
MasonML

Statistical Foundations and Linear Methods

Samuel Schmidgall

Mason Machine Learning VP of Research

Research Project Topics

Samuel Schmidgall (Theoretical Focus)

- Deep Generative Models (ft. Emily Pho)
- Motion Planning using Deep Reinforcement Learning

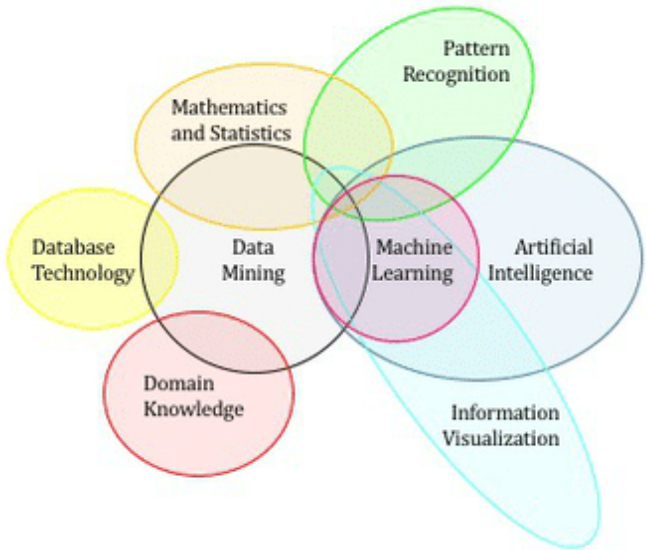
Albert Lam

- Quantitative Finance

Carlos Guerra

- Dimensionality Reduction

What is machine learning?



Statistical Learning	Machine Learning
Computational Statistics	Pattern Recognition
Data Science	Data Mining
Artificial Intelligence	Statistics
Big Data	Decision Theory

What is machine learning?

Machine learning is the scientific study of algorithms and statistical models that computer systems use to effectively perform a specific task without using explicit instructions, relying on models and inference instead.

- Machine Learning is largely a sub-field of Computer Science that evolved from Artificial Intelligence.
- The main focus of Machine Learning is on prediction, rather than inference.

What is machine learning?

Formal Definition of Learning: A computer program is said to learn from experience **E** with respect to some class of tasks **T** and performance measure **P**, if its performance at tasks in **T**, as measured by **P**, improves with experience **E**.

In essence, we're assuming that there is an underlying function f(x) for our data, and the goal of the game is to find the $\hat{f}(x)$ that best approximates f(x).

What is machine learning?

Supervised: Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.

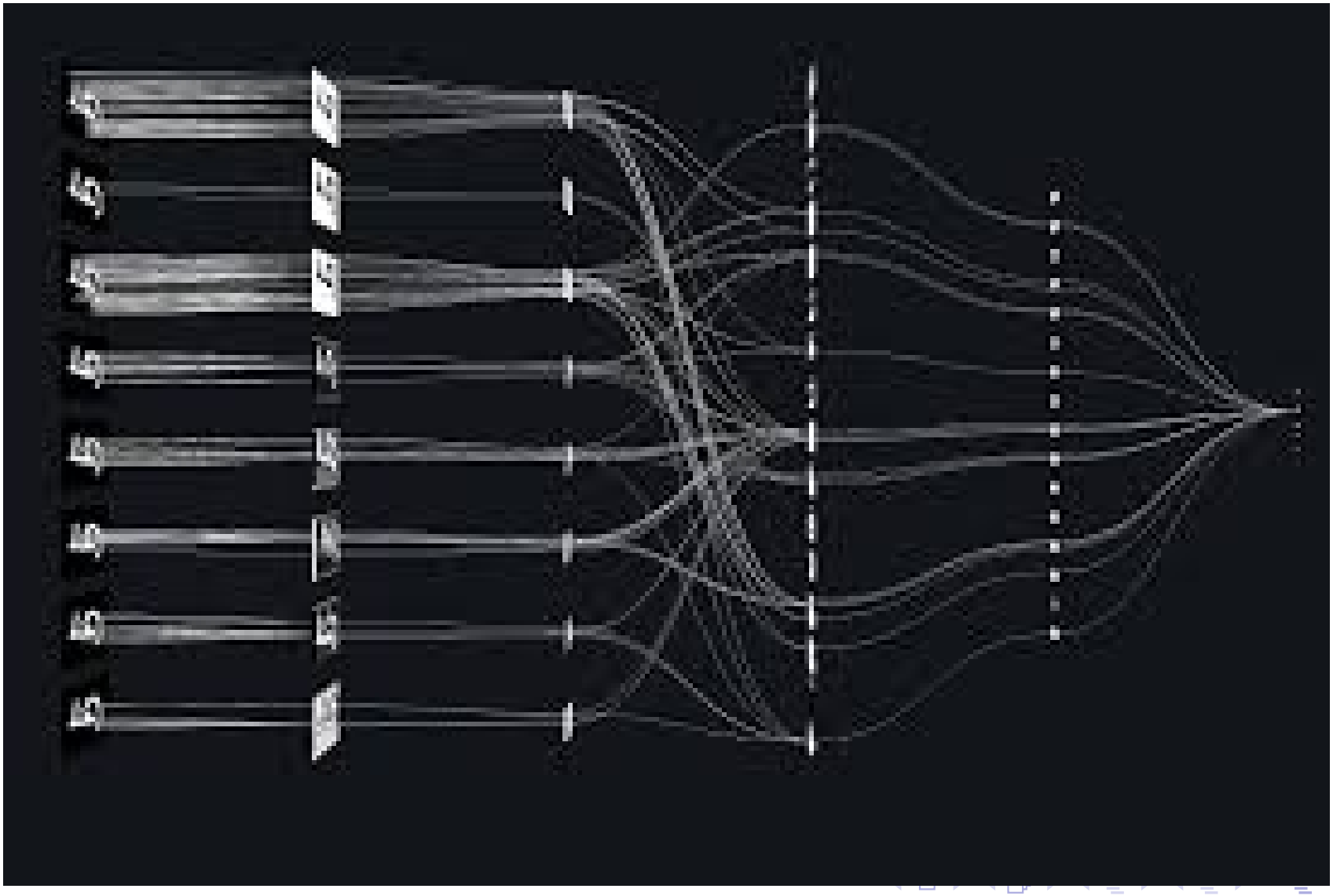
Unsupervised: Unsupervised learning is a branch of machine learning that learns from test data that has not been labeled, classified or categorized.

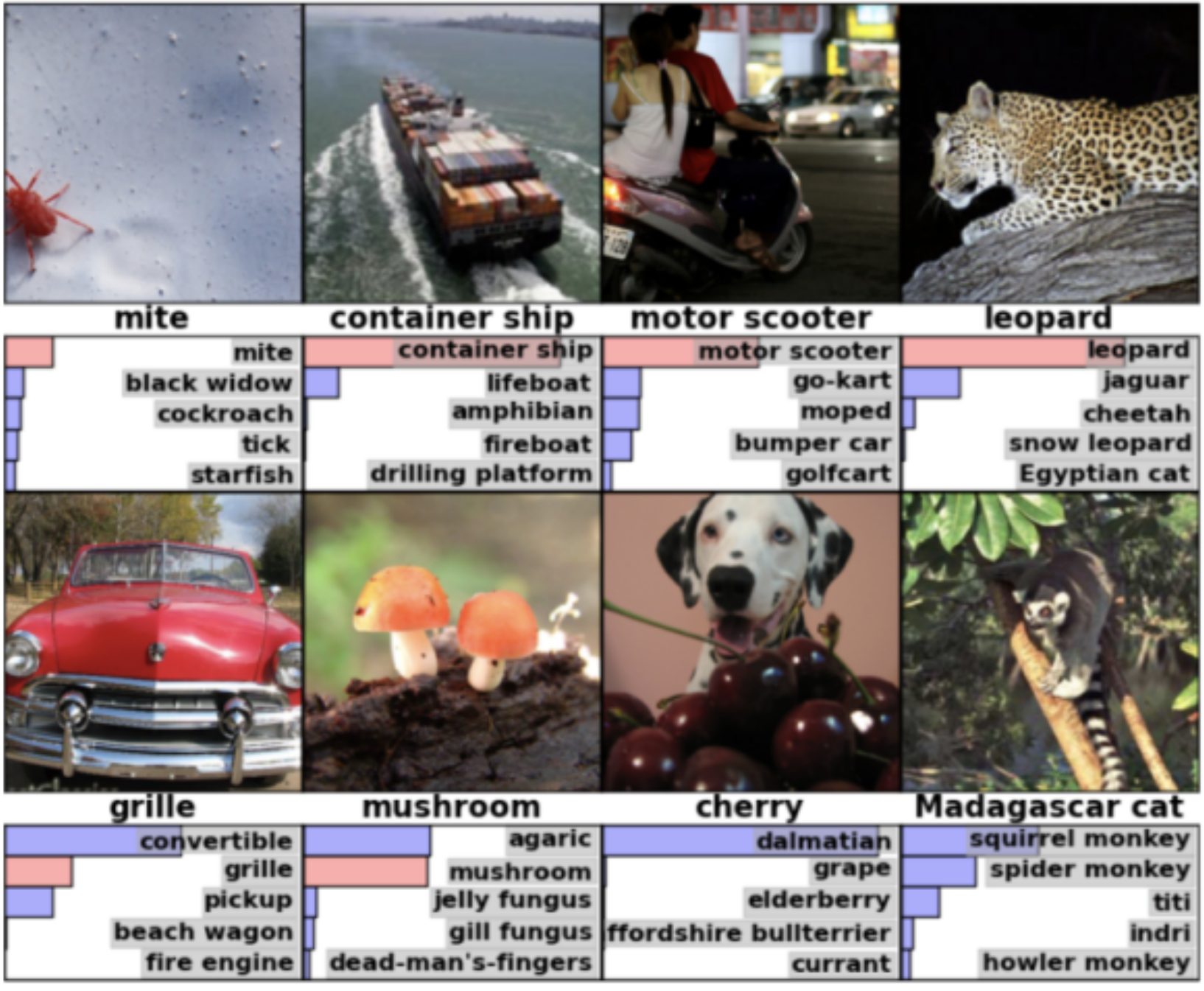
Reinforcement:Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward.

Today we're going to be mostly talking about supervised learning.

Application of Supervised Learning

Have you ever heard of a Neural Network? Neural Networks are the most fun trophy-case example of supervised learning.



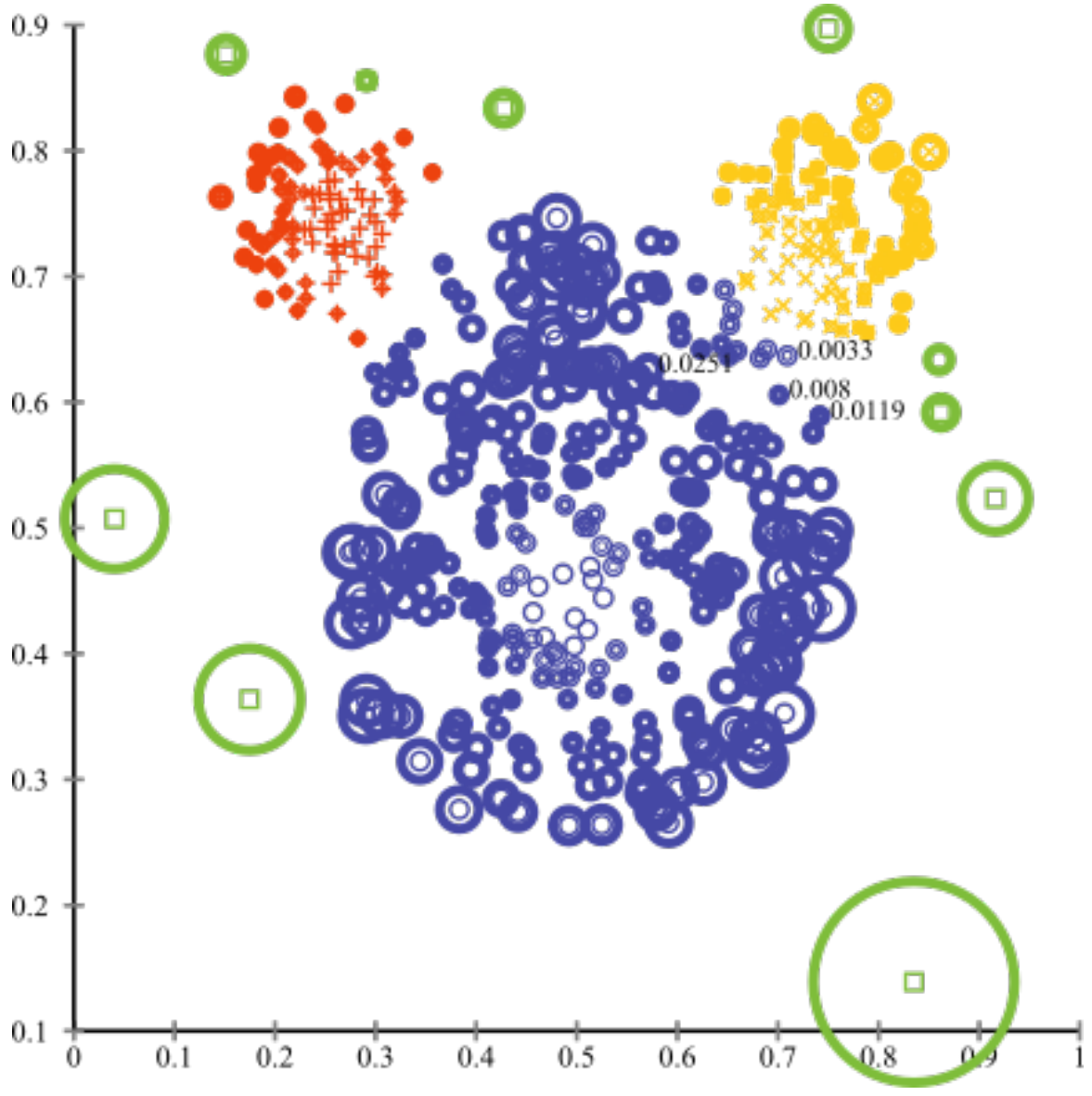

mite
mite
black widow
cockroach
tick
starfish
container ship
container ship
lifeboat
amphibian
fireboat
drilling platform
motor scooter
motor scooter
go-kart
moped
bumper car
golfcart
leopard
leopard
jaguar
cheetah
snow leopard
Egyptian cat
grille
convertible
grille
pickup
beach wagon
fire engine
mushroom
agaric
mushroom
jelly fungus
gill fungus
dead-man's-fingers
cherry
dalmatian
grape
elderberry
ffordshire bullterrier
currant
Madagascar cat
squirrel monkey
spider monkey
titi
indri
howler monkey

Unsupervised Learning

Considering unsupervised learning has no labeled data, how can we do anything with it?

One application is *Cluster Analysis*. Clustering is the task of dividing the population or data points into a number of groups such that data points in the same groups are more similar to other data points in the same group and dissimilar to the data points in other groups.

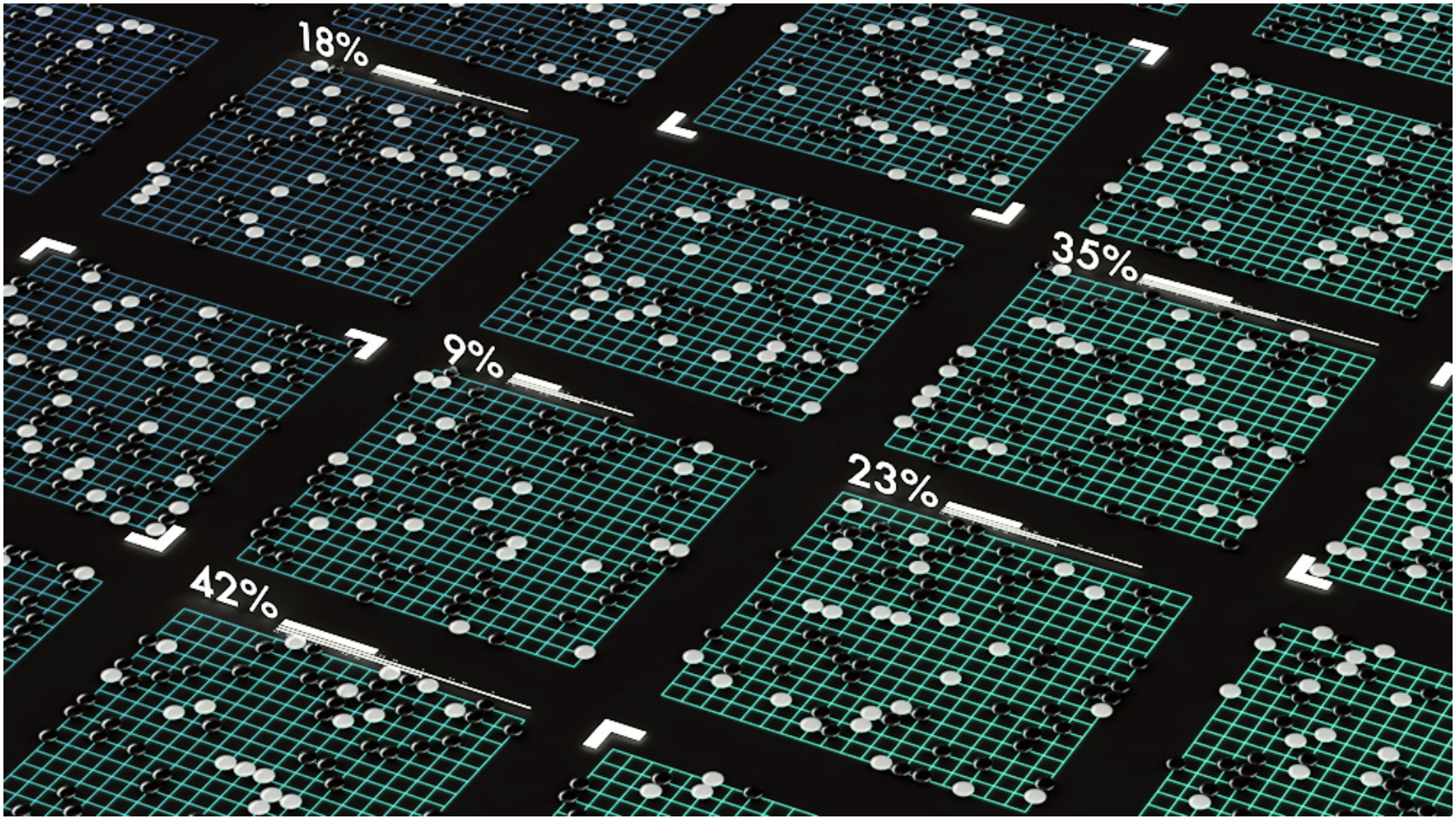
There is an underlying assumption that data points that are similar will be closer together in P-space.


0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
0.0251
0.0033
0.008
0.0119

Reinforcement Learning

Reinforcement Learning is typically used in cases where there is no explicitly labeled data, but the model is able to found out whether or not a sequence of actions lead to positive or negative results.

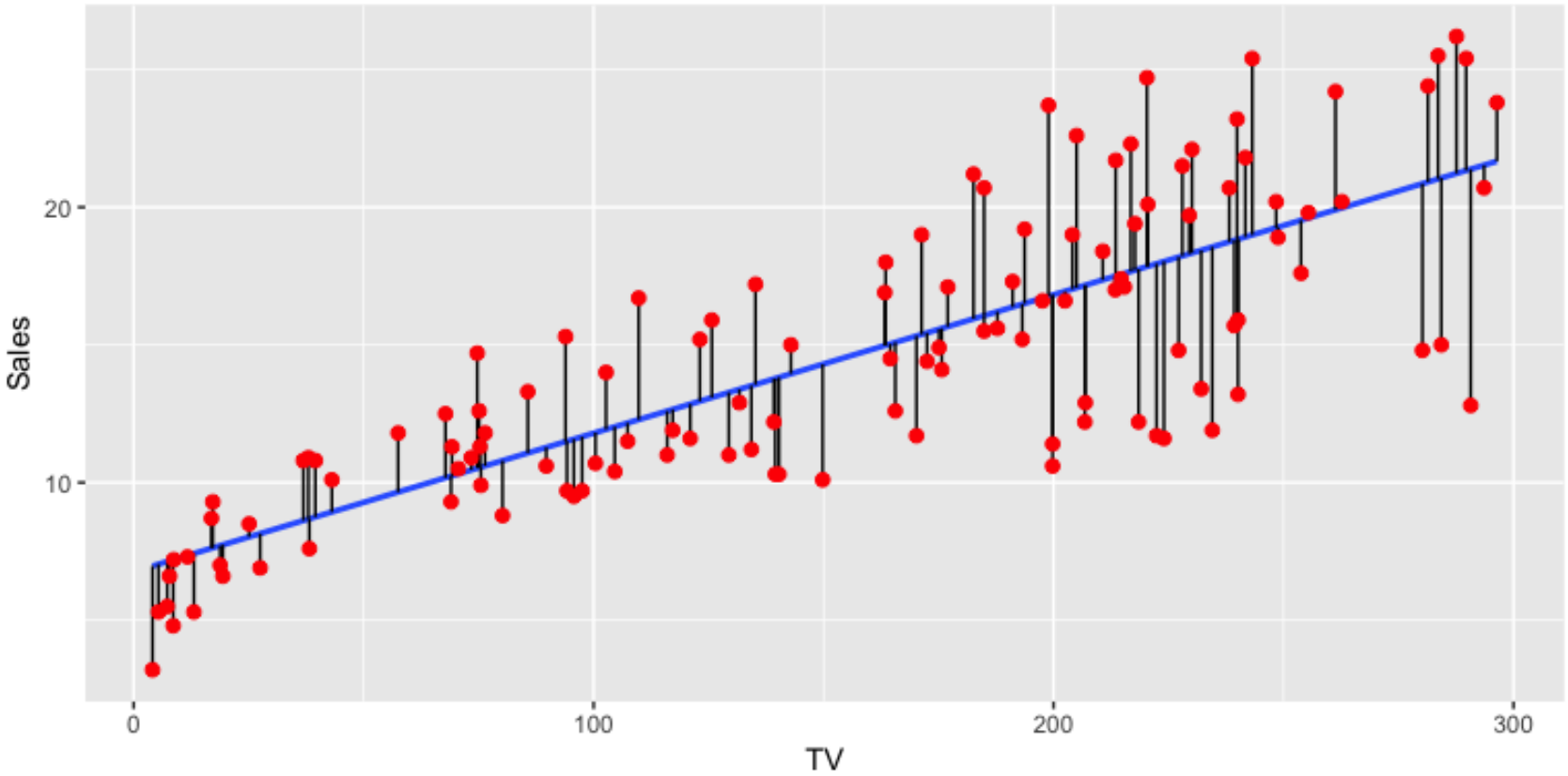
It is most common to see Reinforcement Learning used in Robotics and Game AI (AlphaGO).


18%
35%
9%
23%
42%

Supervised Learning Basics

Let's take a look at some basic Supervised Learning models.

Namely, Linear Regression



Simple Approach to Prediction (Linear Regression)

Given a vector of inputs $\mathbf{X}^T = [x_1, x_2, ..., x_P]$

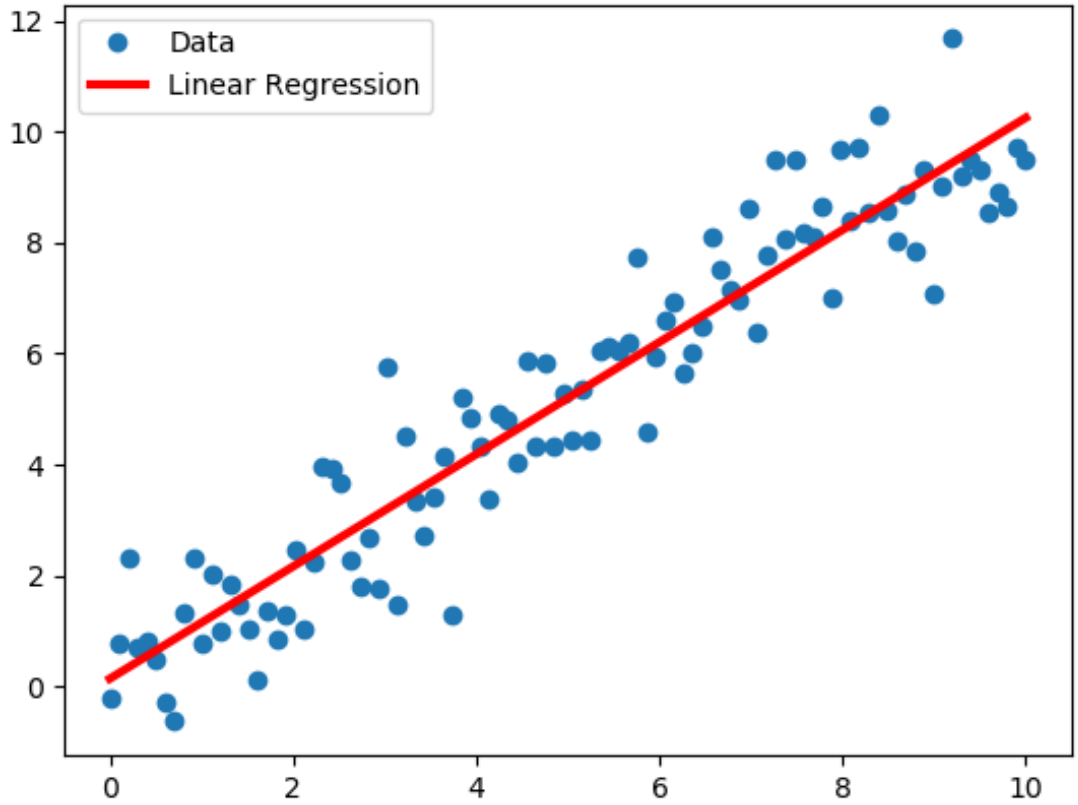
Linear Model: $\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} x_j \hat{\beta}_j$

Meaning that $\hat{\beta}$ is a P dimensional vector with each variable $\hat{\beta}_i$ acting as a multiplier corresponding to a specific variable.

And if we include the constant 1 in $\mathbf{X}$, we can include $\hat{\beta}_0$ in the vector of coefficients $\hat{\beta}$ and write the Linear Model in vector form as:

$$\hat{Y} = \mathbf{X}^T \hat{\beta}$$

Simple Approach to Prediction (Linear Regression)



So, how do we fit the Linear Model to our data?

What will give us the best accuracy?

Simple Approach to Prediction (Linear Regression)

Typically, the method of *least squares* is used.

In this approach we pick the coefficients β to minimize the residual sum of squares:

$$RSS(\beta) = \sum_{i=1}^{N}(y_i - x_i^T\beta)^2$$

Take a moment to think about why we are summing *squares* where x is an NxP Matrix.

Why wouldn't we try and minimize:

$$\sum_{i=1}^{N}|y_i - x_i^T\beta|$$

or

$$\sum_{i=1}^{N}(y_i - x_i^T\beta)^n, n \in > 2\ ?$$

Discuss it with the people around you.

Simple Approach to Prediction (Linear Regression)

The reason that we used the sum of *squares* is because $RSS(\beta)$ is a quadratic function of the parameters, and hence its minimum **always** exists.

To show this, lets think of $RSS(\beta)$ in matrix notation.

$$RRS(\beta) = (y - \mathbf{X}\beta)^T(y - \mathbf{X}\beta)$$

Try to solve for β.

Simple Approach to Prediction (Linear Regression)

Since we are trying to minimize β

$$\frac{\partial RSS}{\partial \beta} = -2\mathbf{X}^T(y - \mathbf{X}\beta)$$

$$-2\mathbf{X}^T(y - \mathbf{X}\beta) = 0$$

$$\mathbf{X}^T y - \mathbf{X}^T\mathbf{X}\beta = 0$$

$$\mathbf{X}^T\mathbf{X}\beta = \mathbf{X}^T y$$

$$\hat{\beta} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T y$$

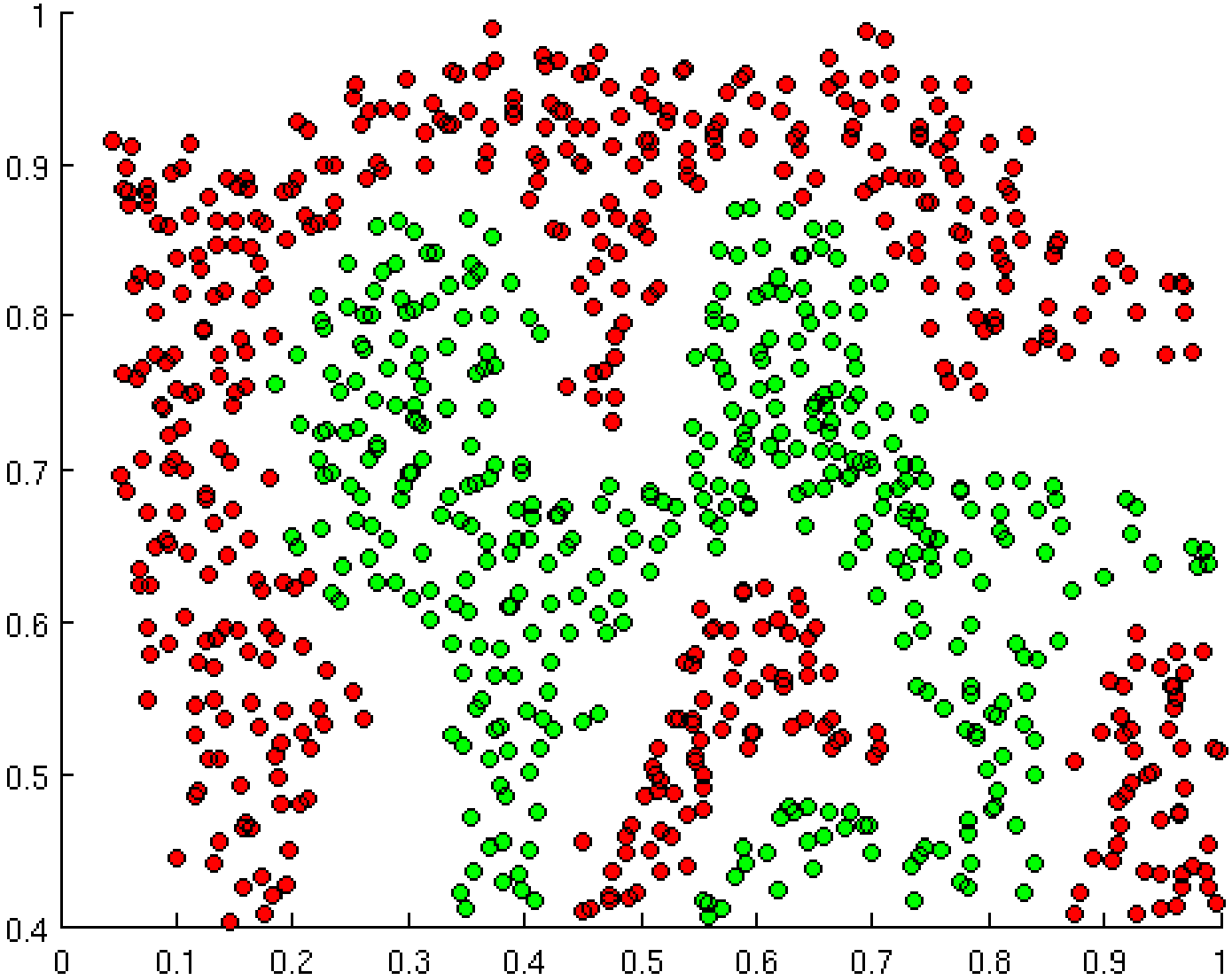
Parametric vs Non-Parametric Models

Linear Regression is considered a **parametric model**. A parametric model assumes that our function f(x) has a finite set of parameters θ.

In the case of Linear Regression, we assumed that there exists a θ of order p, where p is the dimensionality of our input vector, that can approximate the underlying function f(x).

In short, we assumed that the function f(x) was *linear*.

Parametric vs Non-Parametric Models



Parametric vs Non-Parametric Models

Non-Parametric Models assume that the data distribution cannot be defined in terms of a finite set of parameters. But they can often be defined by assuming an infinite dimensional θ.

Nearest Neighbor Method:

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i$$

Where $N_k(x)$ is the neighborhood defined by the k closest points x_i in the training samples (User specifies the k value).

Interactive KNN

K Nearest Neighbors

One fun fact about K Nearest Neighbors is that it is a universal approximator, meaning that as:
$N, k \to \infty$ such that $k/N \to 0, \hat{f}(x) \to E(Y|X = x)$

So, why would we ever use something else? We have a perfect approximation function?

K Nearest Neighbors

For the most part, we often do not have large enough samples. If the linear regression model, or some more structured model is appropriate, then we usually get a more stable estimate than K-Nearest Neighbors.

Bias-Variance Tradeoff

Now that we have seen two very different strategies for function approximation, let's see what the trade-offs are. Keep in mind that this is for unseen data.

$$EPE_k(x_0) = E[(Y - \hat{f}_k(x_0))^2 | X = x_0]$$

$$= Var(\epsilon) + [Bias^2(\hat{f}_k(x_0)) + Var(\hat{f}_k(x_0))]$$

Bias-Variance Tradeoff

$$Var(\epsilon) + [Bias^2(\hat{f}_k(x_0)) + Var(\hat{f}_k(x_0))]$$

There are three terms in this expression. The first term $Var(\epsilon)$ is the irreducible error, the variance of the new test target-and is beyond our control even if we know the true f(x_0)

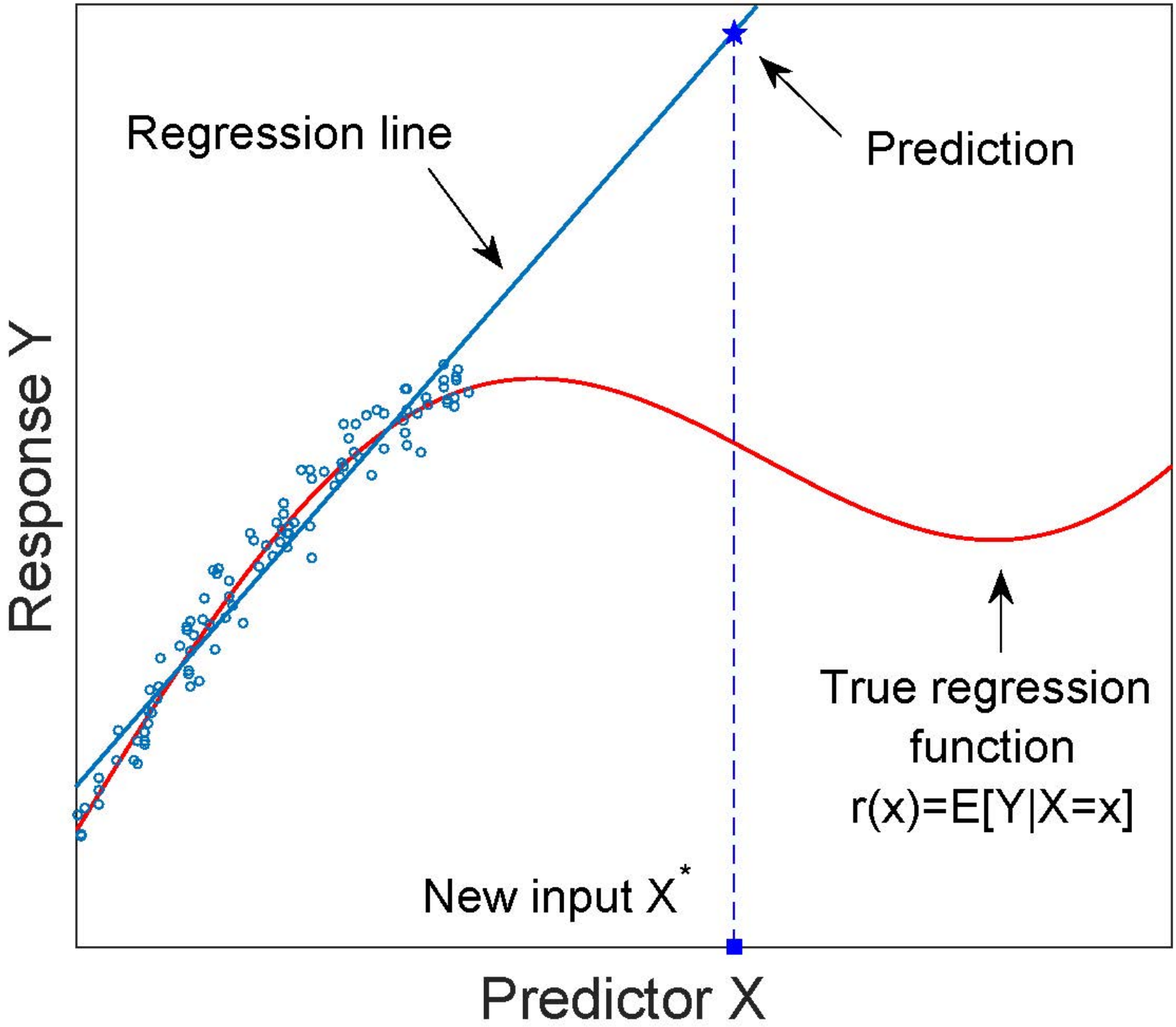
Luckily, the second and third terms are under our control, and make up the *mean squared error* of $\hat{f}(x_0)$, which is broken down into a bias component and a variance component.

Bias

The bias term, $Bias^2(\hat{f}_k(x_0))$, is the squared difference between the true mean f(x_0) and the expected value of the estimate, where the expectation averages the randomness in the training data.

More intuitively, the bias is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).

e.g. This could happen if we try and fit a linear regression model to highly nonlinear data.


Regression line
Prediction
Response Y
True regression
function
r(x)=E[Y|X=x]
New input X*
Predictor X

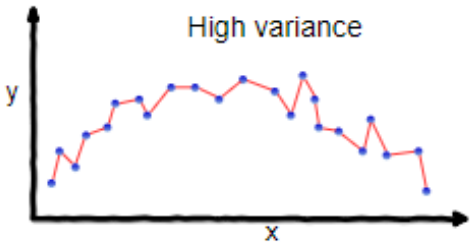
Variance

The variance is an error from sensitivity to small fluctuations in the training set. High variance can cause an algorithm to model the random noise in the training data, rather than the intended outputs (overfitting).

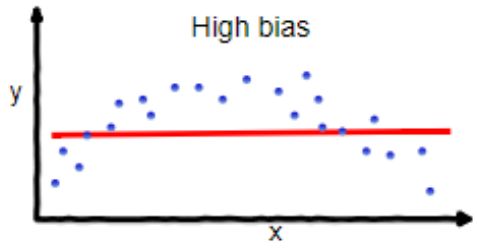
You could imagine that if we had a K-Nearest Neighbor model where K=1 (We choose the closest single neighbor) and we had data with high variance, such as a inexpensive thermometer, our data will misclassify a lot of new data because most of the data in our model is incorrect.

Often times people will call models that over fit easily *flexible* models.

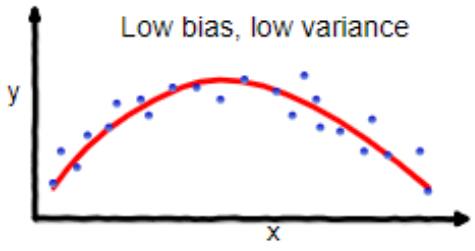
Bias-Variance Tradeoff



overfitting



underfitting

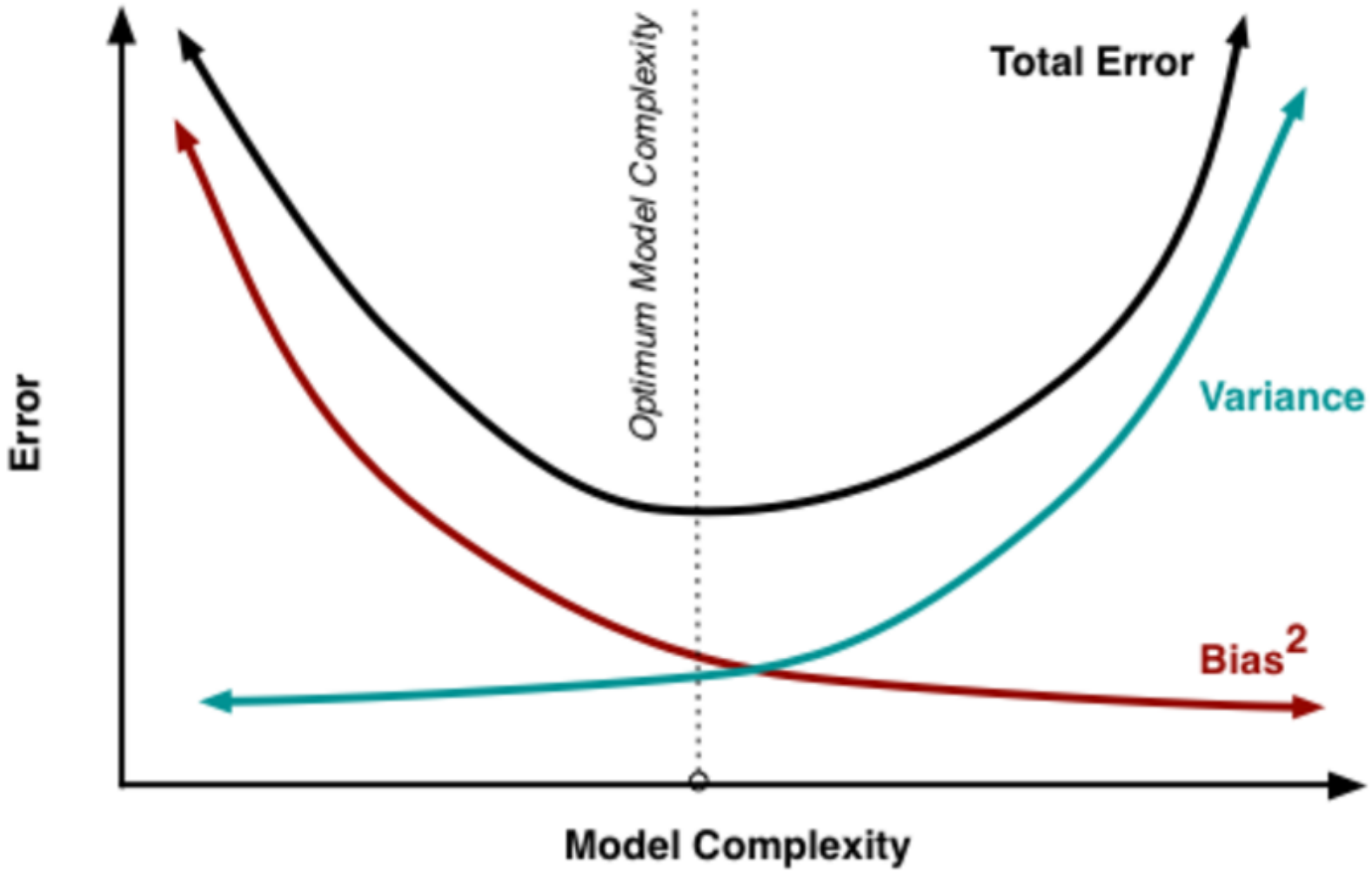


Good balance

We can see that as bias increases, variance decreases. Likewise, as variance increases, bias decreases.

Considering that, there will always exist and optimal bias and variance minimization point.

Bias-Variance Tradeoff



Why do you think that the total error never reaches zero?

Discuss with those around you.

Bias-Variance Tradeoff

The main goal of Machine Learning is to find the point of minimum total error with a computationally efficient model.