Week 5: Neural Nets & SVM

https://www.rambus.com/blogs/fpgas-take-on-convolutional-neural-networks/

Week 4 Review: Trees

- What is the criterion for splitting classification? Regression?
  –
- What is a 1-layer tree called?
  –
- What is different about random forests compared with a single decision tree?
  –
- What are boosting methods fitting after the first tree?
  –

Week 4 Review: Trees

- What is the criterion for splitting classification? Regression?
  - Classification – Gini or Entropy (info. Gain)
  - Regression – R^2 or another SSE-like metric (MAE, MSE, etc)
- What is a 1-layer tree called?
  - A stump
- What is different about random forests compared with a single decision tree?
  - Random forests use many decision trees, using bootstrapping of samples and randomly sampling the features for each tree (mistake here, what is it?)
- What are boosting methods fitting after the first tree?
  - The residuals ((predictions – actual)^2)

Week 4 review quiz

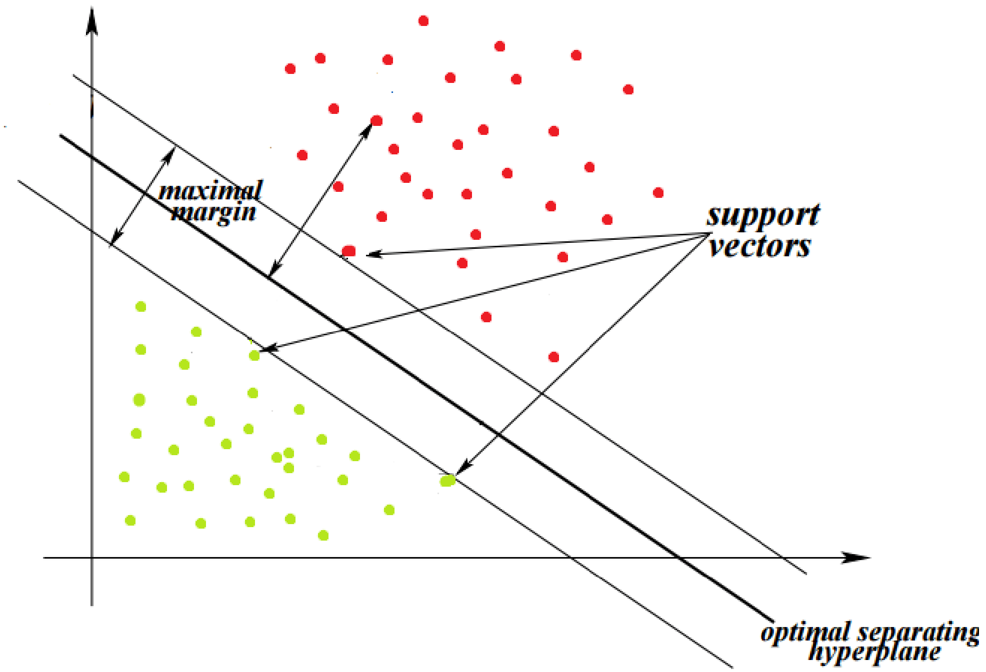
- Using the auto.dt.clean.csv dataset (under week 5), predict mpg from everything else with a random forest
- Create train/test sets
- Try 3 values for mtry, including the default
- Evaluate performance on the train and test set, and check for overfitting
- Plot the feature importances and explain them

SVM

- Support vector machine
- Invented in 1963 by Vladimir N. Vapnik and Alexey Ya. Chervonenkis
  - Vlad and 2 others intro’d the ‘kernel trick’ in 1992
- Works well for fewer data points (<20k) and large number of features

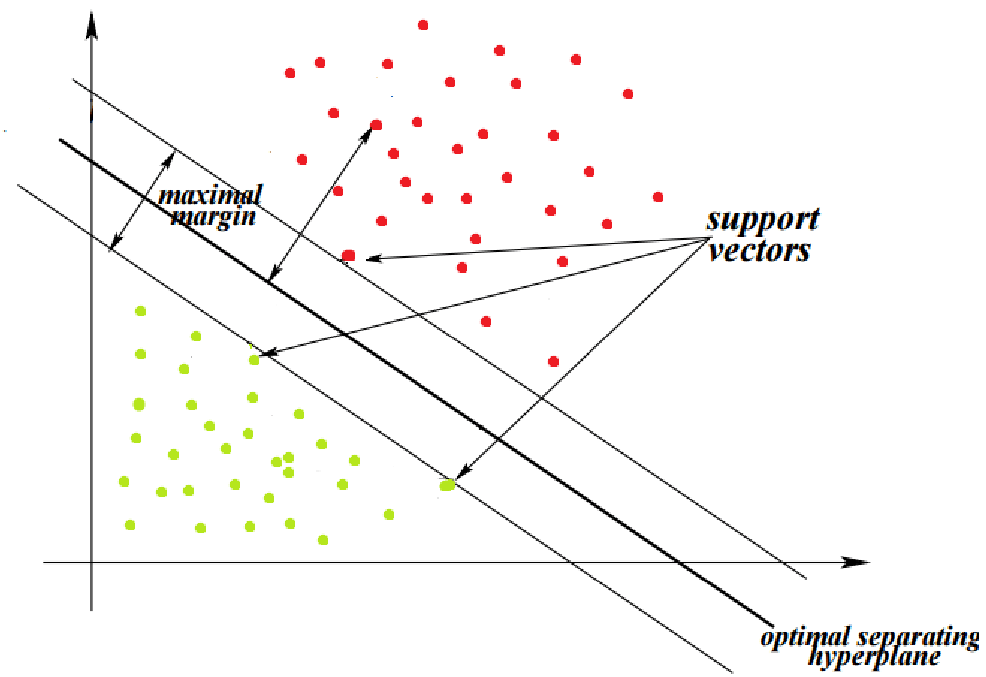
How SVMs work

- Mathematically it finds the optimal separating hyperplane between classes
- A hyperplane is a dot in 1D, a line in 2D, a plane in 3D, and a hyperplane in 4D+



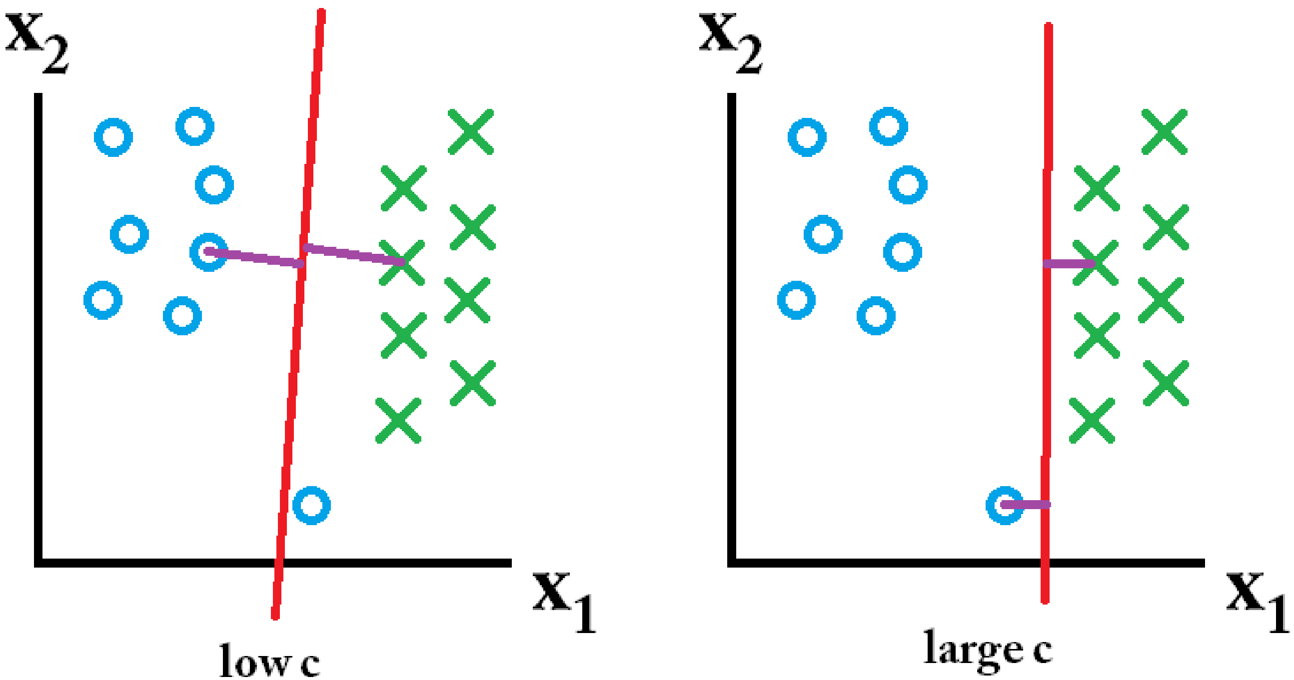
How SVMs work

- Finds the optimal by finding hyperplane with max distance between the 2 nearest points (called support vectors)



Overlapping points

- If classes are overlapping, we can penalize the misclassification. The penalty is proportional to C, so a large C will avoid misclassification

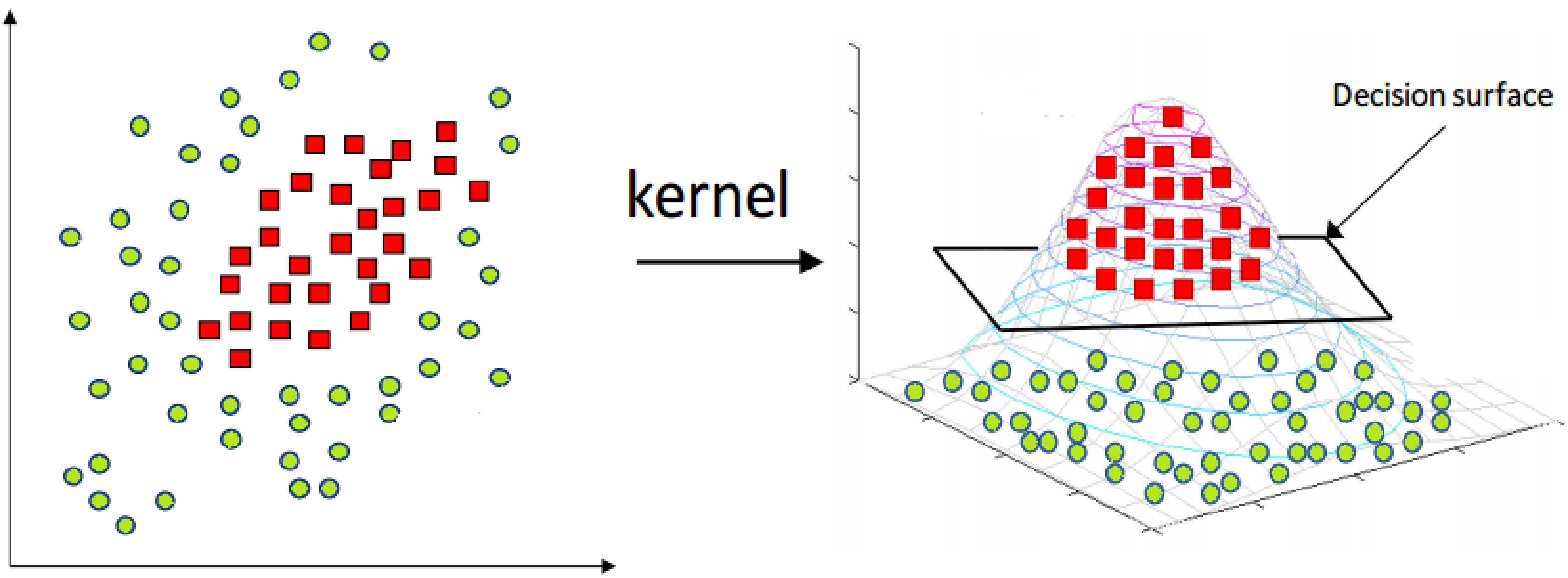


C is a tunable hyperparameter

https://stats.stackexchange.com/a/159051/120921

Kernel Trick

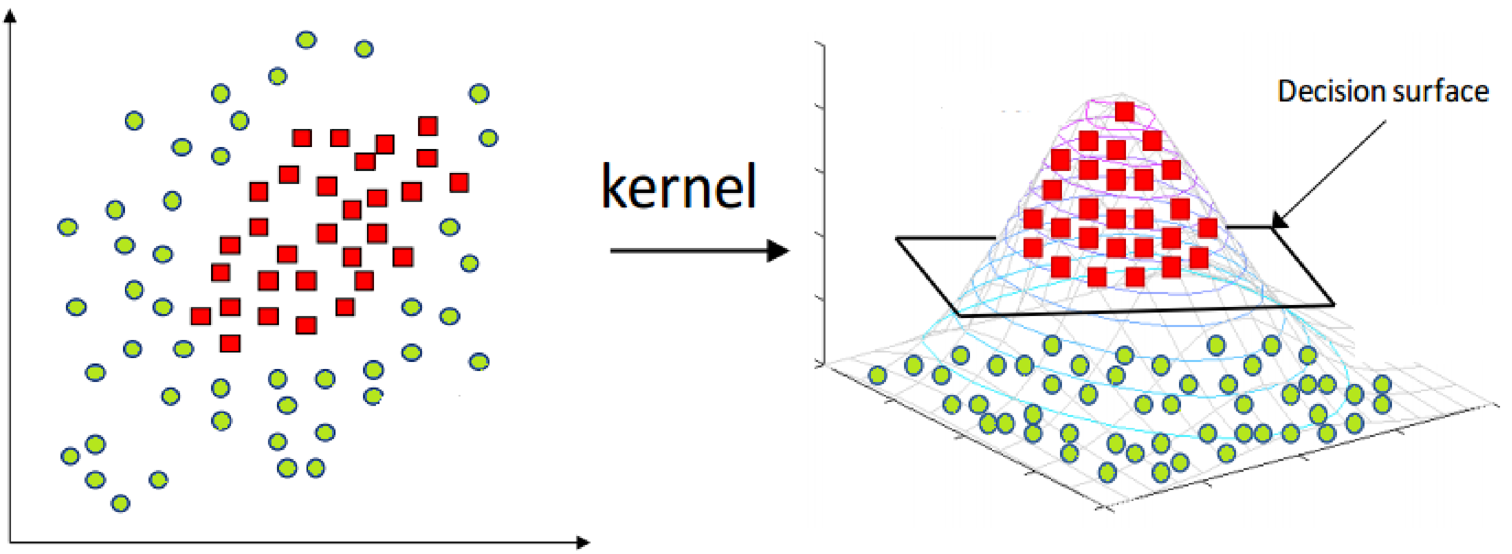
- We can transform the data with a kernel so it is separable



http://blog.hackerearth.com/simple-tutorial-svm-parameter-tuning-python-r

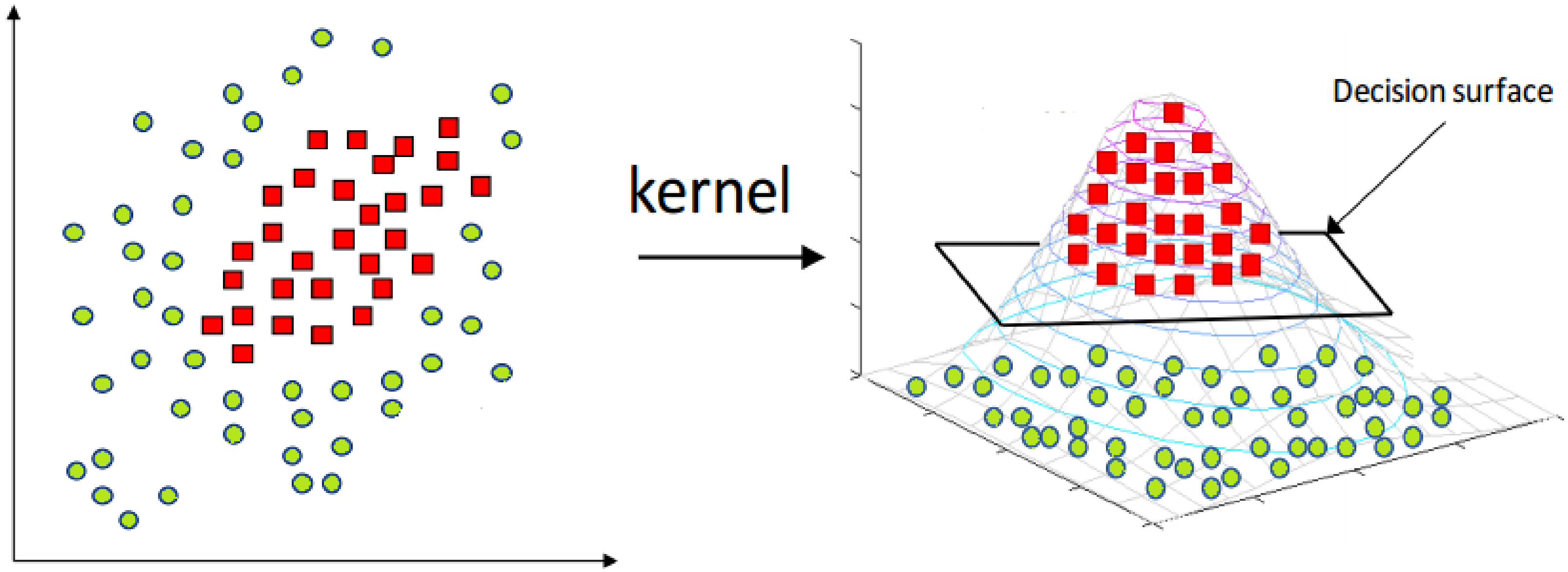
Kernel Trick

- Most common kernels are polynomial (degree of polynomial is a hyperparameter) and radial basis function (RBF, Gaussian)
- These essentially map data from one space to another (it's a bit more complicated than that, but gets into complex math)



http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf

Kernel Trick



$$K(x, x') = \exp\left(\frac{(|x - x'|)^2}{2\sigma}\right)$$

Sigma is a tunable hyperparameter

https://www.youtube.com/watch?v=H_I0pYdzBSk

http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf

SVM Runtime

- The runtime depends on the hyperparameter C
- O(max(n,d) min (n,d)^2), n = number of samples, d = number of features
- In general, n^2*d for RBF, and n*d for linear kernels
- Low C (small misclassification penalty) usually decreases runtime

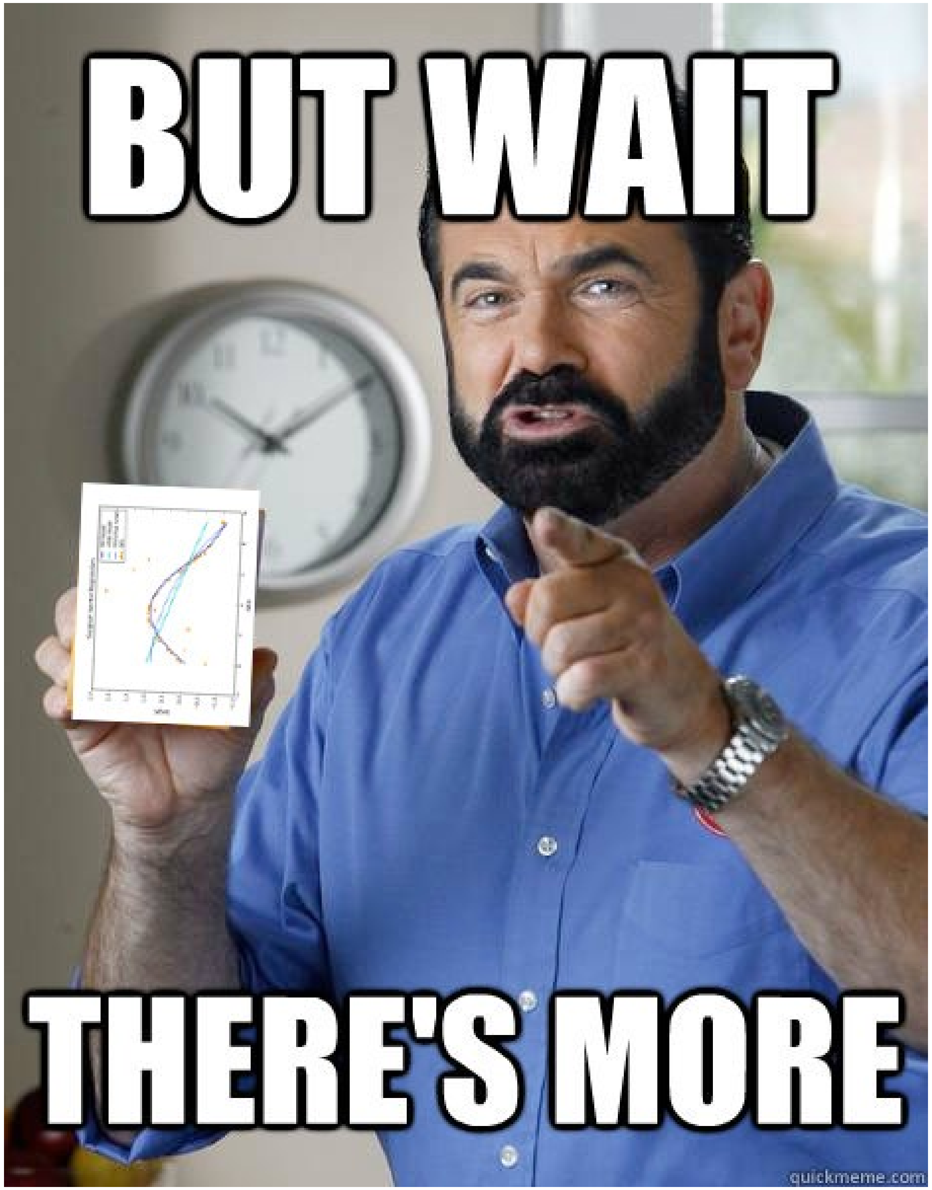
Chapelle, Olivier. "Training a support vector machine in the primal." Neural Computation 19.5 (2007): 1155-1178.

More SVM Resources

- The SVM algorithm has a *lot* more math behind it, here are some follow-up resources

  - http://blog.hackerearth.com/simple-tutorial-svm-parameter-tuning-python-r
  - http://www.robots.ox.ac.uk/~cvrg/bennett00duality.pdf
  - https://www.svm-tutorial.com/
  - https://jakevdp.github.io/PythonDataScienceHandbook/05.07-support-vector-machines.html
  - http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html
  - https://github.com/eriklindernoren/ML-From-Scratch/blob/master/mlfromscratch/supervised_learning/support_vector_machine.py

- Support vector machines (SVMs) can also be used for regression (SVR)
- Similar (complicated) math



https://www.mathworks.com/help/stats/understanding-support-vector-machine-regression.html

http://www.svms.org/regression/SmSc98.pdf

SVM demo/exercise in R

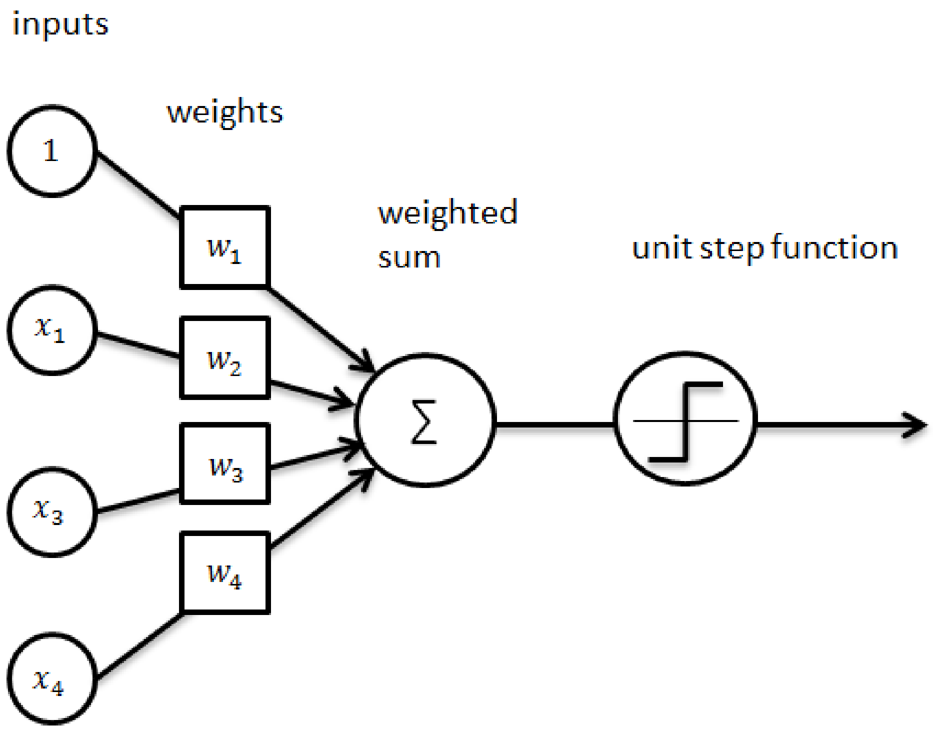
- On heart disease, using caret, kernlab, and e1071 packages

Neural networks

- Can capture most any statistical pattern due to highly nonlinear behavior
- Infinite possibilities for configurations (architecture/topology)
- Requires GPUs (now Google’s TPU as well) to train big nets
- Outperforms all other algorithms when set up correctly
- Difficult to understand what the net is learning/doing

Perceptron

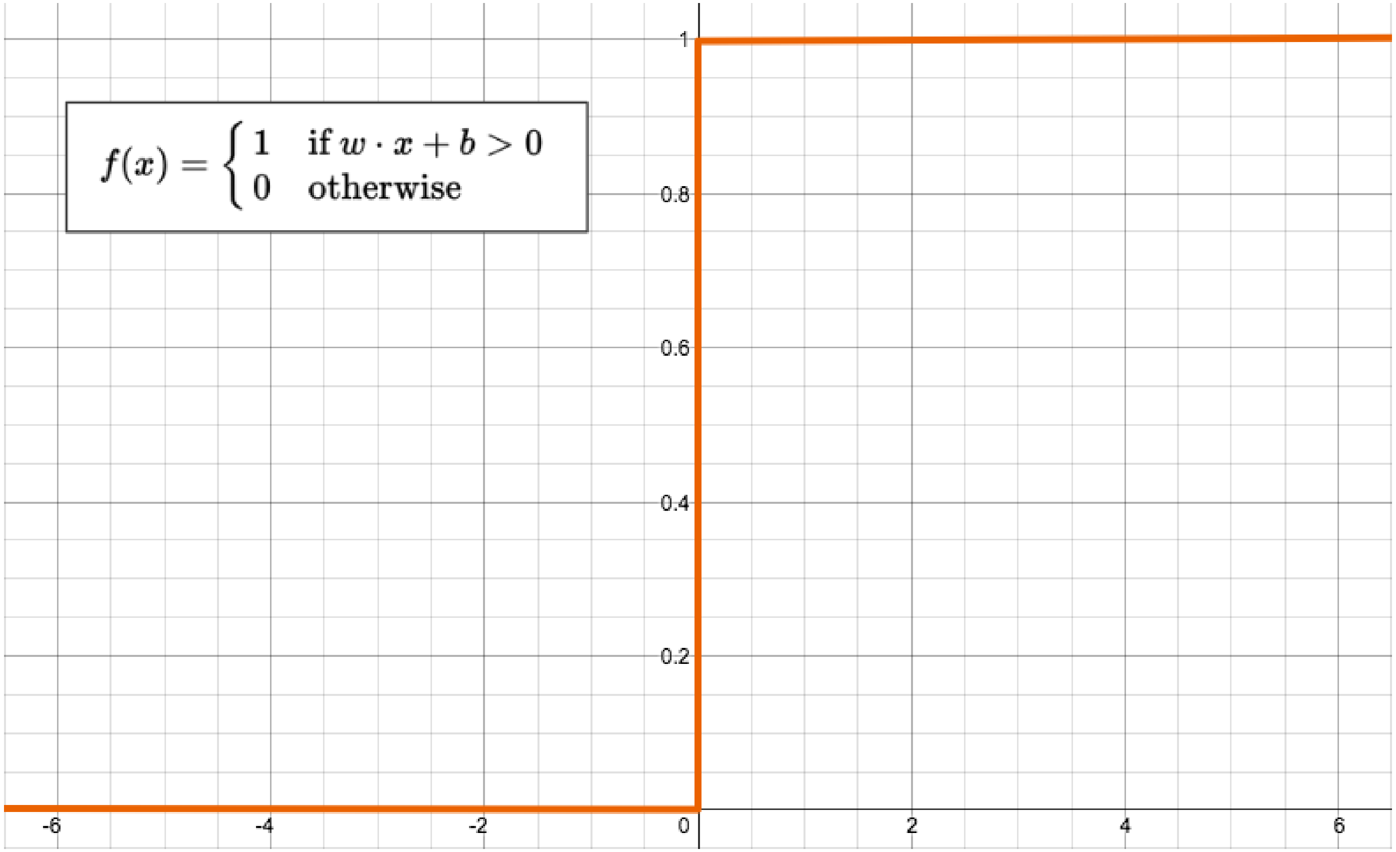
- If the weighted sum is greater than 0, output 1, otherwise 0



http://ataspinar.com/2016/12/22/the-perceptron/

Perceptron

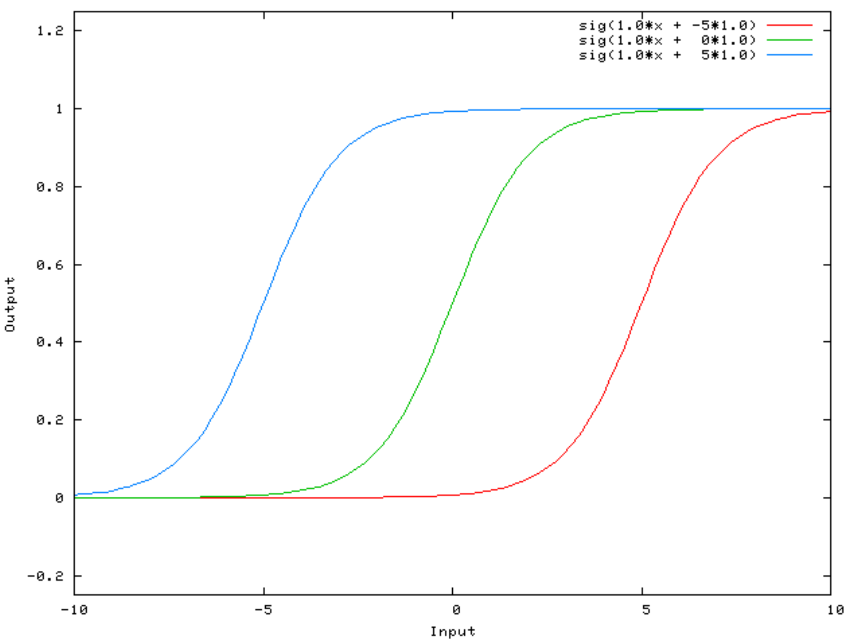
- If the weighted sum is greater than 0, output 1, otherwise 0

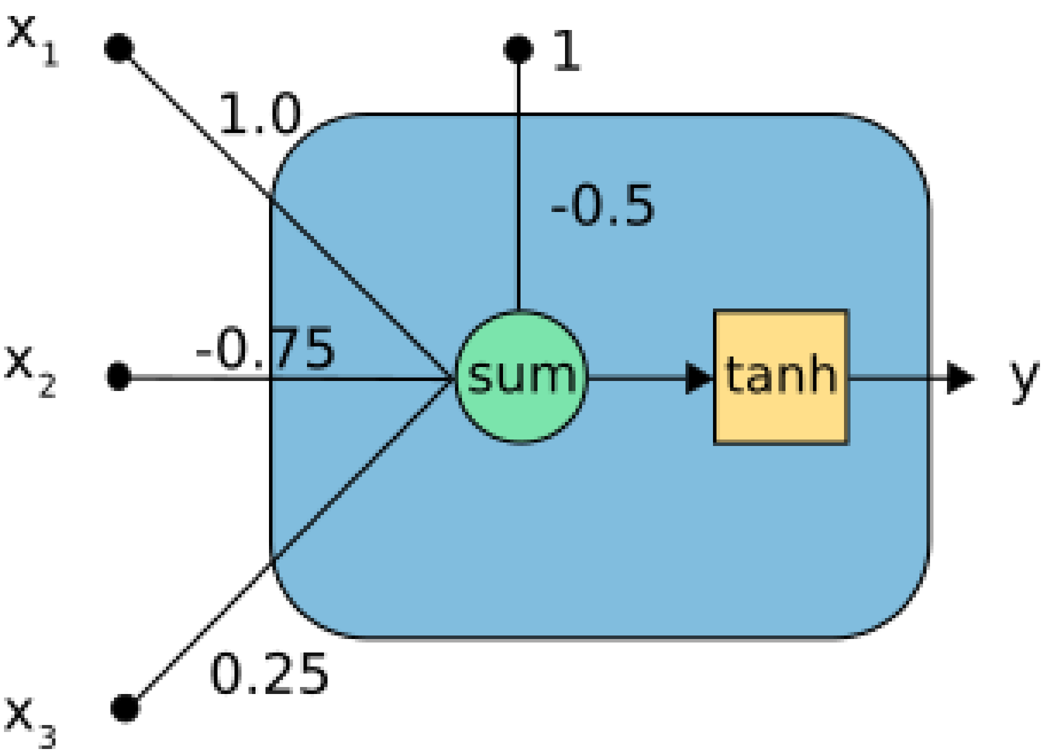


https://appliedgo.net/perceptron/

Activation functions and bias

- Can add a bias to shift the output left or right on the x-axis



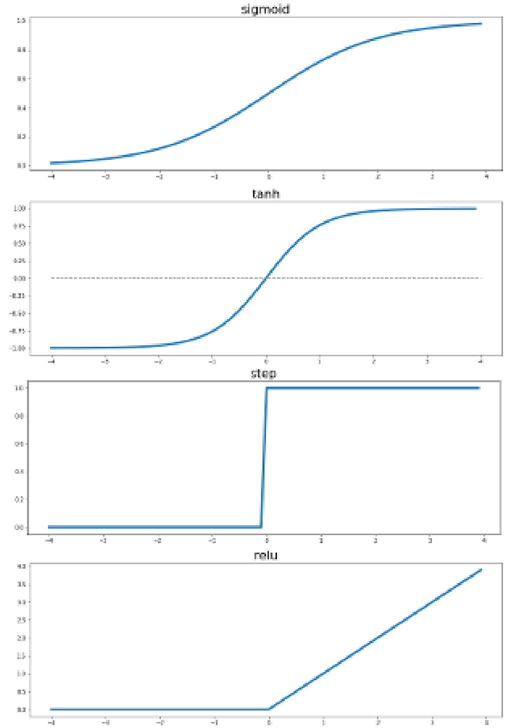


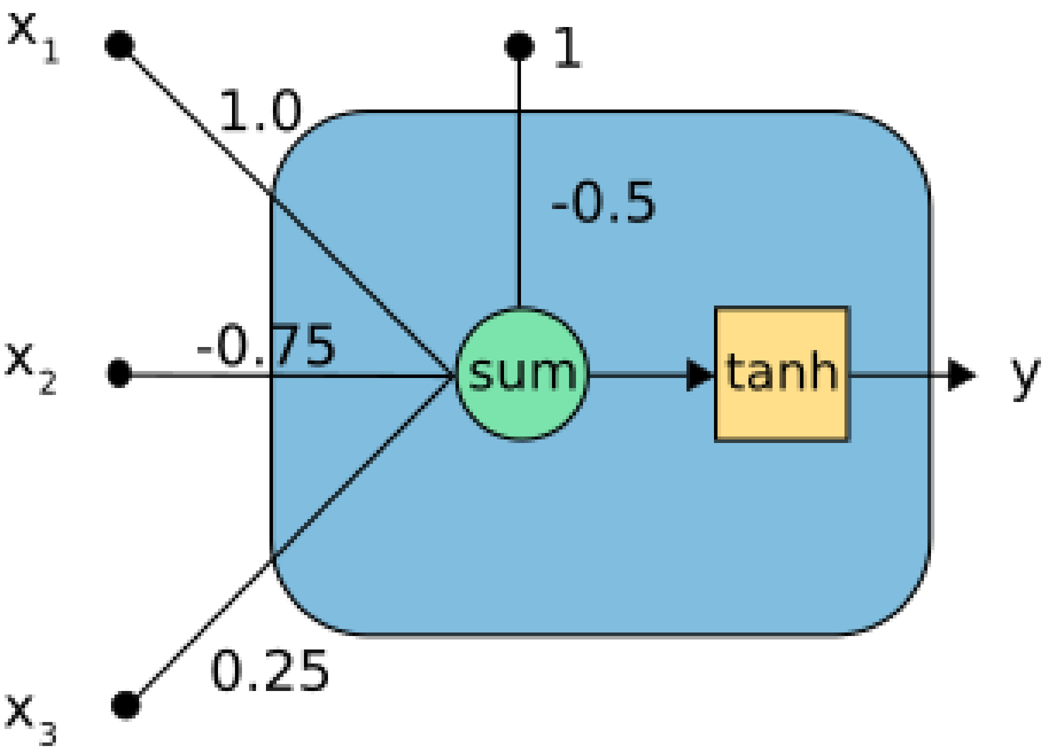
https://stackoverflow.com/a/2499936/4549682

https://www.neuraldesigner.com/blog/perceptron-the-main-component-of-neural-networks

Activation functions and bias

- Can add a bias to shift the output left or right on the x-axis





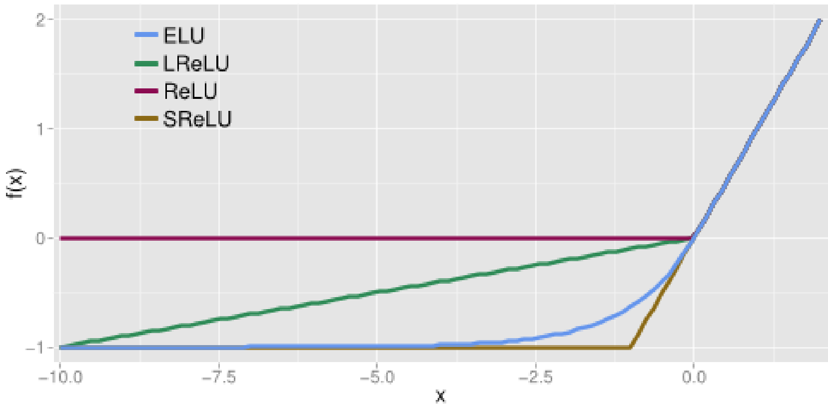
http://www.snee.com/bobdc.blog/2017/09/understanding-activation-funct.html

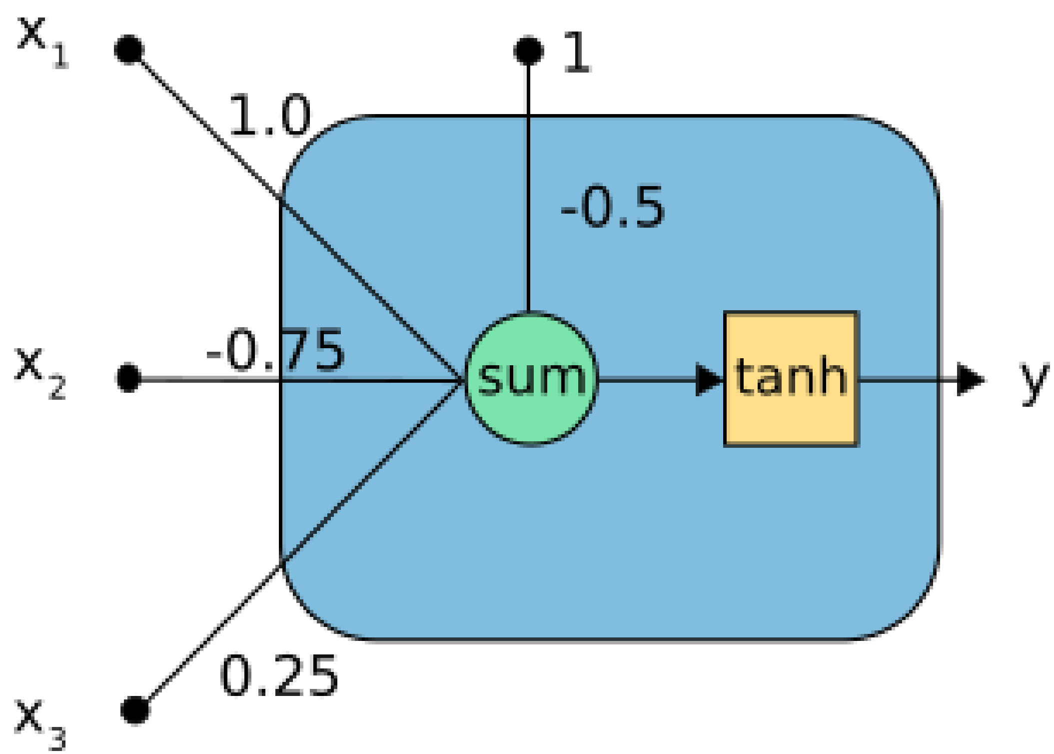
https://en.wikipedia.org/wiki/Activation_function#Comparison_of_activation_functions

https://www.neuraldesigner.com/blog/perceptron-the-main-component-of-neural-networks

Activation functions and bias

- ReLU (rectified linear unit) almost always used, although ELU sometimes works better





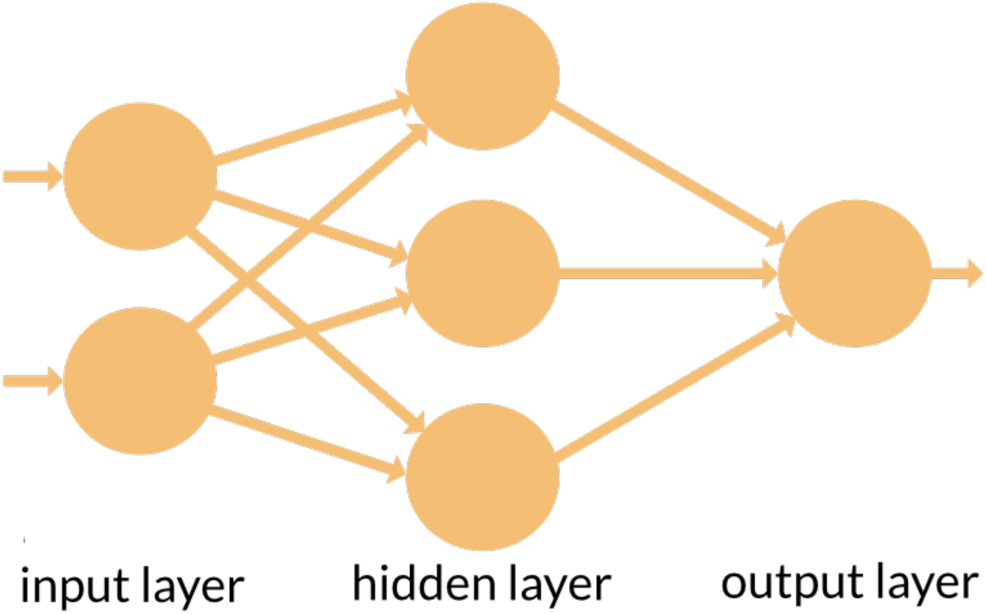
https://arxiv.org/abs/1511.07289

http://laid.delanover.com/activation-functions-in-deep-learning-sigmoid-relu-lrelu-prelu-rrelu-elu-softmax/

https://www.neuraldesigner.com/blog/perceptron-the-main-component-of-neural-networks

Matrix math – dense layers

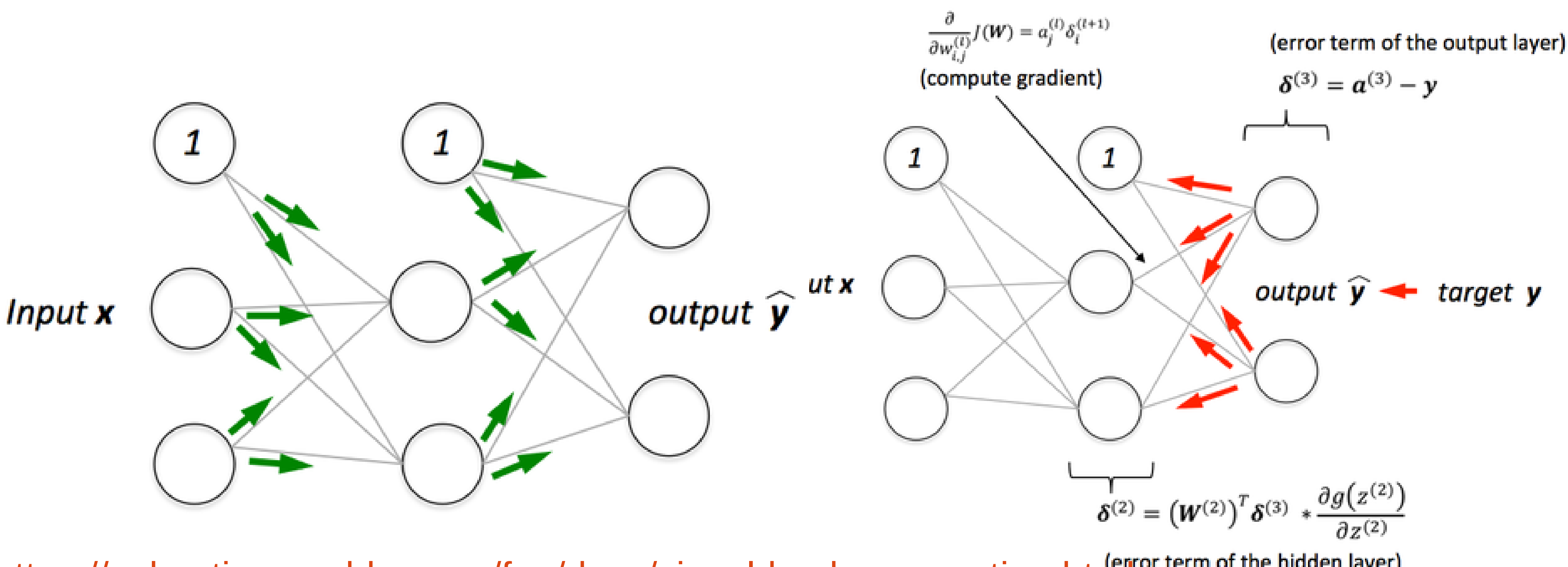
- Now we start stacking neurons on top of each other
- When we have a lot of hidden layers, the network becomes ‘deep’ and we have “deep learning”



https://appliedgo.net/perceptron/

Loss functions and Backpropagation

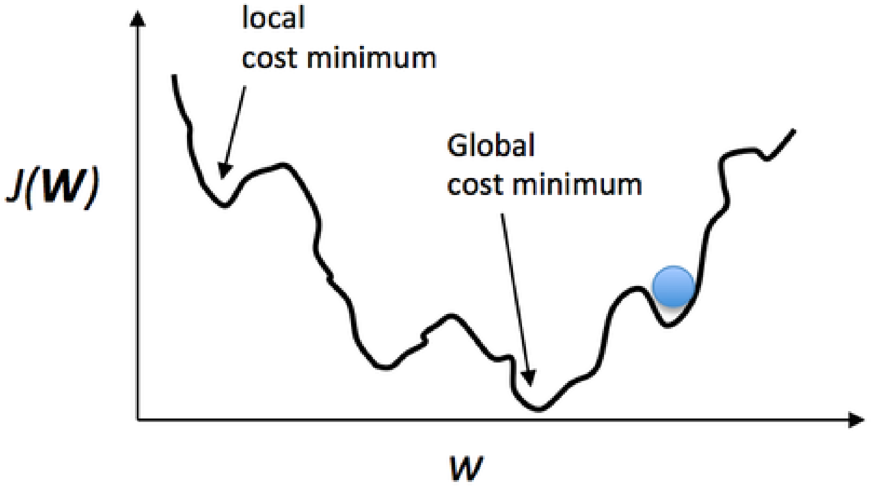
- We need a way to find the best weights in the network
- Make a prediction, calculate error, and move the weights in the direction that minimizes error



https://sebastianraschka.com/faq/docs/visual-backpropagation.html

Loss functions and Backpropagation

- Keep doing this until our weights stop moving much
- Called gradient descent, because we are descending down the slope of the loss function



Link with the math:
http://alexminnaar.com/deep-learning-basics-neural-networks-backpropagation-and-stochastic-gradient-descent.html

https://sebastianraschka.com/faq/docs/visual-backpropagation.html

Loss functions

- For regression, we can use MSE, MAE, or any other custom equation
- For classification, often categorical cross entropy (multiple classes) or binary cross entropy (2 classes)

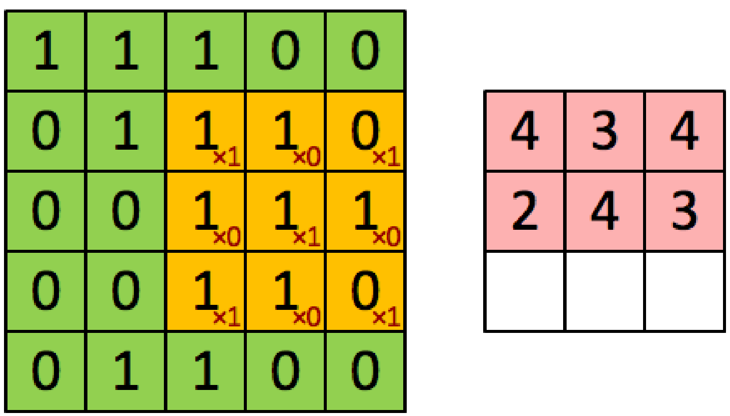
https://keras.io/losses/

http://ml-cheatsheet.readthedocs.io/en/latest/loss_functions.html

https://en.wikipedia.org/wiki/Cross_entropy

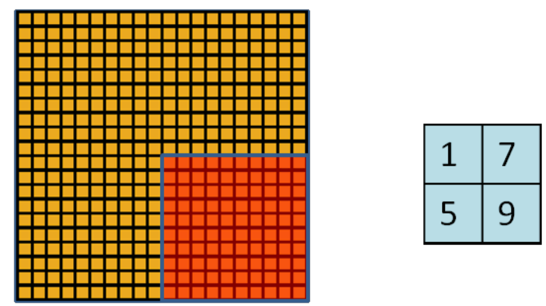
Convolutional and pooling layers

- Used mostly for image recognition, but pattern recognition in general (i.e. for 1D time series)



Image

Convolved Feature



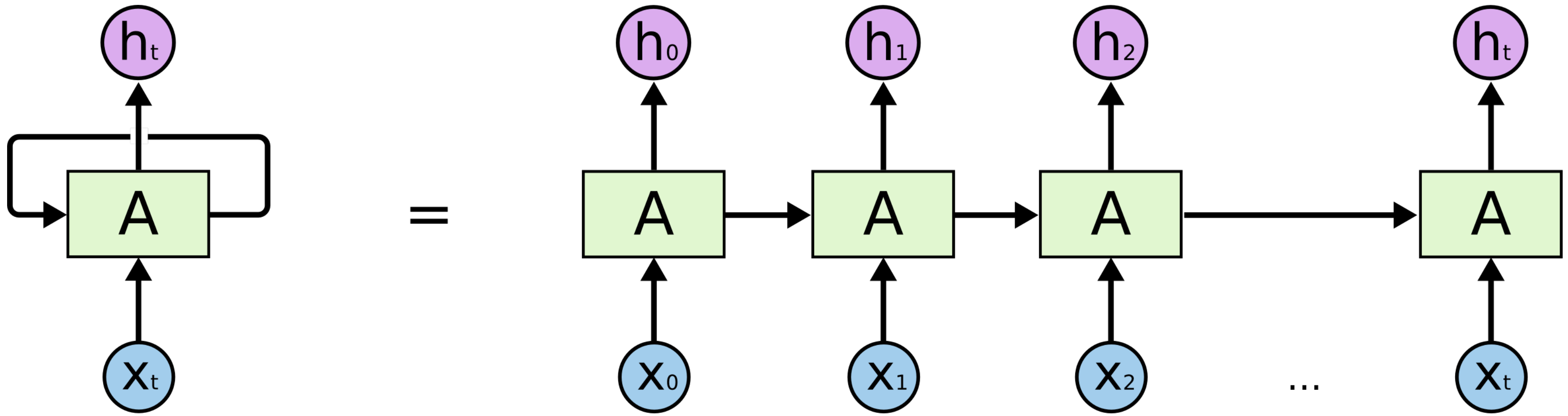
Convolved feature

Pooled feature

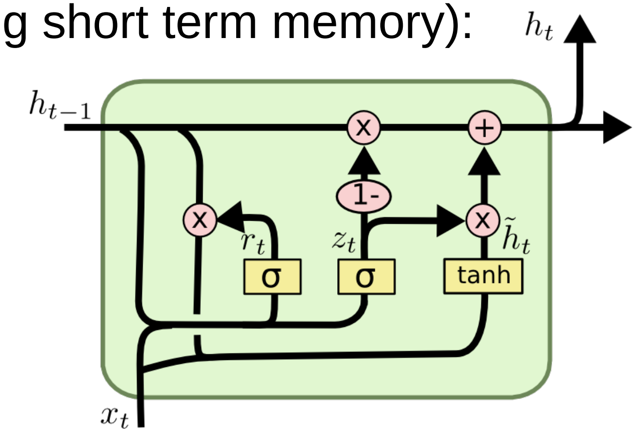
http://ufldl.stanford.edu/tutorial/supervised/FeatureExtractionUsingConvolution/

http://ufldl.stanford.edu/tutorial/supervised/Pooling/

Recurrent networks



LSTM cell (long short term memory):



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

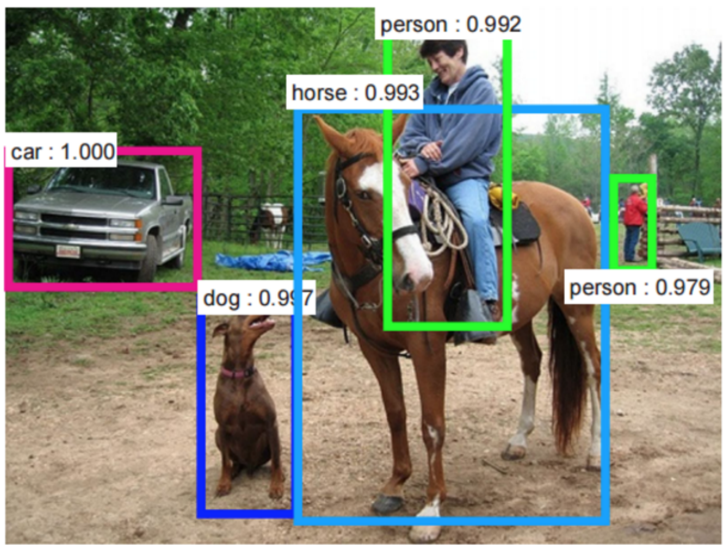
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

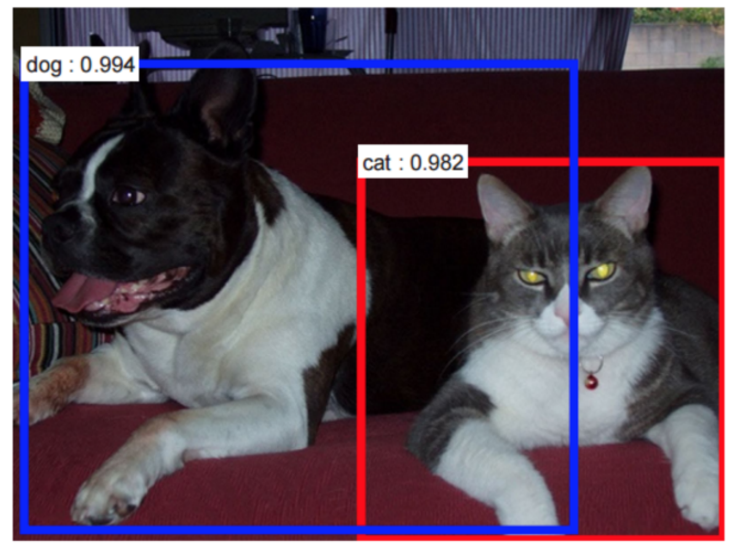
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

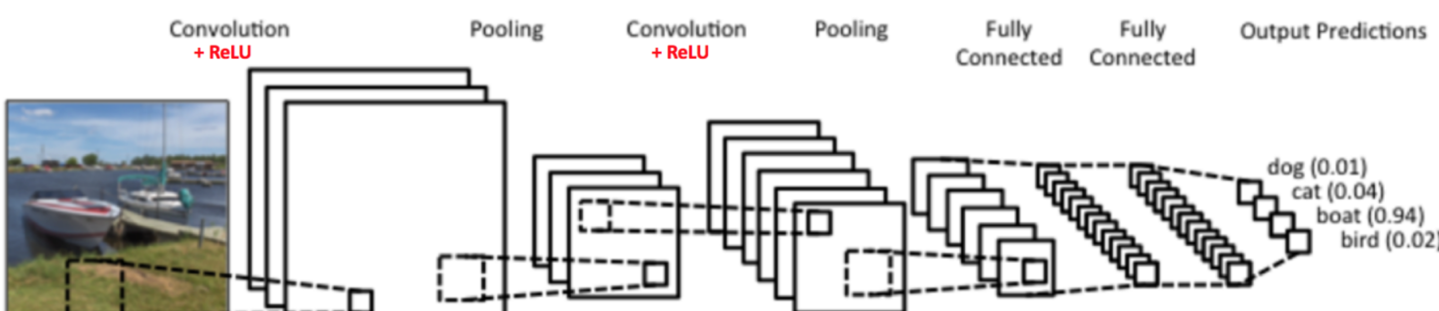
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Image recognition

- These networks are very large and expensive to train







https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Other applications

- Time series analysis/prediction
  - Finance (predicting future trends)
  - Audio (voice recognition/speech generation, generating /recommending music, recognizing songs like Shazam)
  - Electroencephelography (EEG) (brainwaves – predict if you want to move your arm, words you are imagining, etc)
  - Business/commerce data (predicting customer trends or outcomes)
- Generative adversarial networks (GANs)
- Speech/writing generation (RNN/LSTMs)
- Deep Q-learning (reinforcement learning with neural nets)

Interesting applications

- Making art (deepart, deepdream)
- Google’s speech generation improved by neural nets (wavenet)
- Deep reinforcement learning beats Go world champ
- Deep reinforcement learning playing video games, also going for general AI
- Predicting what you are seeing from your brainwaves
- Face recognition
- Creating neural nets...with neural nets

Other neural network topics we didn't cover

- Embedding layers (and word2vec, etc)
- Backpropagation for RNNs
- Residual nets (e.g., ResNet50)
- Transfer learning
- Regularization
- Early stopping
- Optimizers
- Other neural net libraries (lasagna, torch, theano, CNTK...)

Example/project: bike share neural net

- Data from here: https://archive.ics.uci.edu/ml/datasets/bike+sharing+dataset
- Train a regression neural net to predict the amount of bike rentals each hour
- Try at least 5 different architectures, pick the best one and defend your choice
- Post to the week 5 bike share exercise discussion