Last session (2015-10-22)

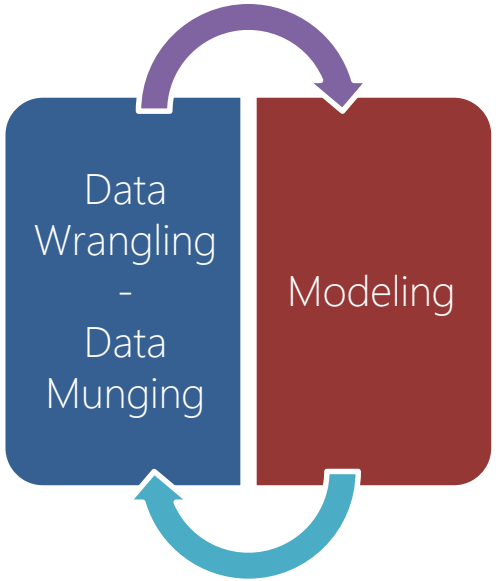
- Big Data Analysis Process
  - Modeling
- Hands-on

Today's session

- Big Data Analysis Process
  - More modeling
- Hands-on – Project 3

Big Data Analysis Process – Main Steps

- Data access
- Data pre-processing / cleaning
- Data transformation / manipulation
- Feature selection
- Feature extraction
- Feature engineering
- Model choice and training
- Model evaluation and tuning
- Model deployment



Modeling

Selected Machine Learning Model Types

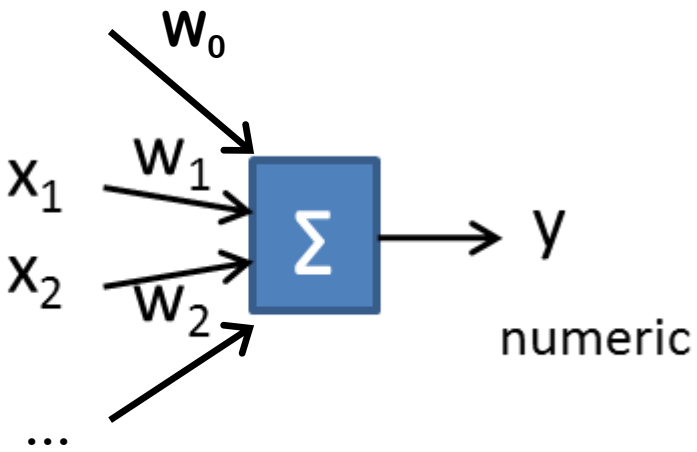
- Linear Regression
- Logistic Regression
- K-Nearest Neighbours
- Decision Trees
- Neural Networks
- Support Vector Machines

- ..... Ensembles of the above

Linear Regression

Tries to model the output variable (target) as a linear combination of the input variables (features)

$y = W_0 + W_1x_1 + W_2x_2 + \ldots$



Linear Regression

Linear regression finds the line that best fits the data points

There are actually a number of different definitions of "best fit," and therefore a number of different methods of linear regression

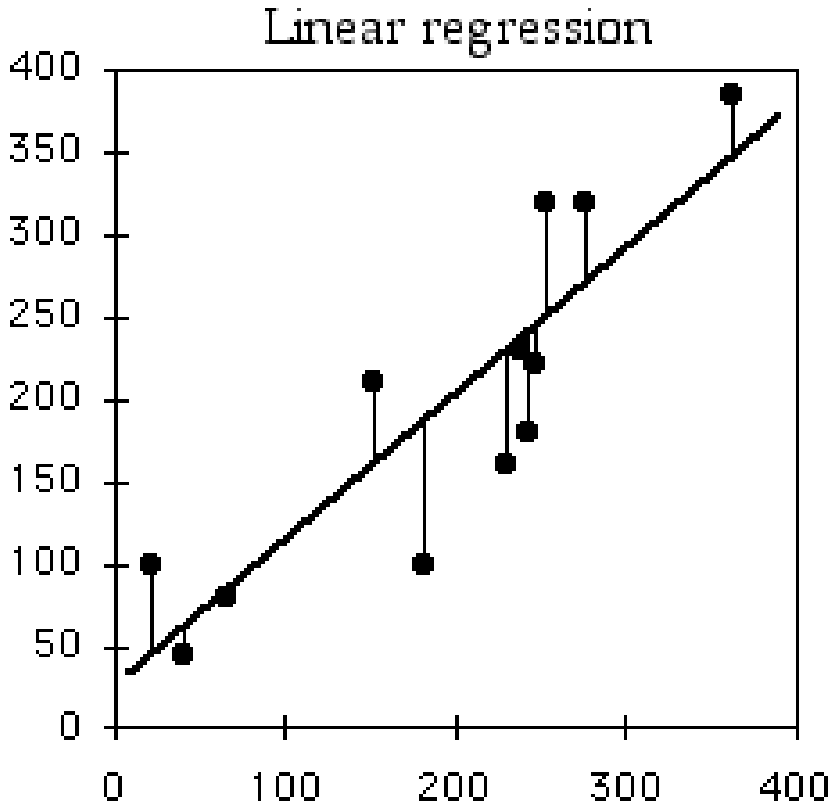
By far the most common is "ordinary least-squares regression"

- minimizes the sum of the squared distances between the points and the line

Least-squares regression

minimizes the sum of the squared distances between the points and the line

$$S = \sum_{i=1}^{n} {r_i}^2$$



Linear Regression in Azure ML

- https://msdn.microsoft.com/en-us/library/azure/dn905978.aspx
- Two solution methods available

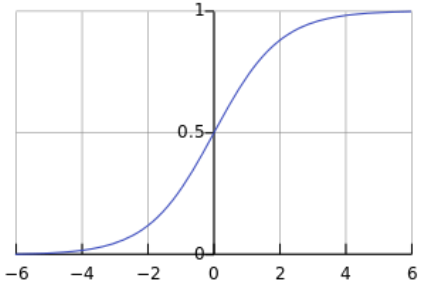
1. Ordinary Least Squares
   - As described earlier
2. Online Gradient Descent
   - Uses a stochastic gradient descent optimizer

What is the equivalent of Linear Regression for a classification problem?

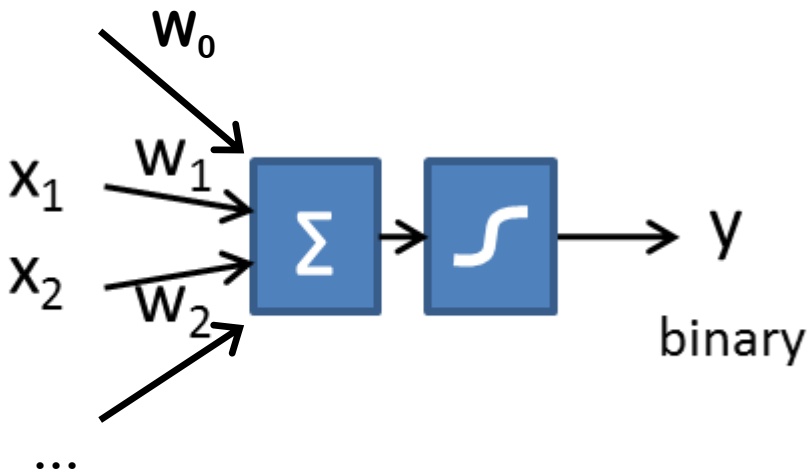
Logistic Regression

In a classification problem, the output is binary rather than numeric

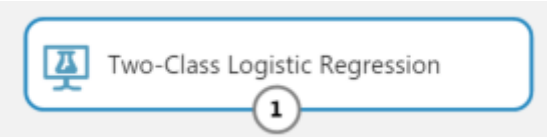
We can imagine doing a linear regression and then compressing the numeric output into a 0..1 range using the sigmoid (logit) function



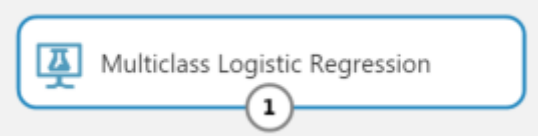
$$S(t) = \frac{1}{1 + e^{-t}}.$$



Logistic Regression in Azure ML



- https://msdn.microsoft.com/en-us/library/azure/dn905994.aspx



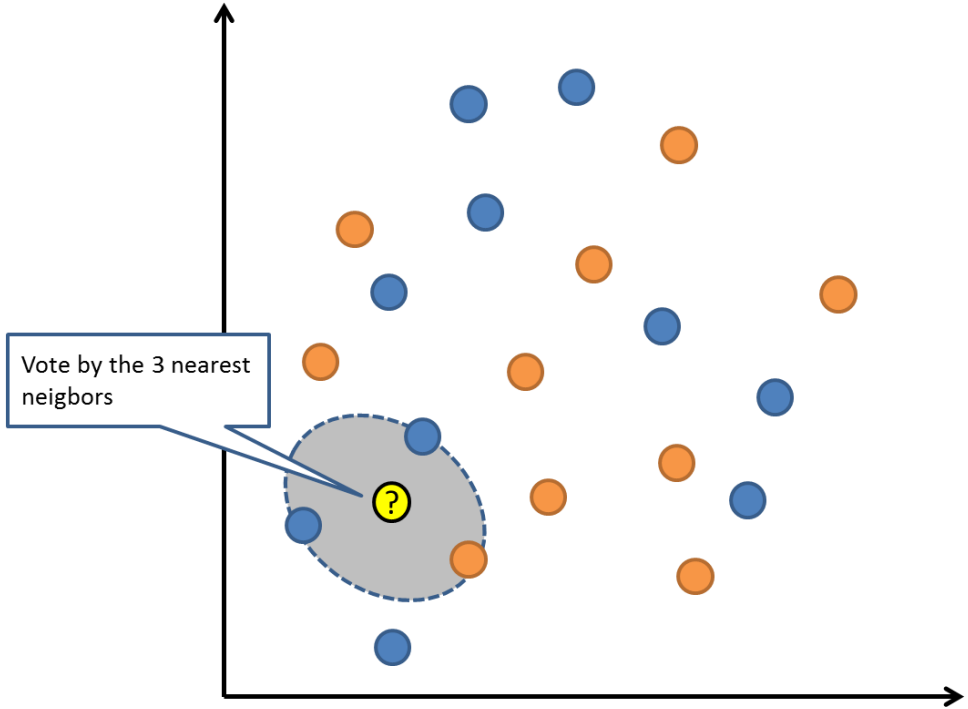
- https://msdn.microsoft.com/en-us/library/azure/dn905853.aspx

k-Nearest-Neighbours - kNN

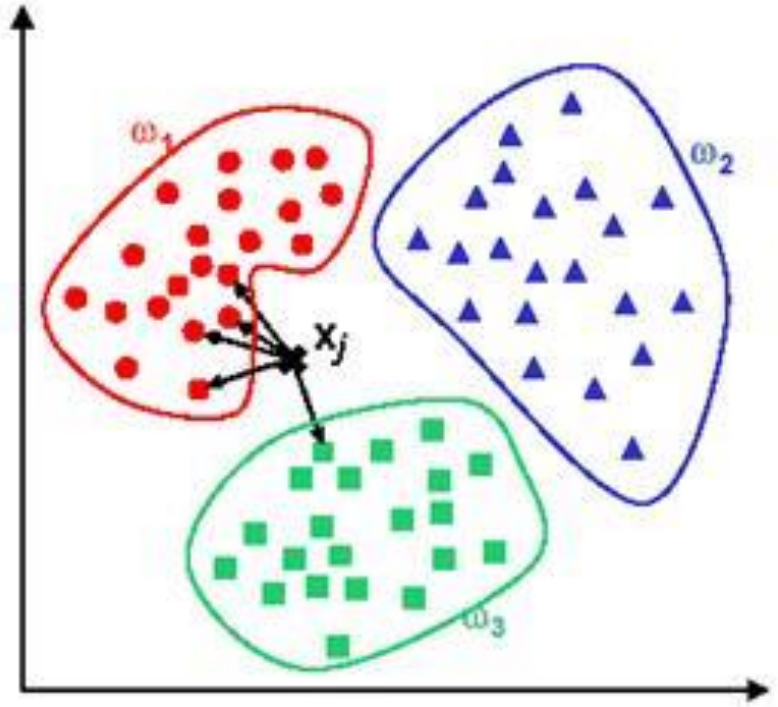
This is also called instance-based learning because it doesn't even learn a single model

The training process involves memorizing all the training data

The voting can also be weighted among the K-neighbors based on their distance from the new data point



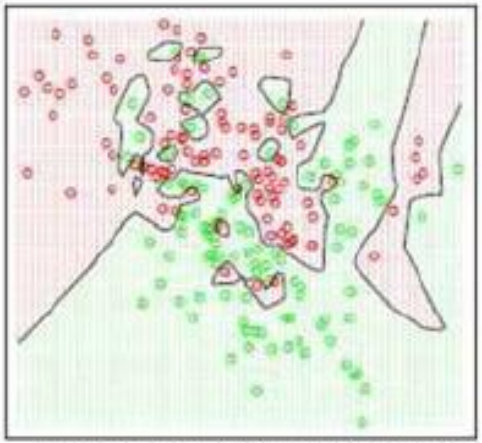
k-Nearest-Neighbours - kNN



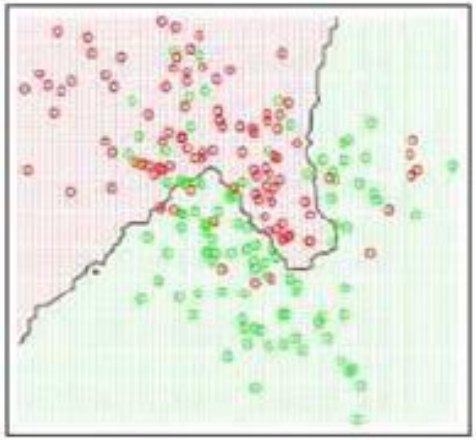
k-Nearest-Neighbours - kNN

Effect of K

K=1



K=15

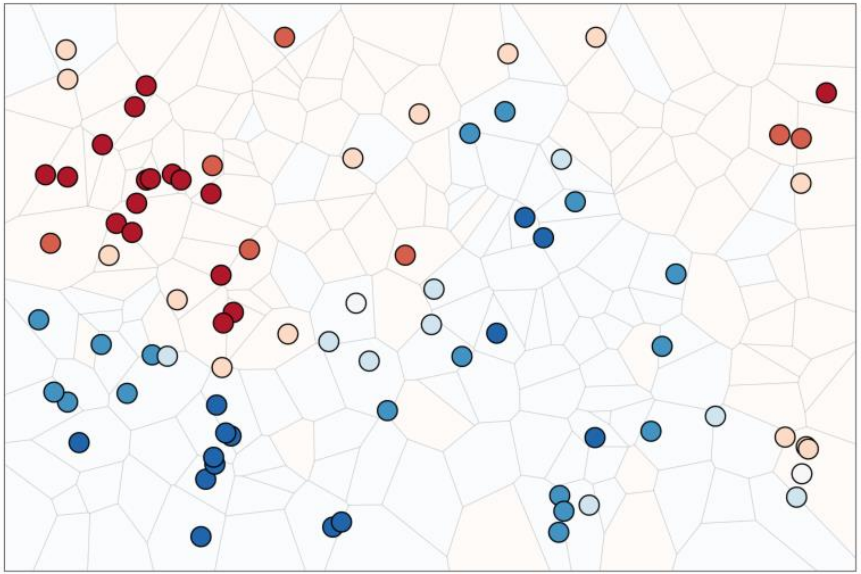


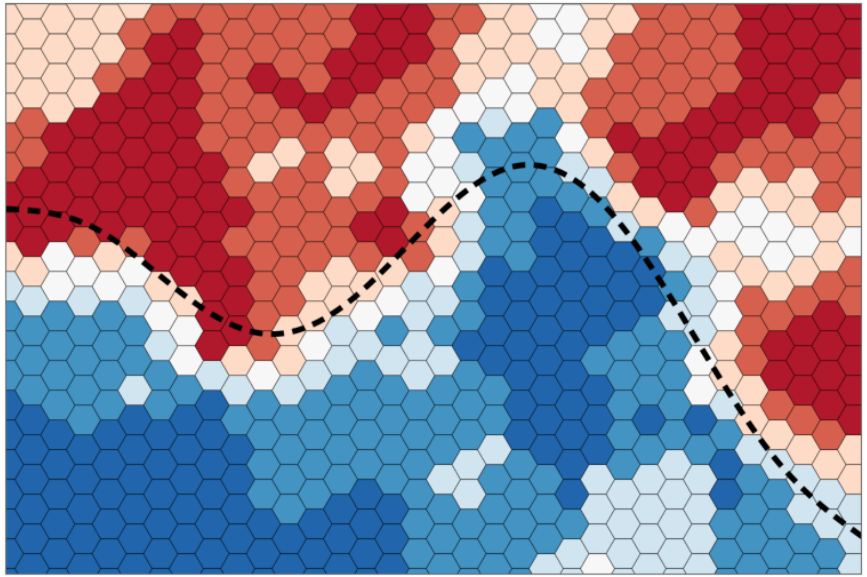
Figures from Hastie, Tibshirani and Friedman (Elements of Statistical Learning)

Larger k produces smoother boundary effect and can reduce the impact of class label noise.

But when k is too large, say k=N, we always predict the majority class

k-Nearest-Neighbours - kNN





http://scott.fortmann-roe.com/docs/BiasVariance.html

k-Nearest-Neighbours in Azure ML

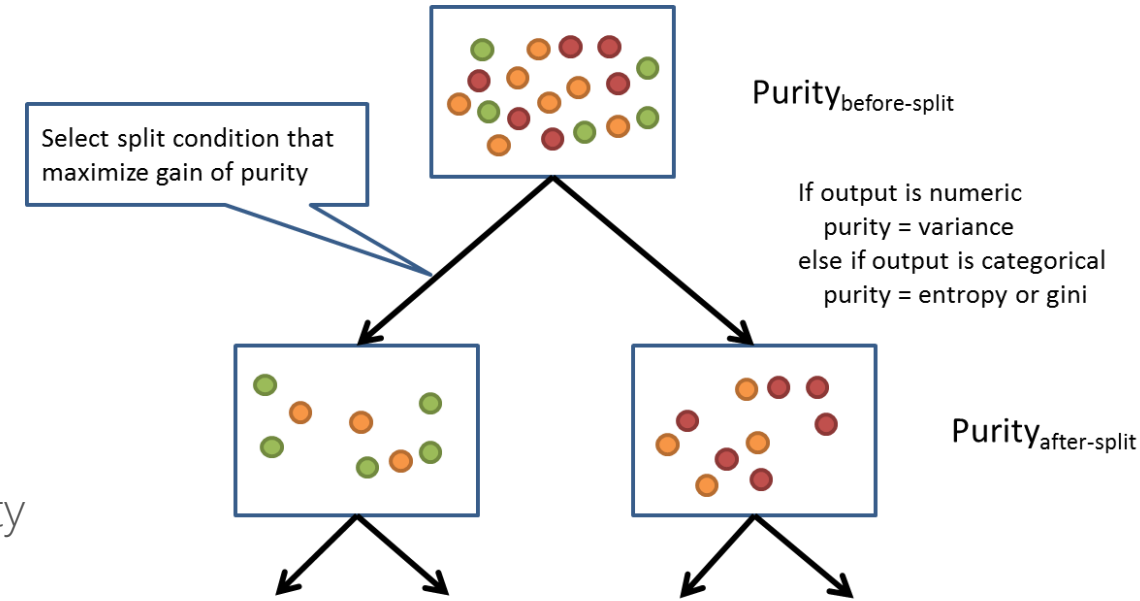
- Not available
- It does not scale well to Big Data!

  (since the model has to "memorize" all the data)

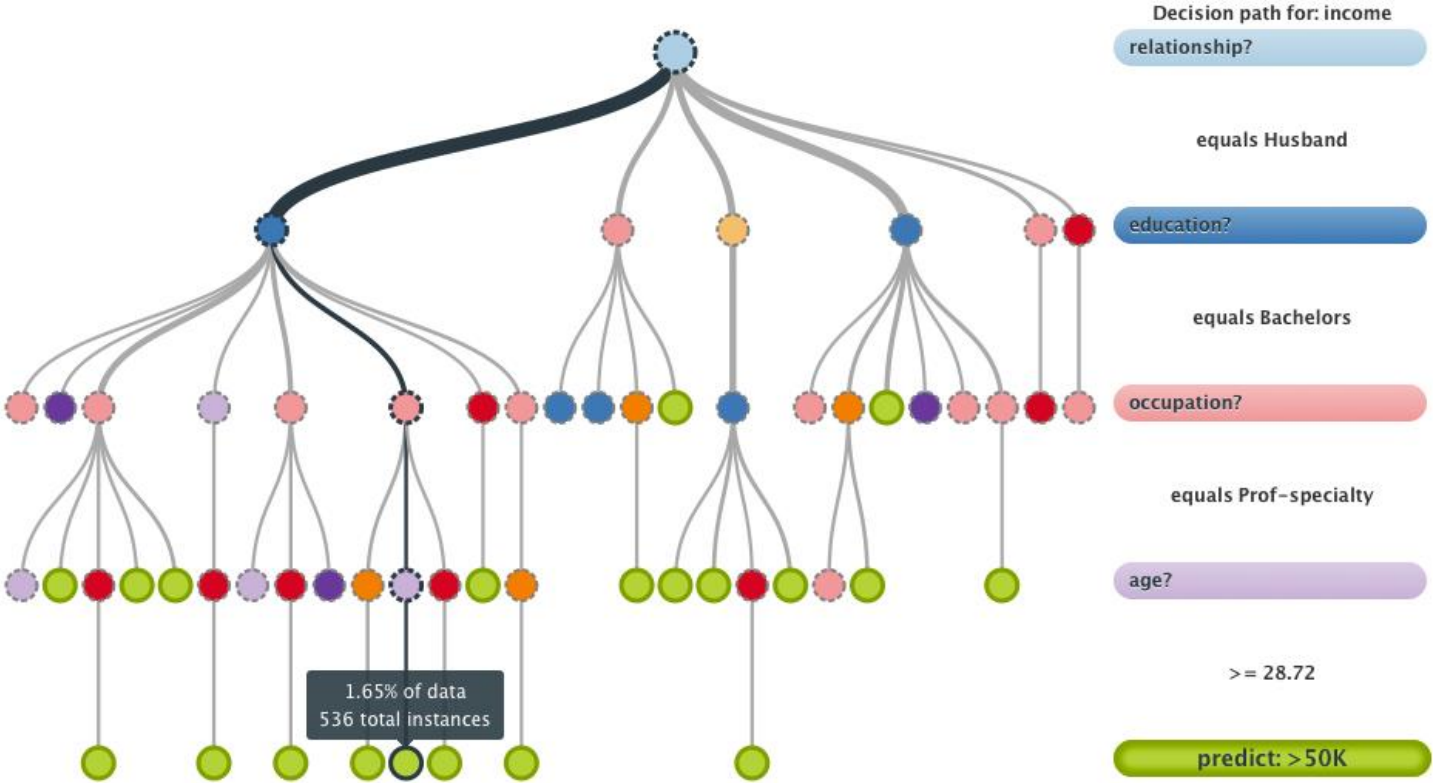
Decision Trees

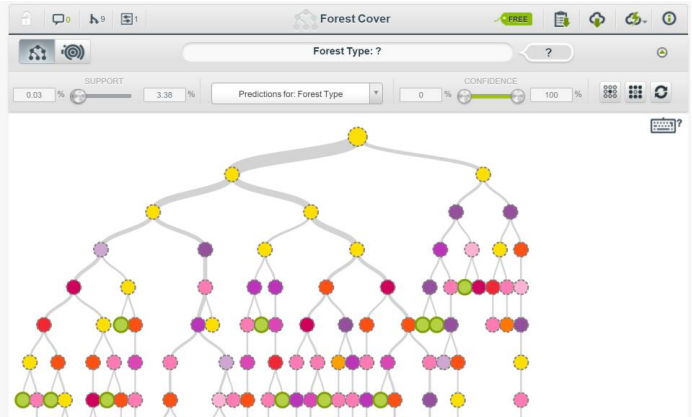
Based on a tree of decision nodes, the learning approach is to progressively divide the training data into buckets of as homogeneous members as possible through the most discriminative dividing criteria possible

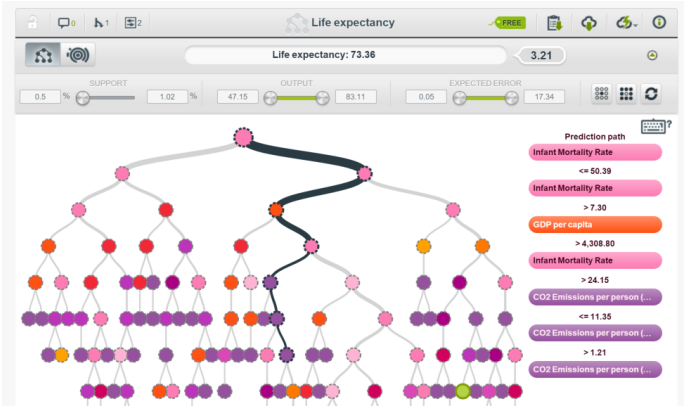
The training process stops when there is no significant gain in purity after further splitting the tree



Decision Trees

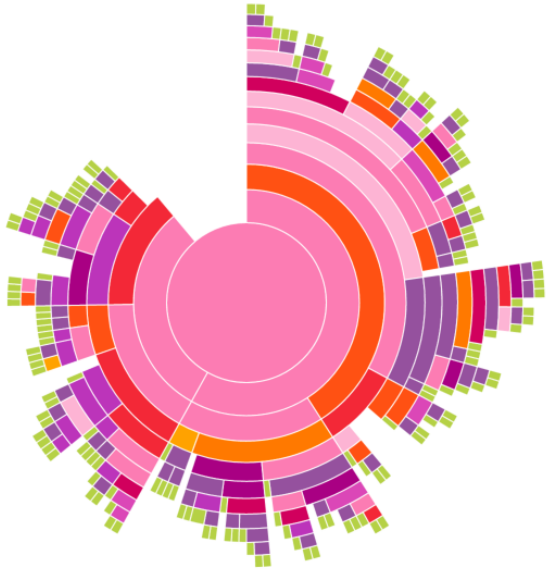






Classification

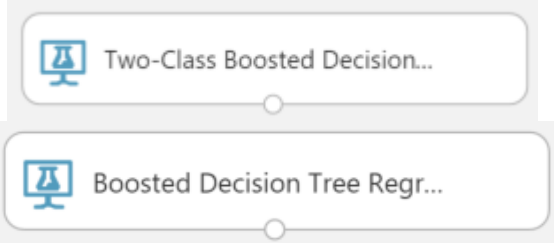
Regression



https://bigml.com/gallery/models

Tree Ensembles in Azure ML

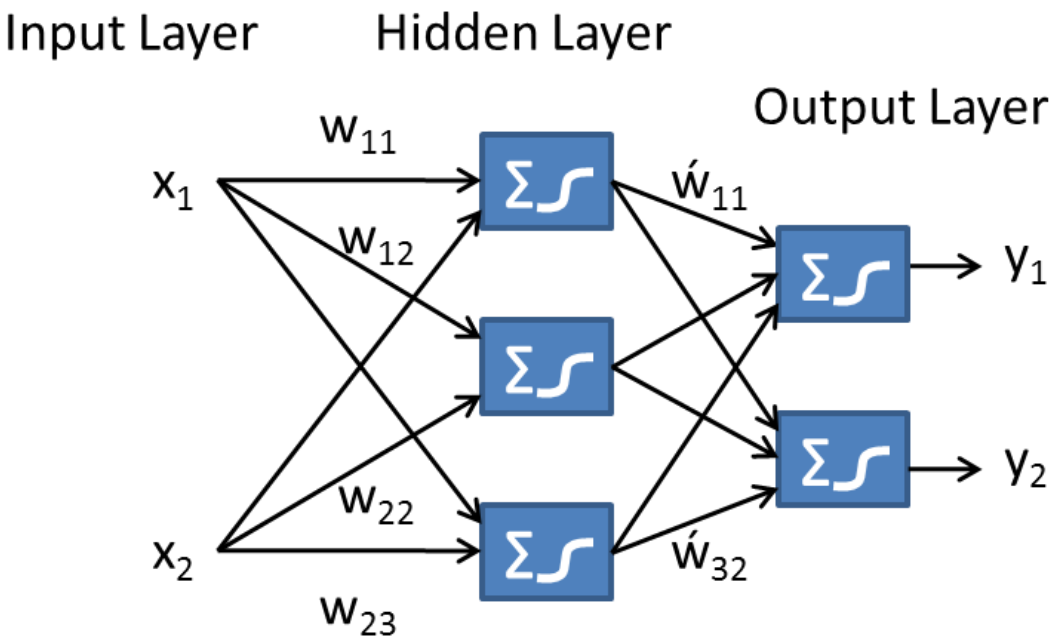
- Decision Trees are prone to overfitting
- Common to use ensembles of decision trees
  - Bagging
    - Each model is trained on a random subset of the data
    - Averaged
    - (https://msdn.microsoft.com/en-us/library/azure/dn905862.aspx)
  - Boosting
    - Incrementally add new models
    - Data previously estimated or classified unaccurately is given higher "weight" in the new model to be added to the ensemble
    - (https://msdn.microsoft.com/en-us/library/azure/dn905801.aspx)



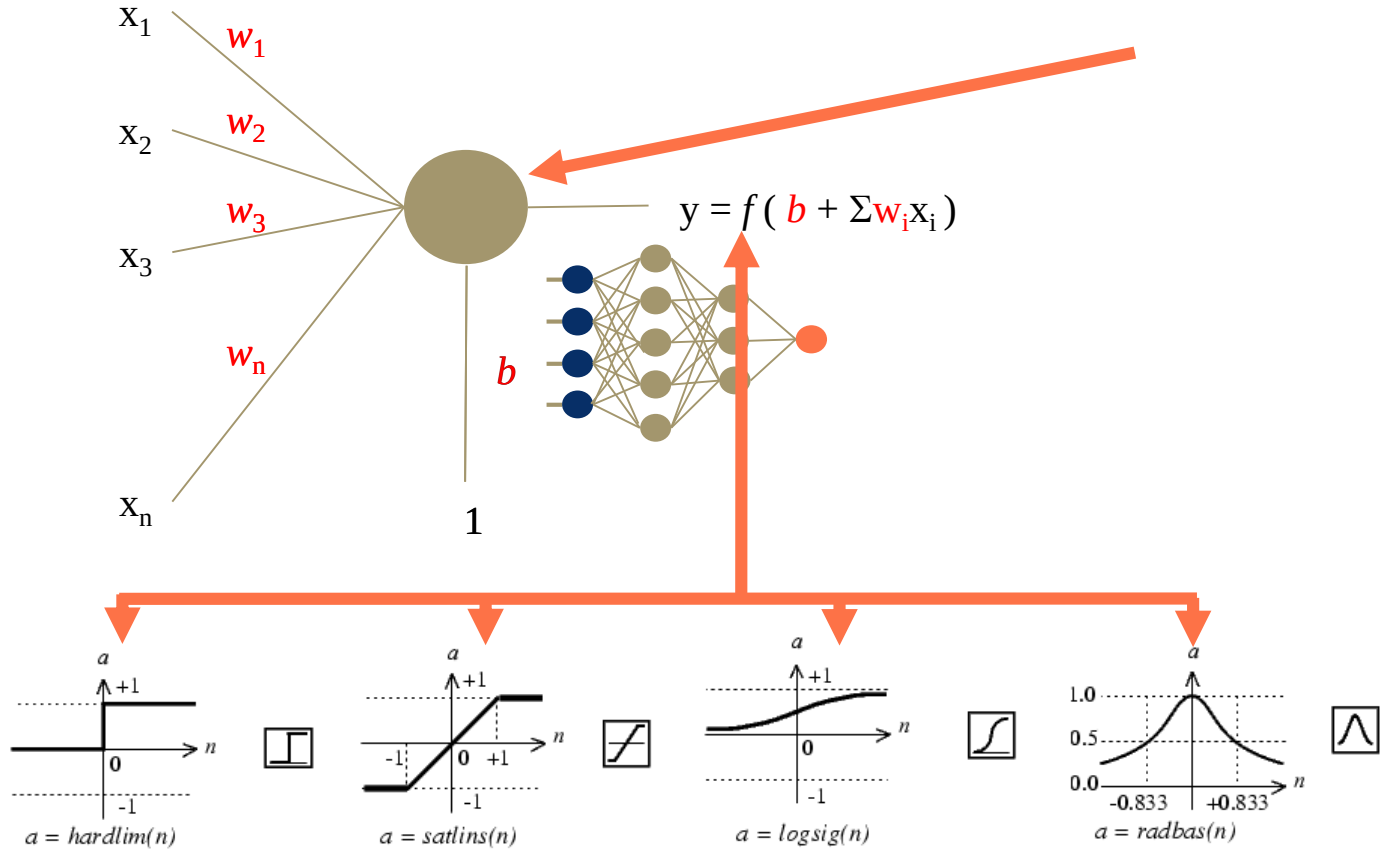
Neural Networks

A Neural Network emulates the structure of a human brain as a network of neurons that are interconnected to each other

Each neuron is (usually) equivalent to a logistic regression unit



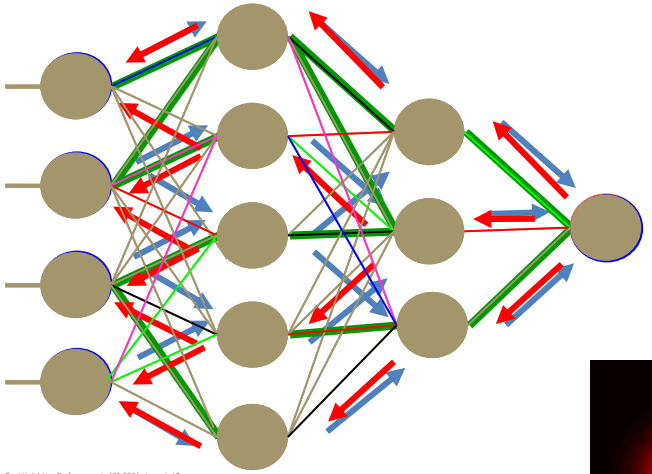
Neuron Model

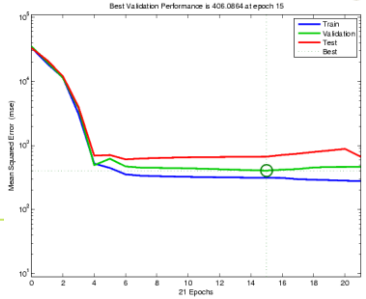


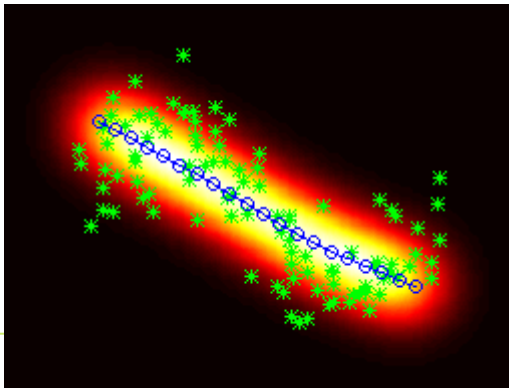
Learning Algorithm - Backpropagation

- Random initialisation of connection weights, w_i, b
- Database of input/output vectors (learning data set)
- Run inputs through network a
- C ch b uts a
- P rough the network and adjust weights
- R ce

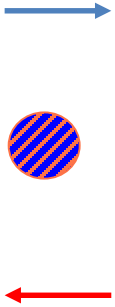
In1	In2	In3	In4	Out
...	...	...	...	...



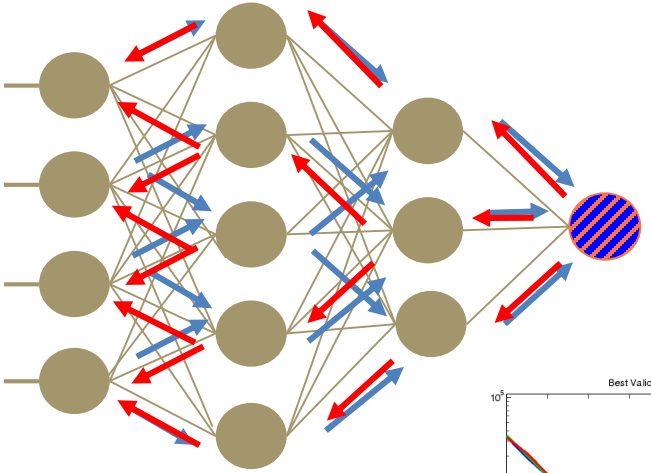


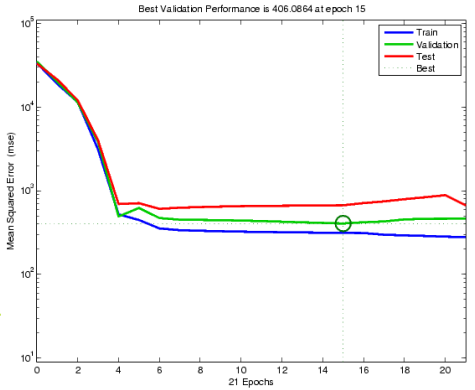


Learning Algorithm - Backpropagation



- Random initialisation of connection weights, w_i, b
- Database of input/output vectors (learning data set)
- Run inputs through network and collect outputs
- Compute error (mismatch between expected outputs and computed outputs)
- Propagate error back through the network and adjust weights
- Repeat until convergence





Neural Networks in Azure ML

- Classification

  2-class

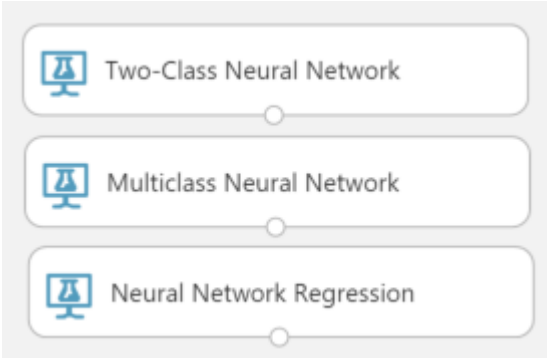
  - https://msdn.microsoft.com/en-us/library/azure/dn905947.aspx

  Multi-class

  - https://msdn.microsoft.com/en-US/library/azure/dn906030.aspx

- Regression

  https://msdn.microsoft.com/en-us/library/azure/dn905924.aspx



- The default neural network is defined as follows:
  - The neural network model has one hidden layer
  - The output layer is fully connected to the hidden layer, and the hidden layer is fully connected to the input layer
  - The number of nodes in the input layer is determined by the number of features in the training data
  - The number of nodes in the hidden layer is determined by the user (with a default value of 100)
  - The number of nodes in the output layer depends on the number of classes (one if it is a regression model)

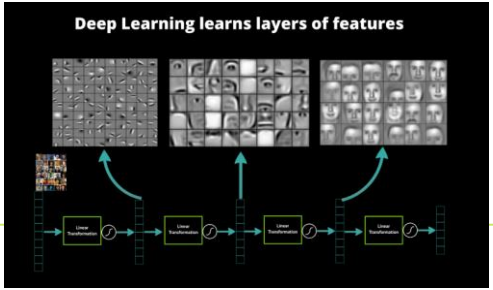
Custom Neural Networks in Azure ML

- Net#

  https://azure.microsoft.com/en-us/documentation/articles/machine-learning-azure-ml-netsharp-reference-guide/

- Net# is a language developed by Microsoft used to define custom neural network architectures in Microsoft Azure ML
  - Create hidden layers and control the number of nodes in each layer
  - Specify how layers are to be connected to each other
  - Define special connectivity structures, such as convolutions and weight sharing bundles
  - Specify different activation functions

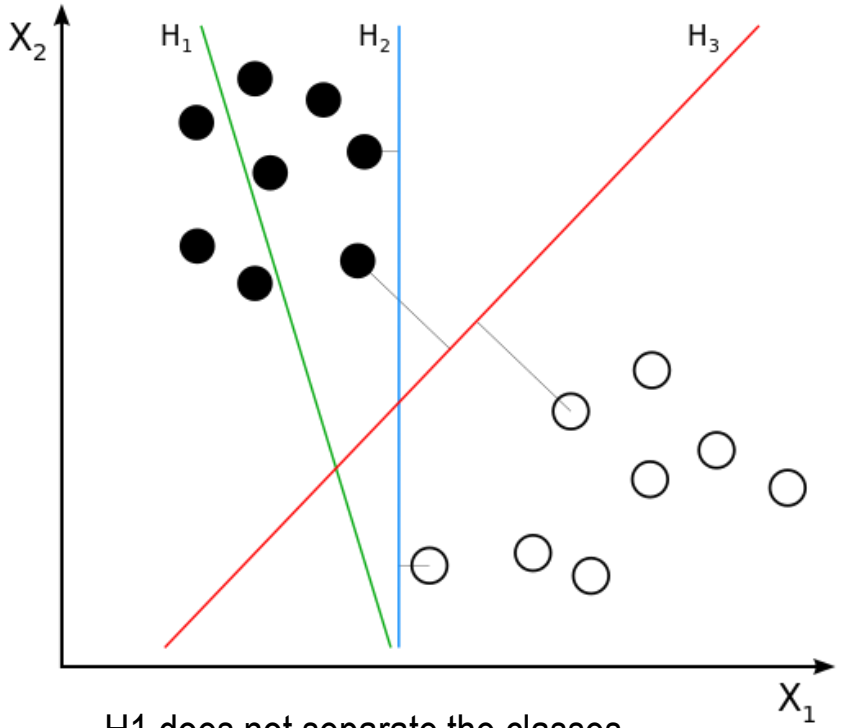
- Can be used to build Deep Neural Networks



Support Vector Machines

A Support Vector Machine provides a binary classification mechanism based on finding a hyperplane between a set of samples with maximum separation

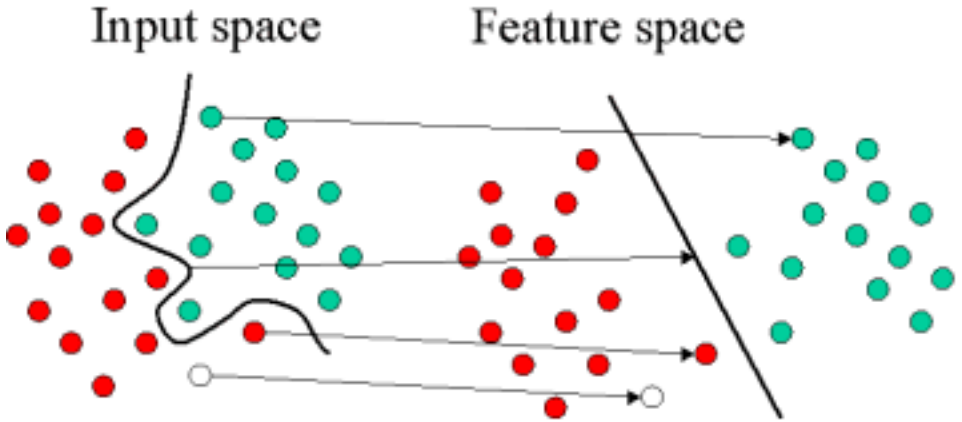
Assumes the data is linearly separable



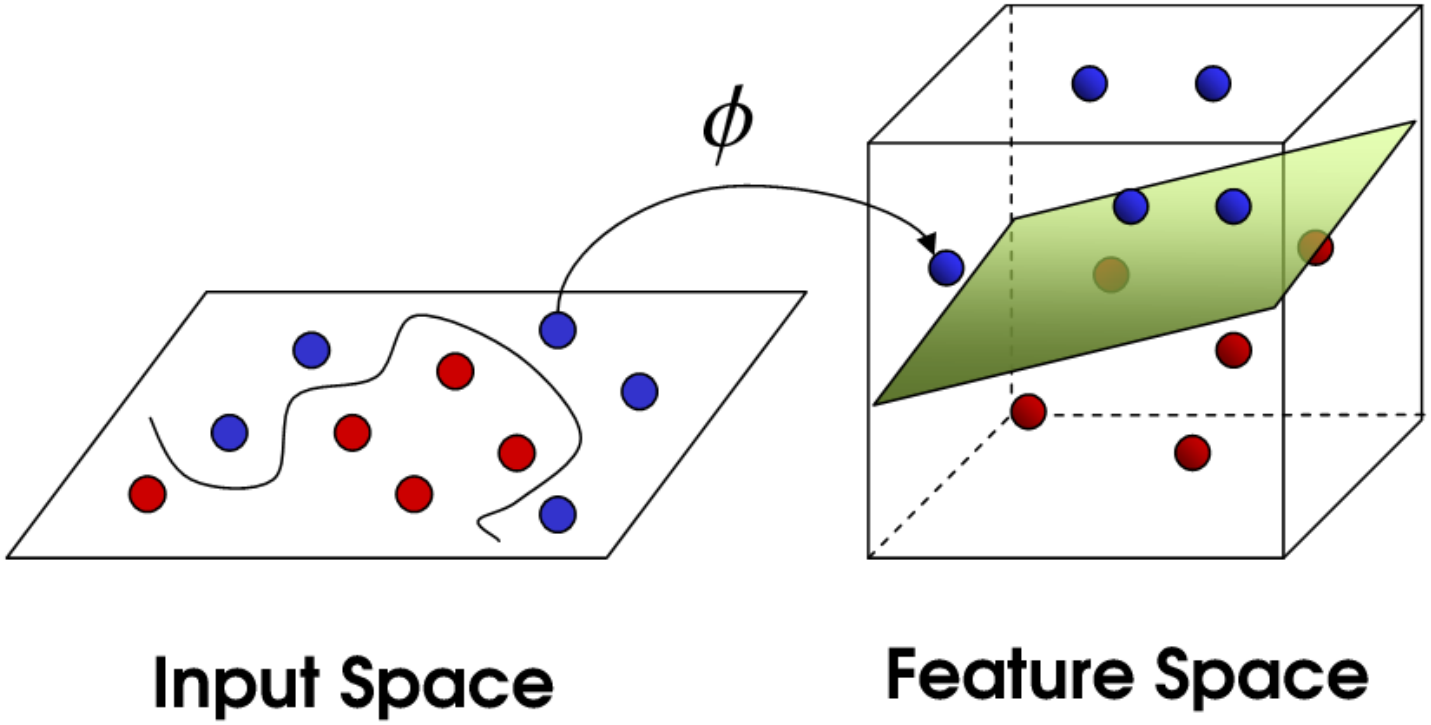
H1 does not separate the classes
H2 does, but only with a small margin
H3 separates them with the maximum margin

Support Vector Machines

If the data distribution is non-linear, the trick is to transform the data to a higher dimension where the data will be linearly separable



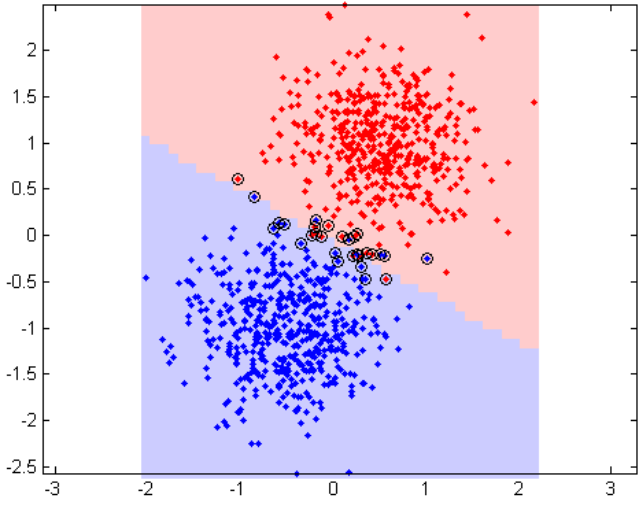
Support Vector Machines

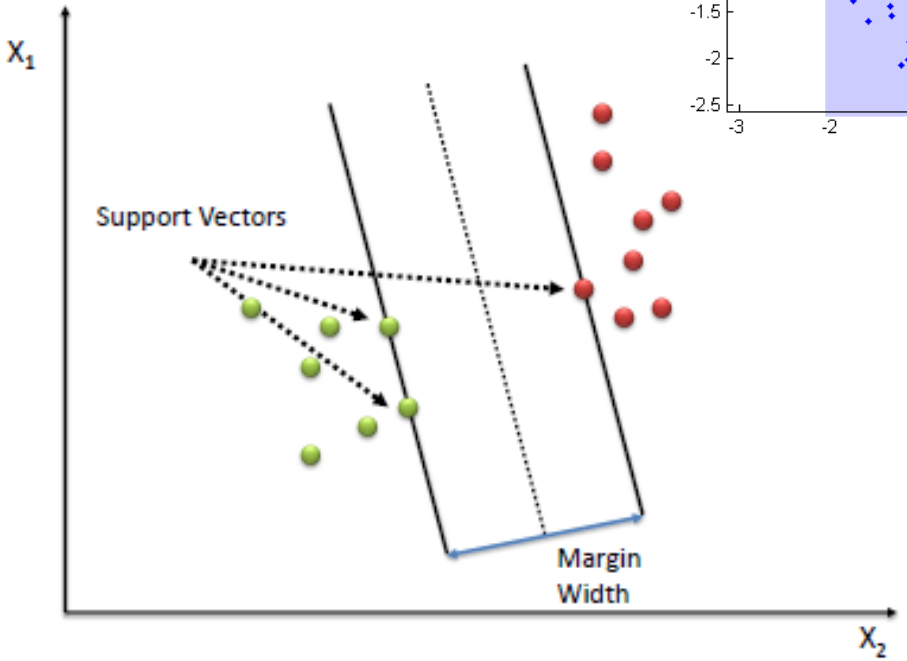


Why "Support Vector" ?

the closest data points are called the **support vectors**

They are the points that define the boundary between the classes





Support Vector Machines in Azure ML

- Classification

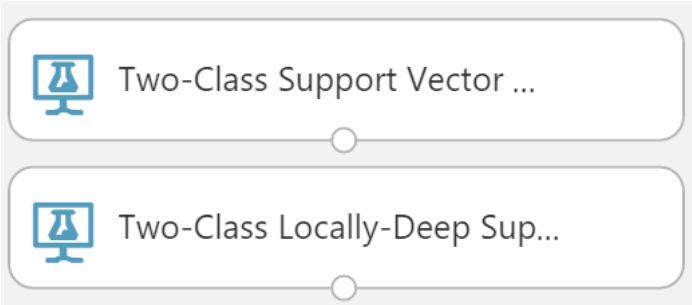
  2-class

  - https://msdn.microsoft.com/en-us/library/azure/dn905835.aspx
  - local deep kernel learning SVM (LD-SVM)
    - https://msdn.microsoft.com/en-US/library/azure/dn913070.aspx

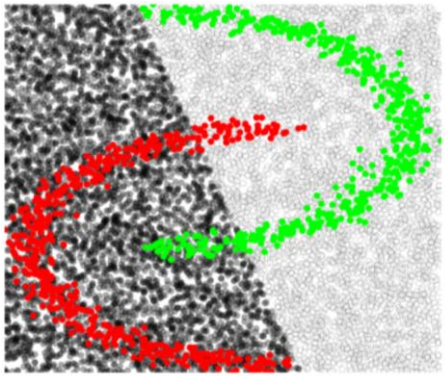
  Multi-class

  - Not available

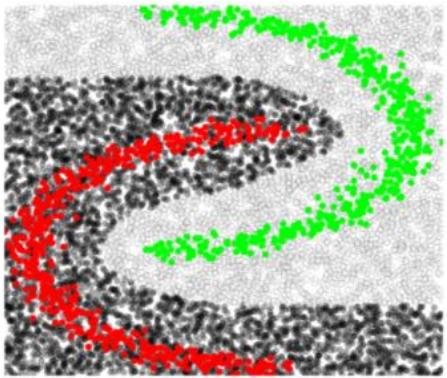
- Regression
  - Not Available



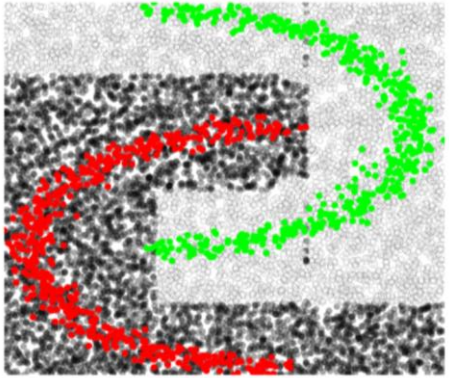
Comparison



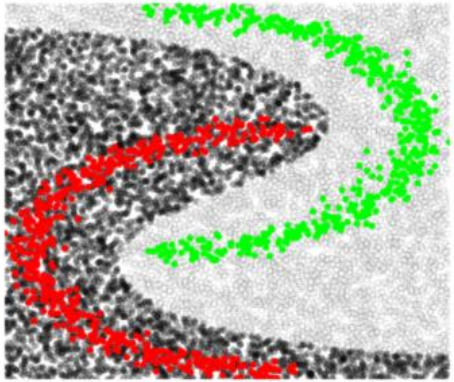
Logistic Regression



K-Nearest Neighbors



Decision Tree



Support Vector Machine

https://github.com/rcompton/ml_cheat_sheet/blob/master/supervised_learning.ipynb