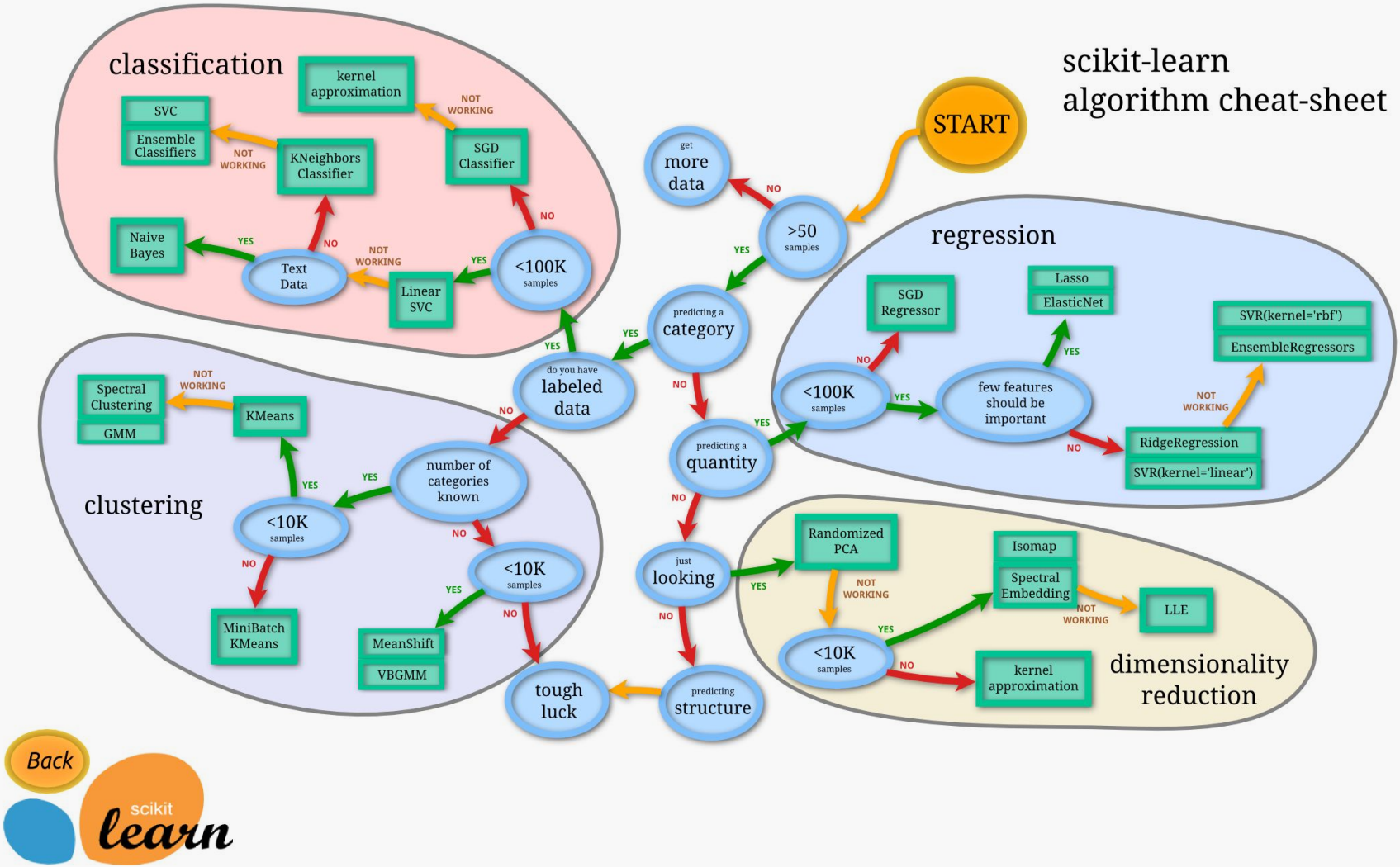
Python Machine Learning In Biology:

Tour of Machine Learning Classifiers

Nichole Bennett


scikit-learn
algorithm cheat-sheet
START
classification
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
KNeighbors Classifier
SGD Classifier
NO
Naive Bayes
YES
Text Data
NO
NOT WORKING
Linear SVC
YES
<100K samples
NO
YES
get more data
NO
>50 samples
YES
predicting a category
YES
NO
do you have labeled data
YES
NO
regression
SGD Regressor
Lasso
ElasticNet
SVR(kernel='rbf')
EnsembleRegressors
NO
<100K samples
YES
few features should be important
YES
NOT WORKING
NO
RidgeRegression
SVR(kernel='linear')
predicting a quantity
YES
NO
clustering
Spectral Clustering
GMM
NOT WORKING
KMeans
YES
number of categories known
YES
NO
<10K samples
NO
MiniBatch KMeans
<10K samples
YES
NO
MeanShift
VBGMM
just looking
YES
NO
Randomized PCA
YES
NOT WORKING
Isomap
Spectral Embedding
NOT WORKING
LLE
<10K samples
YES
NO
kernel approximation
dimensionality reduction
tough luck
predicting structure
Back
scikit learn

Modeling

You probably a good idea of how long it takes you to get to work from your house.

All humans naturally model the world around them.

Over time, your observations about transportation have built up a mental dataset and a mental model that helps you predict what traffic will be like at various times and locations. You probably use this mental model to help plan your days, predict arrival times, and many other tasks.

As scientists we attempt to make our understanding of relationships between different quantities more precise through using data and mathematical/statistical structures. This process is called modeling.

Modeling

Models are simplifications of reality that help us to better understand that which we observe.

In a science setting, models generally consist of an dependent variable (or output) of interest and one or more independent variables (or inputs) believed to have an effect on the dependent variable.

Model-Based Inference

We can use models to conduct inference.

Given a model, we can better understand relationships between an independent variable and the dependent variable or between multiple independent variables.

What are some examples of where inference from a model would be valuable in your research?

Prediction

We can use a model to make predictions, or to estimate an dependent variable's value given at least one independent variable's value.

Predictions can be valuable even if they are not exactly right.

Good predictions are extremely valuable.

What are some examples of where prediction from a model could be valuable for your research?

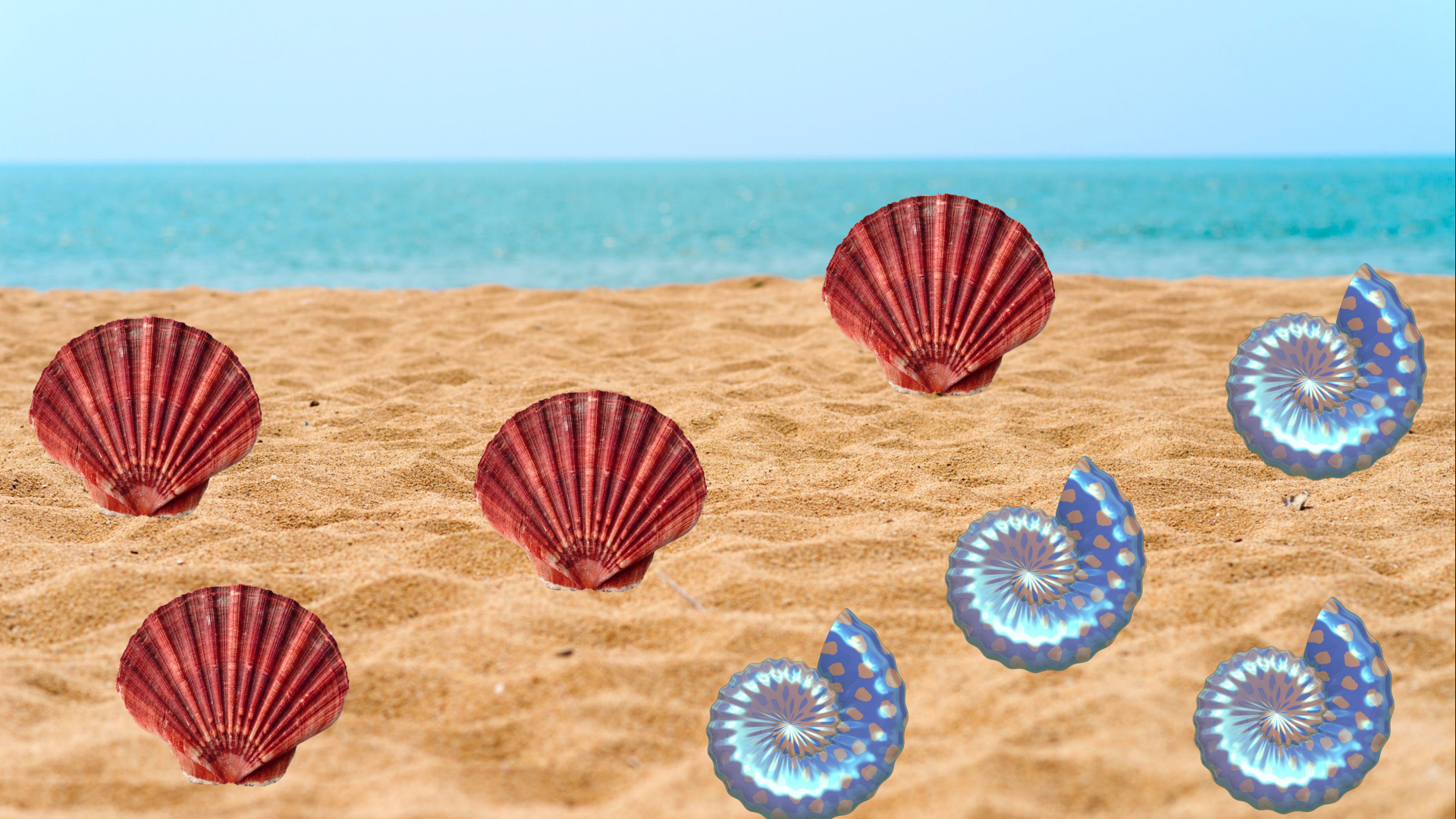
What is the difference between model prediction and inference?

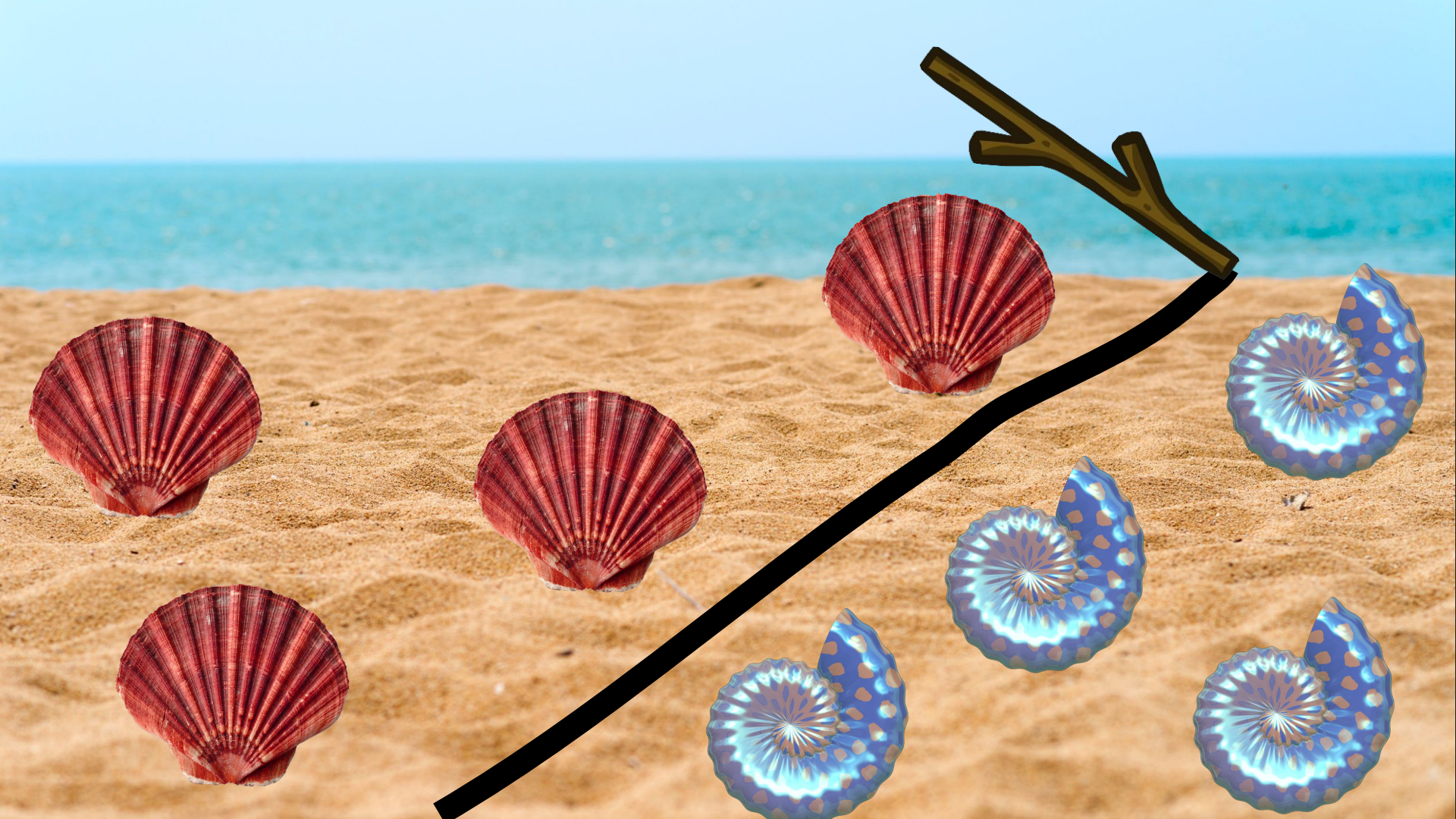
“No Free Lunch” Theorem

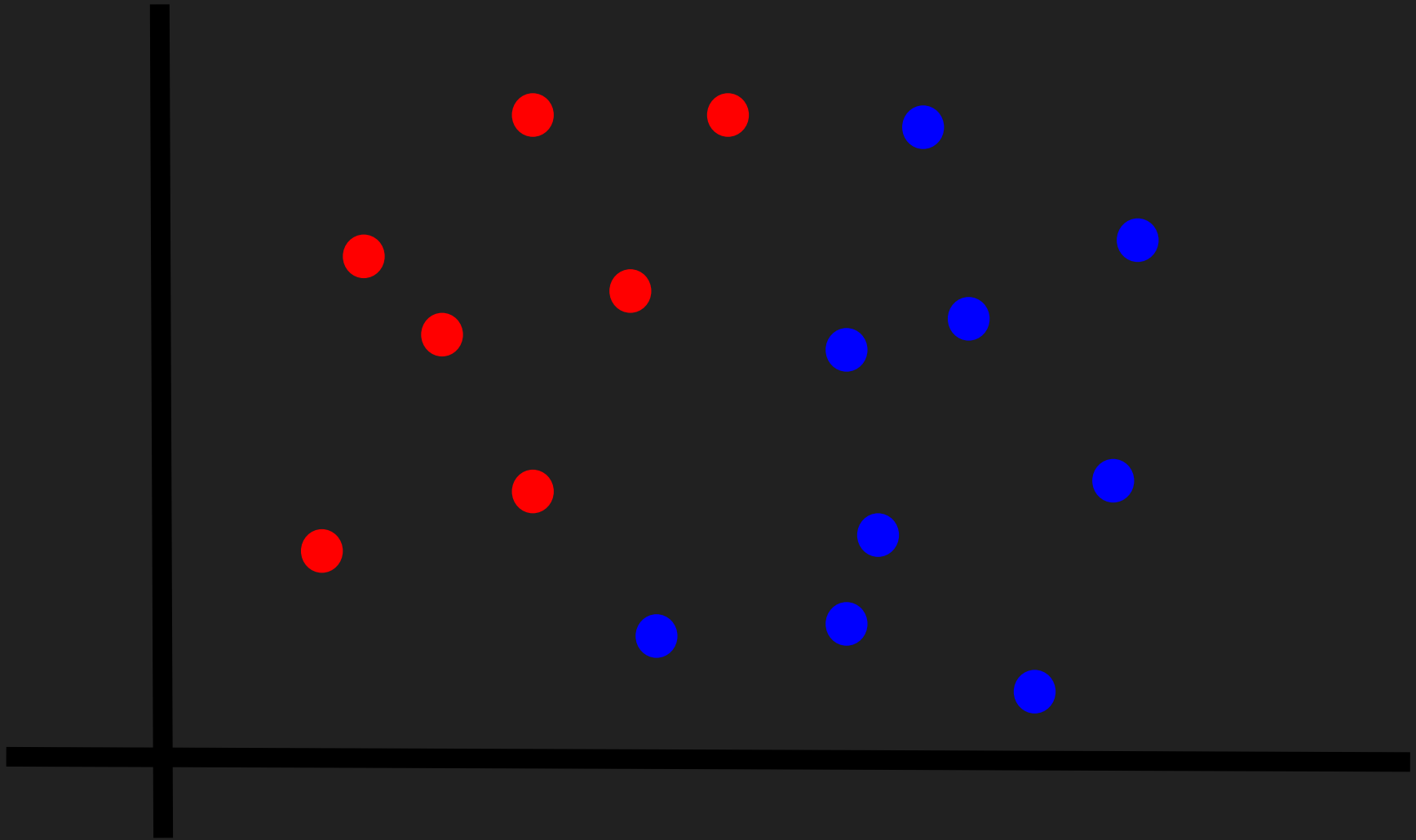
No single classifier works best across all possible scenarios.

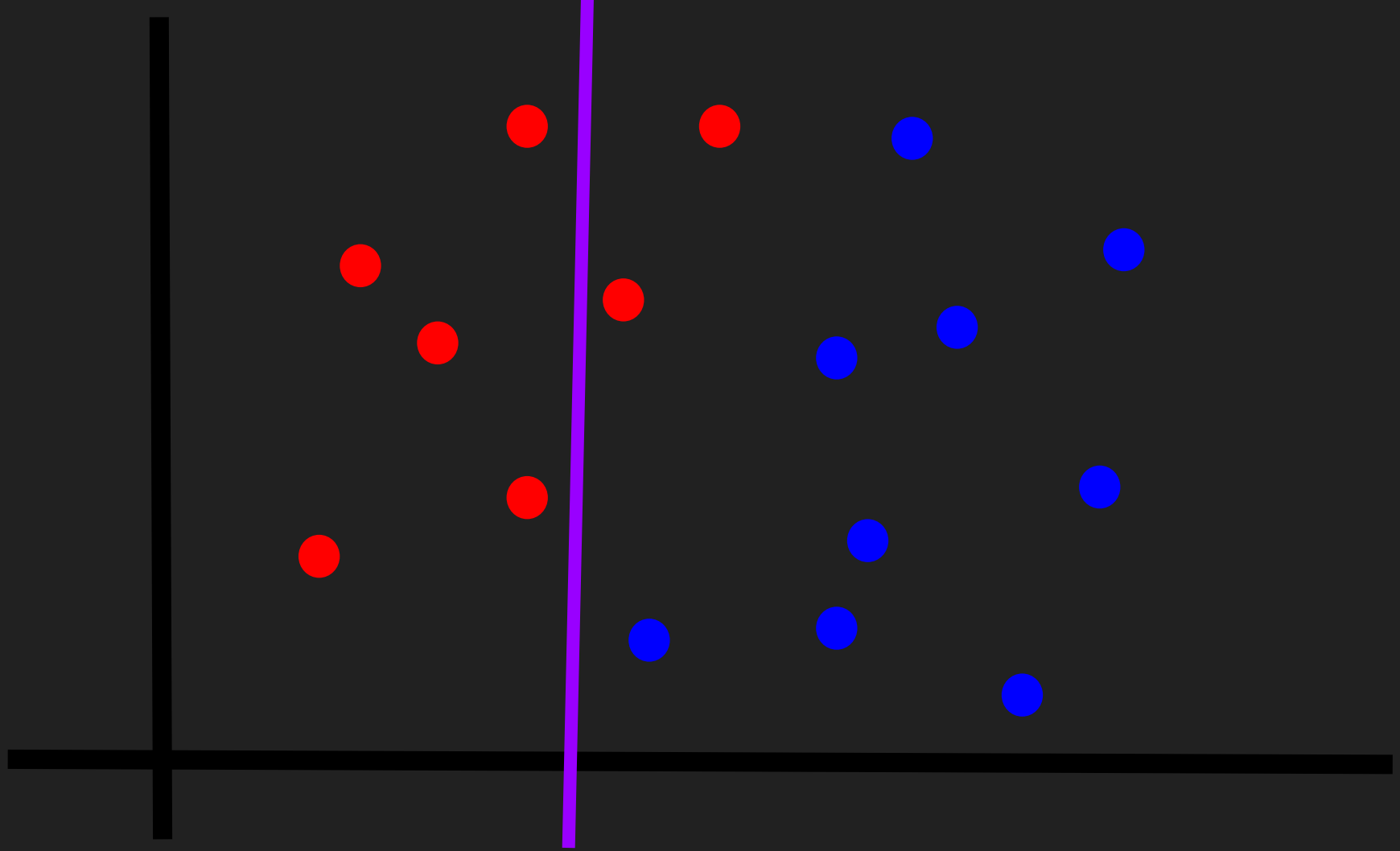
Usually a good idea to compare the performance of a few different algorithms.

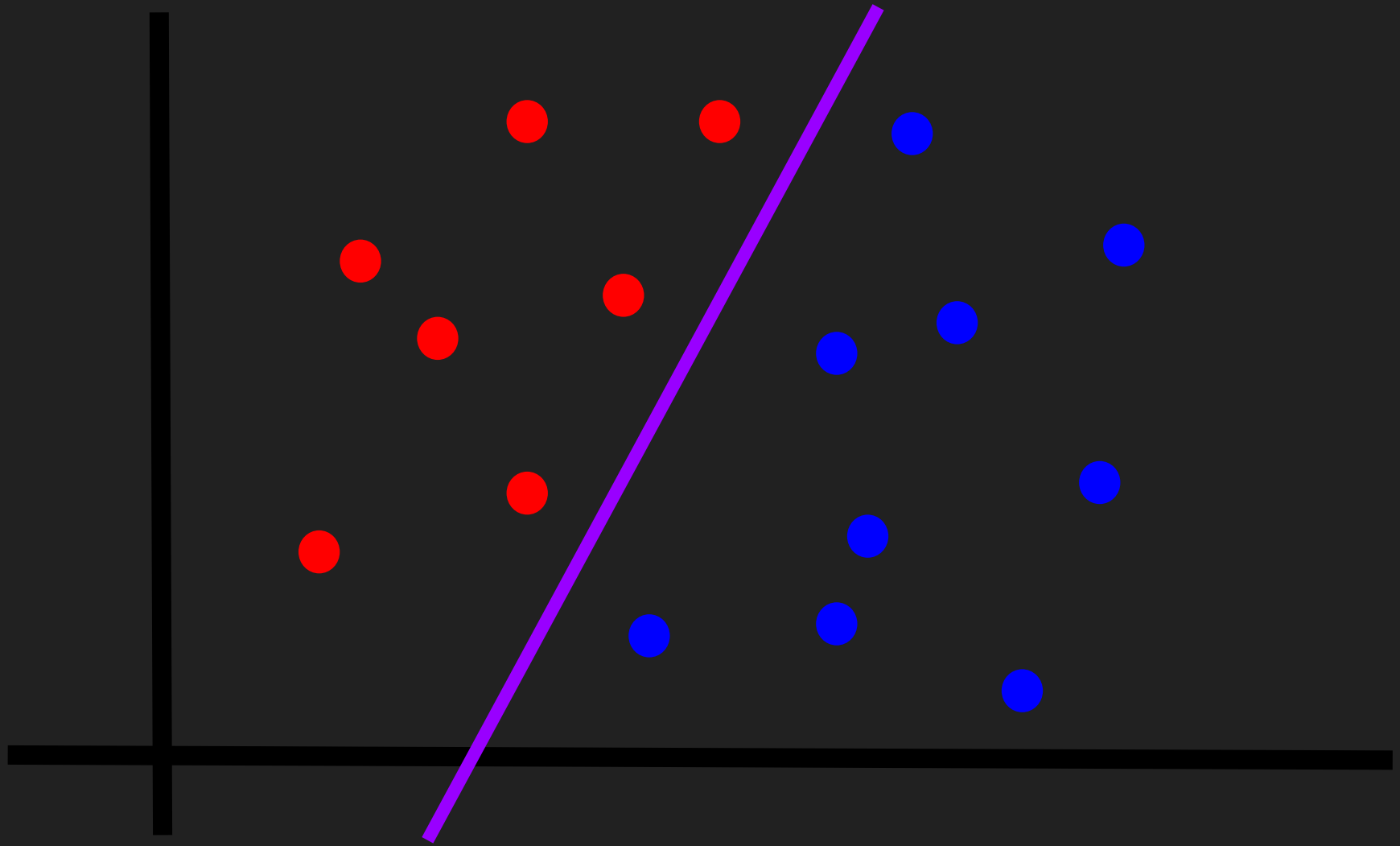
Classifiers





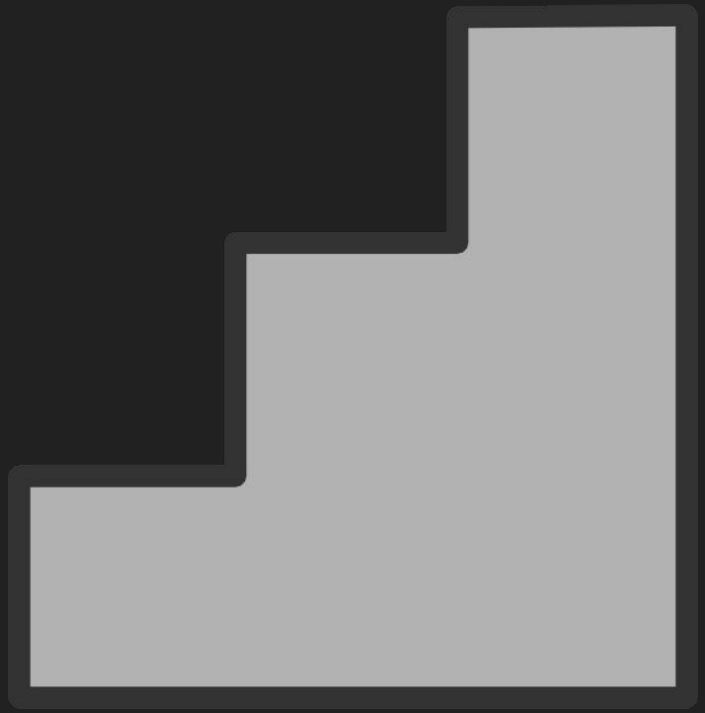






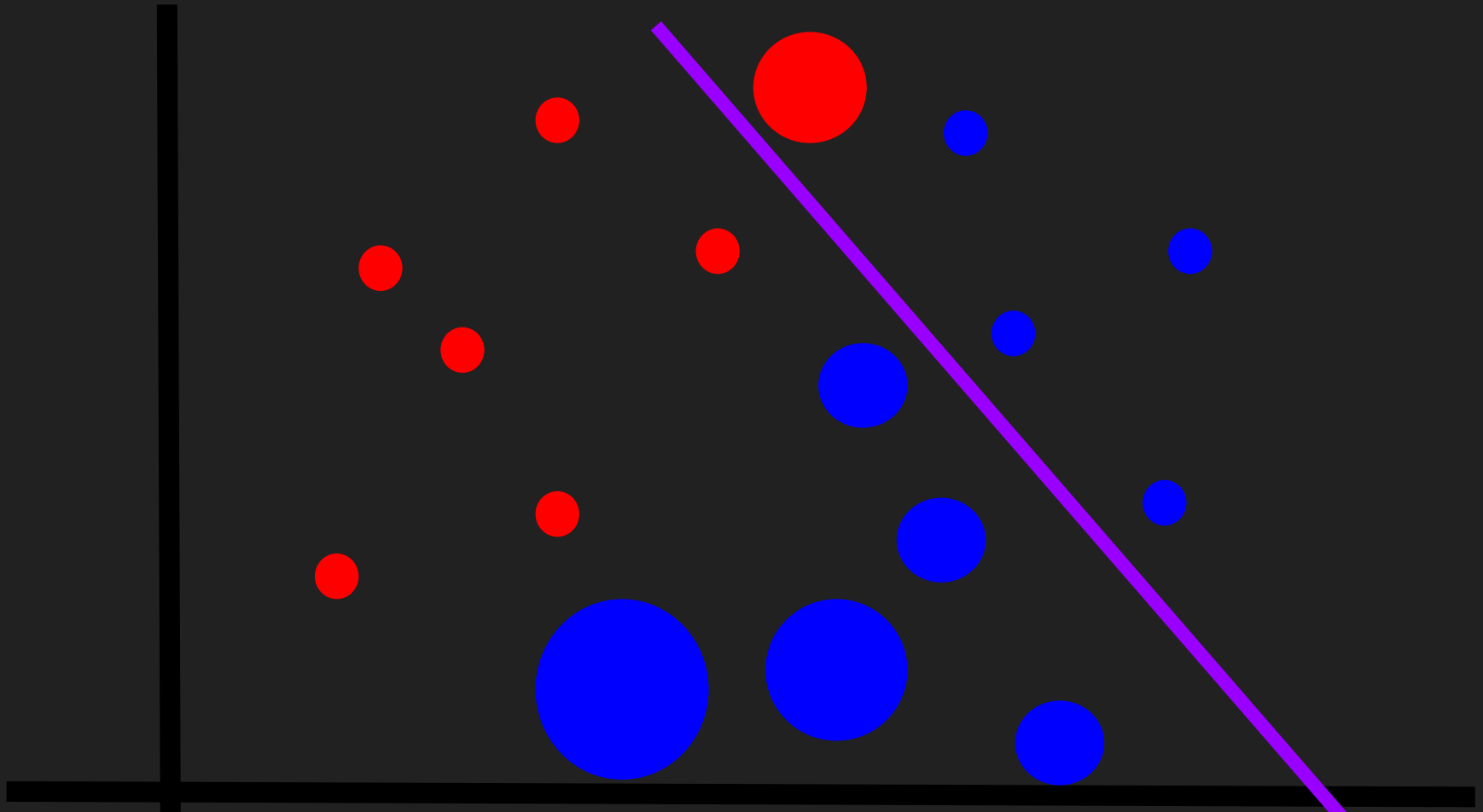
Discrete vs. Continuous Error Functions

Discrete Error Function

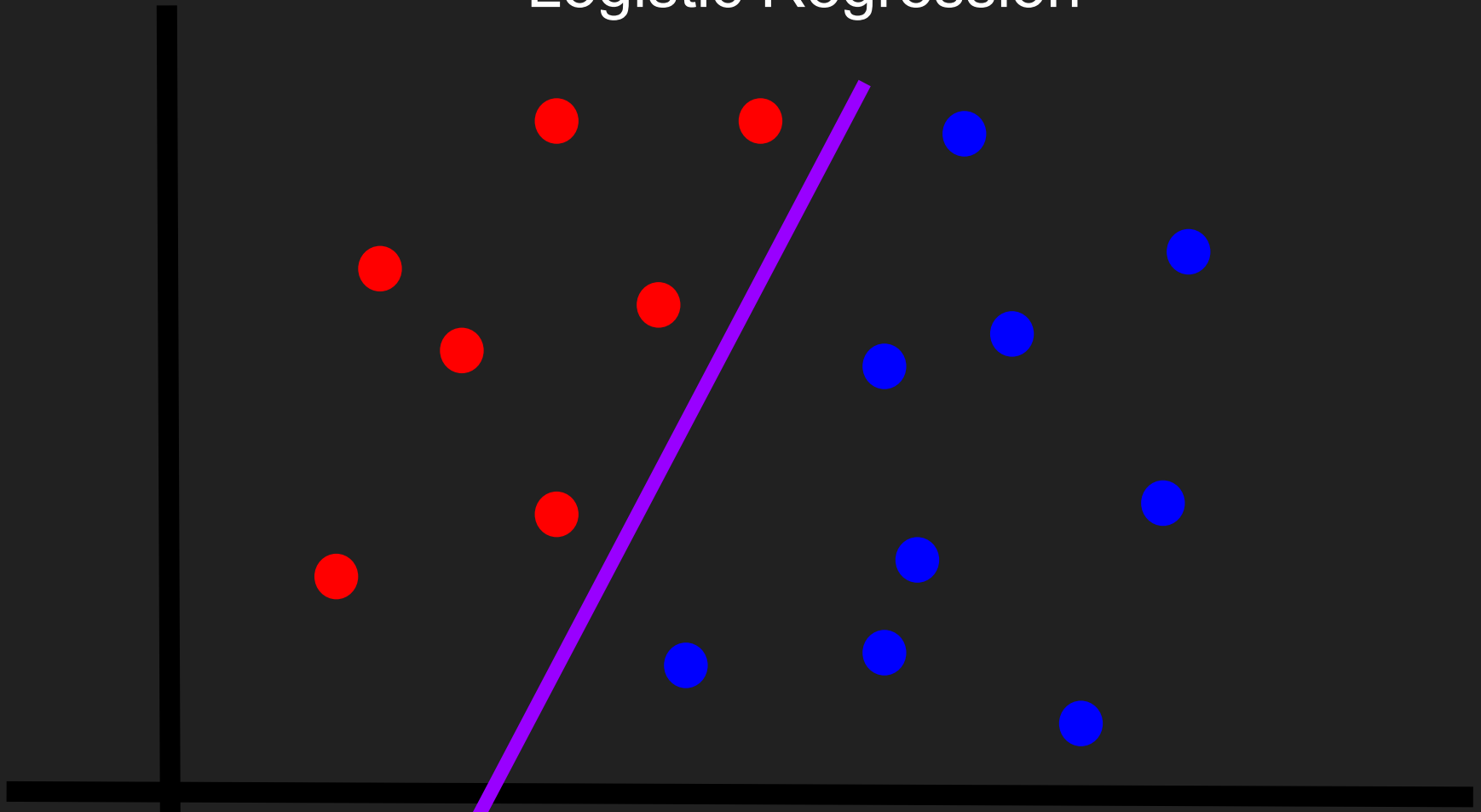


Continuous Error Function

Logistic Regression



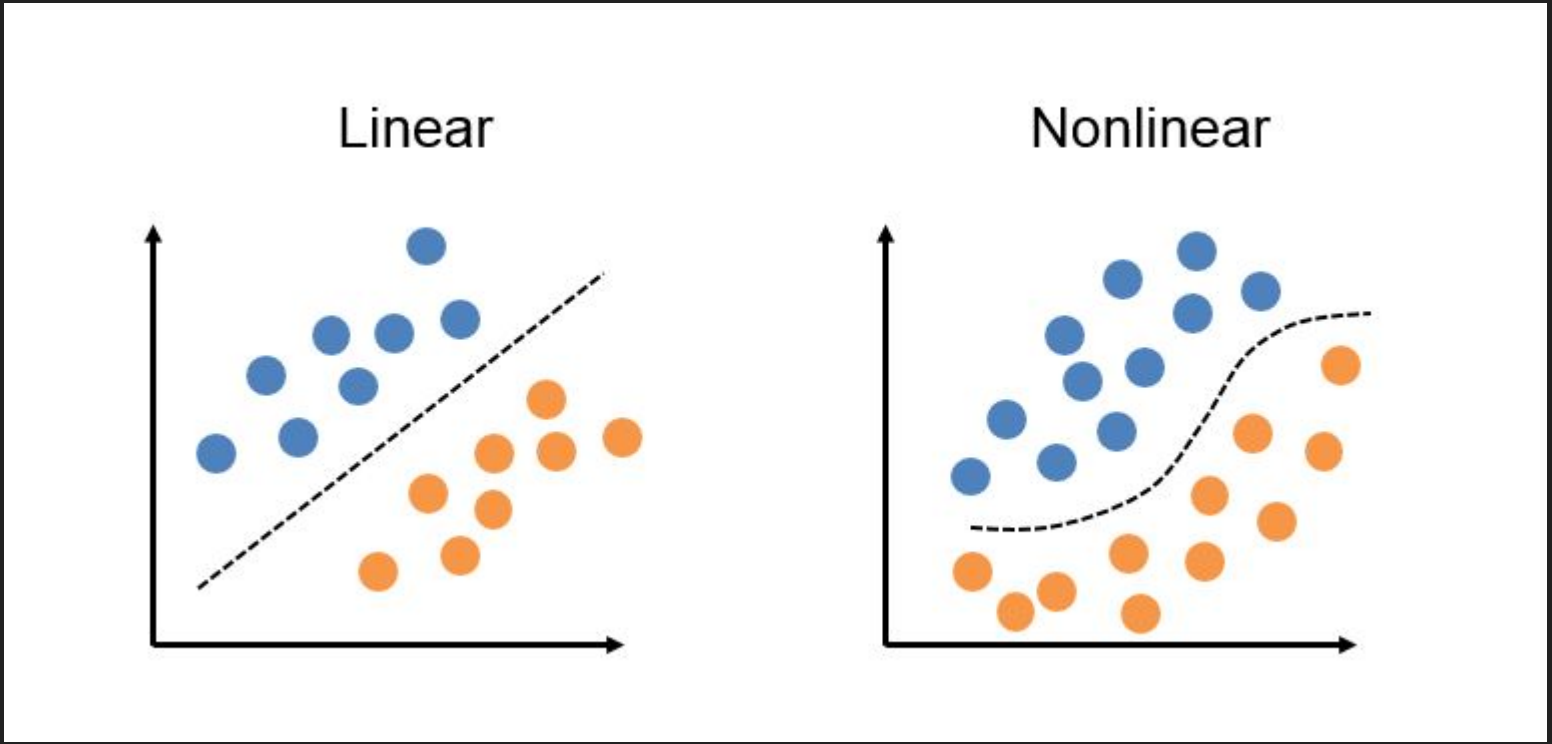
Logistic Regression



Steps for Training a Machine Learning Classifier

1. Selection of features
2. Choosing a performance metric
3. Choosing a classifier and optimization algorithm
4. Evaluating the performance of the model
5. Tuning the algorithm

Quick Note: Linearly Separable vs. Not



Logistic Regression

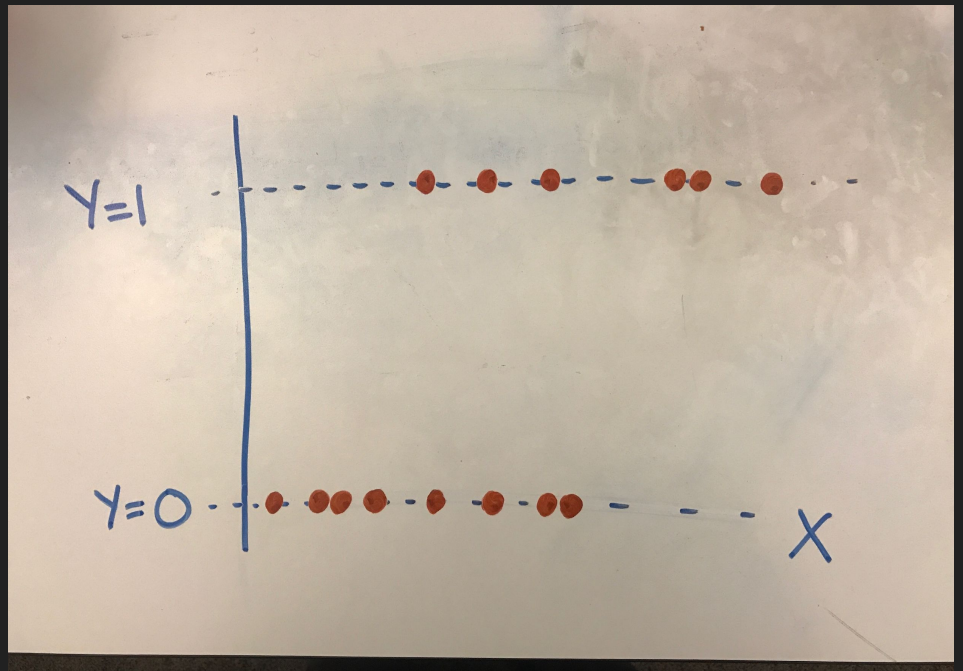
Suppose we have a dataset with two variables in it: one variable we seek to predict (Y) and one independent variable (X) we want to use in order to predict Y. What we would have done previously is to pick a metric (usually mean squared error) and then fit a line of best fit (regression!)

However, what if Y is binary? (not continuous)

What are some examples from our research like this?

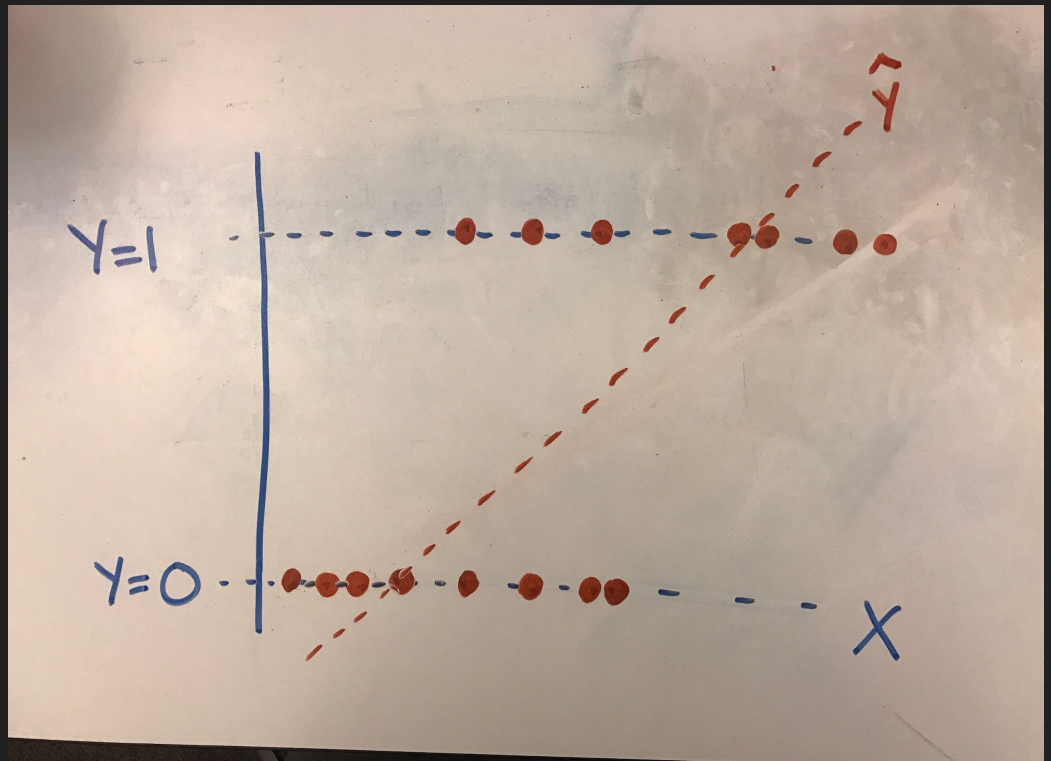
Logistic Regression

After turning our binary variable into a dummy variable appropriate for use in Python, our plot might look something like this.



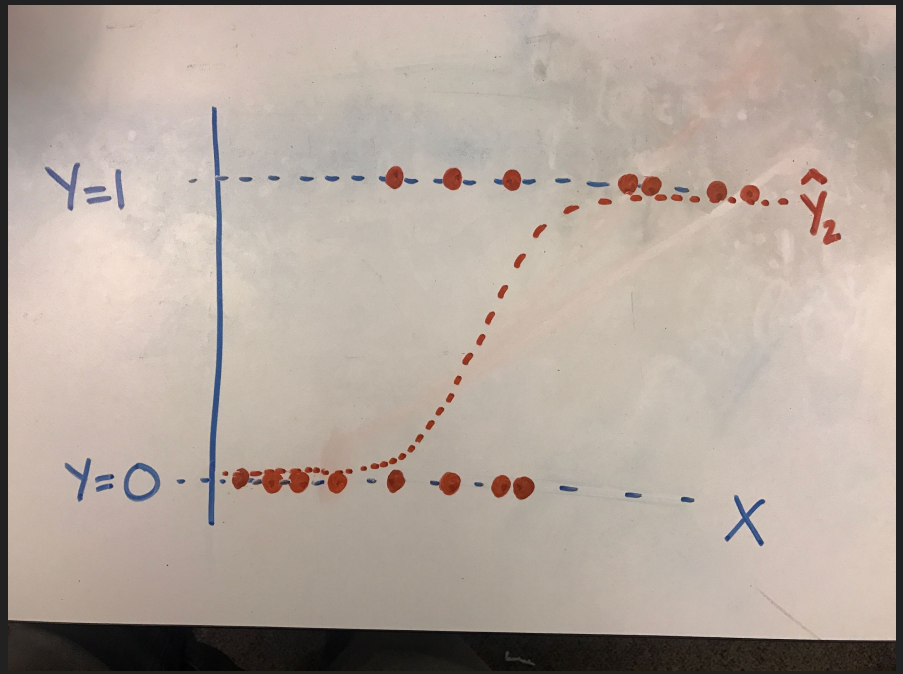
Logistic Regression

If we were to fit a traditional line through it, we'd likely get a line that fits as follows:



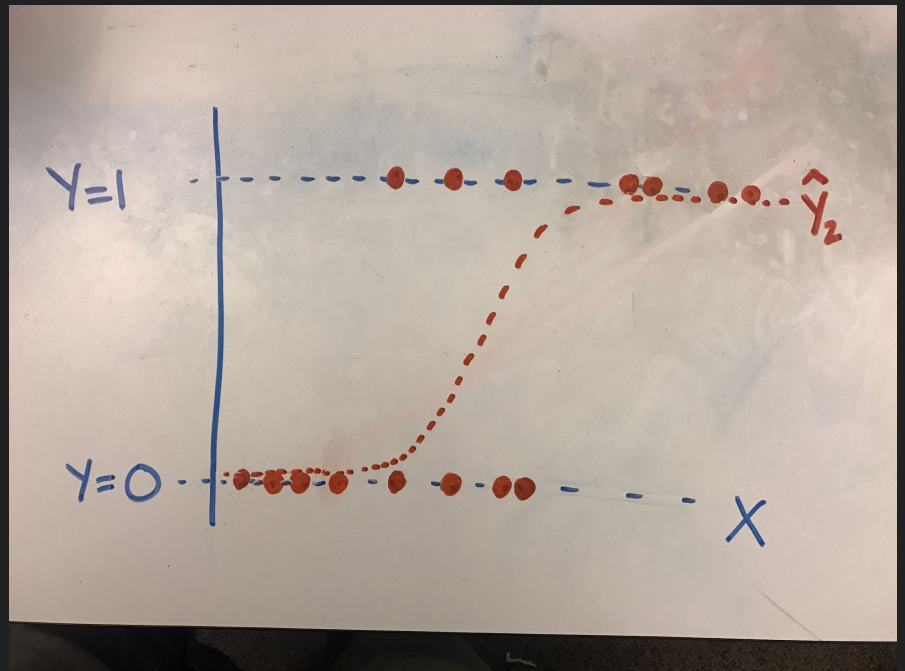
Logistic Regression

Wouldn’t it be great to have a line like this?



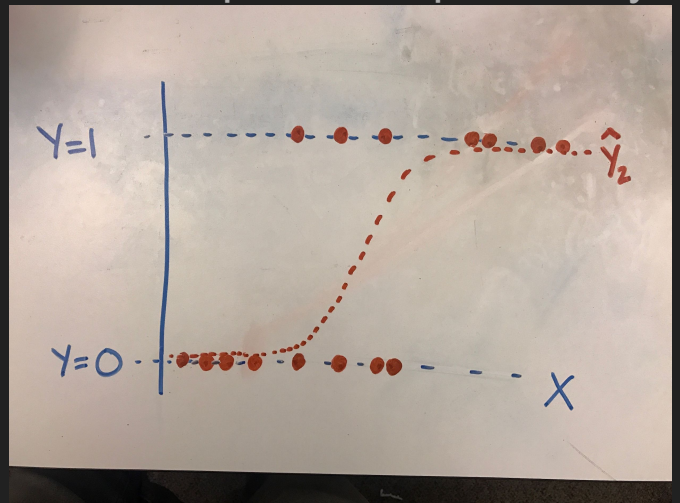
Logistic Regression

Logistic regression measures the relationship between the dependent data point variables and one or more independent data points by estimating probabilities.



Logistic Regression

The goal of logistic regression is to predict the probability that Y is equal to 1 (rather than 0) given certain values of X. Our model's goal is predicting probabilities - rather than target scores - with our independent variables. Once we get these predicted probabilities, we can classify observations as falling into one category or the other based on the predicted probability.



Logistic Regression

Performs well on linearly separable classes

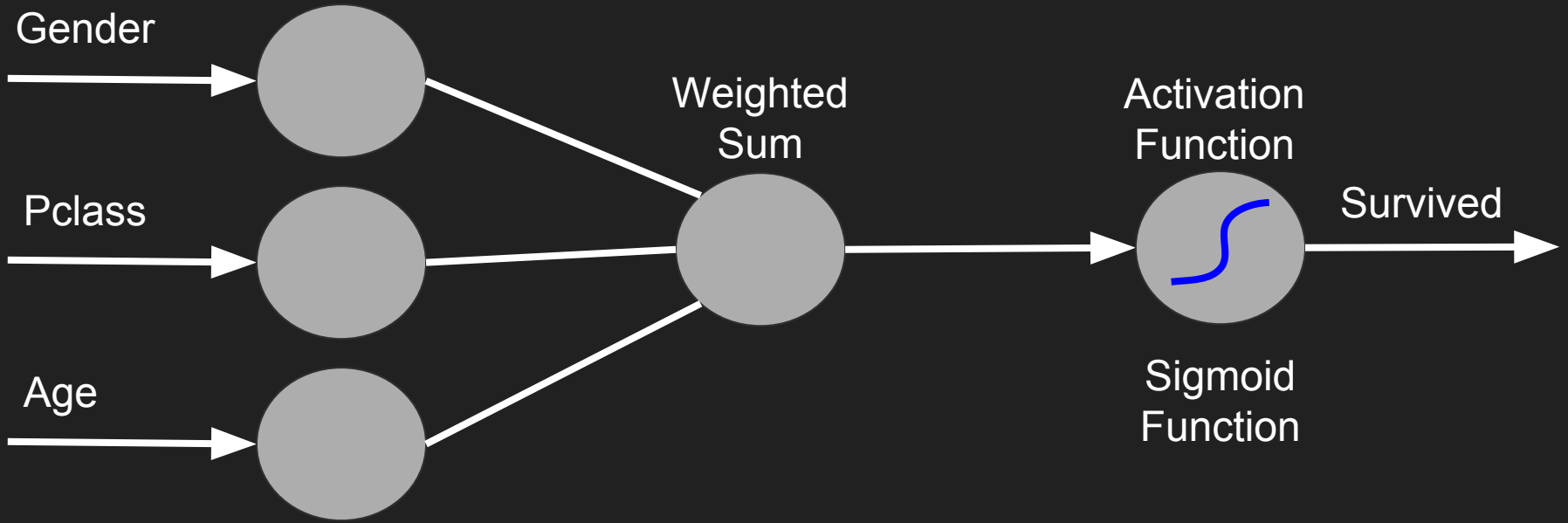
Usually used for binary problems but can be used for multiclass

Logistic is the inverse of a logit function (link function for regression)

Probabilistic model: predict probability that a particular sample belongs to a certain class. (conditional probability it belongs to that class given its features)

Especially useful when are interested in class-membership probability

Logistic Regression on the Titanic Dataset

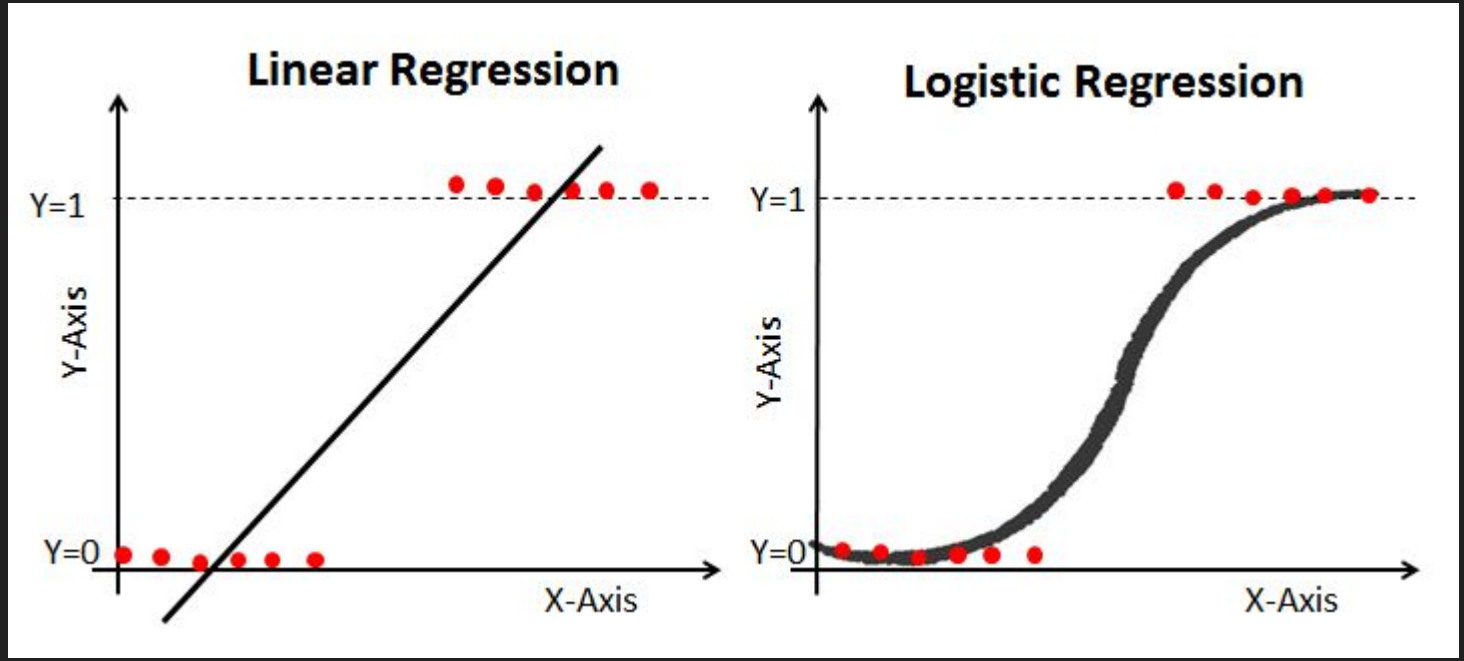


Logistic Regression

Transform the output of a linear regression (Y) which generally can be any value, to a range appropriate for probability - that is, anywhere between 0 and 1.

The specific transformation used to "bend" our linear regression line is called the logit and gives rise to the name "logistic regression." It can predict values between 0 and 1, but does not include either.

Logistic Regression



Logistic Regression

Advantages

Efficient and straightforward (doesn't require high computation power)

Easy to implement

Easily interpretable

Used widely

Doesn't require scaling of features

Provides a probability score for observations

Disadvantages

Not able to handle a large number of categorical features/variables

Vulnerable to overfitting

Can’t solve the non-linear problem

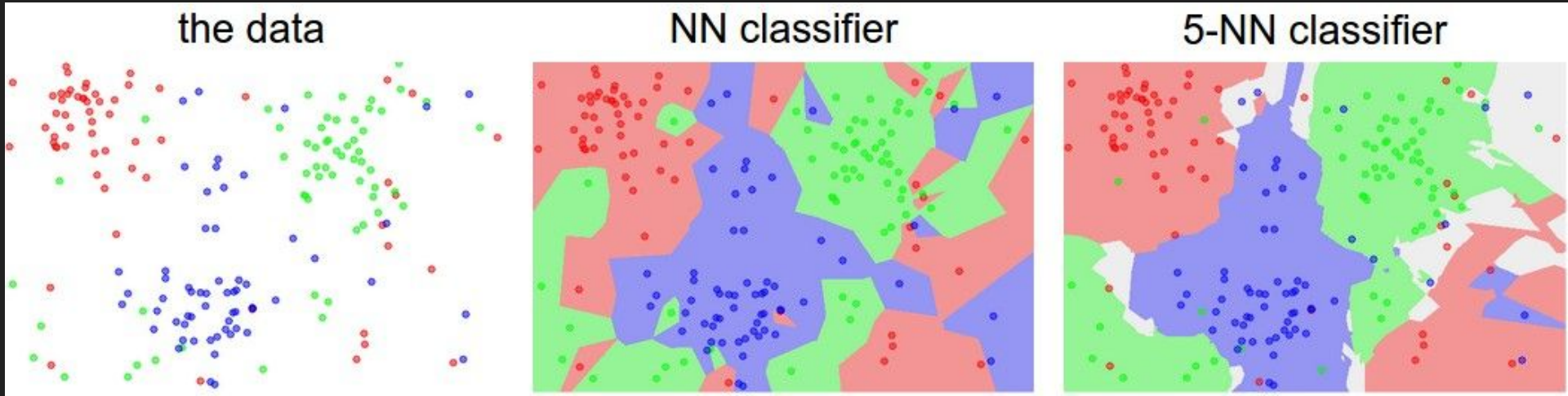
Will not perform well with independent variables that are not correlated to the target variable and are very similar or correlated to each other

KNN Living Demo

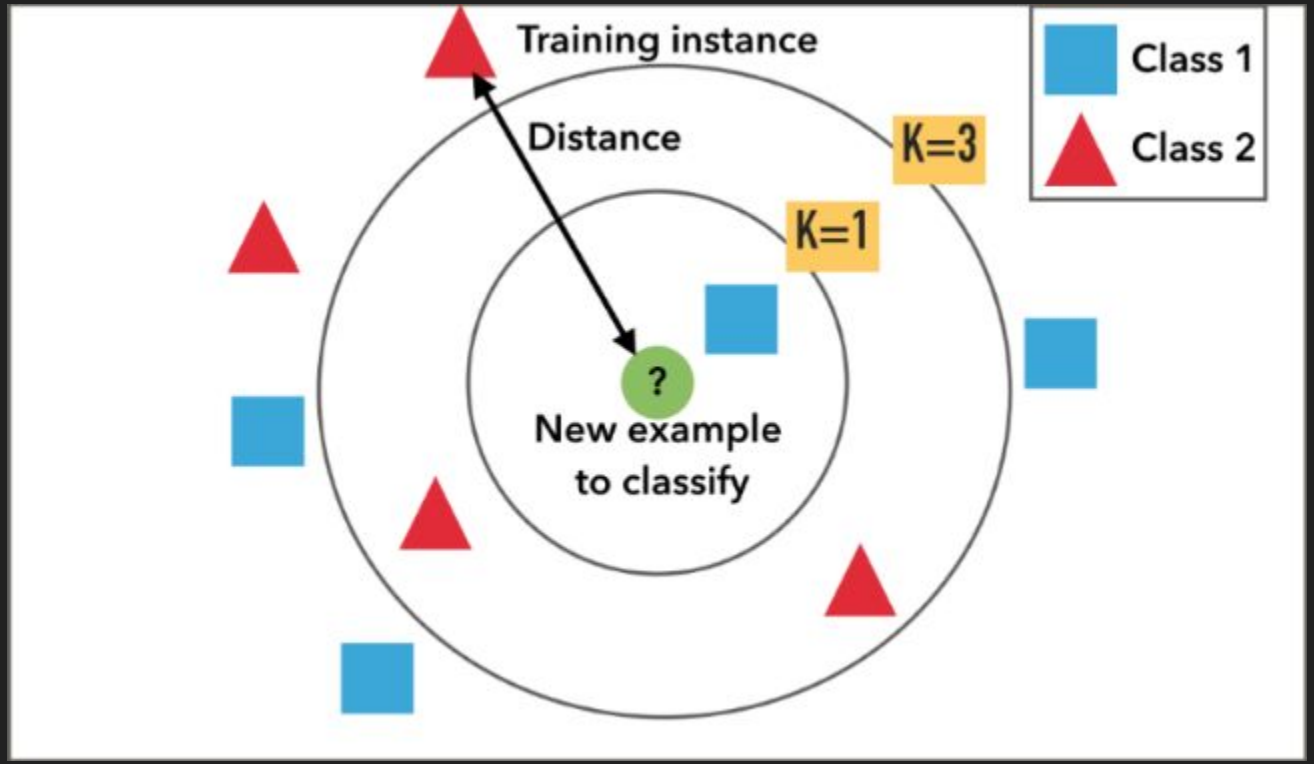
K-nearest Neighbors

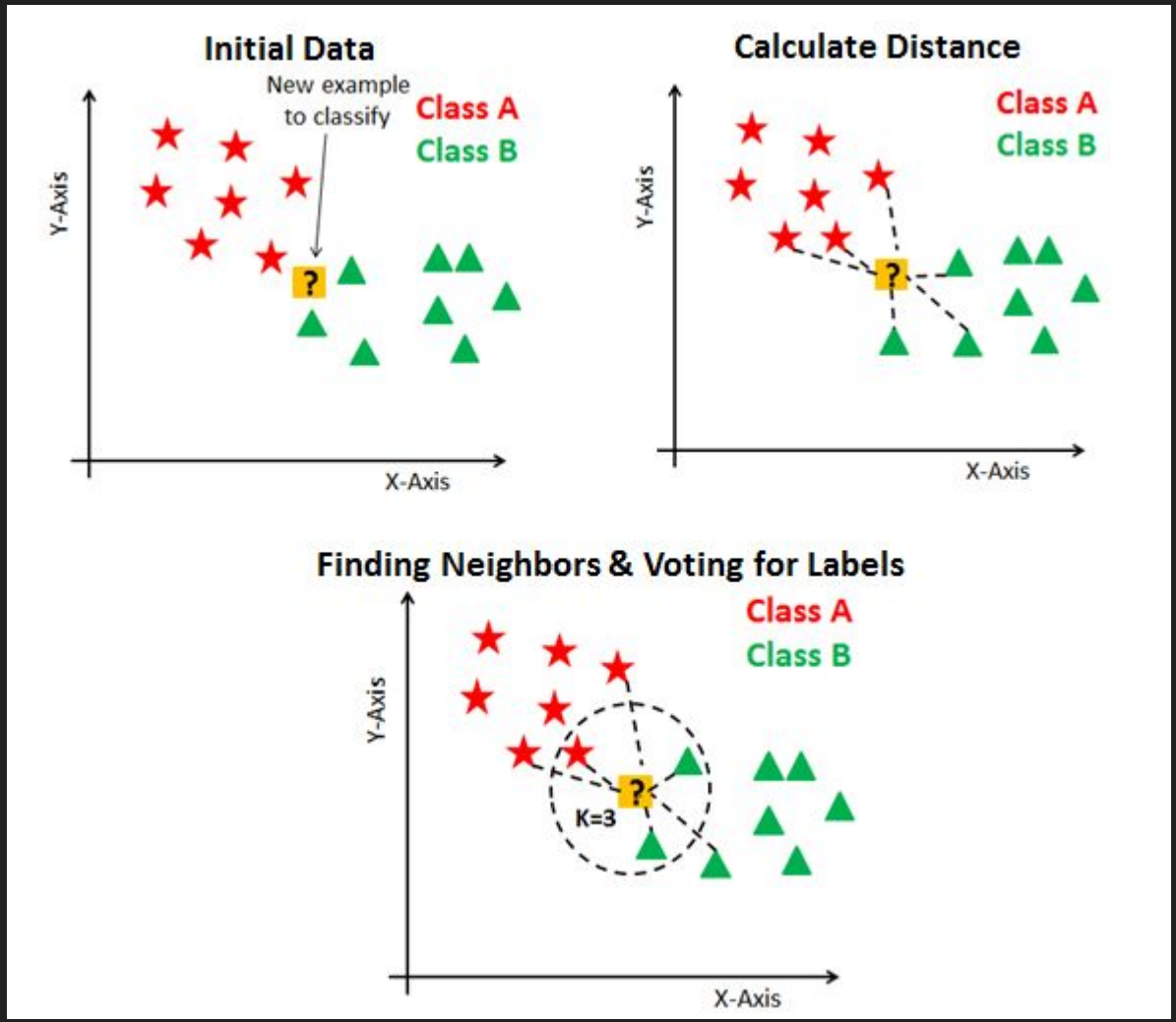
From unknown point, pick k nearest neighbors

Majority class among those k points wins!



K-nearest Neighbors




Initial Data
New example
to classify
Class A
Class B
Y-Axis
X-Axis
?
Calculate Distance
Class A
Class B
Y-Axis
X-Axis
?
Finding Neighbors & Voting for Labels
Class A
Class B
Y-Axis
X-Axis
?
K=3

K-nearest neighbors

Non-parametric, lazy learning algorithm that predicts similarity based on “nearness” (similarity)

- Non-parametric means makes no assumptions about underlying distribution of the data (much of our data doesn’t follow neat distributions)
- Lazy means the training phase is minimal. Uses all (or nearly all) training data. (Testing phase is costly, though)

Spatial algorithm.

K-nearest Neighbors

How do we choose k?

- Balance between under- and over-fitting.
- Curse of dimensionality: feature space becomes increasingly sparse for an increasing number of dimensions. (neighbors may be far!)

How do we choose a distance metric?

- Euclidean distance usually works for real values but important to standardize.

Both k and the distance measure are hyperparameters because we set them. The model doesn’t learn them.

K-nearest Neighbor

Advantages

Simple to understand and explain

Model training is fast

Can be used for classification or regression

Non-linear

Useful for Gene Expression, Protein-Protein Interactions, and 3-D Structure Prediction

Disadvantages

Must store all the training data

Prediction can be slow with large datasets

Sensitive to irrelevant features

Sensitive to the scale of the data

Accuracy may not be competitive with other algorithms

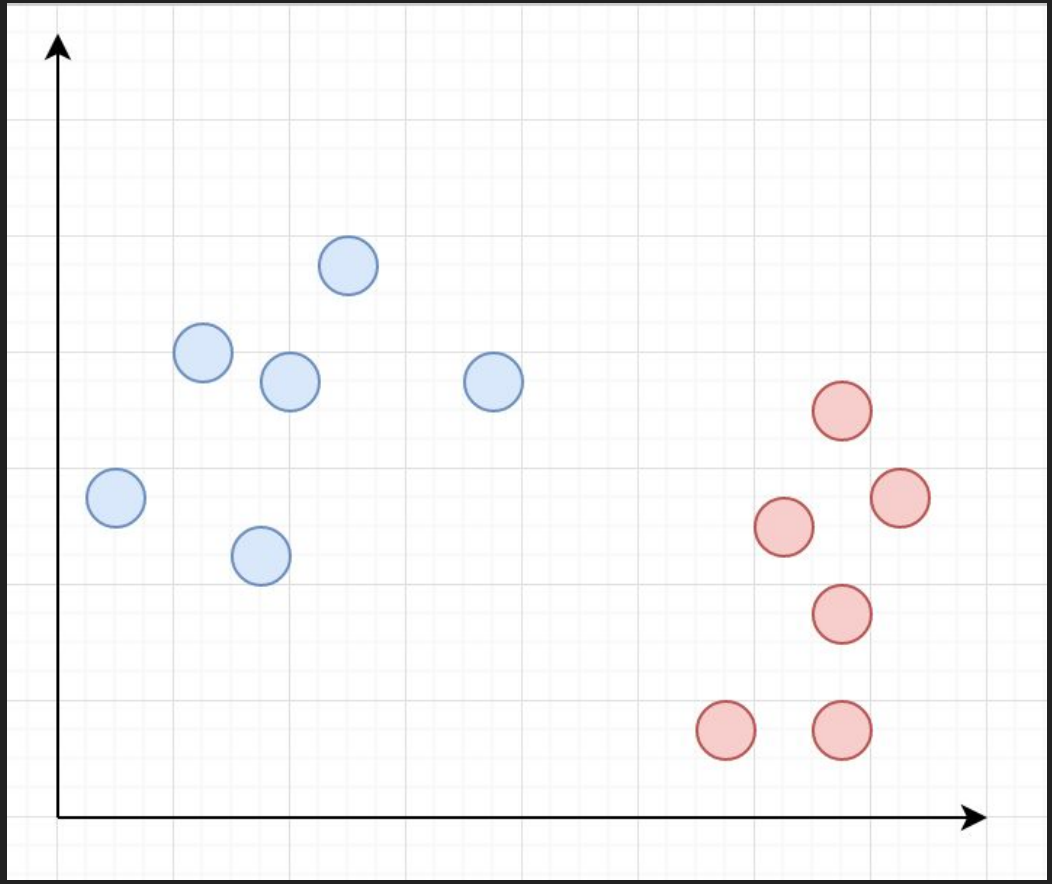
Curse of Dimensionality (works better with fewer features)

SVM Living Demo

Support Vector Machine

Imagine we have red class and blue class, and plotted, they look like this graph.

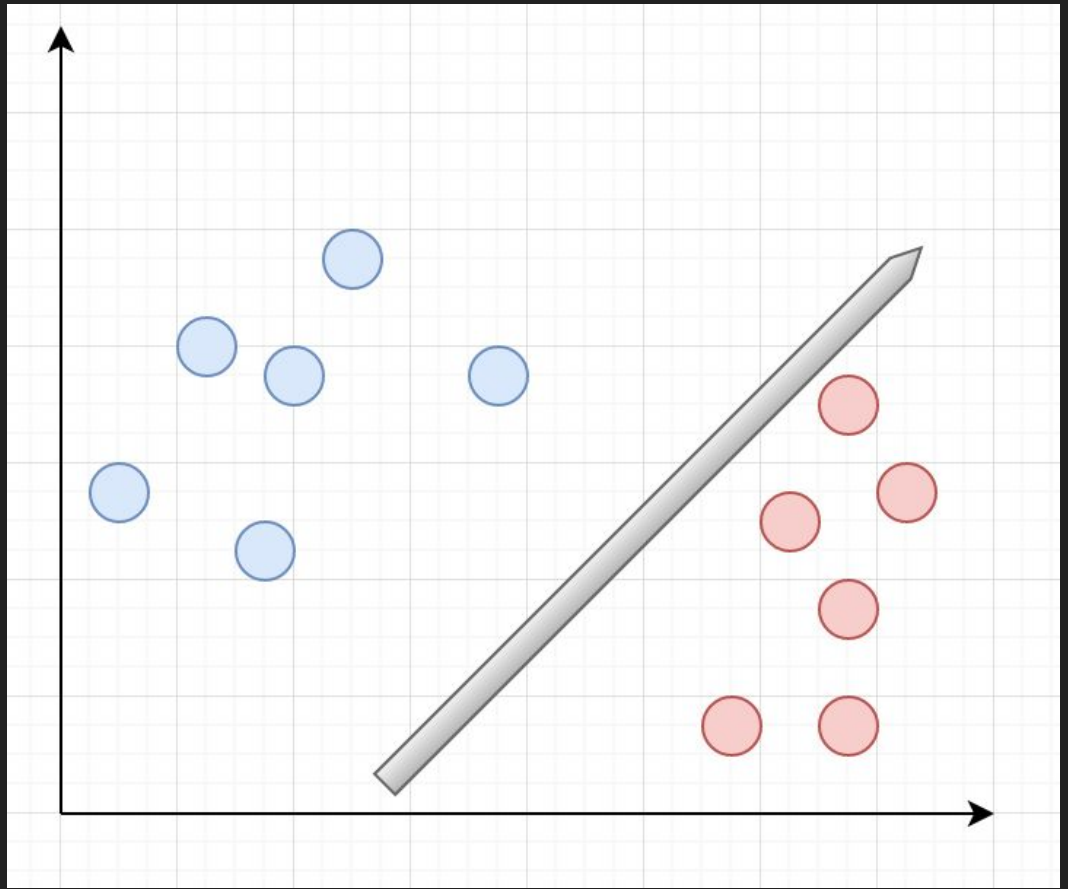
What is the ideal separation boundary given this distribution?



Support Vector Machine

Imagine we have red class and blue class, and plotted, they look like this graph.

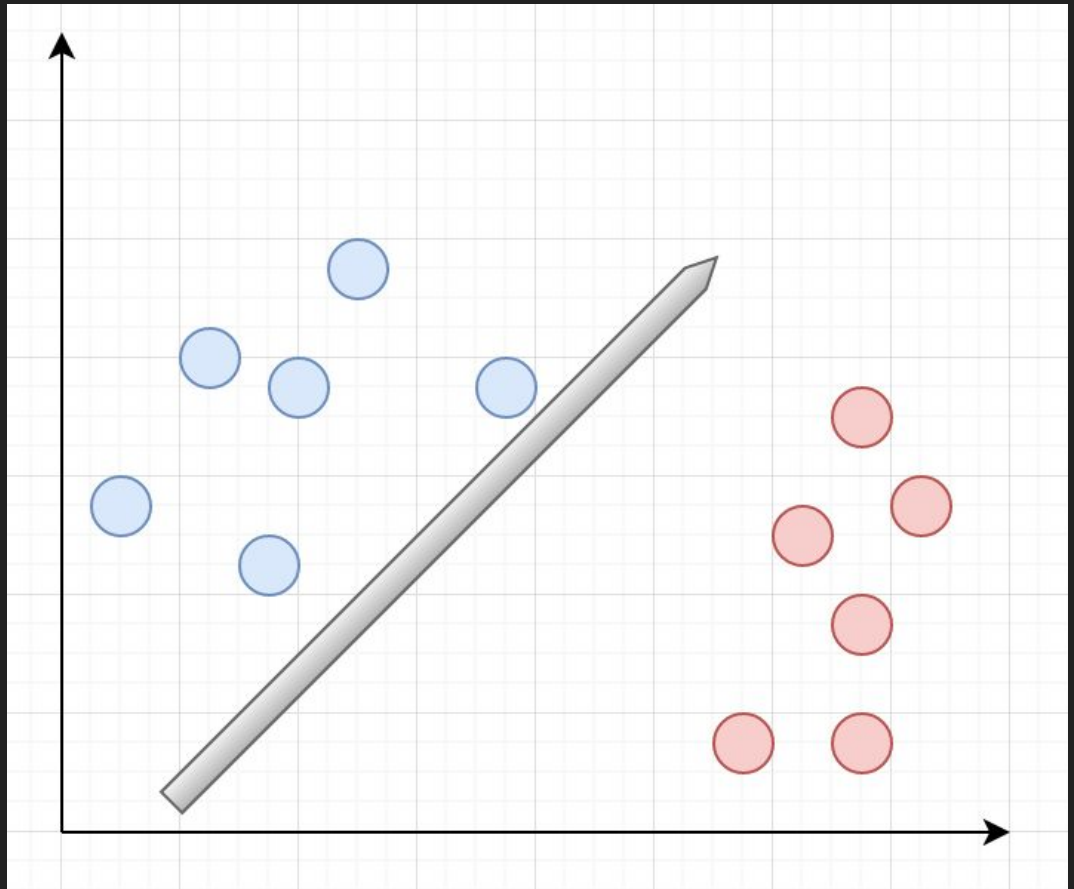
Option 1



Support Vector Machine

Imagine we have red class and blue class, and plotted, they look like this graph.

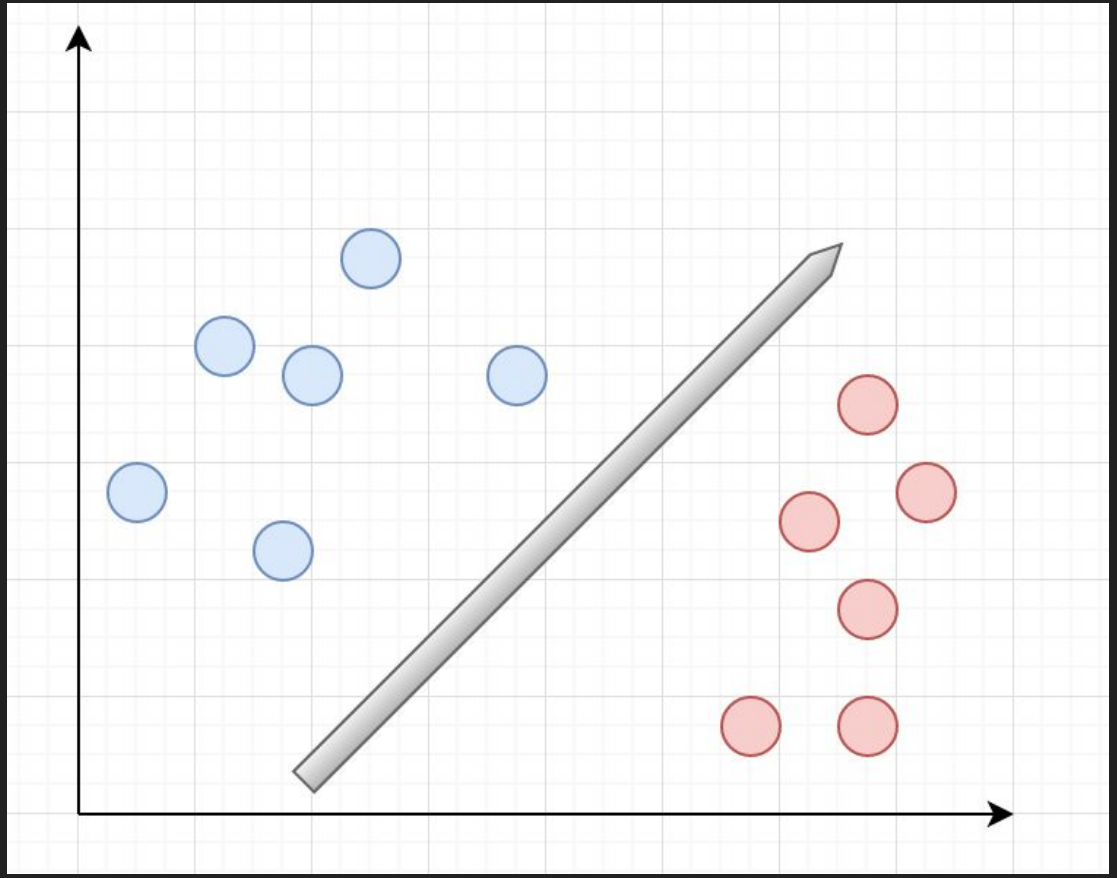
Option 2



Support Vector Machine

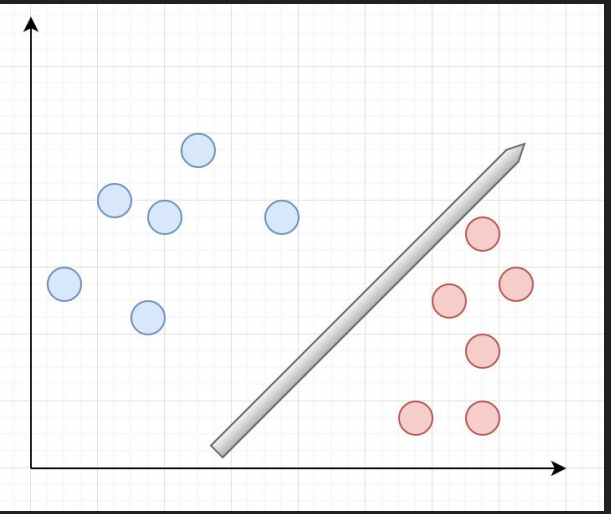
Imagine we have red class and blue class, and plotted, they look like this graph.

Option 3

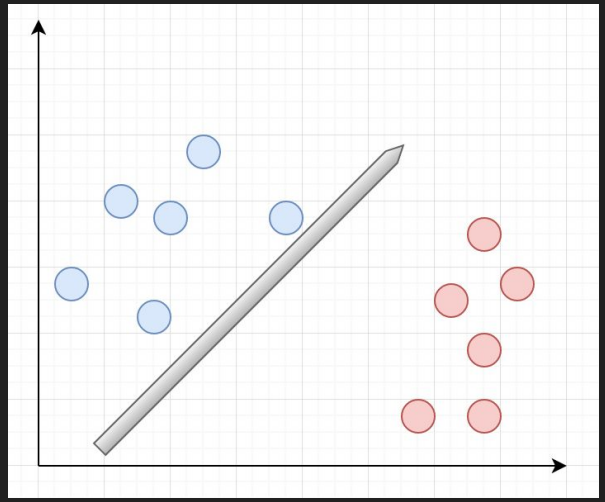


Support Vector Machine

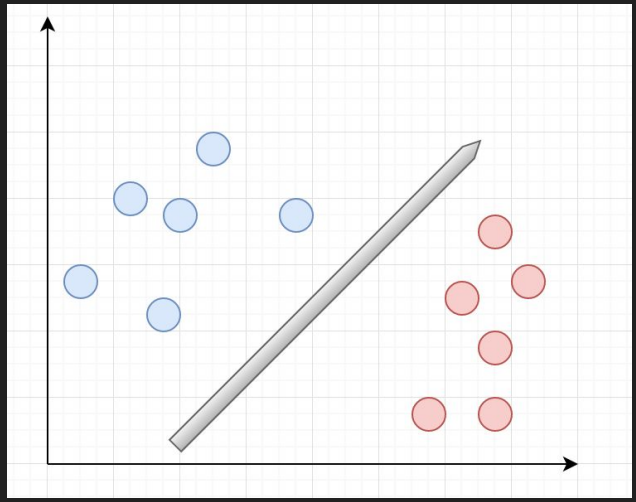
Which is best?



Option 1



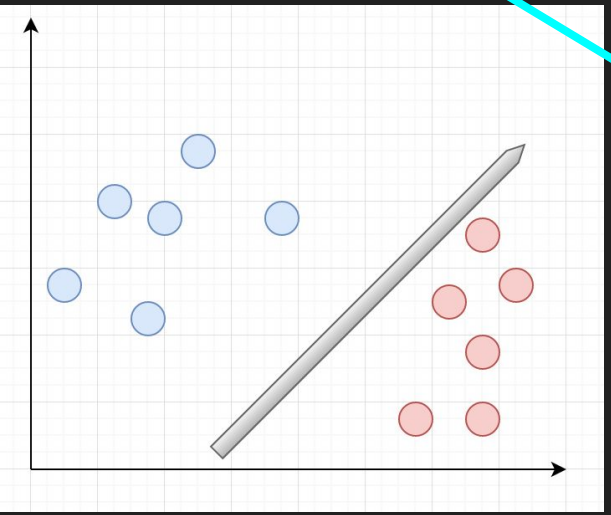
Option 2



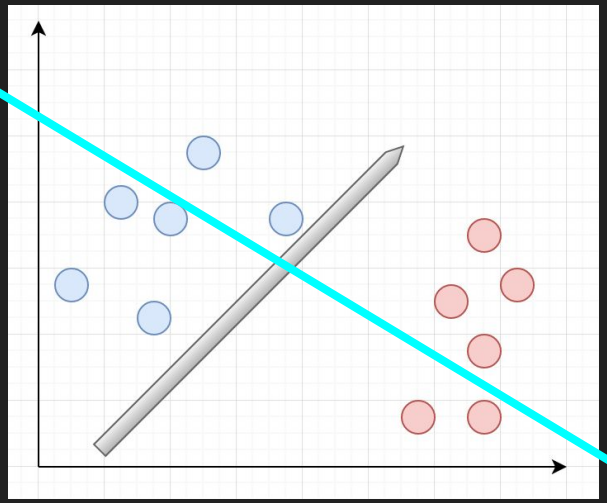
Option 3

Support Vector Machine

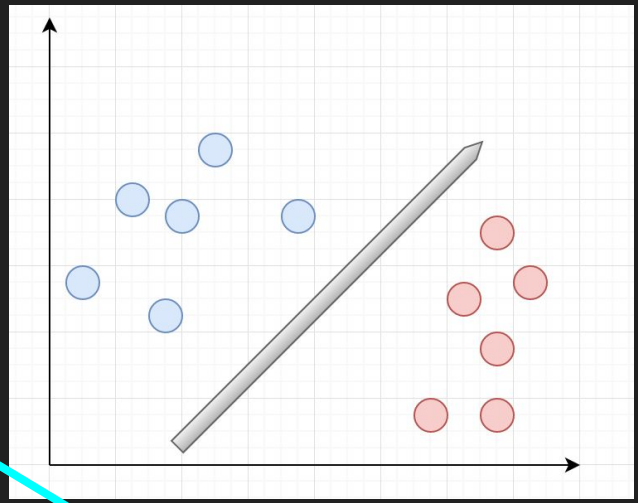
Winner!



Option 1



Option 2

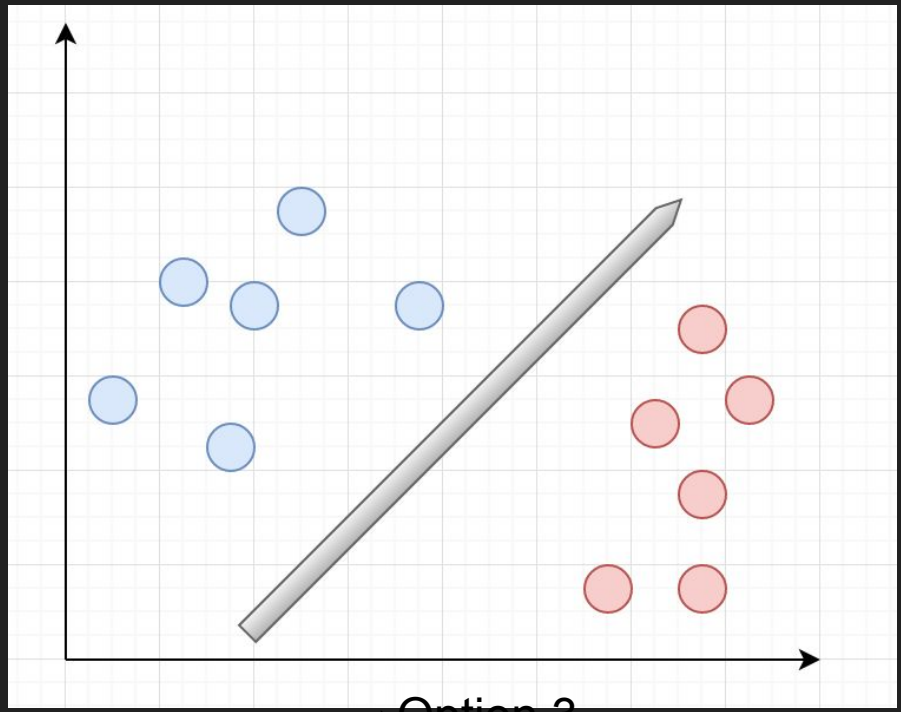


Option 3

Support Vector Machine

Winner. Why?

What specific conditions can you define that makes this optimal?

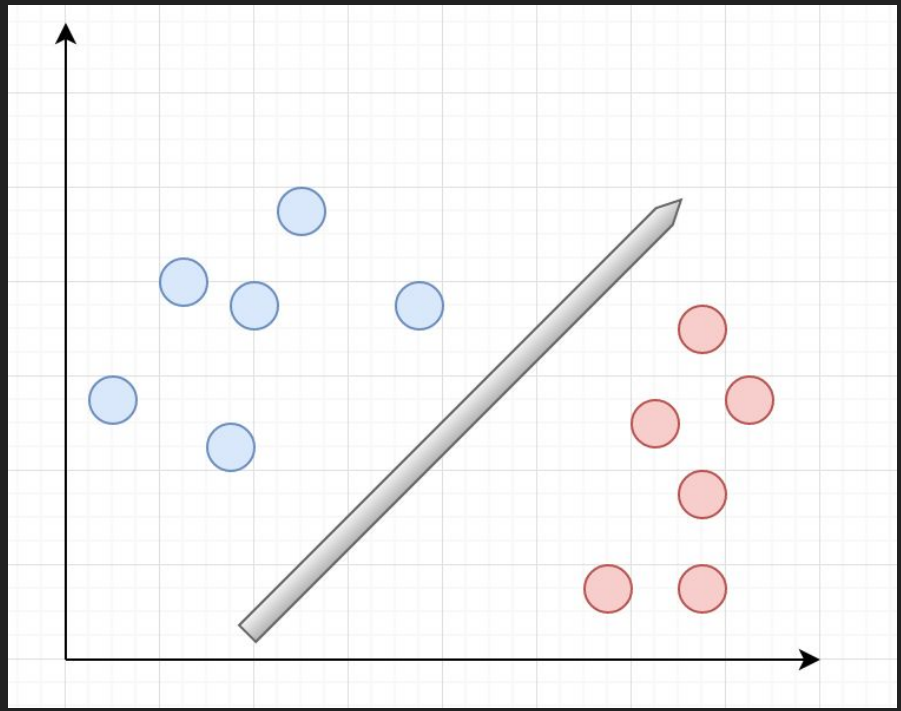


- Option 3

Support Vector Machine

In general, lots of possible solutions for a,b,c.

Support vector machine finds an optimal solution (with respect to what cost?)



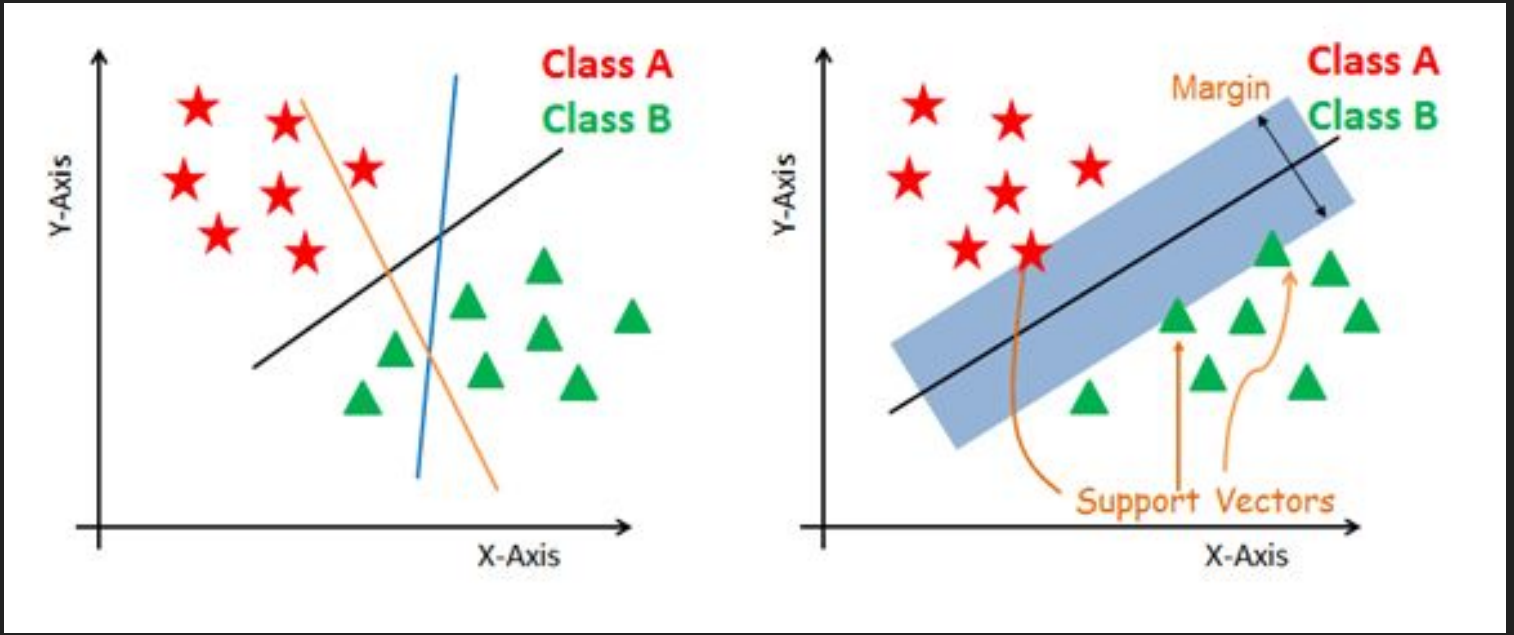
Support Vector Machine

A Support Vector Machine is a binary linear classifier whose decision boundary is explicitly constructed to minimize generalization error

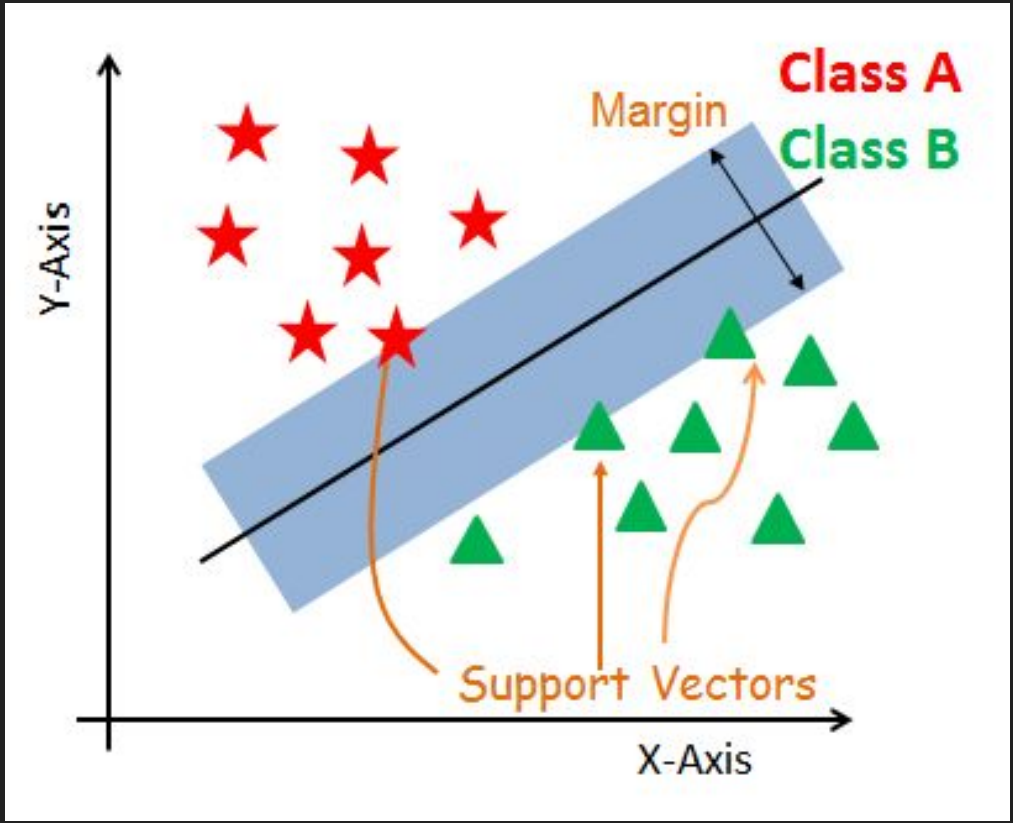
Binary classifier: solves a two-class problem

Linear classifier: creates a linear decision boundary

Support Vector Machines

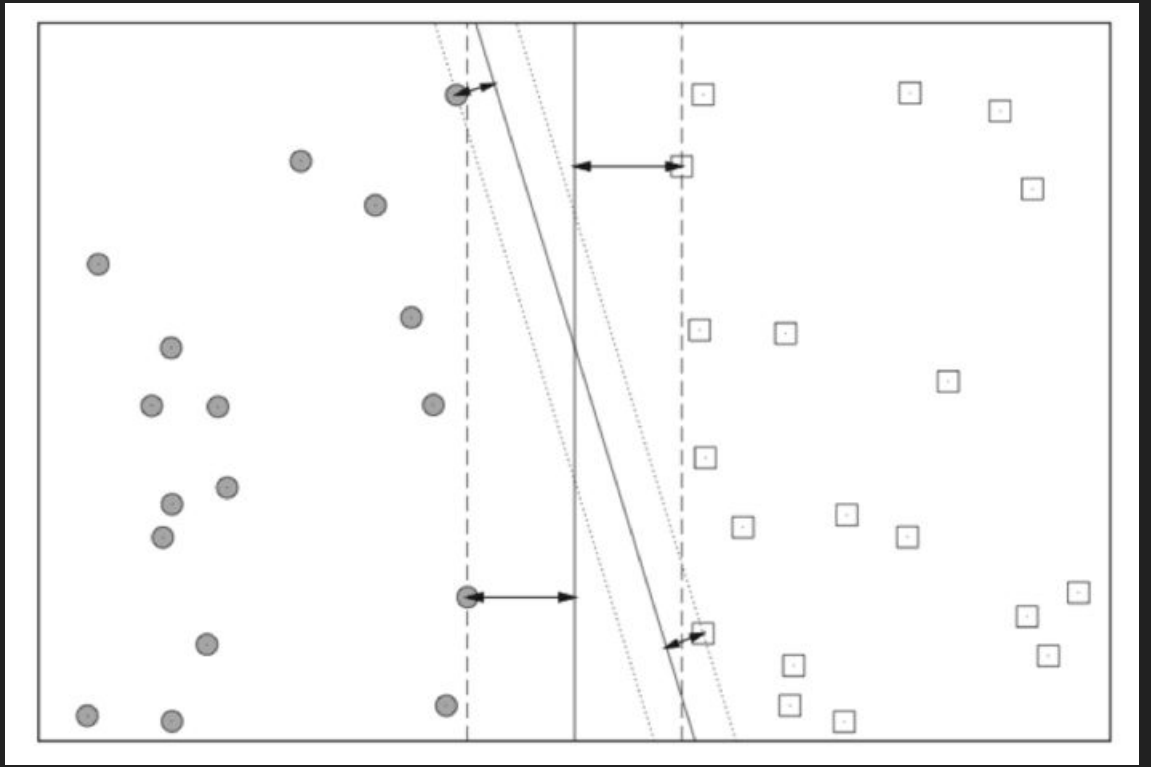


Support Vector Machine



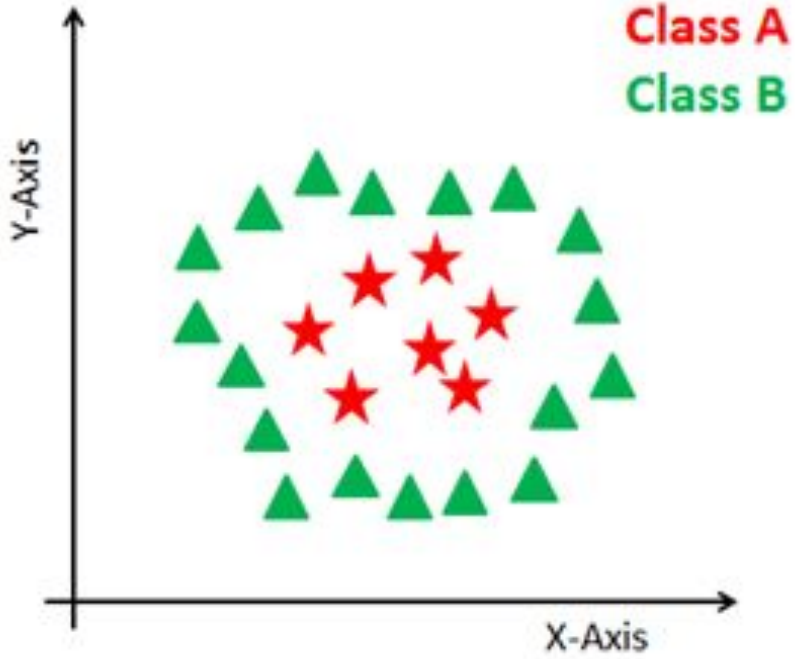
Support Vector Machine

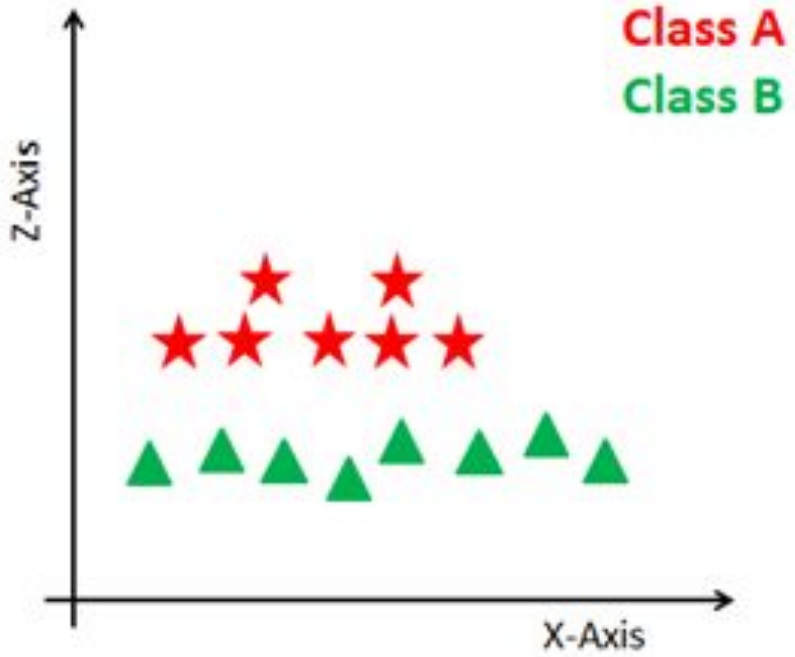
The decision boundary is derived using geometric reasoning (as opposed to the algebraic reasoning we've used to derive other classifiers). The generalization error is equated with the geometric concept of margin, which is the region along the decision boundary that is free of data points.



SVM Pillowcase

Support Vector Machine





Support Vector Machine

The goal of an SVM is to create the linear decision boundary with the largest margin. This is commonly called the maximum margin hyperplane (MMH).

Nonlinear applications of SVM rely on an implicit (nonlinear) mapping that sends vectors from the original feature space K into a higher-dimensional feature space K’. Nonlinear classification in K is then obtained by creating a linear decision boundary in K’. In practice, this involves no computations in the higher dimensional space, thanks to what is called the kernel trick.

Support Vector Machines

Linear Kernel: A linear kernel can be used as normal dot product any two given observations. The product between two vectors is the sum of the multiplication of each pair of input values. K(x, xi) = sum(x * xi)

Polynomial Kernel: A polynomial kernel is a more generalized form of the linear kernel. The polynomial kernel can distinguish curved or nonlinear input space. K(x,xi) = 1 + sum(x * xi)^d

> *Where d is the degree of the polynomial. d=1 is similar to the linear transformation. The degree needs to be manually specified in the learning algorithm.*

Radial Basis Function Kernel The Radial basis function kernel is a popular kernel function commonly used in support vector machine classification. RBF can map an input space in infinite dimensional space. K(x,xi) = exp(-gamma * sum((x – xi^2))

> *Here gamma is a parameter, which ranges from 0 to 1. A higher value of gamma will perfectly fit the training dataset, which causes over-fitting. Gamma=0.1 is considered to be a good default value. The value of gamma needs to be manually specified in the learning algorithm.*

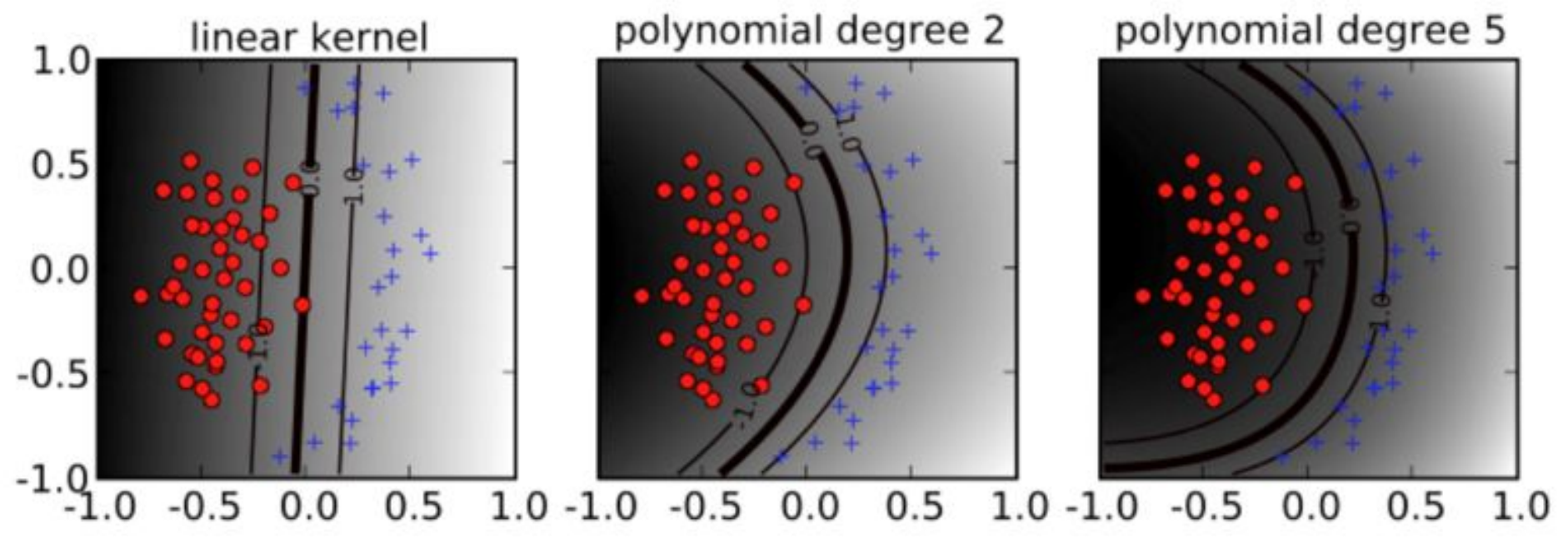
Support Vector Machine: Hyperparameters

Kernel: Polynomial and RBF are useful for non-linear hyperplane.

Regularization: Regularization parameter in python's Scikit-learn C parameter used to maintain regularization. C is the penalty parameter, which represents misclassification or error term. A smaller value of C creates a small-margin hyperplane and a larger value of C creates a larger-margin hyperplane.

Gamma: A lower value of Gamma will loosely fit the training dataset, whereas a higher value of gamma will exactly fit the training dataset, which causes over-fitting.

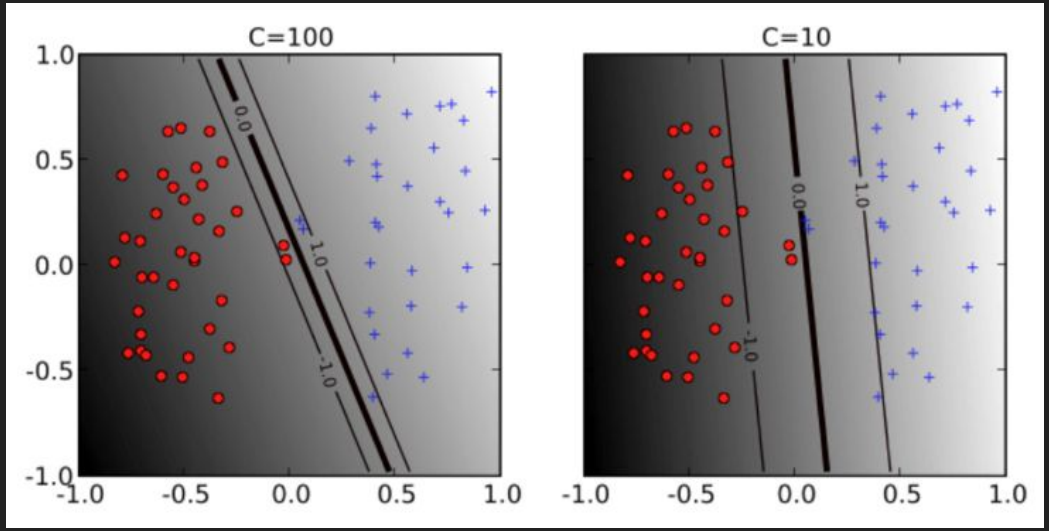
Support Vector Machine



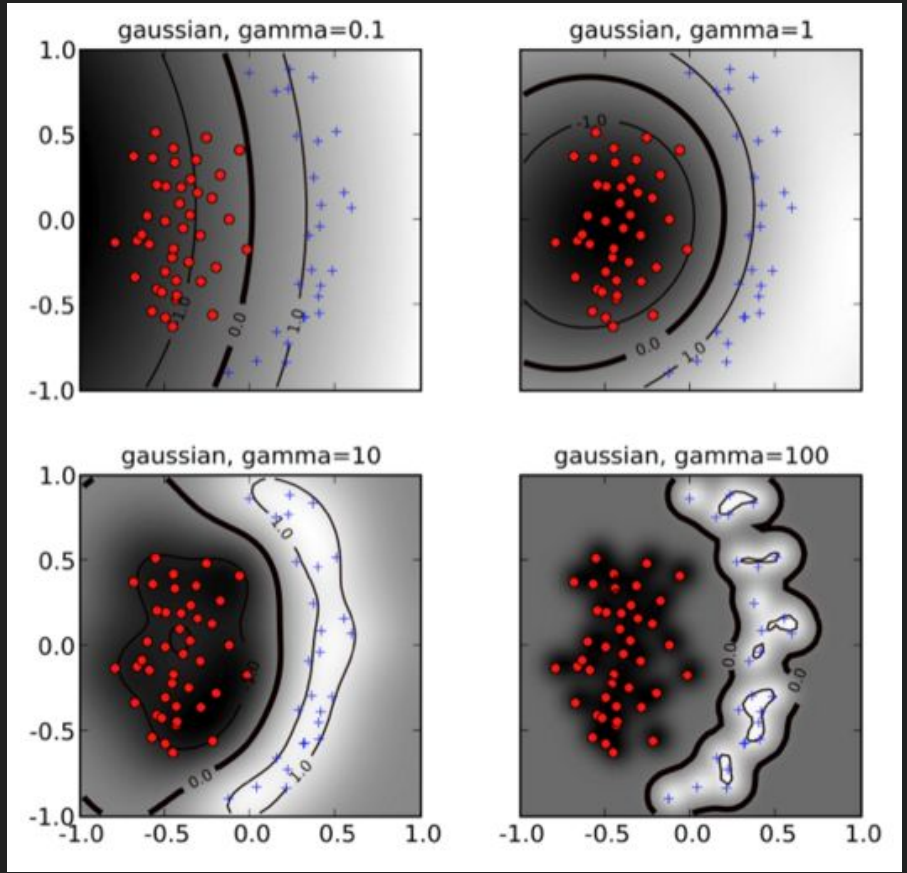
Support Vector Machines

Soft margin, slack variables-class overlap is achieved by relaxing the minimization problem or softening the margin.

The hyper-parameter C (soft-margin constant) controls the overall complexity by specifying penalty for training error. This is regularization.



Support Vector Machine



Support Vector Machines

Advantages

Good accuracy and perform faster prediction.

Use less memory because they use a subset of training points in the decision phase.

SVM works well with a clear margin of separation and with high dimensional space.

Disadvantages

Not suitable for large datasets because of its high training time

Works poorly with overlapping classes.

Sensitive to the type of kernel used.

Decision Trees

In the game "20 Questions," an individual or group has the opportunity to identify an unknown object by asking a series of up to twenty "yes" or "no" questions.

We can think about all possible "target objects" and then ask questions that help us to quickly pare down the number of objects so that we can identify the true target in twenty of fewer questions.

When playing this game, what are some good strategies? What about poor strategies?

For example, suppose that I am trying to guess the item "marker" I might ask a series of questions like:

- Is the object in this room? (Yes.)
- Is the object within five feet of me? (Yes.)
- Is the object larger than a loaf of bread? (No.)
- Is there only one of these objects in the room? (No.)
- Do you hold this object when you use it? (Yes.)
- Is the object a pen? (No.)
- Is the object a book? (No.)
- Is the object a marker? (Yes!)

Decision Trees

Let’s play a game!

https://en.akinator.com/

How does this game work?

Decision Trees

Decision trees are a widely popular and powerful machine learning technique for both classification and regression problems.

To perform classification or regression, decision trees make sequential, hierarchical decisions about the outcome variable based on the predictor data.

Break down our data by asking a series of questions.

They are great if we care about interpretability.

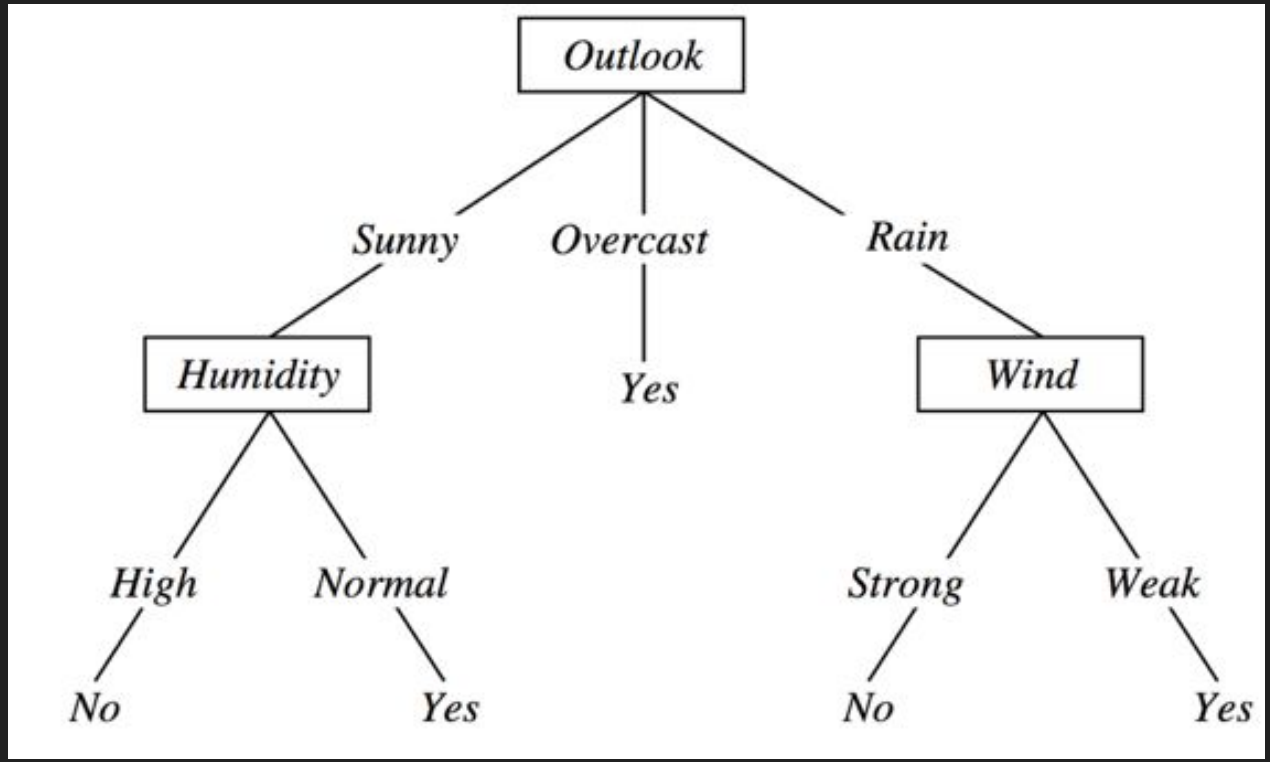
Decision Tree

Decision tree models are hierarchical and non-parametric.

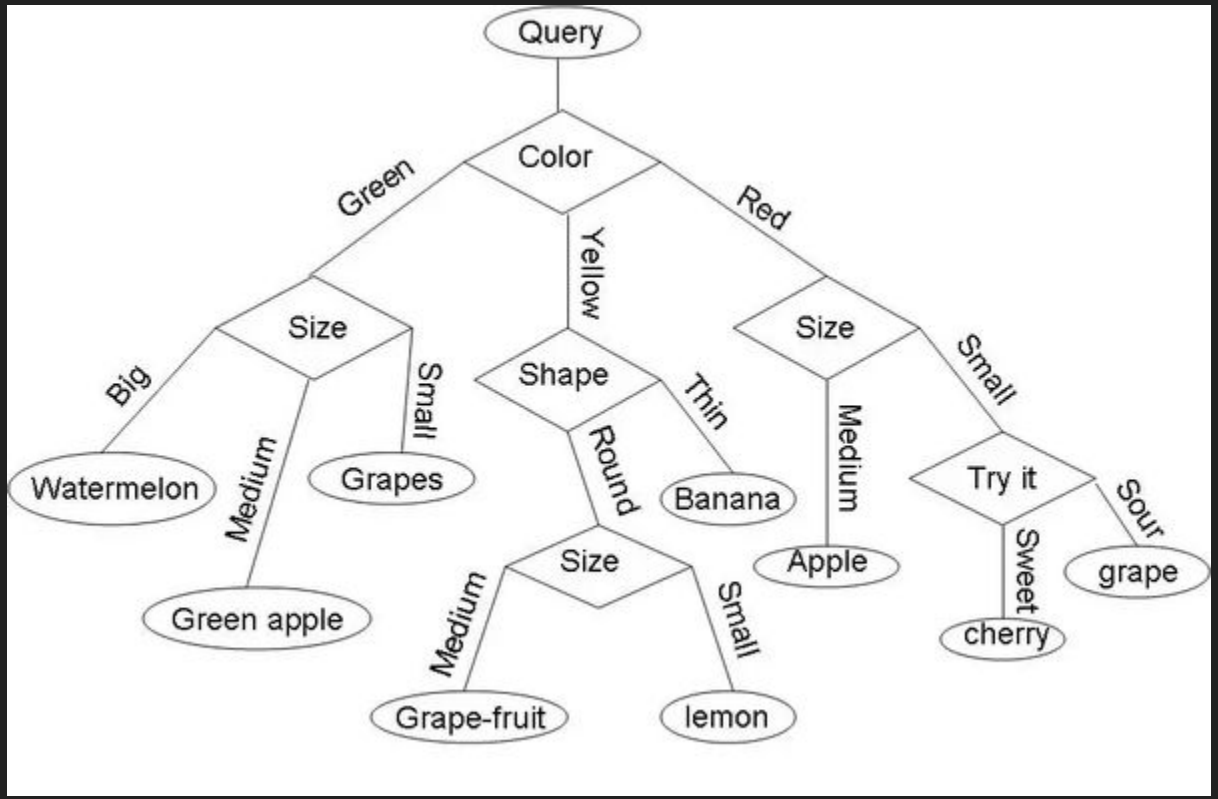
Hierarchical means that the model is defined by a sequence of questions which yield a class label or value when applied to any observation. Once trained, the model behaves like a recipe, a series of "if this then that" conditions that yields a specific result for our input data.

Non-parametric methods stand in contrast to models like logistic regression or ordinary least squares regression. There are no underlying assumptions about the distribution of the data or the errors. Non-parametric models essentially start with no assumed parameters about the data and construct them based on the observed data.

Decision Tree



Fruit Decision Tree



Decision Trees

Splits can be binary way or multi-way.

Features can be categorical or continuous.

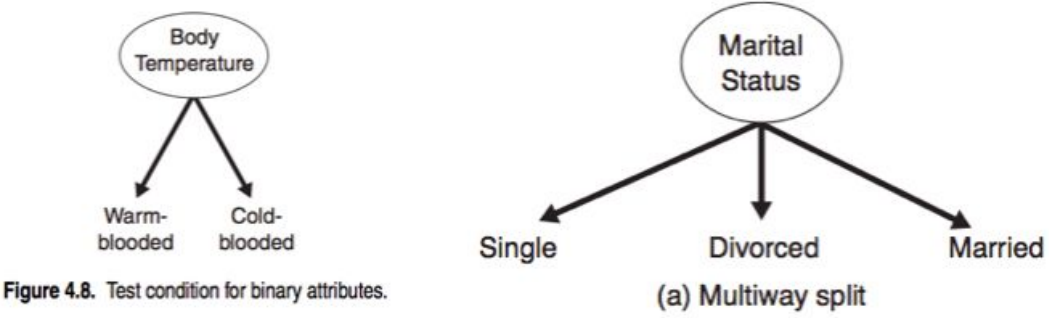


Figure 4.8. Test condition for binary attributes.

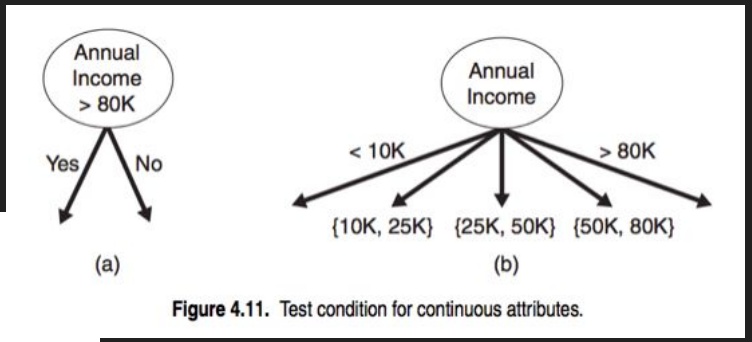


Figure 4.11. Test condition for continuous attributes.

Decision Trees

Start at the tree root and split the data on the feature that results in the largest information gain.

Repeat this splitting procedure until all the leaves are pure.

> “Pure” means all the samples at each node belong to the same class.

This can result in a very “deep” tree with many nodes (overfitting).

> So we usually want to prune trees by setting a maximum depth.

Decision Trees

Use an objective function to maximize information gain--splitting the nodes on the most informative features.

Information gain is the difference between the impurity of the parent node and the sum of the child node impurities. (Lower impurity of child nodes, the higher the information gain)

Decision Trees

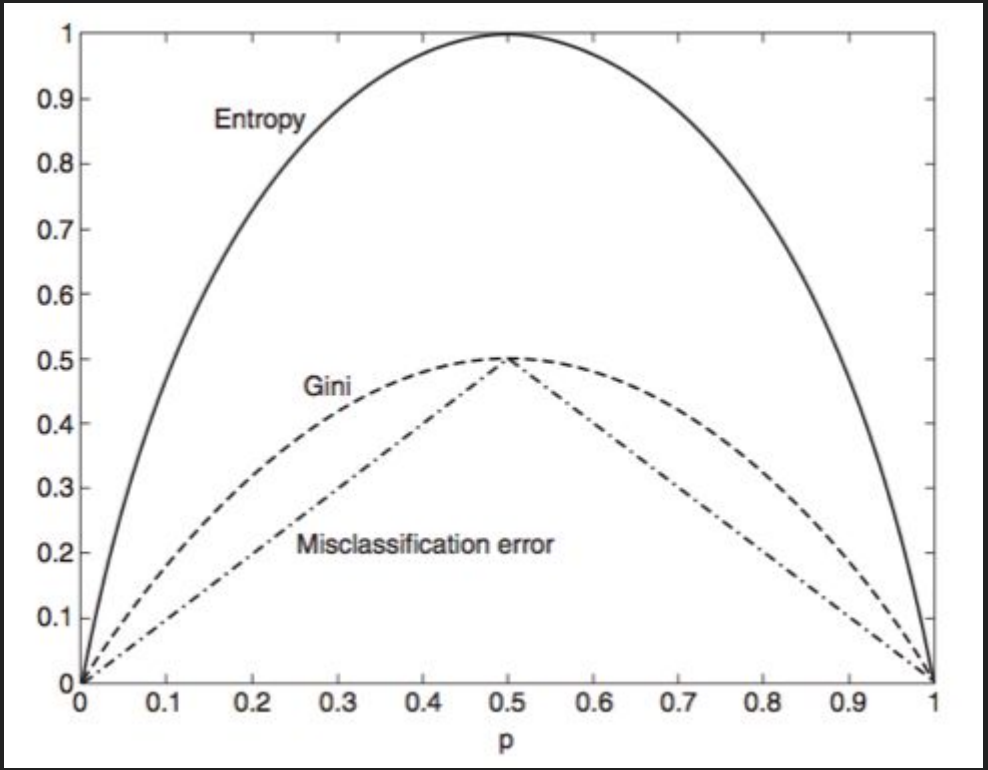
Three impurity measures (splitting criteria):

- Gini impurity (most used)
- Entropy
- Classification error

In practice, Gini and entropy yield pretty similar results and it’s more worth your time experimenting with different pruning cut-offs than with different impurity measures.

Classification error is useful for pruning but not for growing trees.

Decision Trees



Decision Trees

Addressing overfitting:

A stopping criterion determines when to no longer construct further nodes.

We can stop when all records belong to the same class, or when all records have the same attributes. This maximizes variance at the expense of bias, leading to overfitting.

Setting a maximum depth: A simple way to prevent overfitting is to set a hard limit on the "depth" of the decision tree.

Minimum observations to make a split: An alternative to maximum depth (and can be used at the same time), is to specify the minimum number of datapoints required to make a split at a node.

Decision Trees: Advantages

- Simple to understand and interpret.
- Requires little data preparation.
- Able to handle both numerical and categorical data.
- Uses a white box model (boolean logic)
- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.
- Robust. Performs well even if its assumptions are somewhat violated by the true model from which the data were generated.
- Performs well with large datasets. Large amounts of data can be analyzed using standard computing resources in reasonable time.
- Once trained can be implemented on hardware and has extremely fast execution.

Decision Trees: Disadvantages

- Locally-optimal--practical decision-tree learning algorithms are based on heuristics such as the greedy algorithm where locally-optimal decisions are made at each node.
- Such algorithms cannot guarantee to return the globally-optimal decision tree.
- Overfitting--decision-tree learners can create over-complex trees that do not generalize well from the training data.
- There are concepts that are hard to learn because decision trees do not express them easily.In such cases, the decision tree becomes prohibitively large.
- Decision tree learners create biased trees if some classes dominate. It is therefore recommended to balance the dataset prior to fitting with the decision tree.

Classifiers: Pros and Cons

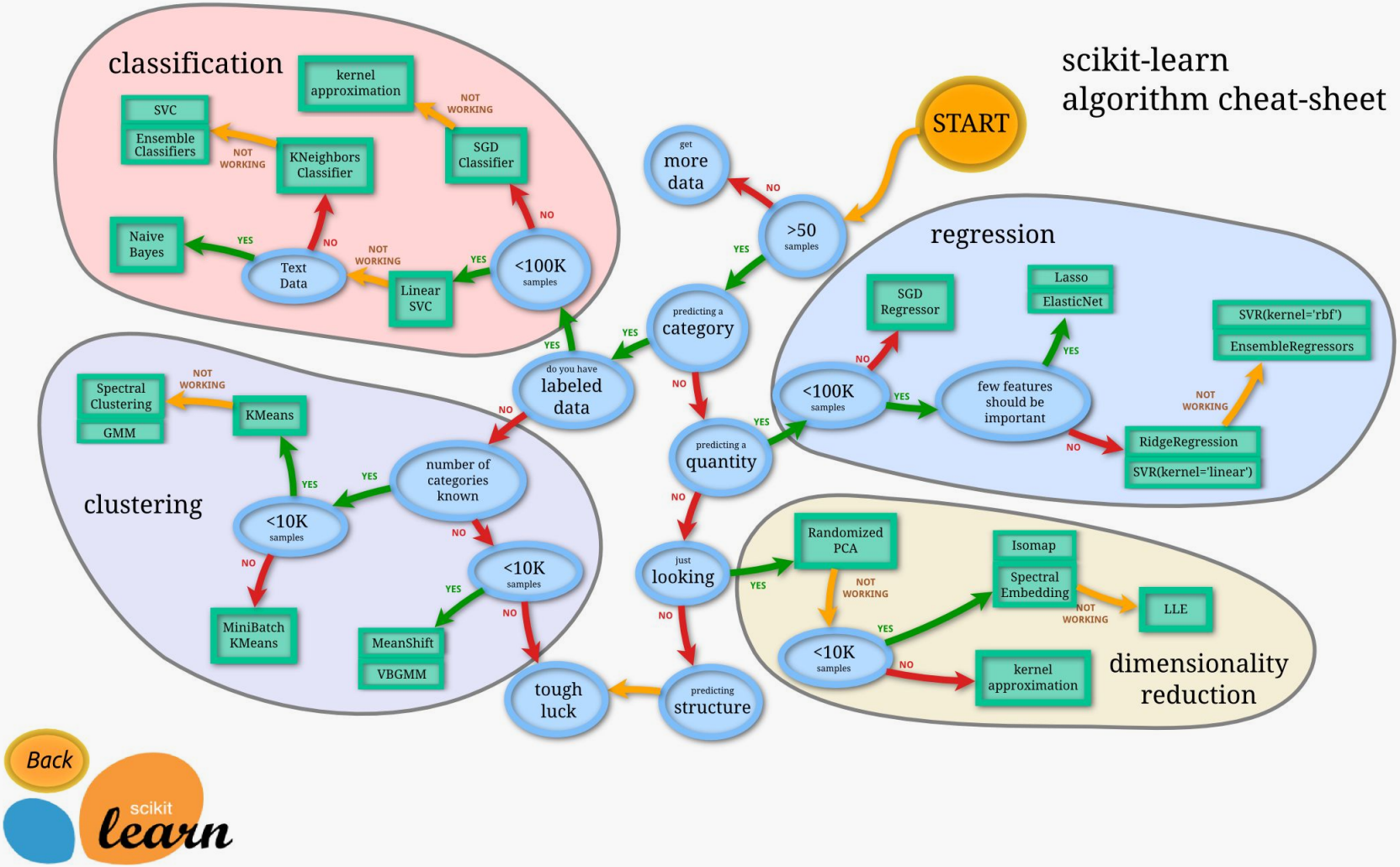
Decision trees are good if we care about interpretability but overfit easily.

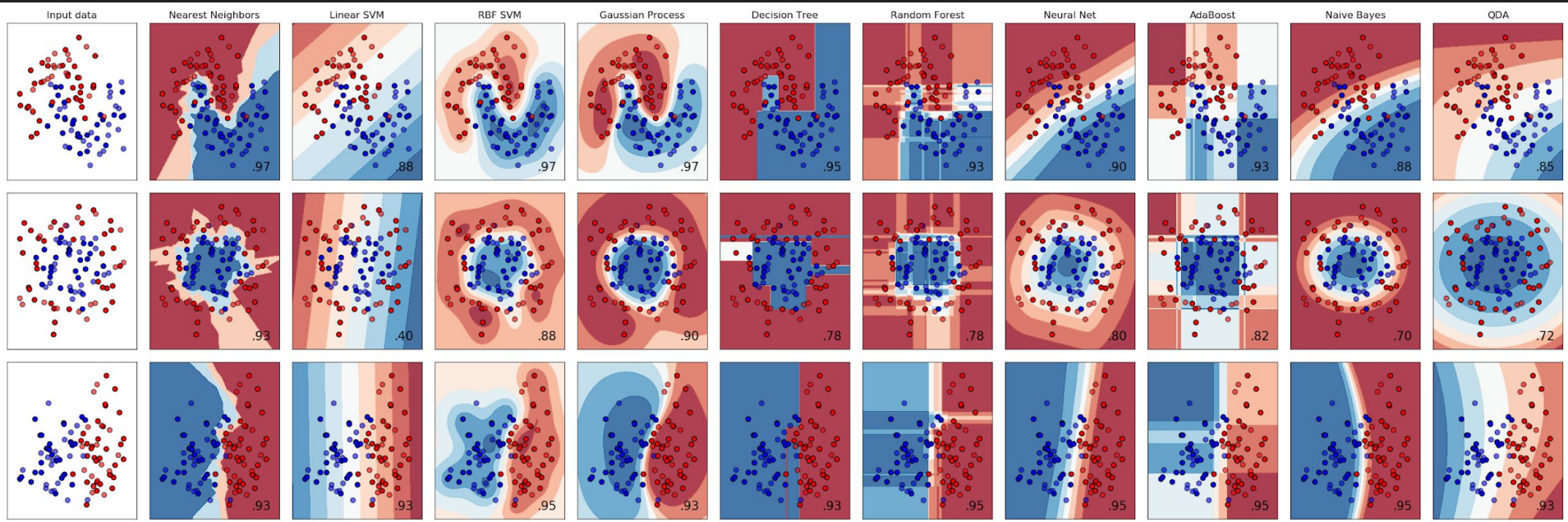
Logistic regression allows us to predict probabilities.

Support vector machines are powerful linear models that can be extended to nonlinear problems but have lots of parameters to tune.

Ensemble methods (e.g. random forest) don’t need much parameter tuning and don’t overfit easily.

K-nearest neighbors makes predictions without any model training but is computationally expensive


scikit-learn
algorithm cheat-sheet
START
get more data
NO
>50 samples
YES
predicting a category
YES
NO
do you have labeled data
YES
NO
predicting a quantity
YES
NO
just looking
YES
NO
predicting structure
tough luck
classification
kernel approximation
NOT WORKING
SVC
Ensemble Classifiers
NOT WORKING
KNeighbors Classifier
SGD Classifier
NO
Naive Bayes
YES
Text Data
NO
NOT WORKING
Linear SVC
YES
<100K samples
YES
clustering
Spectral Clustering
GMM
NOT WORKING
KMeans
YES
number of categories known
YES
<10K samples
NO
MiniBatch KMeans
NO
<10K samples
YES
MeanShift
VBGMM
NO
regression
SGD Regressor
NO
<100K samples
YES
few features should be important
YES
Lasso
ElasticNet
NO
SVR(kernel='rbf')
EnsembleRegressors
NOT WORKING
RidgeRegression
SVR(kernel='linear')
dimensionality reduction
Randomized PCA
YES
NOT WORKING
<10K samples
YES
Isomap
Spectral Embedding
NOT WORKING
LLE
NO
kernel approximation
Back
scikit learn


Input data
Nearest Neighbors
Linear SVM
RBF SVM
Gaussian Process
Decision Tree
Random Forest
Neural Net
AdaBoost
Naive Bayes
QDA
.97
.88
.97
.97
.95
.93
.90
.93
.88
.85
.93
.40
.88
.90
.78
.78
.80
.82
.70
.72
.93
.93
.95
.93
.93
.95
.95
.95
.95
.93