

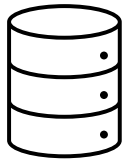
# Foundations – Machine Learning

Foundations

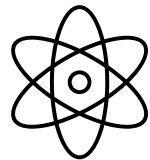


# Machine Learning

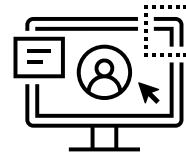
Before we jump into Neural Networks, Tensorflow, Keras API etc... its a good idea to understand a few fundamental ideas regarding machine learning.



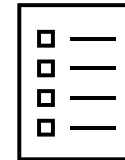
**Learning from  
Data**



**Adaptability &  
Generalization**



**Improvement  
with Experience**



**Task specific  
customization**

# Machine Learning

The following foundational knowledge areas are covered:

What is Machine Learning?

What is Deep Learning?

Supervised vs Unsupervised Learning

Supervised Learning Process

Evaluating Performance

Overfitting

# Machine Learning

## What is Machine Learning?

- Machine learning is a method of data analysis that automates analytical model building.
- Using algorithms that iteratively learn from data, machine learning allows computers to find hidden insights without being explicitly programmed where to look.

# What is it used for?

- Fraud detection.
- Web search results.
- Real-time ads on web pages
- Credit scoring.
- Prediction of equipment failures.
- New pricing models.
- Network intrusion detection.
- Recommendation Engines
- Customer Segmentation
- Text Sentiment Analysis
- Customer Churn
- Pattern and image recognition.
- Email spam filtering

# What are Neural Networks?

- Neural Networks are a way of modeling biological neuron systems mathematically.
- These networks can then be used to solve tasks that many other types of algorithms can not (e.g. image classification)
- Deep Learning simply refers to neural networks with more than one hidden layer.

# Machine Learning

There are different types of machine learning we will focus on during the next sections of the course:

**Supervised Learning**

**Unsupervised Learning**

# Machine Learning

## Machine Learning

Automated  
analytical models

## Neural Networks

A type of machine  
learning  
architecture  
modeled after  
biological neurons

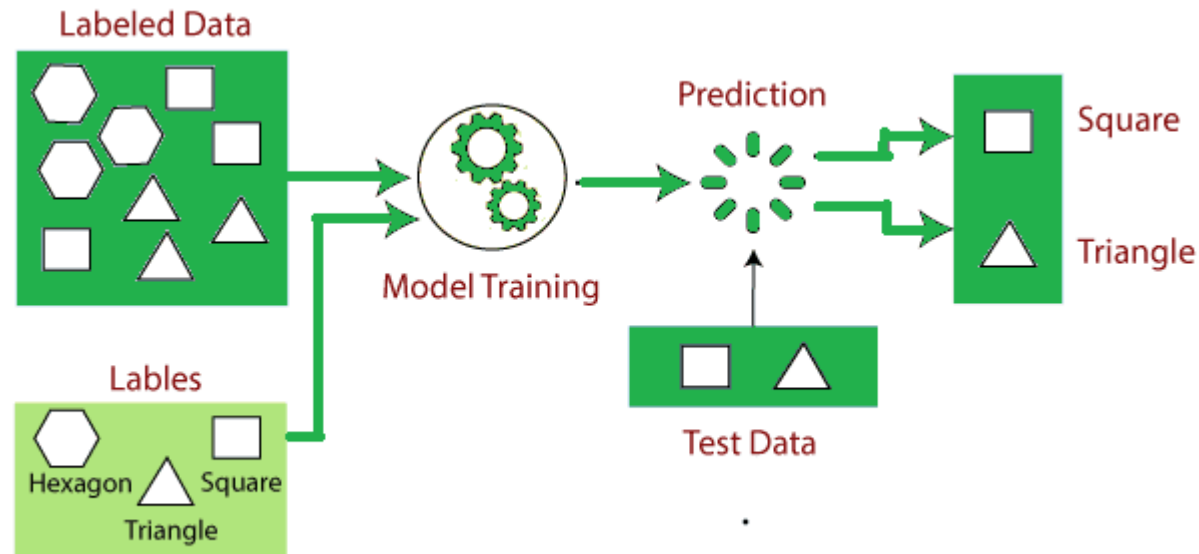
## Deep Learning

A neural network  
with more than one  
hidden layer



# Machine Learning

- Let's begin by learning about one of the most common machine learning tasks- Supervised Learning!



# Supervised Learning

Foundations



# Supervised Learning

- **Supervised learning** algorithms are trained using **labeled** examples, such as an input where the desired output is known.
- For example, a segment of text could have a category label, such as:
  - **Spam vs. Legitimate Email**
  - **Positive vs. Negative Movie Review**

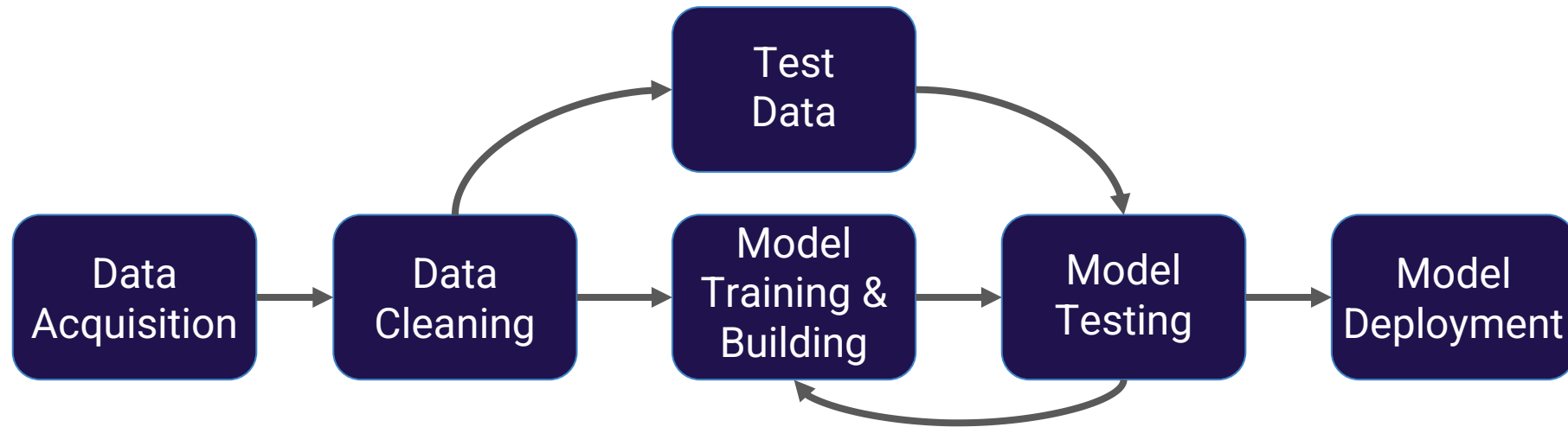
# Supervised Learning

- The network receives a set of inputs along with the corresponding correct outputs, and the algorithm learns by comparing its actual output with correct outputs to find errors.
- It then modifies the model accordingly.

# Supervised Learning


- Supervised learning is commonly used in applications where historical data predicts likely future events.

# Machine Learning Process



# Machine Learning Process

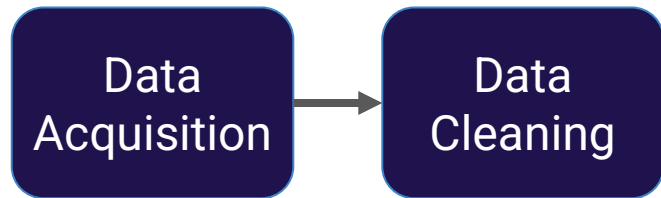
Get your data! Customers, Sensors, etc...



Data  
Acquisition

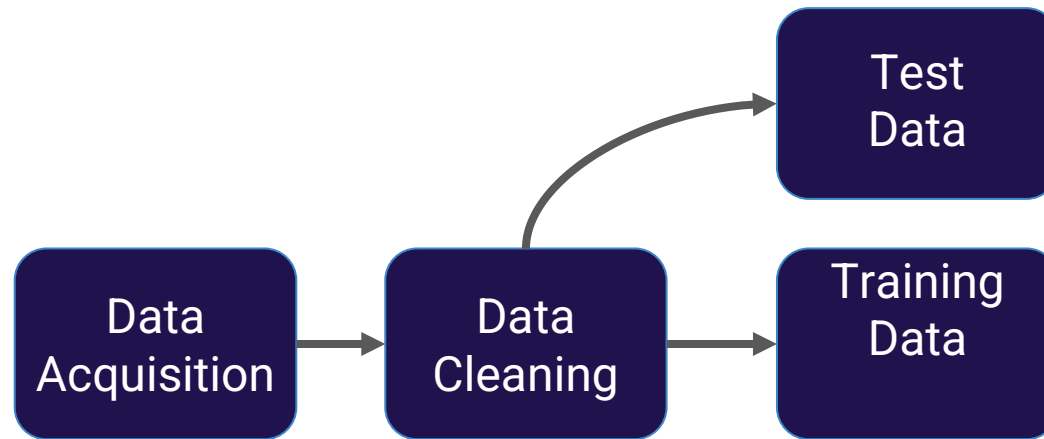
# Machine Learning Process

- Clean and format your data (using Pandas)

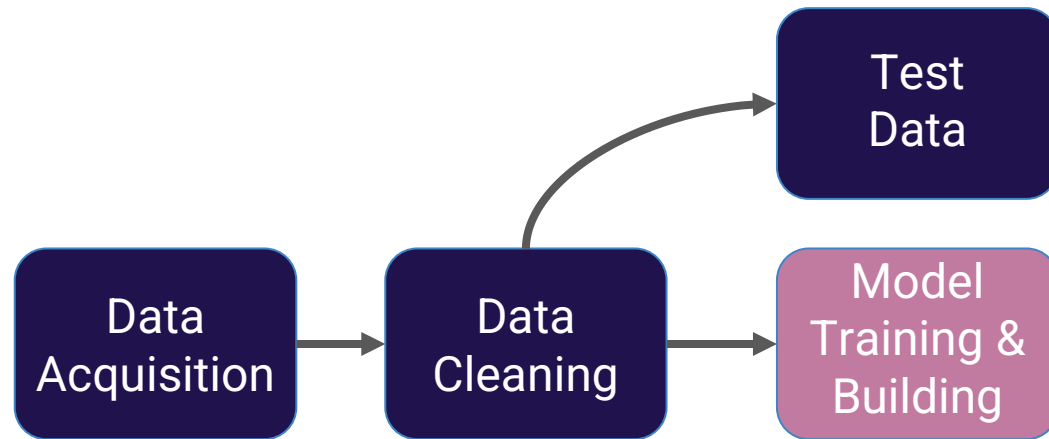




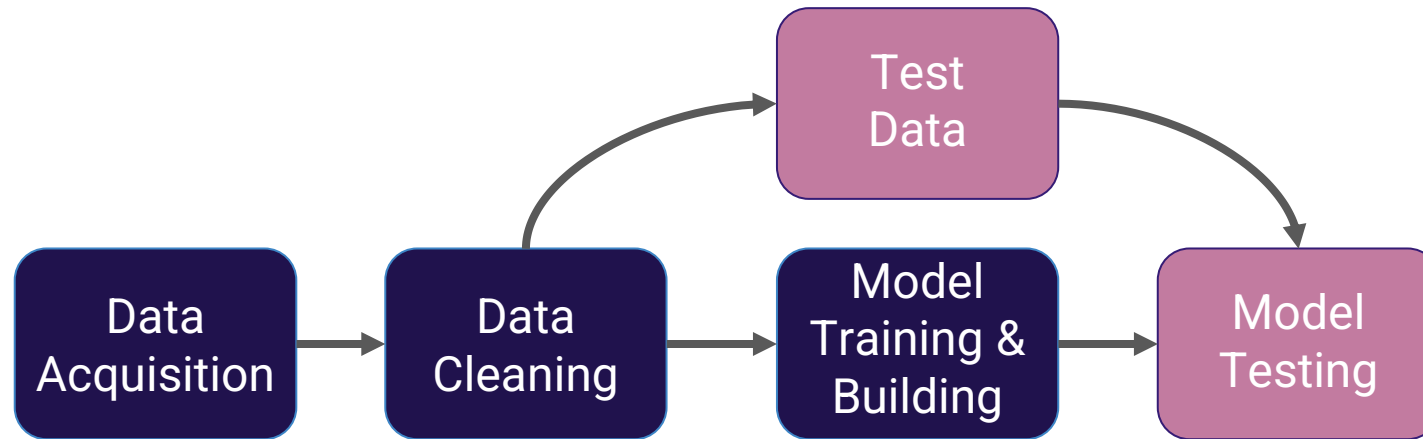
# Machine Learning Process



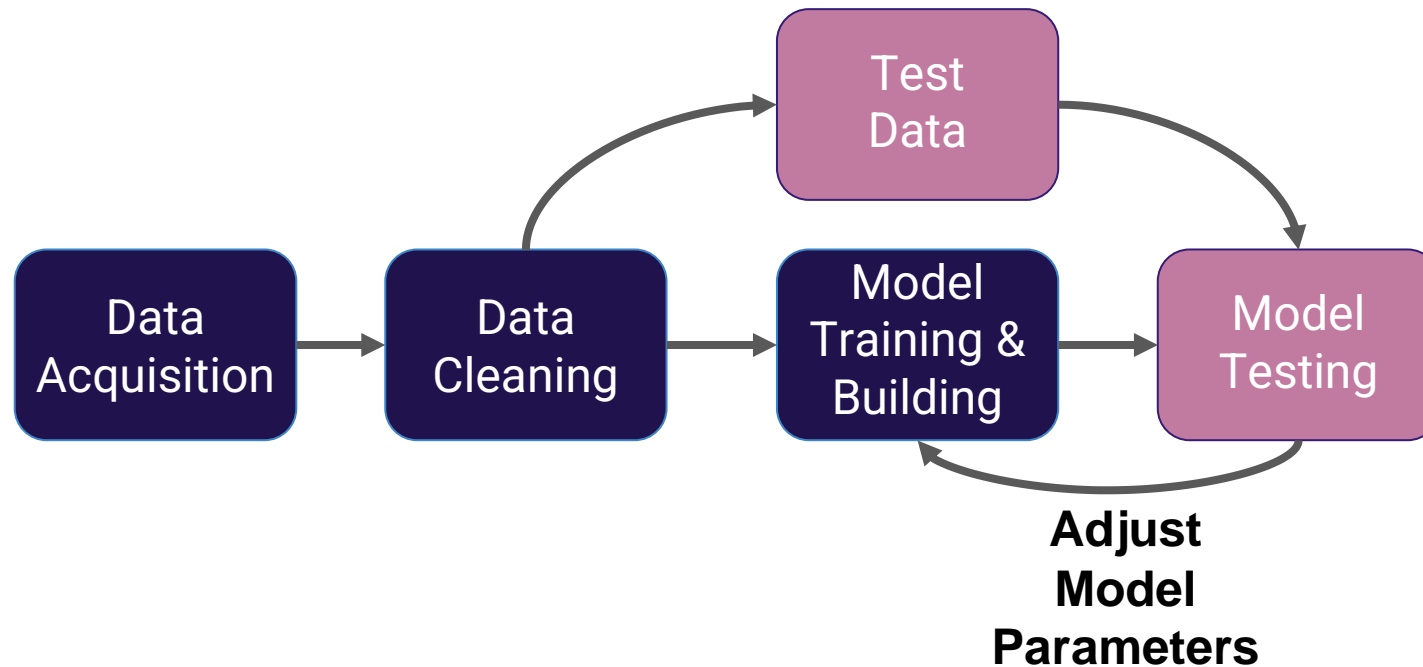
# Machine Learning Process



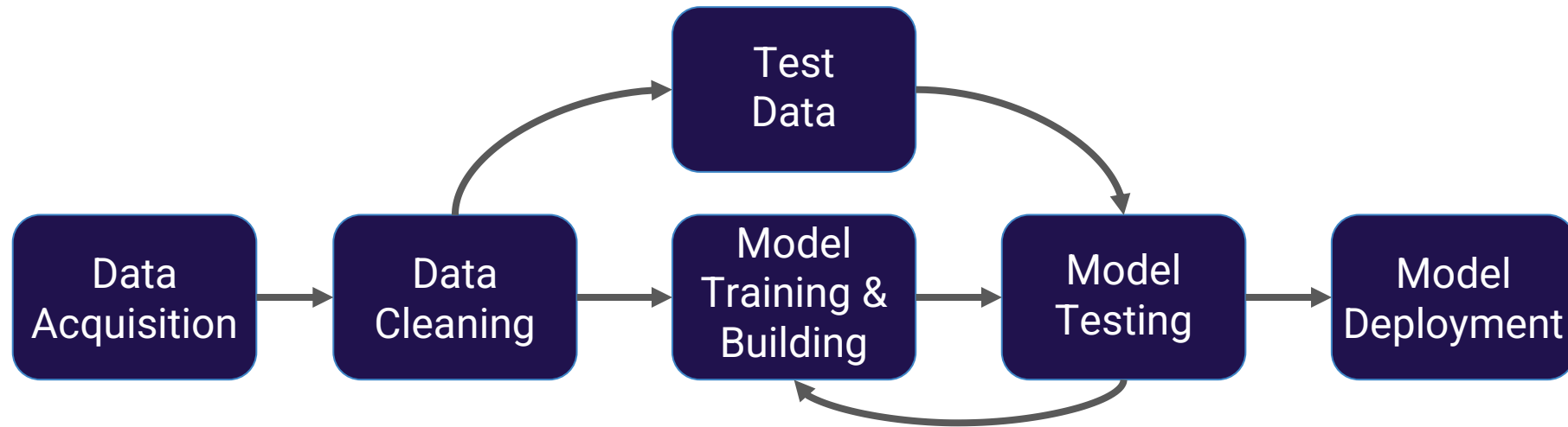
# Machine Learning Process



# Machine Learning Process



# Machine Learning Process



# Supervised Learning

- What we just showed is a simplified approach to supervised learning, it contains an issue!
- Is it fair to use our single split of the data to evaluate our models performance?
- After all, we were given the chance to update the model parameters again and again.

# Supervised Learning

- To fix this issue, data is often split into **3 sets**
  - Training Data
    - Used to train model parameters
  - Validation Data
    - Used to determine what model hyperparameters to adjust
  - Test Data
    - Used to get some final performance metric

# Supervised Learning

- This means after we see the results on the **final test set** we don't get to go back and adjust any model parameters!
- This final measure is what we label the true performance of the model to be.



# Supervised Learning

- In this course, in general we will simplify our data by using a simple **train/test split**.
- We will simply train and then evaluate on a test set (leaving the option to students to go back and adjust parameters).
- After going through the course, you will be able to easily perform another split to get **3 data sets** if you desire.

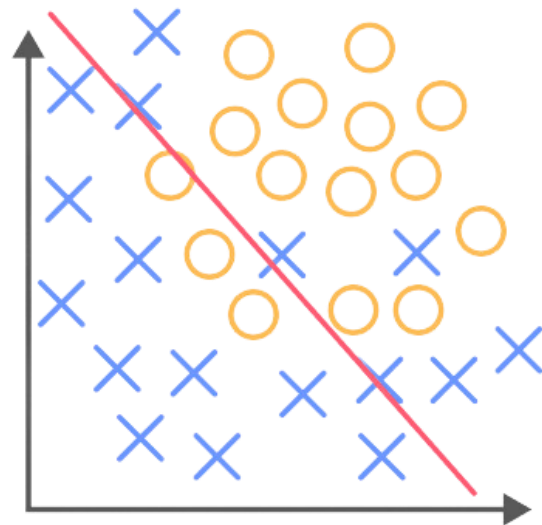
# Overfitting & Underfitting



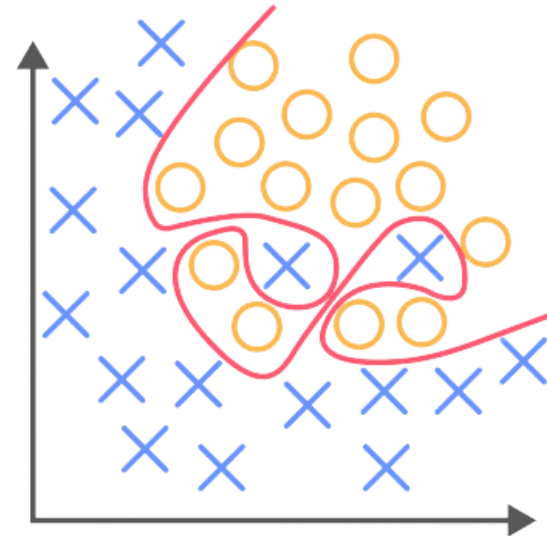
Foundations

# Machine Learning

- Now that we understand the full process for supervised learning, let's touch upon the important topics of **overfitting** and **underfitting**.



Underfitting



Overfitting

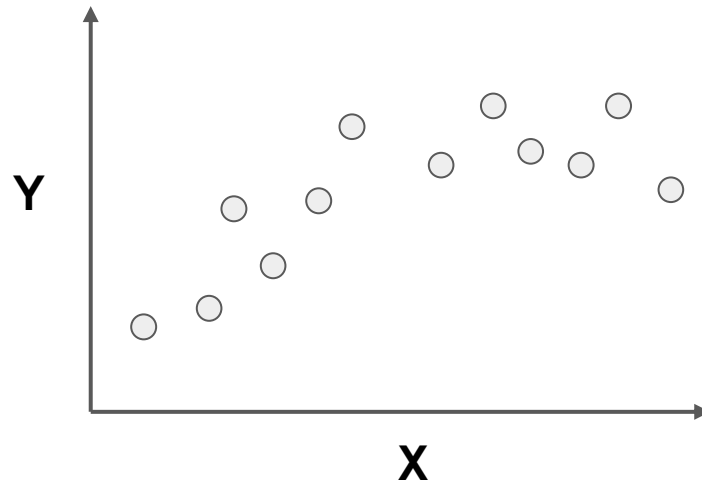
# Machine Learning

## Overfitting

- The model fits too much to the noise from the data.
- This often results in **low error on training sets but high error on test/validation sets.**

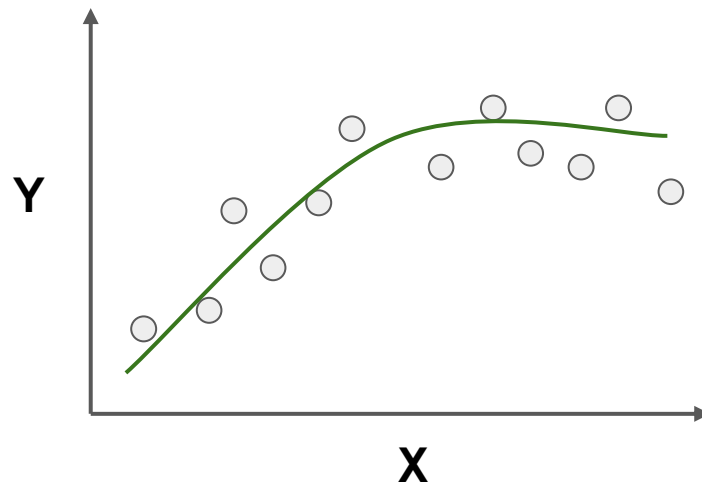
# Machine Learning

Data



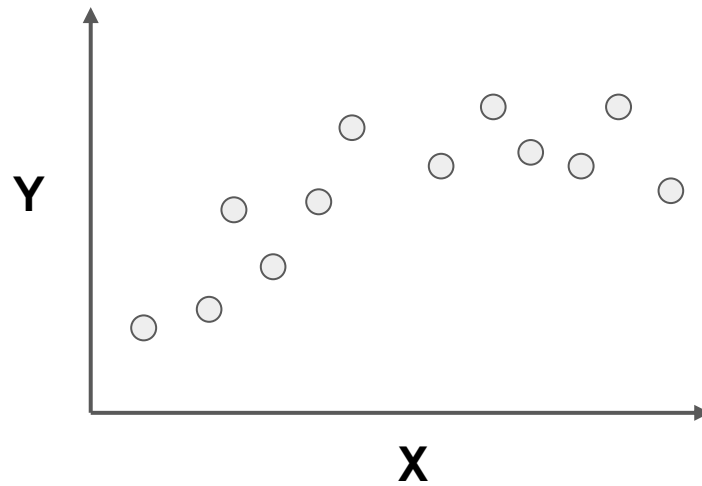
# Machine Learning

**Good Model**



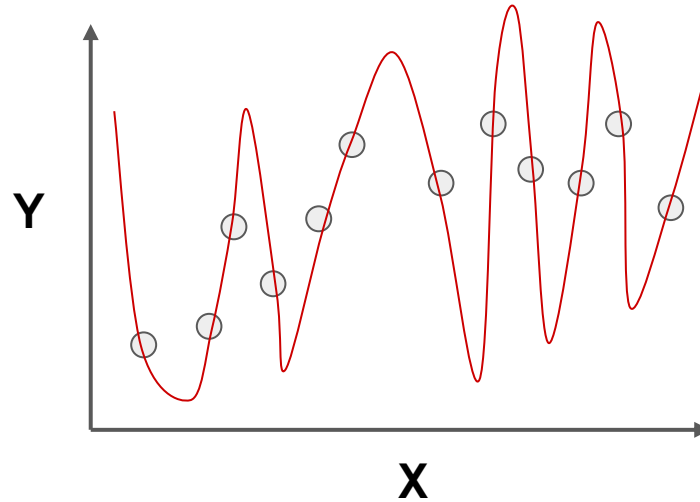
# Machine Learning

- **Overfitting**



# Machine Learning

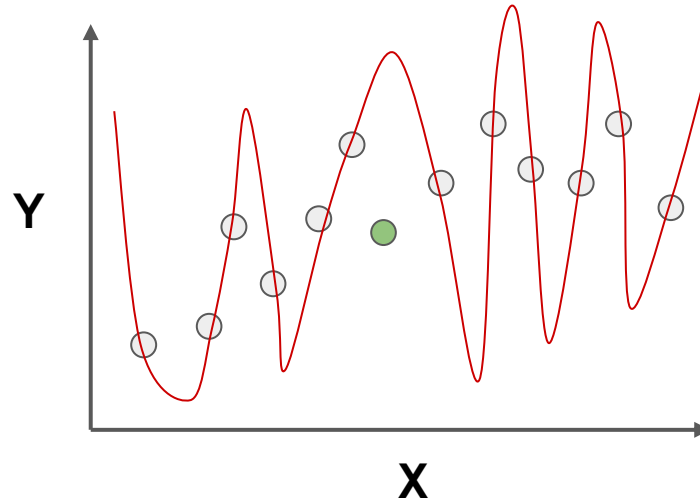
- **Overfitting**





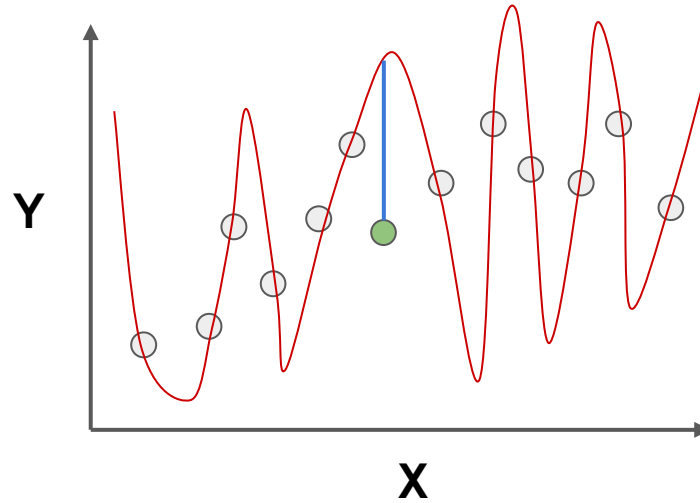
# Machine Learning

- **Overfitting**



# Machine Learning

- **Overfitting**



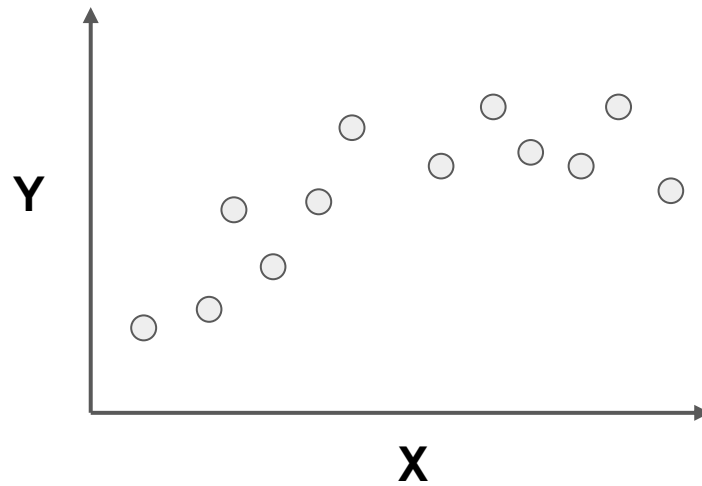
# Machine Learning

## Underfitting

- Model does not capture the underlying trend of the data and does not fit the data well enough.
- Low variance but high bias.
- Underfitting is often a result of an excessively simple model.

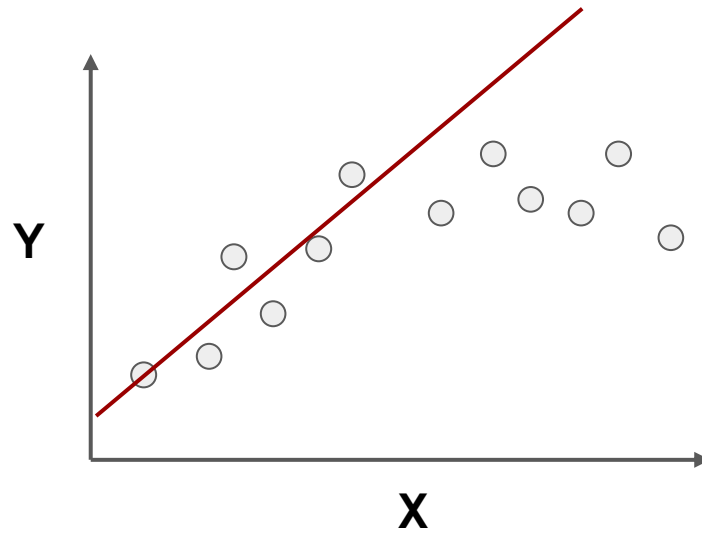
# Machine Learning

**Data**



# Machine Learning

## Underfitting

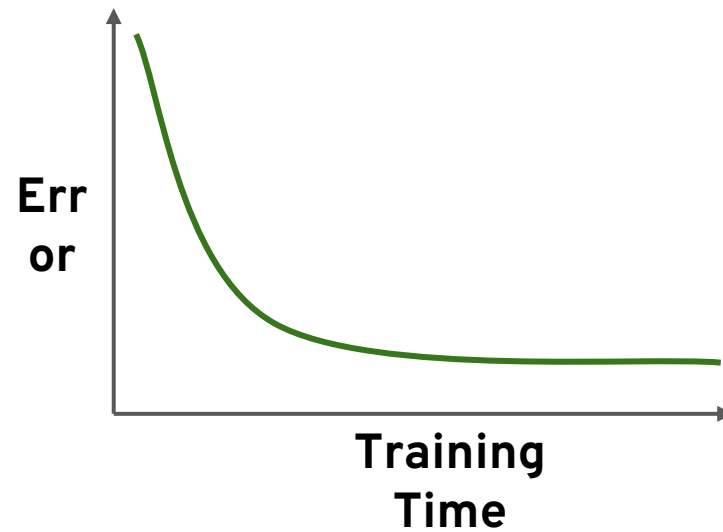


# Machine Learning

- This data was easy to visualize, but how can we see underfitting and overfitting when dealing with multi dimensional data sets?
- First let's imagine we trained a model and then measured its error over training time.

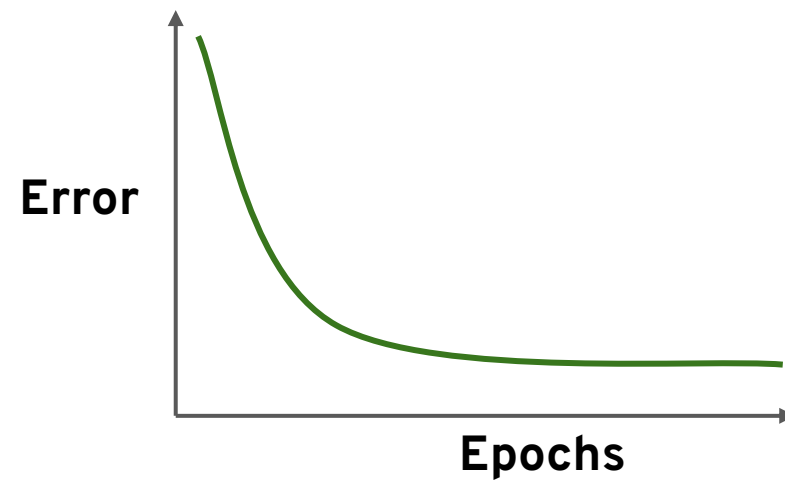
# Machine Learning

- Good Model



# Machine Learning

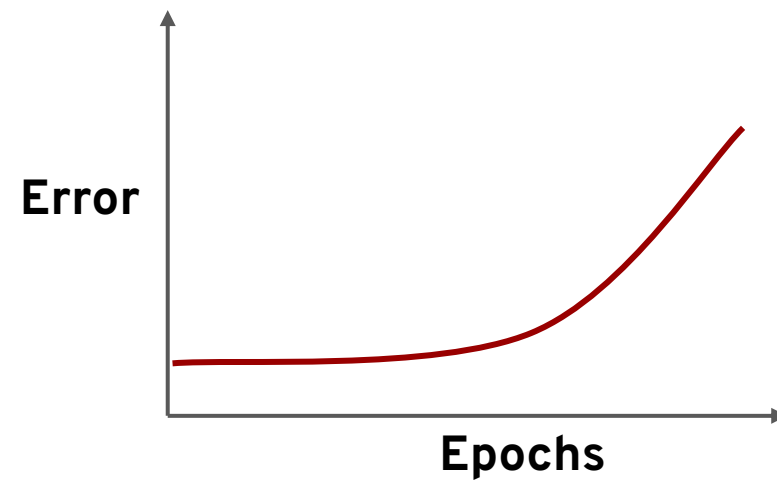
- Good Model





# Machine Learning

- Bad Model



# Machine Learning

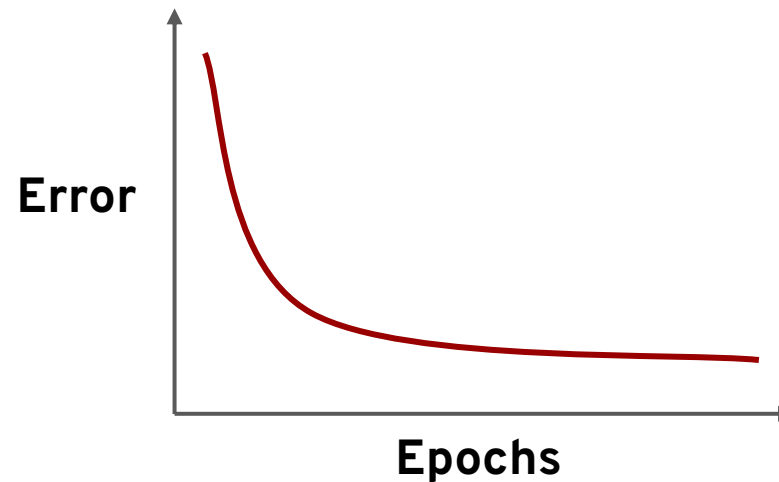
When thinking about **overfitting** and **underfitting** we want to keep in mind the relationship of model performance on the training set versus the test/validation set.

# Machine Learning

Let's imagine we split our data into a **training set** and a **test set**

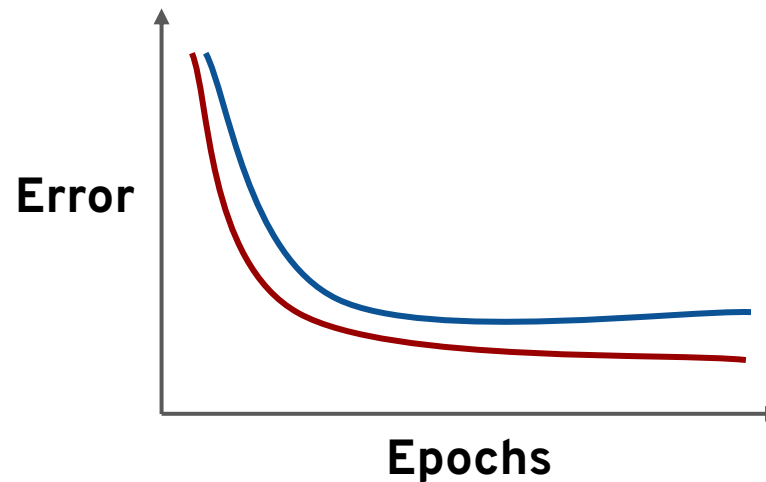
# Machine Learning

We first see performance on the **training set**



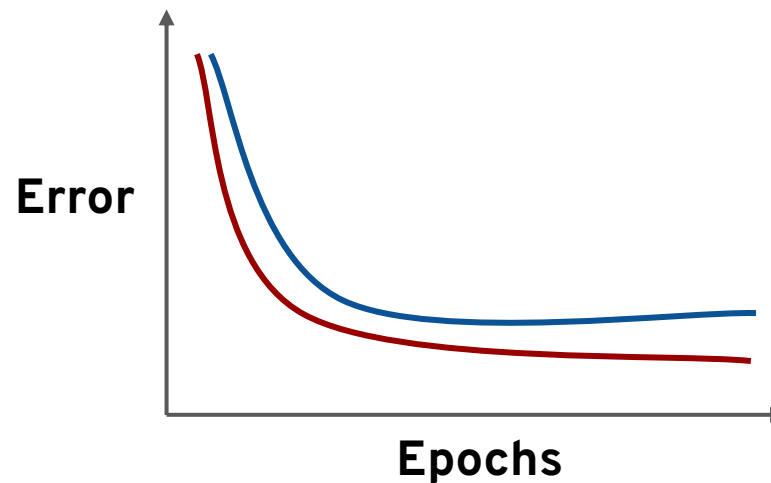
# Machine Learning

Next we check performance on the **test set**



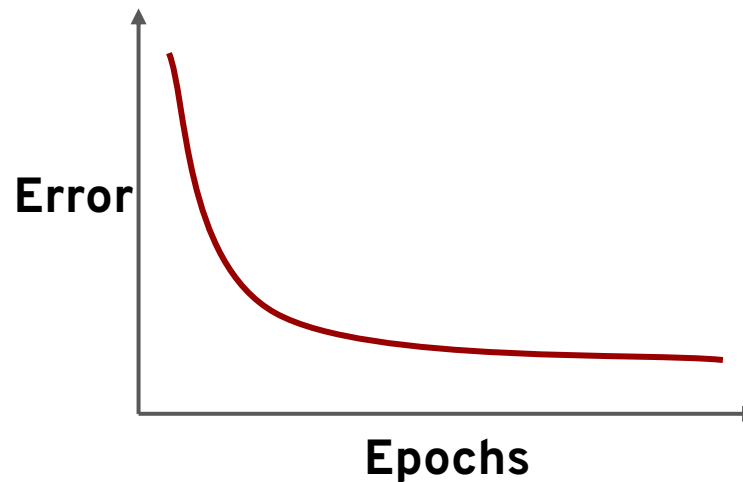
# Machine Learning

Ideally the model would perform well on both, with similar behavior.



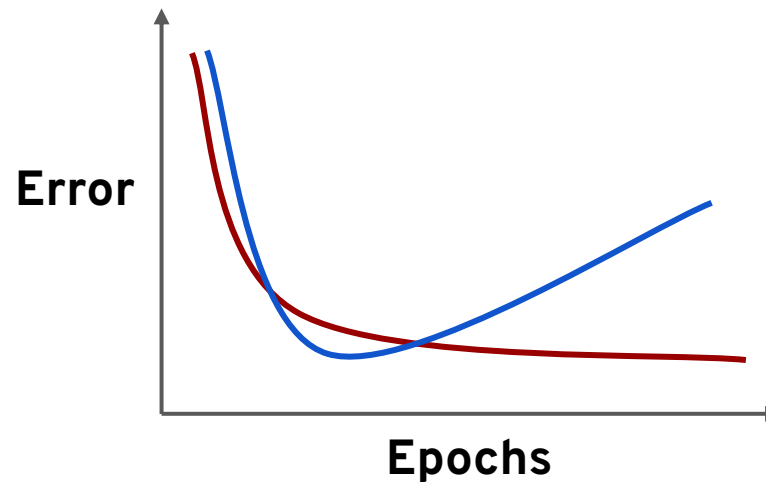
# Machine Learning

But what happens if we overfit on the training data? That means we would perform poorly on new test data!



# Machine Learning

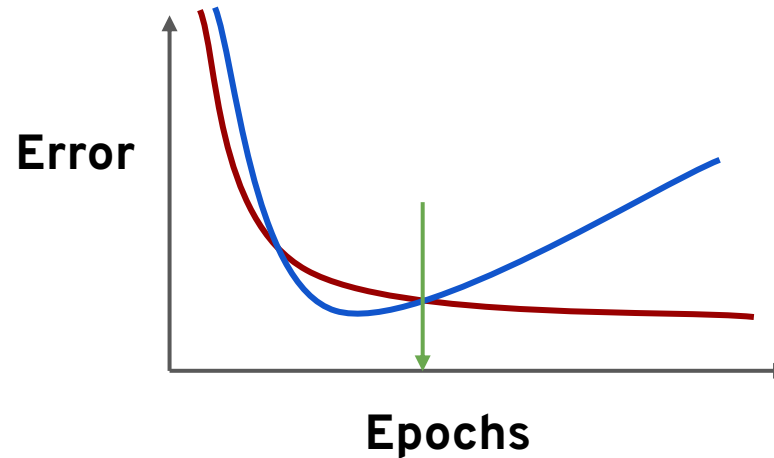
But what happens if we overfit on the training data? That means we would perform poorly on new test data!





# Machine Learning

This is a good indication of training too much on the training data, you should look for the point to cut off training time!



# Machine Learning

- We'll check on this idea again when we actually begin creating models!
- For now just be aware of this possible issue!

# Evaluating Performance - Classification



**Foundations**

# Model Evaluation

- We just learned that after our machine learning process is complete, we will use performance metrics to evaluate how our model did.
- Let's discuss classification metrics in more detail!

# Model Evaluation

The key classification metrics we need to understand are:

- Accuracy
- Recall
- Precision
- F1-Score

# Model Evaluation

But first, we should understand the reasoning behind these metrics and how they will actually work in the real world!

# Model Evaluation

Typically in any classification task your model can only achieve two results:

- Either your model was **correct** in its prediction.
- Or your model was **incorrect** in its prediction.

# Model Evaluation

- Fortunately incorrect vs correct expands to situations where you have multiple classes.
- For the purposes of explaining the metrics, let's imagine a **binary classification** situation, where we only have two available classes.



# Model Evaluation

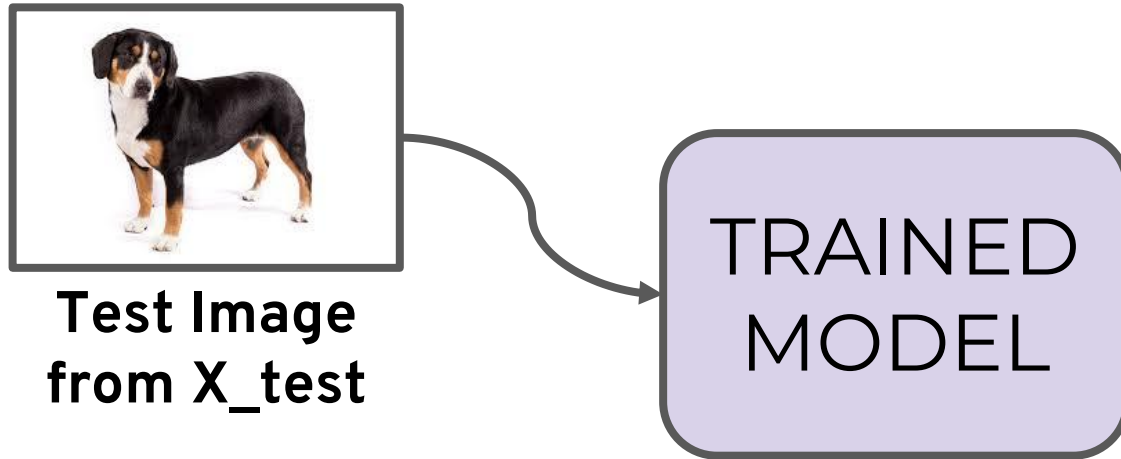
- In our example, we will attempt to predict if an image is a dog or a cat.
- Since this is supervised learning, we will first **fit/train** a model on **training data**, then **test** the model on **testing data**.
- Once we have the model's predictions from the **X\_test** data, we compare it to the **true y values** (the correct labels).

# Model Evaluation



TRAINED  
MODEL

# Model Evaluation



# Model Evaluation



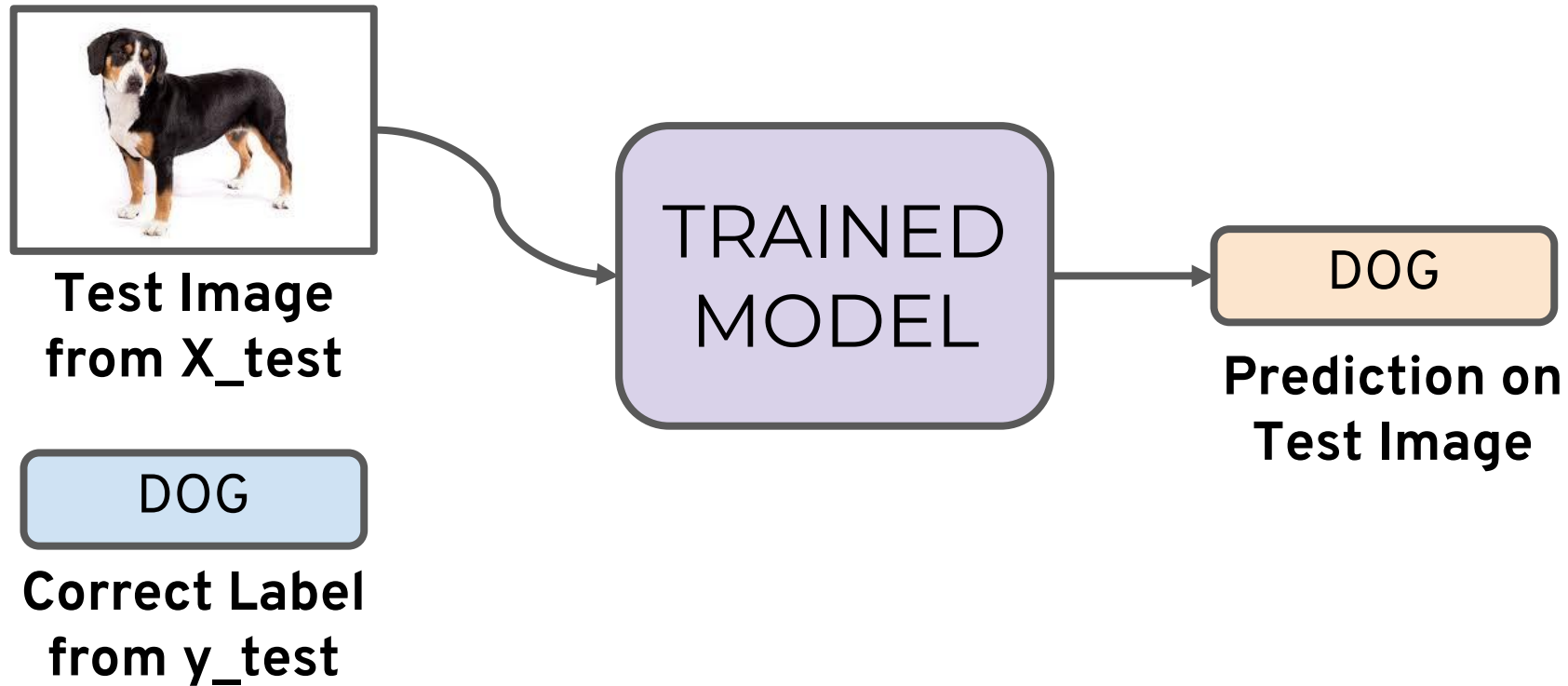
**Test Image  
from  $X_{\text{test}}$**

DOG

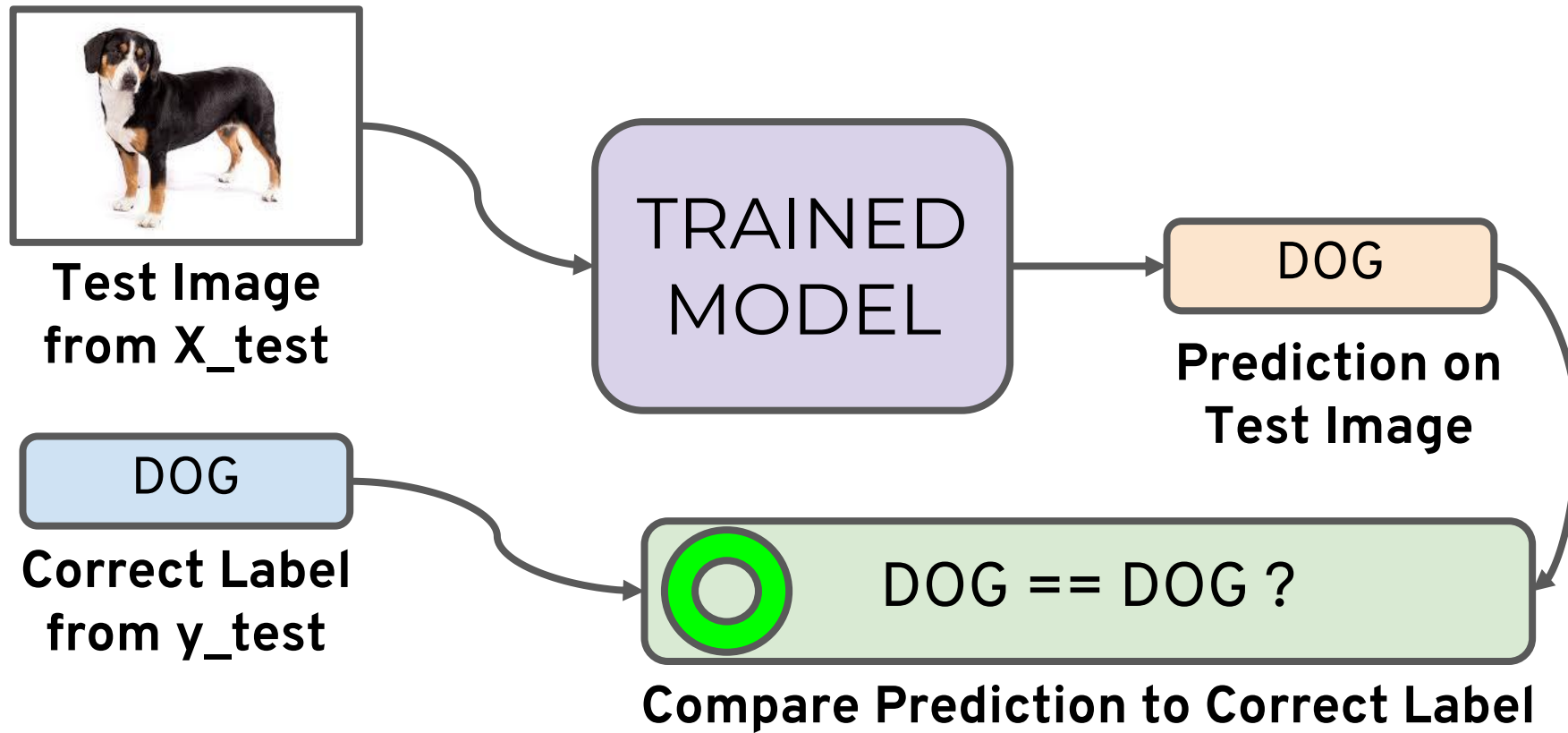
**Correct Label  
from  $y_{\text{test}}$**

TRAINED  
MODEL

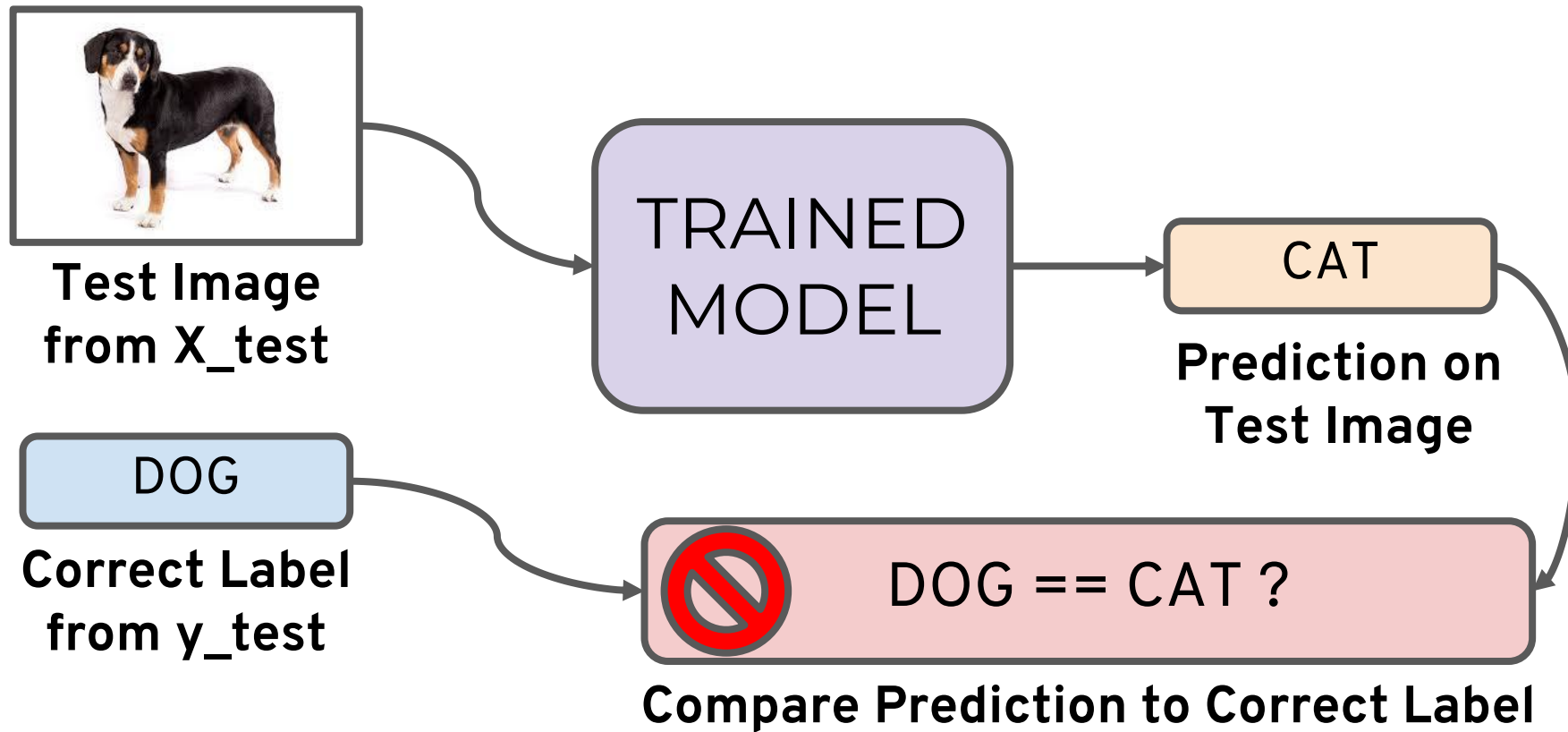
# Model Evaluation



# Model Evaluation



# Model Evaluation



# Model Evaluation

- We repeat this process for all the images in our  $X$  test data.
- At the end we will have a count of correct matches and a count of incorrect matches.
- The key realization we need to make, is that in the real world, not all incorrect or correct matches hold equal value!



# Model Evaluation

- Also in the real world, a single metric won't tell the complete story!
- To understand all of this, let's bring back the 4 metrics we mentioned and see how they are calculated.
- We could organize our predicted values compared to the real values in a confusion matrix.

# Model Evaluation

## Accuracy

- Accuracy in classification problems is the number of correct predictions made by the model divided by the total number of predictions.

# Model Evaluation

## Accuracy

- For example, if the  $X_{\text{test}}$  set was 100 images and our model **correctly** predicted 80 images, then we have **80/100**.
- **0.8 or 80% accuracy.**

# Model Evaluation

## Accuracy

- Accuracy is useful when target classes are well balanced
- In our example, we would have roughly the same amount of cat images as we have dog images.

# Model Evaluation

## Accuracy

- Accuracy is **not** a good choice with **unbalanced** classes!
- Imagine we had 99 images of dogs and 1 image of a cat.
- If our model was simply a line that always predicted **dog** we would get 99% accuracy!

# Model Evaluation

## Accuracy

- Imagine we had 99 images of dogs and 1 image of a cat.
- If our model was simply a line that always predicted **dog** we would get 99% accuracy!
- In this situation we'll want to understand **recall** and **precision**

# Model Evaluation

## Recall

- Ability of a model to find all the relevant cases within a dataset.
- The precise definition of recall is the number of true positives **divided by** the number of true positives plus the number of false negatives.

# Model Evaluation

## Precision

- Ability of a classification model to identify only the relevant data points.
- Precision is defined as the number of true positives divided by the number of true positives plus the number of false positives.



# Model Evaluation

## Recall and Precision

- Often you have a trade-off between Recall and Precision.
- While recall expresses the ability to find all relevant instances in a dataset, precision expresses the proportion of the data points our model says was relevant actually were relevant.

# Model Evaluation

## F1-Score

- In cases where we want to find an optimal blend of precision and recall we can combine the two metrics using what is called the F1 score.

# Model Evaluation

## F1-Score

- The F1 score is the harmonic mean of precision and recall taking both metrics into account in the following equation:

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

# Model Evaluation

## F1-Score

- We use the harmonic mean instead of a simple average because it punishes extreme values.
- A classifier with a precision of 1.0 and a recall of 0.0 has a simple average of 0.5 but an F1 score of 0.

# Model Evaluation

We can also view all correctly classified versus incorrectly classified images in the form of a confusion matrix.

# Confusion Matrix

		predicted condition	
total population		prediction positive	prediction negative
true condition	condition positive	<b>True Positive (TP)</b>	<b>False Negative (FN)</b> (type II error)
	condition negative	<b>False Positive (FP)</b> (Type I error)	<b>True Negative (TN)</b>

# Confusion Matrix

		predicted condition			
		total population	prediction positive	prediction negative	Prevalence $= \frac{\Sigma \text{ condition positive}}{\Sigma \text{ total population}}$
true condition	condition positive	True Positive (TP)	False Negative (FN) (type II error)	True Positive Rate (TPR), Sensitivity, Recall, Probability of Detection $= \frac{\Sigma \text{ TP}}{\Sigma \text{ condition positive}}$	
	condition negative	False Positive (FP) (Type I error)	True Negative (TN)	False Positive Rate (FPR), Fall-out, Probability of False Alarm $= \frac{\Sigma \text{ FP}}{\Sigma \text{ condition negative}}$	
		Positive Predictive Value (PPV), Precision $= \frac{\Sigma \text{ TP}}{\Sigma \text{ prediction positive}}$	False Omission Rate (FOR) $= \frac{\Sigma \text{ FN}}{\Sigma \text{ prediction negative}}$	Positive Likelihood Ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	
		Accuracy $= \frac{\Sigma \text{ TP} + \Sigma \text{ TN}}{\Sigma \text{ total population}}$	False Discovery Rate (FDR) $= \frac{\Sigma \text{ FP}}{\Sigma \text{ prediction positive}}$	Negative Predictive Value (NPV) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ prediction negative}}$	Negative Likelihood Ratio (LR-) $= \frac{\text{FNR}}{\text{TNR}}$

# Model Evaluation

- The main point to remember with the confusion matrix and the various calculated metrics is that they are all fundamentally ways of comparing the predicted values versus the true values.
- What constitutes “good” metrics, will really depend on the specific situation!



# Model Evaluation

- Still confused on the confusion matrix?
- No problem! Check out the Wikipedia page for it, it has a really good diagram with all the formulas for all the metrics.
- Throughout the training, we'll usually just print out metrics (e.g. accuracy).

# Model Evaluation

- Let's think back on this idea of:
  - What is a good enough accuracy?
- This all depends on the context of the situation!
- Did you create a model to predict presence of a disease?
- Is the disease presence well balanced in the general population? (Probably not!)

# Model Evaluation

- Often models are used as quick diagnostic tests to have **before** having a more invasive test (e.g. getting urine test before getting a biopsy)
- We also need to consider what is at stake!

# Model Evaluation

- Often we have a precision/recall trade off, We need to decide if the model will should focus on fixing False Positives vs. False Negatives.
- In disease diagnosis, it is probably better to go in the direction of False positives, so we make sure we correctly classify as many cases of disease as possible!

# Model Evaluation

- All of this is to say, machine learning is not performed in a “vacuum”, but instead a collaborative process where we should consult with experts in the domain (e.g. medical doctors)

# Evaluating Performance - Regression

Foundations



# Evaluating Regression

- Let's take a moment now to discuss evaluating Regression Models
- Regression is a task when a model attempts to predict continuous values (unlike categorical values, which is classification)

# Evaluating Regression

- You may have heard of some evaluation metrics like accuracy or recall.
- These sort of metrics aren't useful for regression problems, we need metrics designed for **continuous** values!



# Evaluating Regression

- For example, attempting to predict the price of a house given its features is a **regression task**.
- Attempting to predict the country a house is in given its features would be a classification task.

# Evaluating Regression

**Let's discuss some of the most common evaluation metrics for regression:**

- Mean Absolute Error
- Mean Squared Error
- Root Mean Square Error

# Evaluating Regression

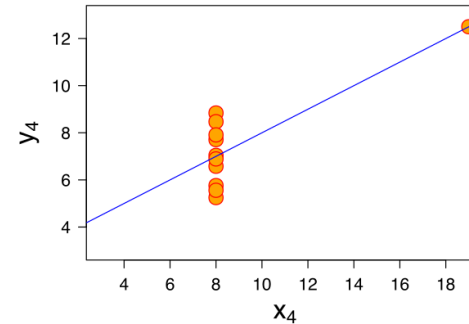
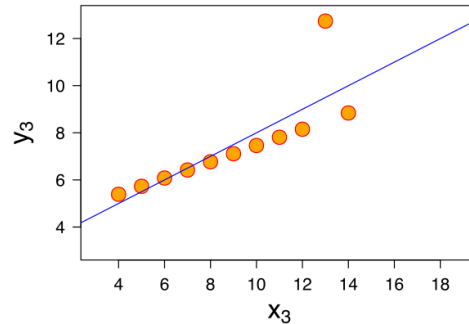
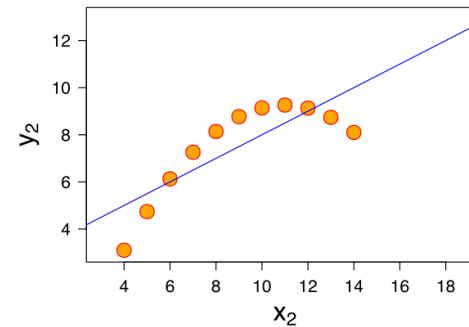
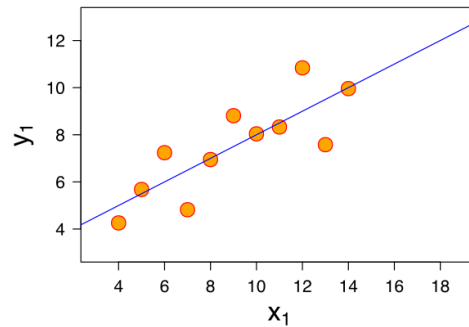
## Mean Absolute Error (MAE)

- This is the mean of the absolute value of errors.
- Easy to understand

$$\frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

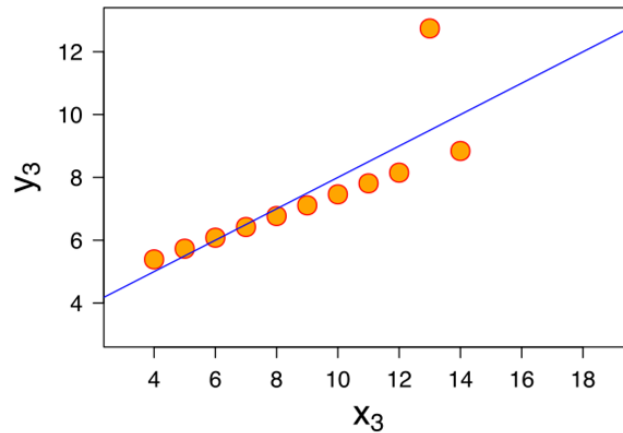
# Evaluating Regression

MAE won't punish large errors however.



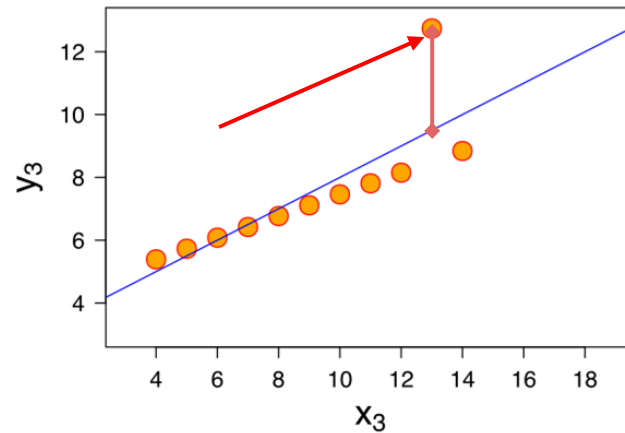
# Evaluating Regression

MAE won't punish large errors however.



# Evaluating Regression

We want our error metrics to account for these!



# Evaluating Regression

## Mean Squared Error (MSE)

- This is the mean of the squared errors.
- Larger errors are noted more than with MAE, making MSE more popular.

$$\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

# Evaluating Regression

## Root Mean Square Error (RMSE)

- This is the root of the mean of the squared errors.
- Most popular (has same units as y)

$$\sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$



# Machine Learning

- Most common question from students:
  - “Is this value of RMSE good?”
- Context is everything!
- A RMSE of \$10 is fantastic for predicting the price of a house, but horrible for predicting the price of a candy bar!

# Machine Learning

- Compare your error metric to the average value of the label in your data set to try to get an intuition of its overall performance.
- Domain knowledge also plays an important role here!

# Machine Learning

- Context of importance is also necessary to consider.
- We may create a model to predict how much medication to give, in which case small fluctuations in RMSE may actually be very significant.

# Machine Learning

You should now feel comfortable with the various methods of evaluating a regression task.

# Unsupervised Learning

Foundations



# Machine Learning

- We've covered supervised learning, where the **label was known** due to **historical labeled data**.
- But what happens when we don't have historical labels?

# Machine Learning

There are certain tasks that fall under unsupervised learning:

- Clustering
- Anomaly Detection
- Dimensionality Reduction

# Machine Learning

## Clustering

- Grouping together **unlabeled** data points into categories / clusters
- Data points are assigned to a cluster based on similarity



# Machine Learning

## Anomaly Detection

- Attempts to detect outliers in a dataset
- For example, fraudulent transactions on a credit card.

# Machine Learning

## Dimensionality Reduction

- Data processing techniques that reduces the number of features in a data set, either for compression, or to better understand underlying trends within a data set.

# Machine Learning

## Unsupervised Learning

- It's important to note, these are situations where we **don't** have the correct answer for historical data!
- Which means evaluation is much harder and more nuanced!

# Unsupervised Process



# Machine Learning

Later on in the course, we'll explore unsupervised learning processes with specialized neural network structures, such as autoencoders.