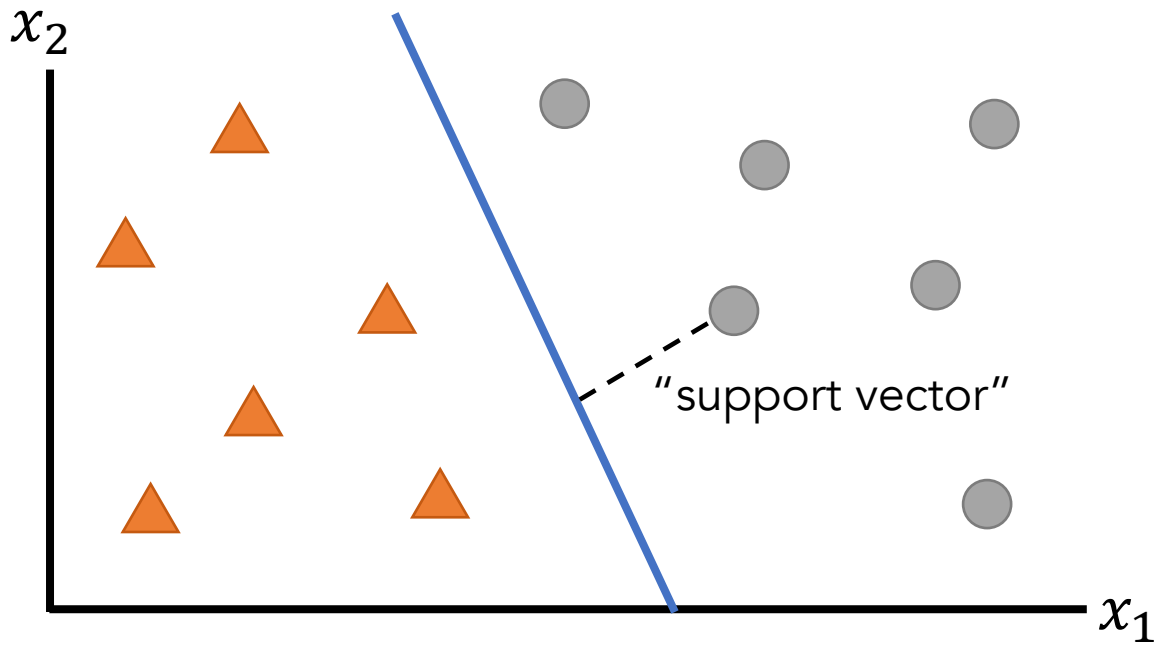
Uses a collection of decision trees
(forest = multiple trees) to make a decision



# Classification: Support Vector Machine (SVM)

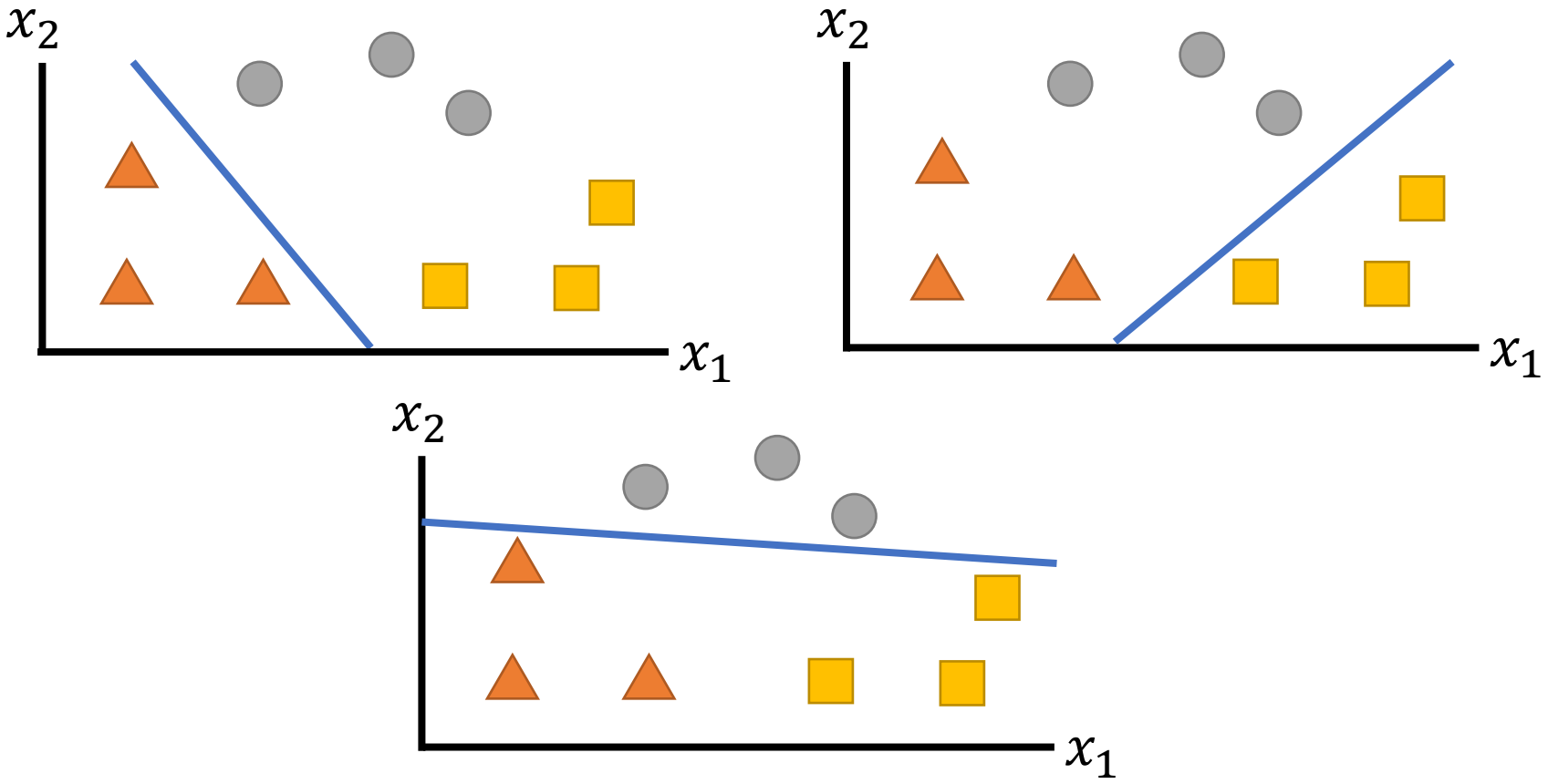
Separates two different classes with a hyperplane that maximizes separation of classes

$$y \geq w_1 x_1 + w_2 x_2 + \cdots + b$$



# Classification: Support Vector Machine (SVM)

Can separate between multiple classes by creating multiple one-versus-rest classifiers



# Regression ↔ Classification

Models for classification sometimes have an analogue for regression

| Classification | Regression |
| --- | --- |
| **Support Vector Machine (SVM)**<br>Classifies based on which side of the decision boundary the input is on | **Support Vector Regression (SVR)**<br>Regresses based on how far away the input is from the decision boundary |
| **Decision Tree**<br>Leaf nodes correspond to classes | **Decision Tree Regression**<br>Leaf nodes correspond to continuous values |

# Regression ↔ Classification

Regression can be changed to classification with thresholds

- Threshold a logistic regression at 0.5
- Bilirubin is borderline if >1.3 mg/dl, severe if >3 mg/dl

Regression gives more information

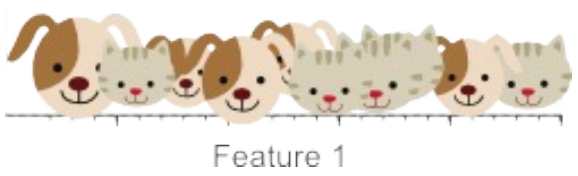
- Diagnosis would be different if bilirubin is 3.1 mg/dl vs. 15 mg/dl, but they are both in the “severe” class

# How To Do Machine Learning In 7 Easy Steps

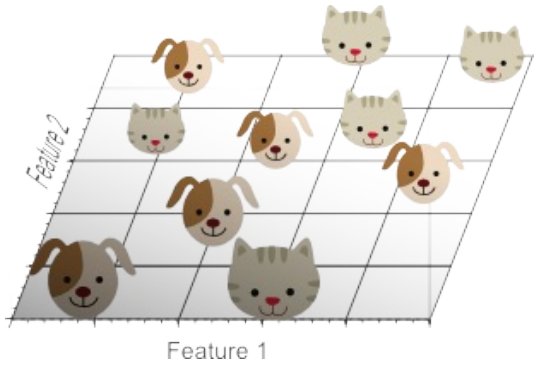
1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Curse of Dimensionality

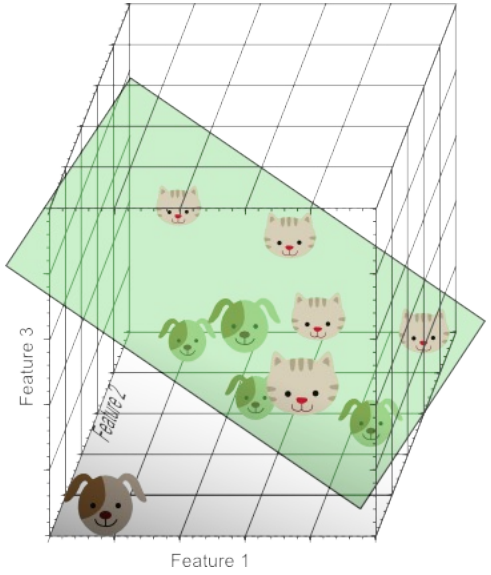
The more features we use, the higher the likelihood that we can successfully separate the classes perfectly, but…




1 feature =
50% accuracy



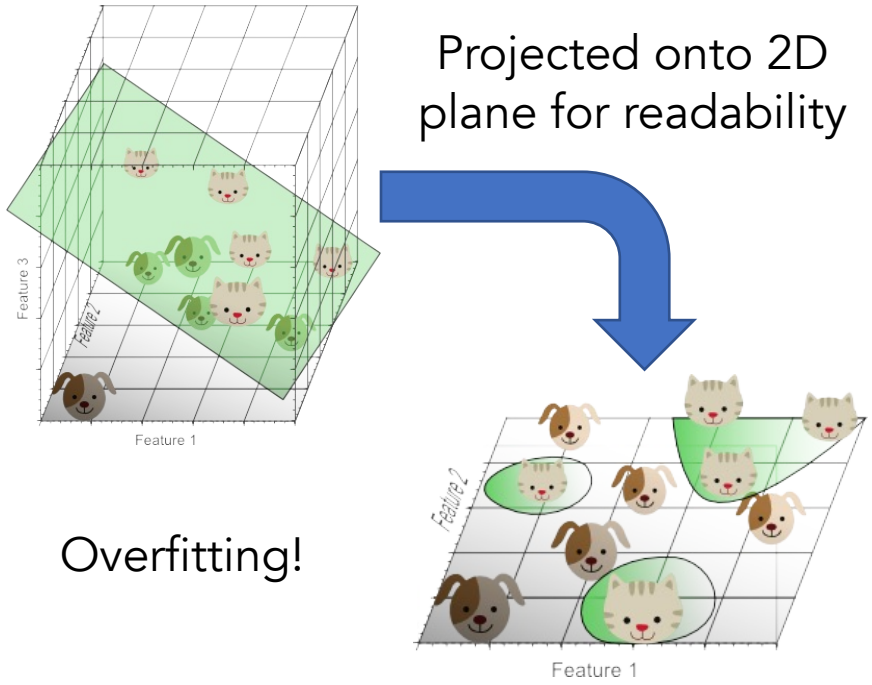

2 features =
70% accuracy




3 features =
100% accuracy

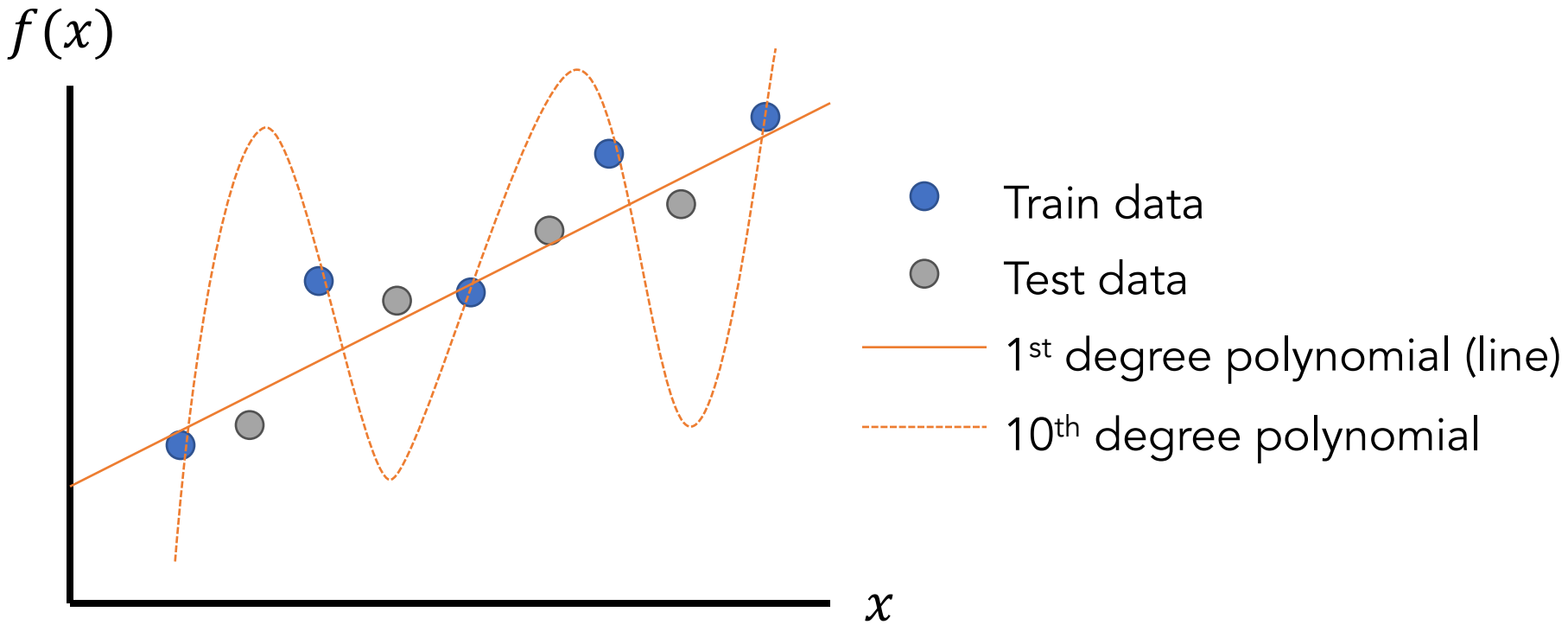
# Curse of Dimensionality

…using too many features can allow the model to learn special cases that do not generalize to unseen data (overfitting)



# Overfitting

Polynomial fit, $y = w_1 x^n + w_2 x^{n-1} + \cdots + b$



# Feature Selection

There are many automated techniques you can use to pick the most important features

Some techniques care about the labels, while others just care about properties of the data itself

# Variance Threshold

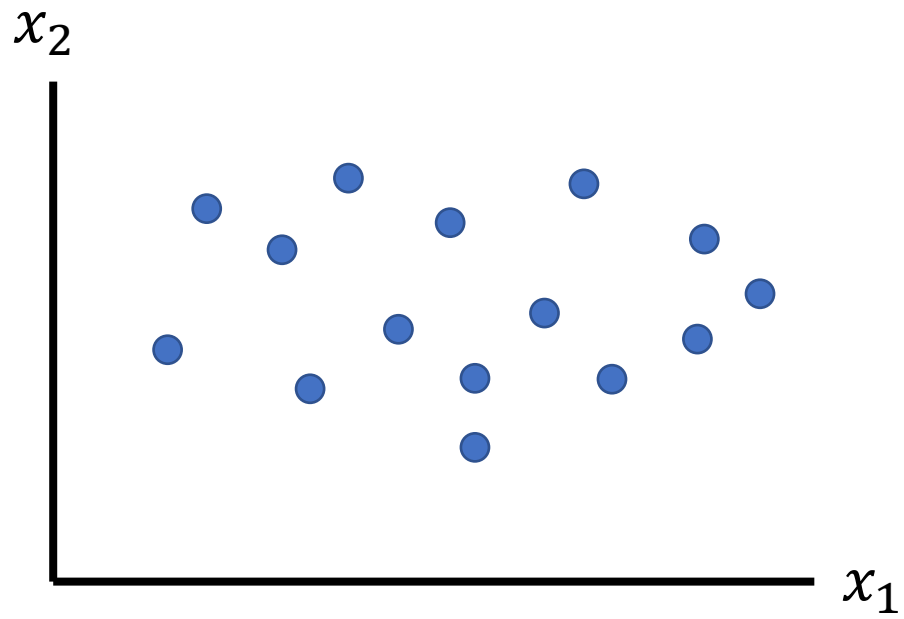
Only keep features that are spread apart (high variance) to keep the data as "unique" as possible

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

$\sigma^2$ = variance

$\bar{x}$ = mean value

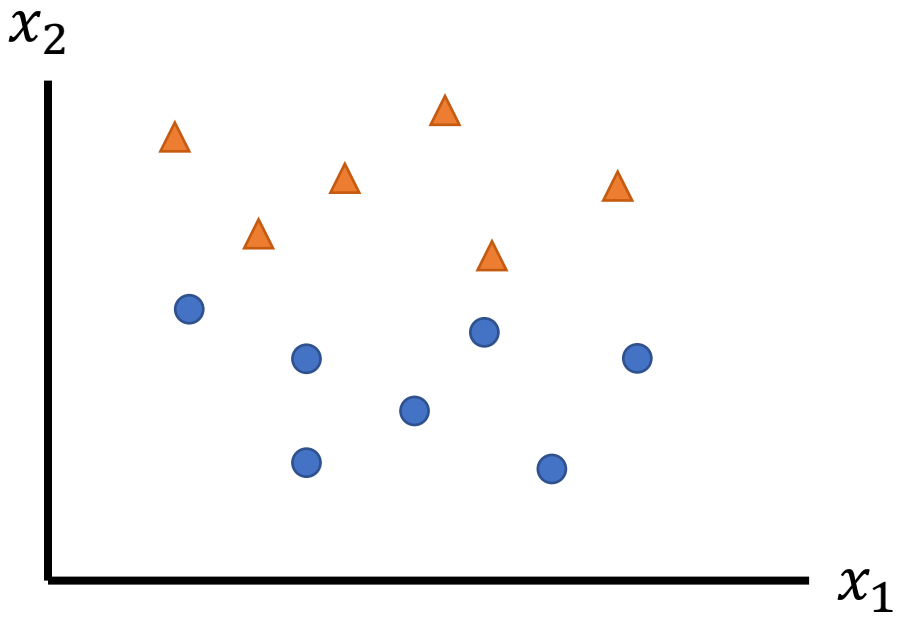
$N$ = number of points

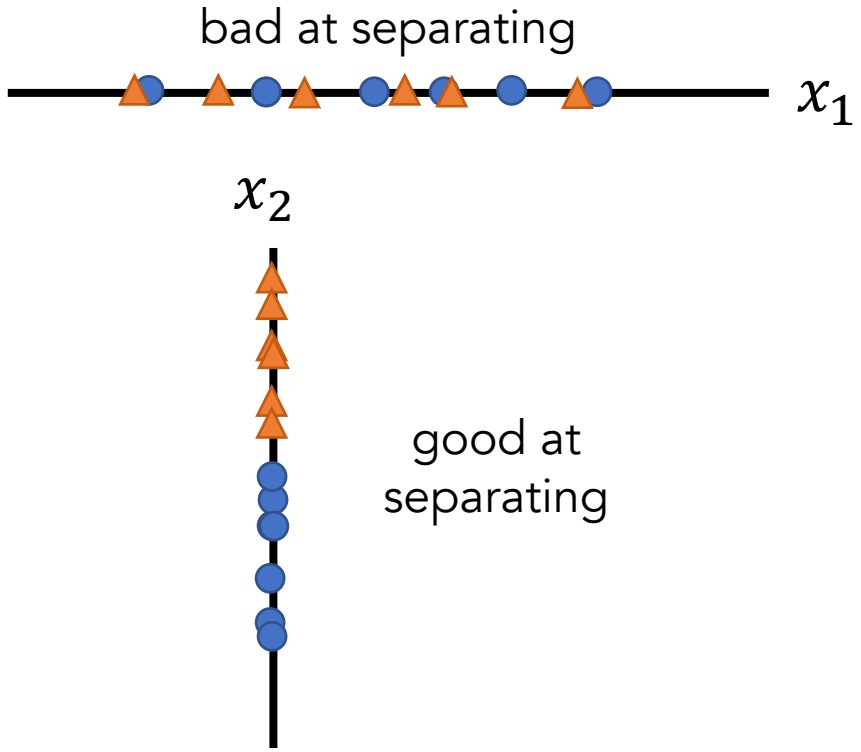



Which feature has the higher variance? $x_1$

# Univariate Feature Selection

Performs statistical tests using one feature at a time and picks the ones that work best



# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Hyperparameters

A property of the specific model that you are using that can be adjusted to affect how well it works

| Model | Examples of Hyperparameters |
| --- | --- |
| **Nearest Neighbors** | **Number of neighbors:** how many neighbors to use for calculation<br>**Radius of neighbors:** the radius within which neighbors can be considered |
| **Decision Tree** | **Max depth:** how deep is the tree allowed to be<br>**Min samples for split**: the minimum of samples required to justify a new decision |

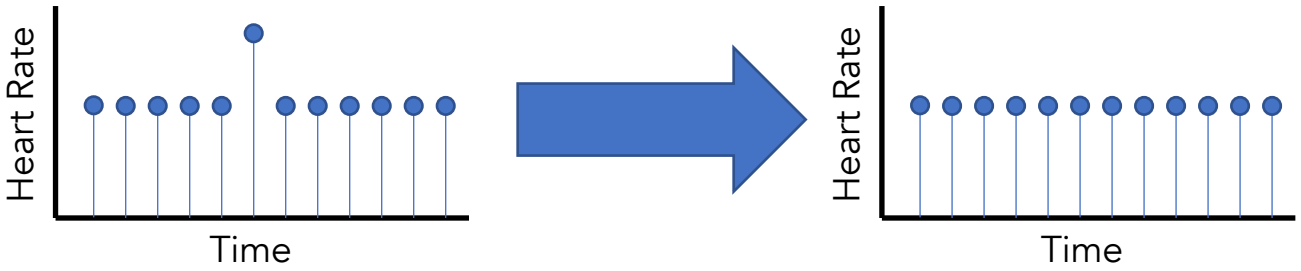
# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Outputs

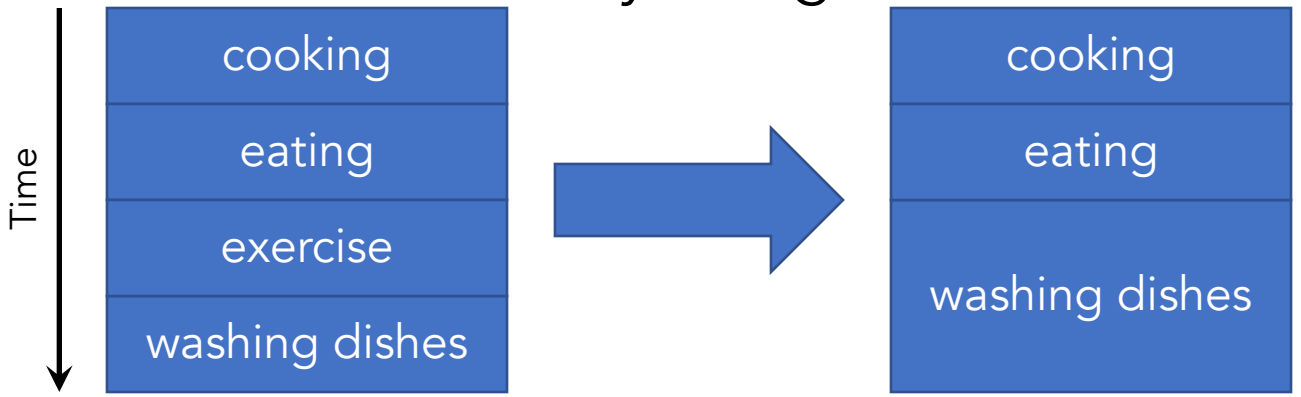
The results of your machine learning model don't have to be final

Ex: smoothing in heart rate tracking




How can someone's heart rate jump by 40 bpm for 1 second?

Ex: context in activity recognition



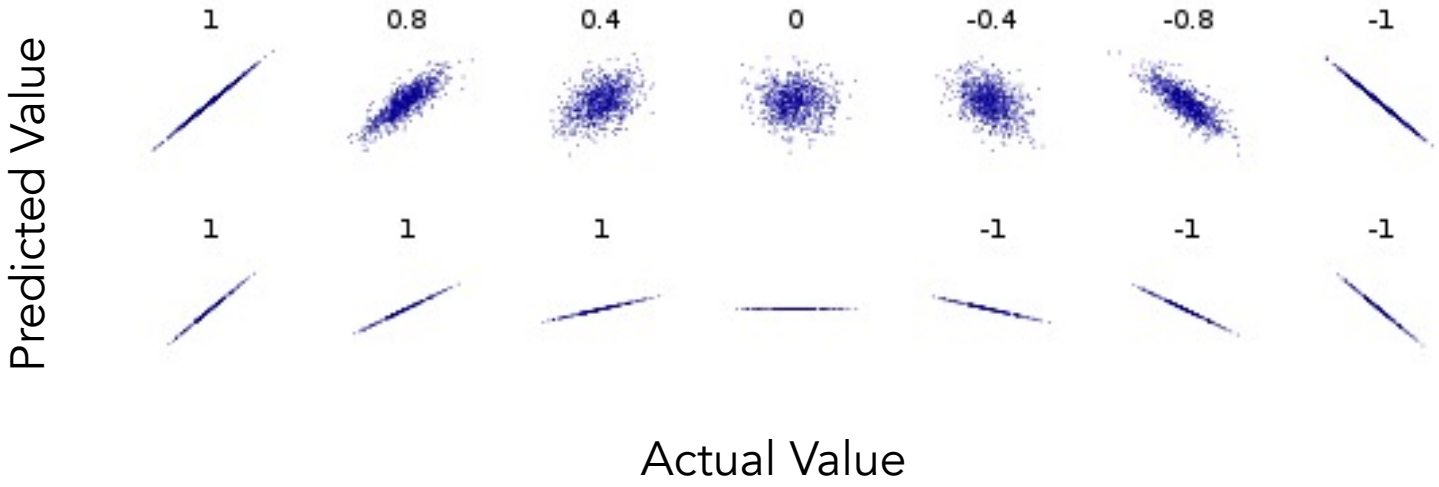

Who exercises between eating and washing dishes?

# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Regression Results

Pearson Correlation: measures the correlation between two normally-distributed variables




Spearman Correlation: measures the correlation between the rank/ordering of two variables

# Binary Classification Results

## Confusion Matrix

| | Positive | Negative |
|---|---|---|
| **Positive** | True positive (TP) | False positive (FP) |
| **Negative** | False negative (FN) | True negative (TN) |

## Overall Performance

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN}$$

$$\text{F1-score} = \frac{2*TP}{2*TP+FP+FN}$$

# Binary Classification Results

## Confusion Matrix

| | Positive | Negative |
|---|---|---|
| **Positive** | True positive (TP) | False positive (FP) |
| **Negative** | False negative (FN) | True negative (TN) |

## Information Retrieval

$$\text{Precision} = \frac{TP}{TP+FP}$$

What fraction of items that were returned were actually relevant to my query?

$$\text{Recall} = \frac{TP}{TP+FN}$$

What fraction of items that were relevant to my query were actually returned?

# Binary Classification Results

## Confusion Matrix

| | Positive | Negative |
|---|---|---|
| **Positive** | True positive (TP) | False positive (FP) |
| **Negative** | False negative (FN) | True negative (TN) |

## Diagnostics

Sensitivity $= \frac{TP}{TP+FN}$

What fraction of people with the given health condition were given a positive test result?

Specificity $= \frac{TN}{TN+FP}$

What fraction of people without the given health condition were given a negative test result?

# Multi-Class Classification Results

| | Dog | Cat | Human |
|---|---|---|---|
| Dog | DD | CD | HD |
| Cat | DC | CC | HC |
| Human | DH | CH | HH |

$$\text{Overall Accuracy} = \frac{DD + CC + HH}{CD + HD + DC + HC + DH + CH}$$

# Multi-Class Classification Results

| | Dog | Cat | Human |
|---|---|---|---|
| Dog | DD | CD | HD |
| Cat | DC | CC | HC |
| Human | DH | CH | HH |

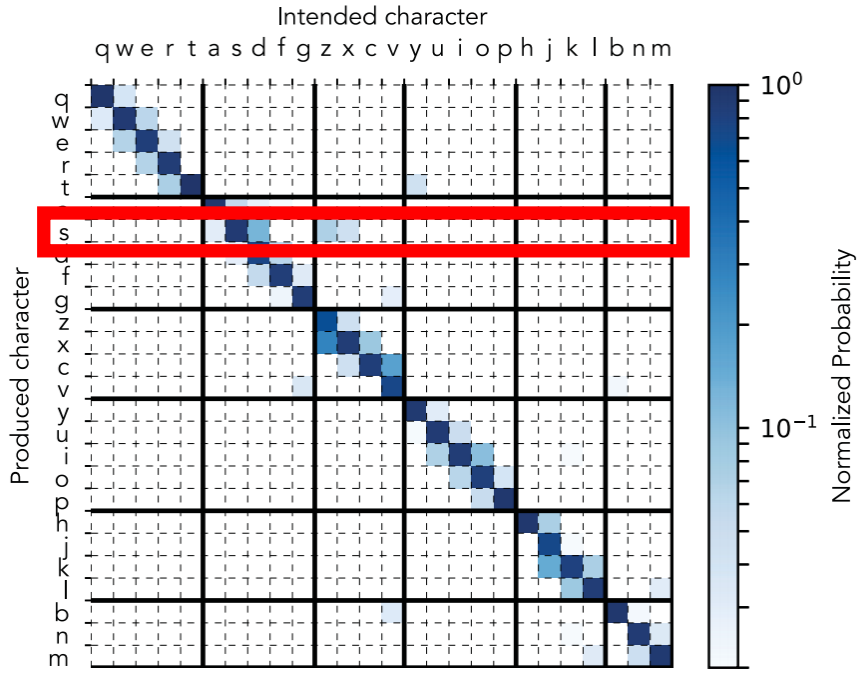
| | Dog | Non-Dog |
|---|---|---|
| Dog | DD | CD+HD |
| Non-Dog | DC+CH | CC+HC+ CH+HH |

3-class problem → 2-class problem

$$\text{Precision at identifying dogs} = \frac{TP}{TP+FP} = \frac{DD}{DD+CD+HD}$$

$$\text{Recall at identifying dogs} = \frac{TP}{TP+FN} = \frac{DD}{DD+DC+CH}$$

# Multi-Class Classification Results



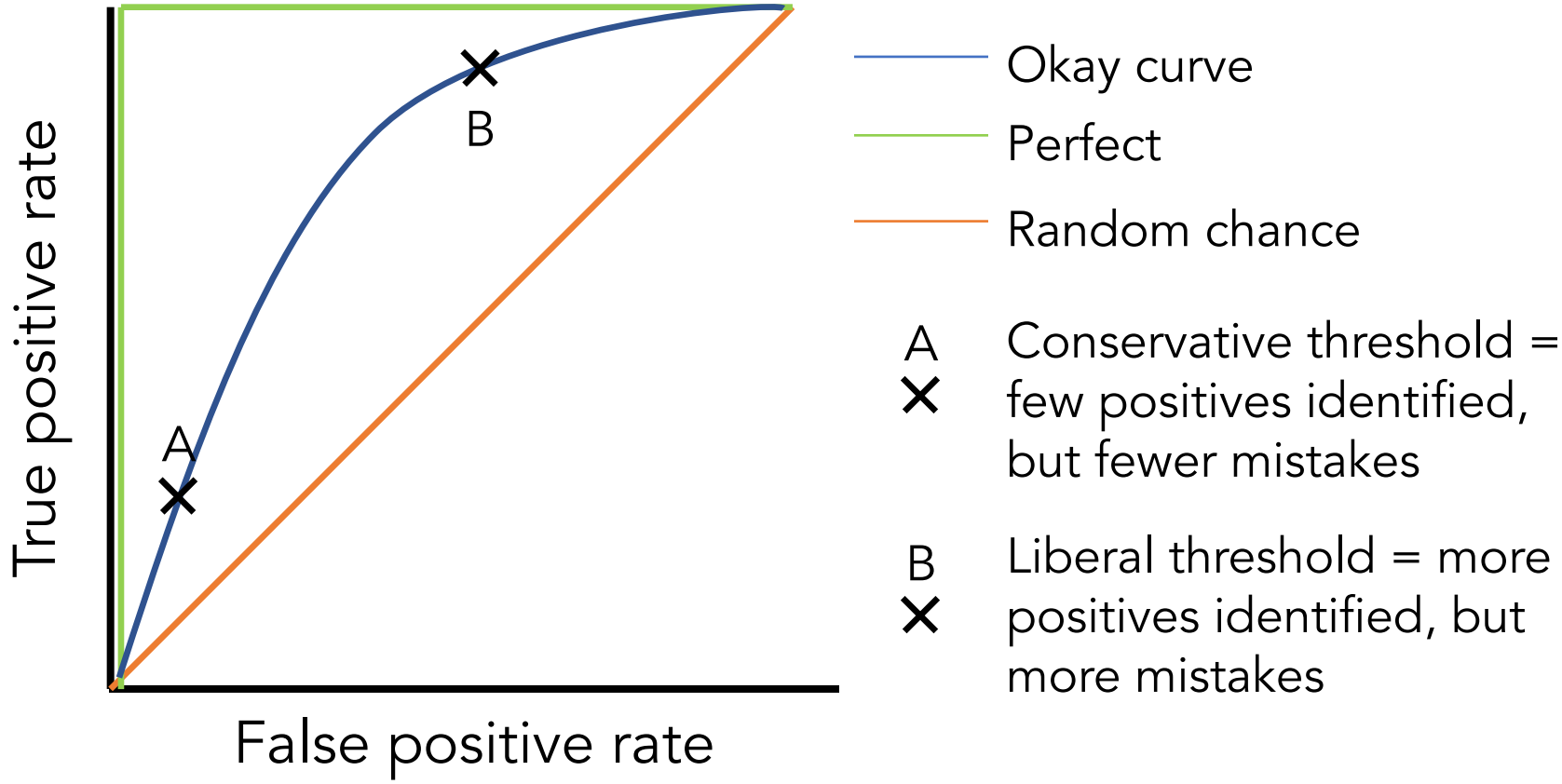

What characters are being confused with the letter ‘s’?

‘a’, ‘d’, ‘z’, ‘x’


Why might this be the case?

# ROC Curve

ROC = Receiver Operating Characteristic




Shows how a classifier's performance changes depending on different thresholds

# How To Do Machine Learning In 7 Easy Steps

1. Create your features and labels
2. Decide how the data should be split for training and testing
3. Select an appropriate model based on the kind of problem you want to solve
4. Add feature selection, if applicable
5. Add a mechanism for tuning hyperparameters, if applicable
6. Add post-processing, if applicable
7. Use an appropriate method for visualizing or interpreting accuracy

# Questions?