

Cloud Guru - 6 - Algorithms

<https://acloud.guru/course/aws-certified-machine-learning-specialty/learn/1efec92d-271f-f097-0c26-236b41ab42b1/f4cd49ae-be82-06a0-7672-7edc64126b7f/watch?backUrl=~2Fcourses&backUrl=~2Fcourses&backUrl=~2Fcourses,~2Fcourses>

Algorithm vs Heuristic

Algorithm	Heuristic
Set of steps to follow to solve a specific problem intended to be repeatable with the same outcome.	A mental short-cut or “rule of thumb” that provides guidance on doing a task but does not guarantee a consistent outcome.

Algorithm = troubleshooting steps to narrow down possibilities to the correct one
Ex: for electrical pb - go through a set of steps to see where the fault is



Heuristic = educating guess

Ex: We try a new pizza place. It could be a good or bad pizza, but we don't know. We are not going to try to see how they make the tough, what is their recipe, match against what we know,... We are willing to take a chance here
We use heuristic a lot in our life



When to use algorithm - trying to **drive out Bias**

Trying to remove assumptions and past experience.

In contrast, computers can't make these assumptions. They don't have that intuition required by a heuristic.

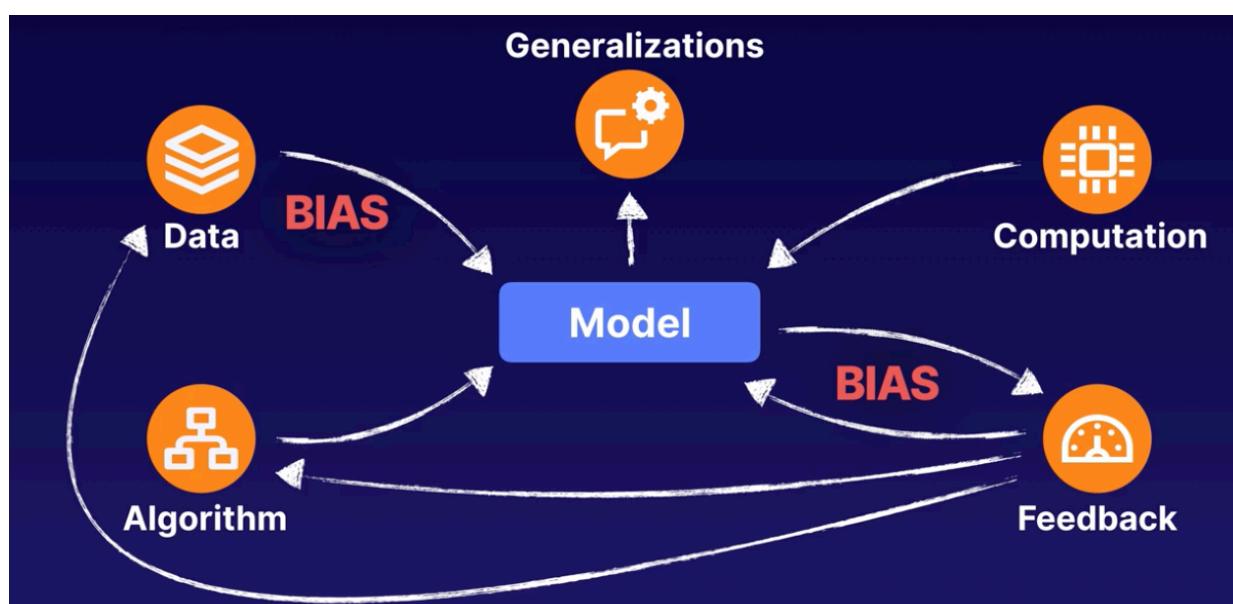
=> **Algorithm** is a **finite process with known input and expected outputs**

We can use this in ML as we want repeatable generalization.

Now **Bias** can still creep in into ML process:

- by data we select when training or testing, or when we exclude a chunk of data
- feedback loop

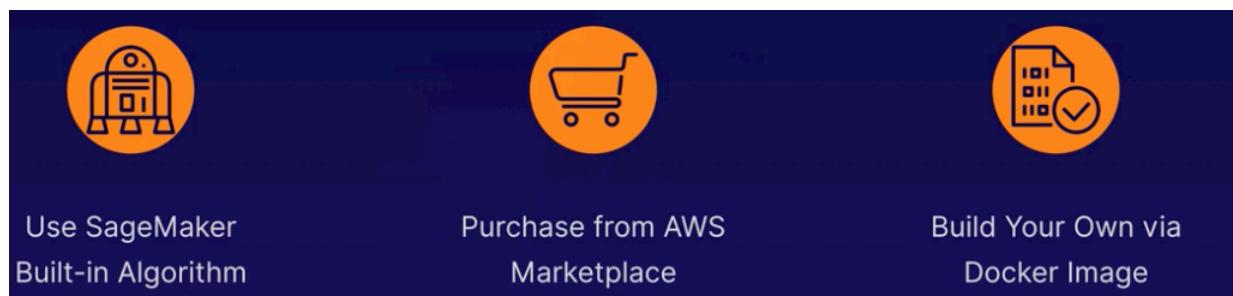
We might skew the model to fulfill our assumptions in the first place



3 types of Learning:

	Supervised Learning	Unsupervised Learning	Reinforcement Learning
Training	Training Data and Testing Data	No Training	How to Maximize Reward
Discrete	Classification	Clustering	Simulation-based Optimization
Continuous	Regression	Reduction of Dimensionality	Autonomous Devices

Algorithms:



Regression

Example: linear equation: $y = 2x + 3$

Given x , we can figure out y and vice-versa

We can plot a line too.

Seeing values on that line, we can estimate/predict what is the y value further down on that line



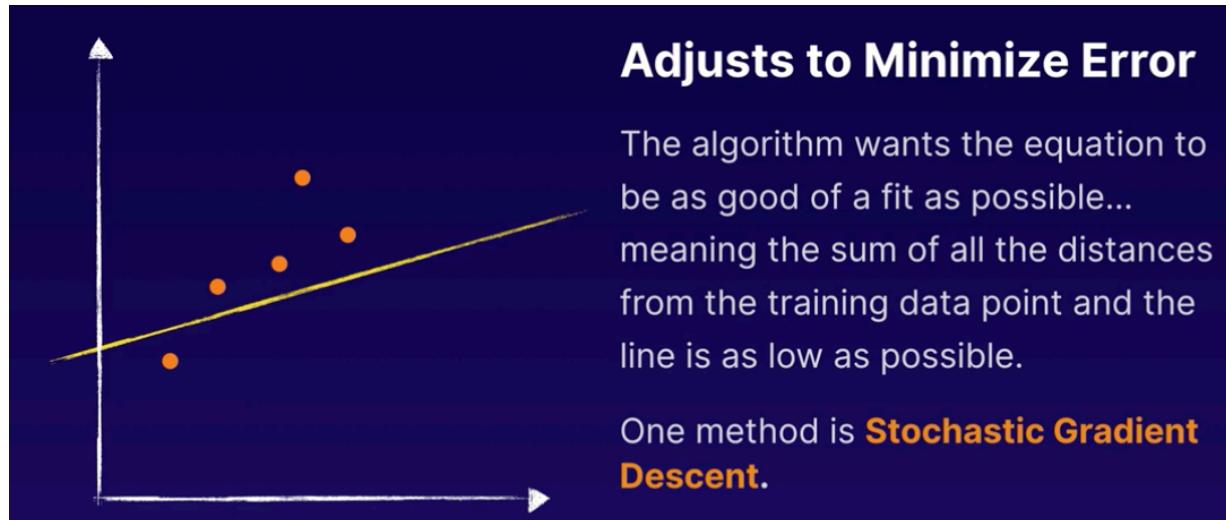
Linear Learner Algorithm



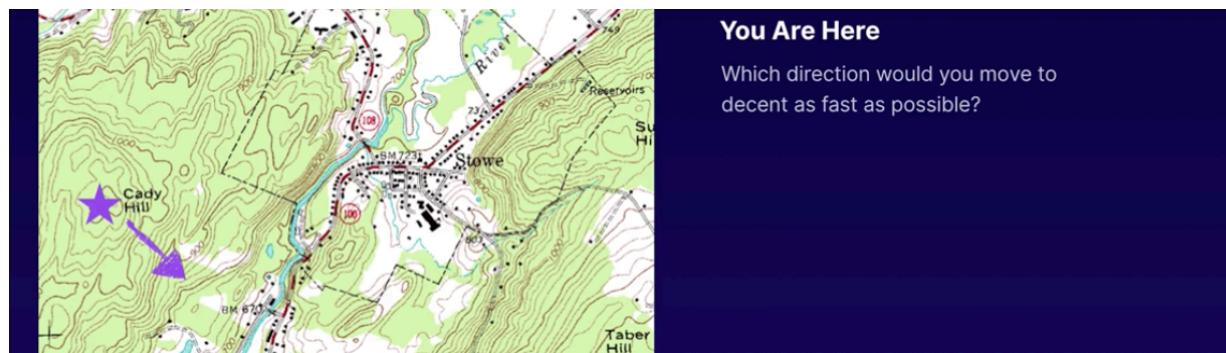
Linear models are supervised learning algorithms for regression, binary classification or multiclass classification problems. You give the model labels (x,y) with x being high-dimensional vector and y is a numeric label. The algorithm learns a linear function, or, for classification problems, a linear threshold function, and maps a vector x to an approximation of label y .



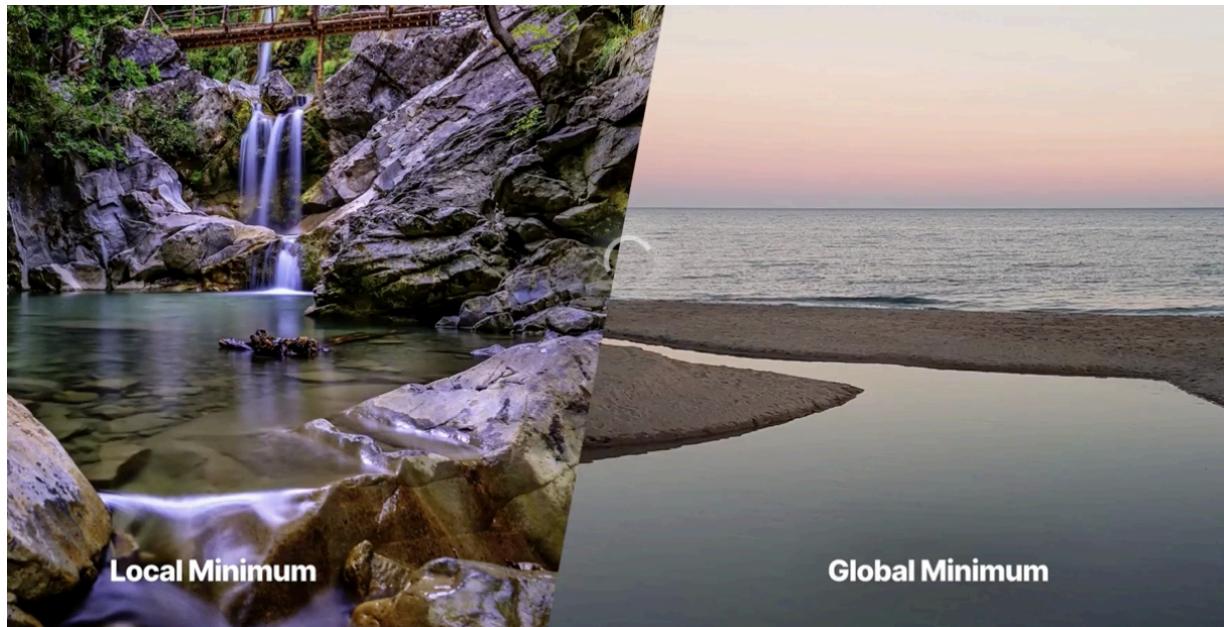
To use this algorithm you need a number or list of numbers which yields some other number...the answer you're after. You can use it to predict a specific value or a threshold for grouping purposes.



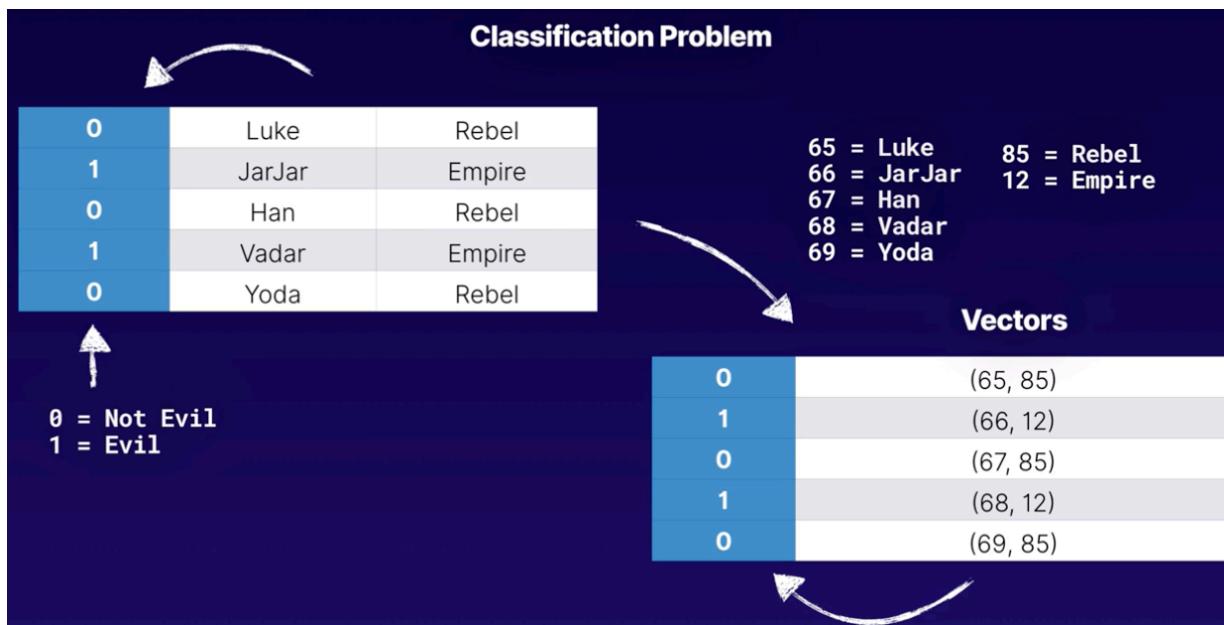
Ex of topographic map to explain Stochastic Gradient Descent
Map is the error.



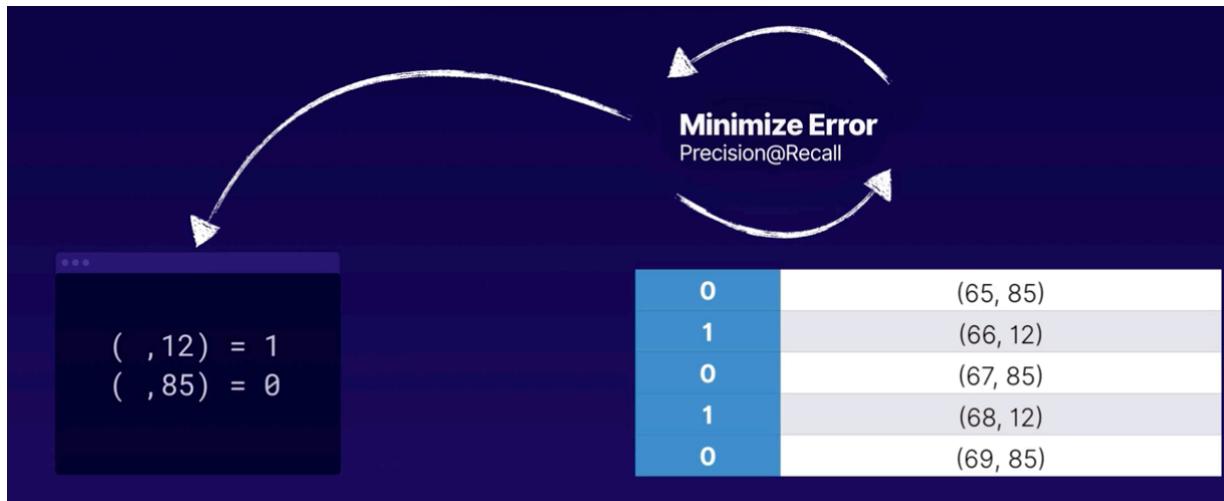
Another idea - idea coming down
Some may be stuck in pools, some may make their way all the way to the ocean



For linear regression, we need numbers -> convert the data and end up with vectors



Linear learning will learn through trial/error process, is that the 2nd value in the vector is the most accurate to predict outcome, whether somebody is a 0 or 1



1	Very Flexible Linear Learner can be used to explore different training objectives and choose the best one. Well suited for discrete or continuous inferences.	Built-in Tuning Linear Learner algorithm has an internal mechanism for tuning hyperparameters separate from the automatic model tuning feature.	Good First Choice If your data and objective meets the requirements, Linear Learner is a good first choice to try for your model.
---	---	---	---

Examples when to use Linear Learner Algorithm - Use Cases:

- **Predict quantitative value based on given numeric input**

Example: Based on last five years ROI from marketing spend, what can we expect to be this year's ROI?

- **Discrete Binary Classification Problems**

Example: Based on past customer response, should I mail this particular customer? Yes or No?

- **Discrete Multiclass Classification Problems**

Example: Based on past customer response, how should I reach this customer? Email, Direct Mail, or Phone Call?

It works well when we have good continuous data.

With sparse dataset, need to adjust this with Factorization Machine

(34, 4, 56, 34, 6, 73, 3, 4, 12, 65, 63...)

Linear Learner



(34, 4, , 34, 6, , 3, , 12, 65, , ...)

Factorization Machines

▲ ▲ ▲ ▲

a sparse dataset



Factorization Machines Algorithm

General purpose supervised learning algorithm for both binary classification and regression. Captures interaction between features with high dimensional sparse datasets.

SUPERVISED



To use this algorithm you need a number or list of numbers which yields some other number...the answer you're after. You can use it to predict a specific value or a threshold for placing into one of two groups. It is a good choice when you have "holes" in your data.

Things to know about Factorization Machines Algorithm

1

Considers only pair-wise features.

Amazon SageMaker's implementation of factorization machines will only analyze relationships of two pairs of features at a time.

4

Really needs LOTS of data.

To make up for the sparseness, needs lots of data. Recommended dimension of the input feature space is between 10,000 and 10,000,000.

2

CSV is not supported.

CSV is not a good choice for sparse dimensions. File and Pipe mode training are supported using recordIO-protobuf format with Float32 tensors.

5

CPUs rock sparse data better.

AWS recommends CPUs with factorization machines for the most efficient experience.

3

Doesn't work for Multiclass Problems.

Factorization Machines algorithm can be run in either binary classification mode or regression mode.

6

Don't perform well on dense data.

Other algorithms are much more performant when you have a full set of data.

Example: Movie recommendation engine
4 movies watched by 4 people.
Nobody has watched all movies —> perfect situation for a recommender engine

Objective: We want to find a movie that Dante is going to like

Dante

(1, , , 0)

0 = Did not like the movie
1 = Liked the movie

Bob

(, 1, , 0)

“Clerks”

Jay

(, , , 1)

“Mallrats”

Veronica

(1, 1, 0,)

“Dogma”

“Clerks 2”

Caitlin

(1, 0, , 1)

=> do one hot encoding that creates a matrix for us to evaluate with movies and people that like them

p_Bob	p_Jay	p_Veronica	p_Caitlin	m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2
1	0	0	0	0	1	0	0
0	1	0	0	0	0	0	1
0	0	1	0	1	0	0	0
0	0	1	0	0	1	0	0
0	0	0	1	1	0	0	0
0	0	0	1	0	0	0	1

1,0,0,0,0,1,0,0
 0,1,0,0,0,0,0,1 X
 0,0,1,0,1,0,0,0
 0,0,1,0,0,1,0,0
 0,0,0,1,1,0,0,0
 0,0,0,1,0,0,0,1

To note if we had 100,000 viewers and 10,000 films, we would end up with a really huge matrix, filled for the most with 0

That is a huge waste and why Factorization Machine does not support CSV => rather they use recordIO protobuf with Float32 Tensor

p_Bob	p_Jay	p_Veronica	p_Caitlin	m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2
1	0	0	0	0	1	0	0
0	1	0	0	0	0	0	1
0	0	1	0	1	0	0	0
0	0	1	0	0	1	0	0
0	0	0	1	1	0	0	0
0	0	0	1	0	0	0	1

p_Dante	m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2
1	1	0	0	0

(1, , ,0)

Start building a profile of Dante's movies preferences vs other's preference, and summing that

p_Bob	p_Jay	p_Veronica	p_Caitlin	m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2		
1	0	0	0	0	1	0	0		
0	1	0	0	0	0	0	1		
0	0	1	0	1	0	0	0		
0	0	1	0	0	1	0	0		
0	0	0	1	1	0	0	0		
0	0	0	1	0	0	0	1		
p_Dante	m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2	Sum	m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2
1	1	0	0	0	-4	-1	1	0	0
						-1	0	0	1
						0	0	0	0
						-1	1	0	0
						0	0	0	0
						-1	0	0	1
						-4	2	0	2

We want anything positive

m_Clerks	m_Mallrats	m_Dogma	m_Clerks_2
-1	1	0	0
-1	0	0	1
0	0	0	0
-1	1	0	0
0	0	0	0
-1	0	0	1
Sum	-4	2	Nope
Dante	(1, , , 0)		
I recommend “Mallrats”.			

Use case for Factorization Machine Algorithm:

- **High Dimensional Sparse Data Sets**

Example: Click-stream data on which ads on a webpage tend to be clicked given known information about the person viewing the page.

- **Recommendations**

Example: What sort of movies should we recommend to a person who has watched and rated some other movies?

Clustering

Unsupervised - no need for training data

Unsupervised algorithms that aim to group things such that they are with other things more similar than different.

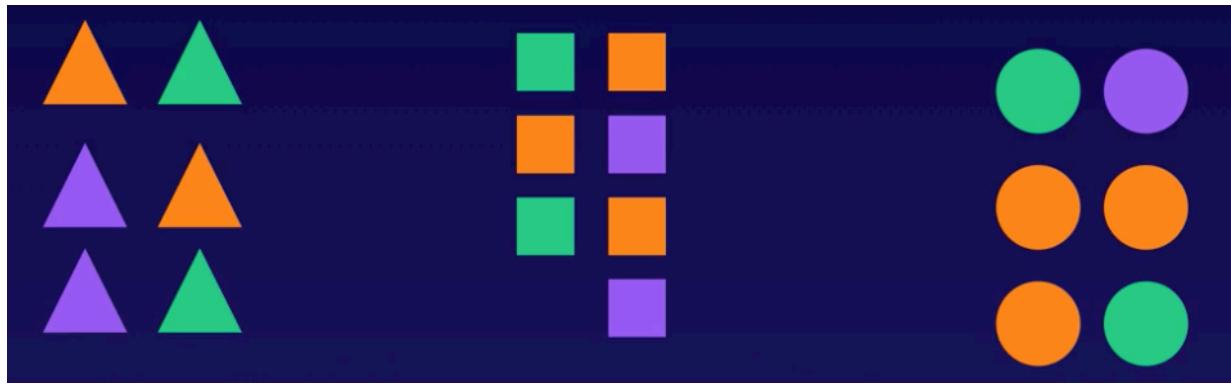


Specify which column a clustering algorithm need to pay attention to to group the data together

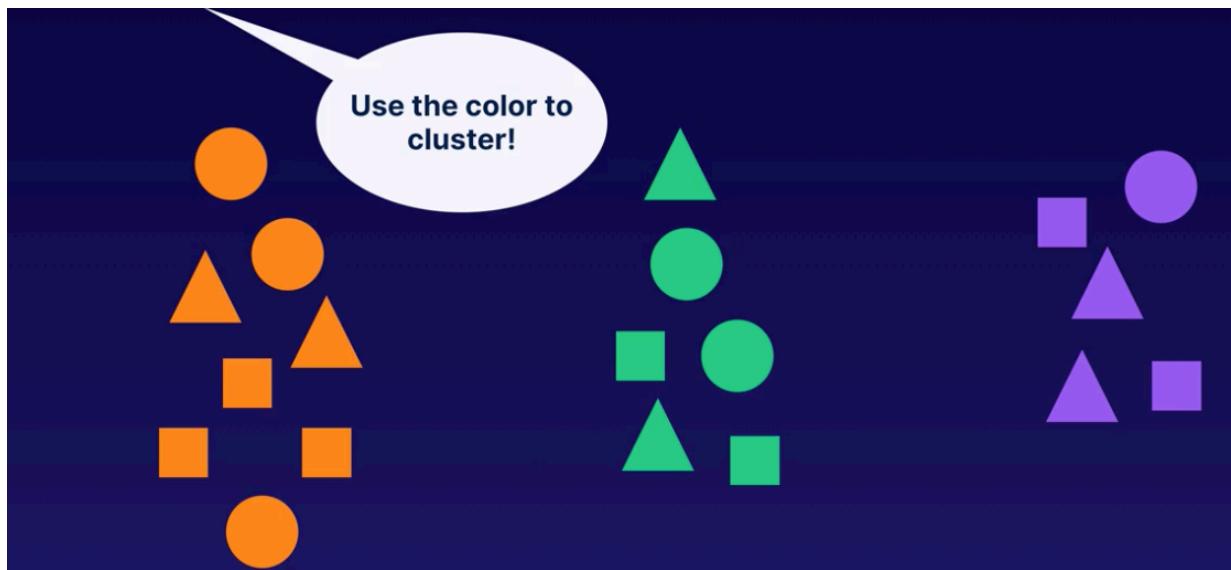
Use the number of sides to cluster!



Clustering Algorithm



We could also cluster based on color



Popular clustering algorithm: K-Means

K-Means



Unsupervised algorithm that attempts to find discrete groupings within data, where members of a group are as similar as possible to one another and as different as possible from members of other groups. The Euclidean distance between these points represents the similarity of the corresponding observations.

UNSUPERVISED



K-Means will take in a list of things with attributes. You specify which attributes indicate similarity and the algorithm will attempt to group them together such that they are with other similar things. "Similarity" is calculated based on the distance between the identifying attributes.

SageMaker K-means - things to know:

1

Expects Tabular Data

Rows represent the observations that you want to cluster and the columns represent attributes of the observations.

4

CPU Instances Recommended

GPU instances can be used but SageMaker's K-Means can only use one GPU.

2

Define the Identifying Attributes.

You must know enough about the data set to propose attributes that will define similarity. If you have no idea, there are ways around this too!

5

Training is Still a Thing

You want to be sure your model is still accurate and using the best identifying attributes. Your data just doesn't have labels.

3

SageMaker uses a Modified K-Means

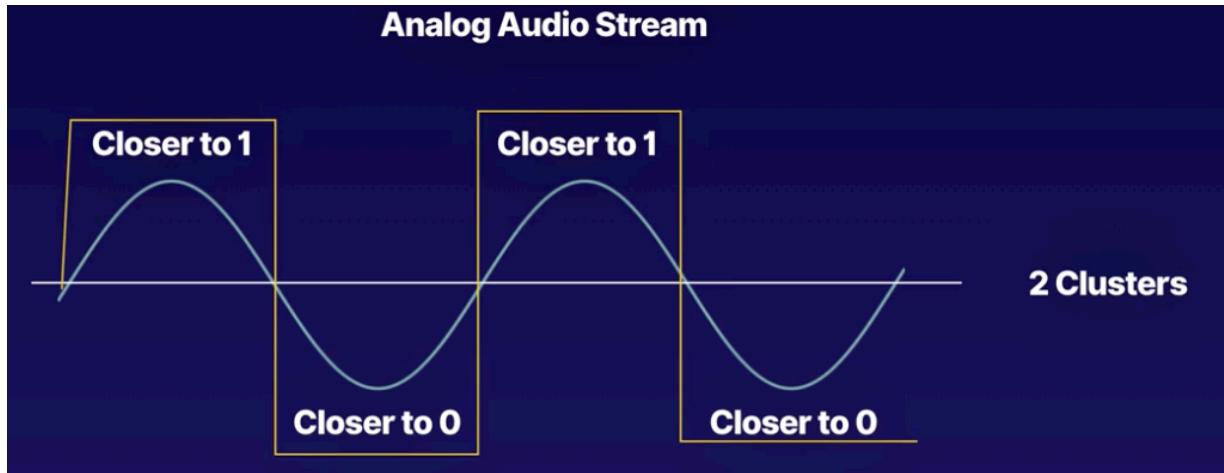
AWS uses a modified version of the web-scale K-Means algorithm, which it claims to be more accurate.

6

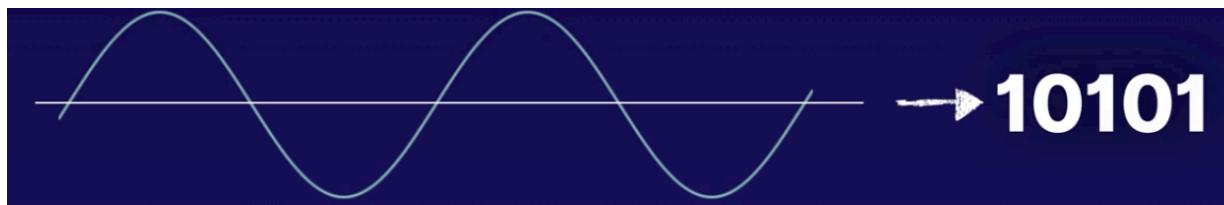
You Define Number of Features and Clusters

You must define the number features for the algorithm to analyze and the number of clusters you want.

Example of application: audio signal processing



Translate audio signal into digital signal with 0 and 1



The digitization process clips a bit of the fidelity from the analog



Other example: cluster the MNIST dataset
Store coordinates of white/black pixels and build vectors

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7	7
8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8	8
9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9	9

Classification



Popular classification algorithm in SM: K-Nearest Neighbor

K-Nearest Neighbor



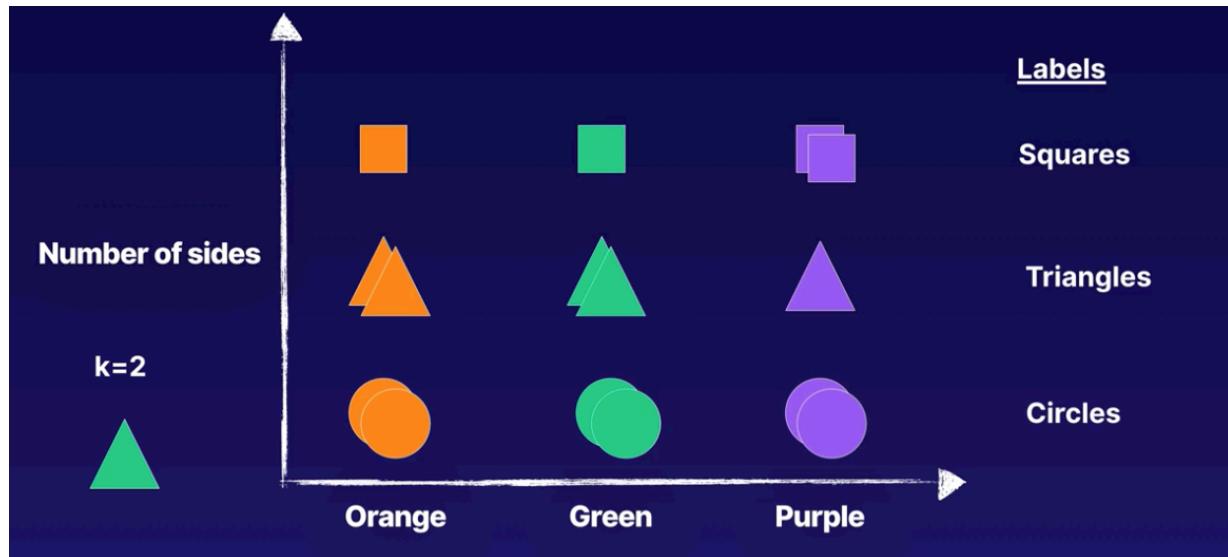
SUPERVISED

An index-based, non-parametric method for classification or regression. For classification, the algorithm queries the k points that are closest to the sample point and returns the most frequently used label as the predicted label. For regression, the algorithm queries the k closest points to the sample point and returns the average of the feature values as predicted value.



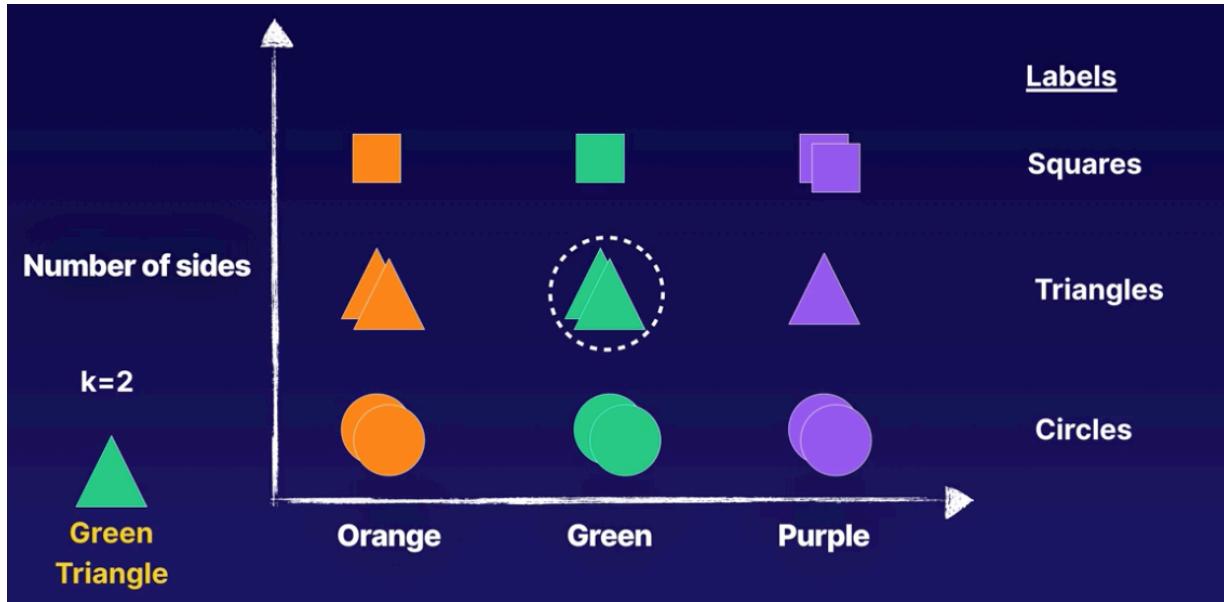
Predicts the value or classification based on that which you are closest. It can be used to classify or to predict a value (average value of nearest neighbors).

Example:



Hyper parameter: $k = 2$

Find the 2 nearest neighbors to this input shape (a triangle)



- | | | |
|--|--|--|
| 1
You choose the number of "neighbors".
You include a value for k , or in other words, the number of closest neighbors to use for classifying. | 2
KNN is a <i>lazy</i> algorithm.
Does not use training data points to generalize but rather uses them to figure out who's nearby. | 3
The training data stays in memory.
KNN doesn't "learn" but rather uses the training dataset to decide on similar samples. |
|--|--|--|

Use cases for K-nearest neighbors:

- **Credit Ratings**

Example: Group people together for credit risk based on attributes they share with others of known credit usage.

- **Product Recommendations**

Example: Based on what someone likes, recommend similar items they might also like.

K-nearest neighbor is a bit a matter of stereotyping
So be aware of Bias

Example when classification has gone wrong:

Redlining

The practice of literally drawing lines around neighborhoods and classifying those as:

- A - Best
- B - Still Desirable
- C - Definitely Declining
- D - Hazardous

Private and public entities would then use those redline maps to deny home loans, insurance, and other services for those less desirable areas – regardless of the qualifications of the applicant.

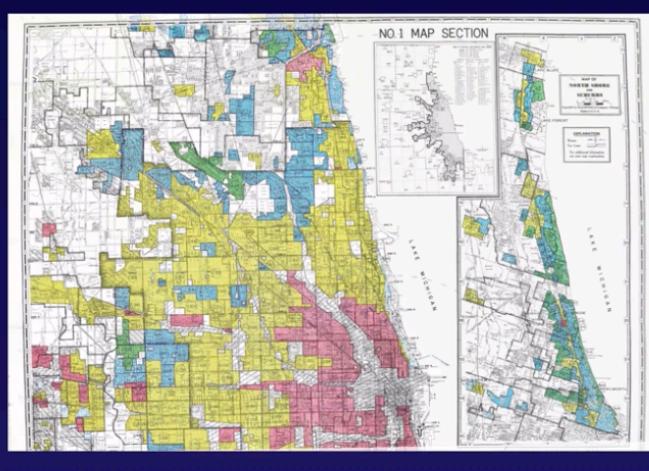


Image Analysis