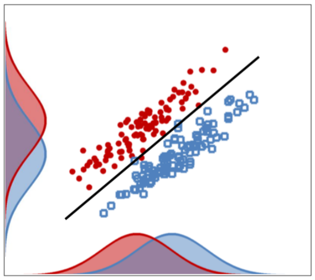
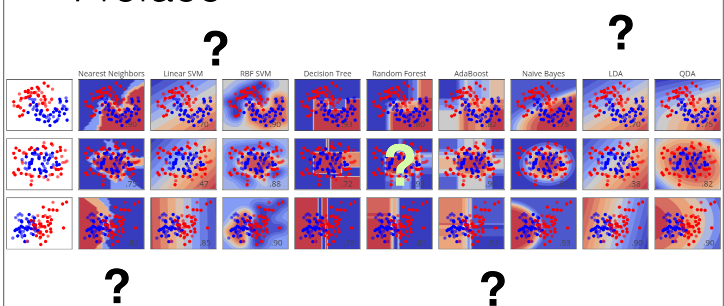


Algorithm Selection

Two-day workshop
Duke University
Adam Chekroud

Preface

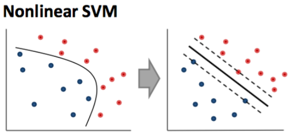


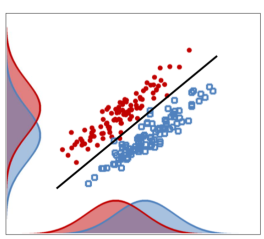
- “Algorithm Selection: Try all of them*”

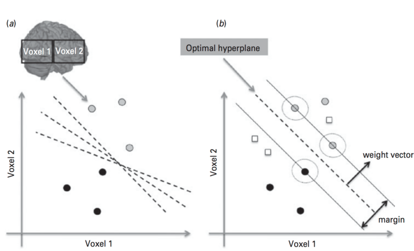
* only caveat: with appropriate train-test precautions

Objectives

Nonlinear SVM



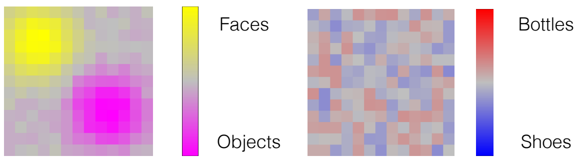




(Supervised) Machine Learning

Working definition: using computer algorithms to identify patterns in data that help us **predict** things we care about

- Most accurate mapping from observables to an outcome



Neuroscience has readily adopted the notion of "distributed processing"

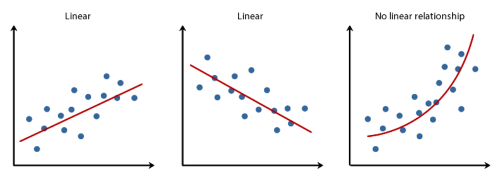
—> Multivariate approach is more sensitive in capturing complex signal

Supervised Learning

Very different to traditional inference!

- Often, we must leave interpretability at the door…

While regression was all about estimating beta…

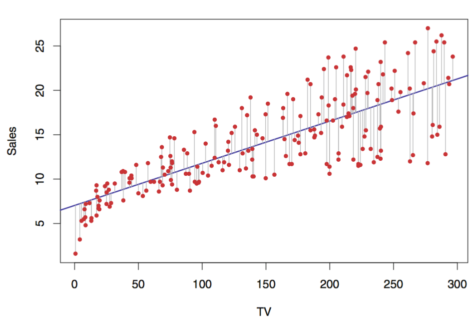


….ML is all about y-hat

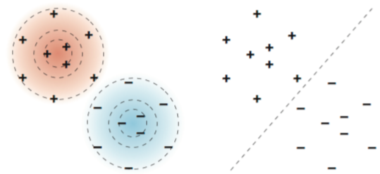
Supervised Learning

Algorithms are, for the most part, data agnostic:

- Medical diagnosis, computer vision, spam filtering, neuroscience, genetics, tailoring ads, selling you stuff, filling ur amazon wishlist.



How do we choose an algorithm?



ML algorithms come with different “flavours”

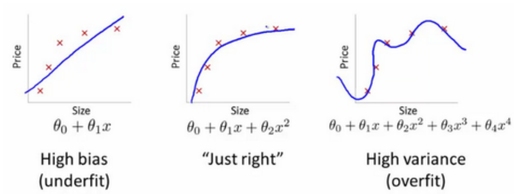
Three broad aspects to consider (at least)

- Accuracy
- Feasibility
- Linearity (or, the nature of the problem)

Choosing an algorithm

Accuracy

- More accuracy is not always better (!!)
- Is an approximation sufficient?
- Is an approximation better?!
  - Simpler models less likely to over fit



c.f. Bias-variance tradeoff

Choosing an algorithm

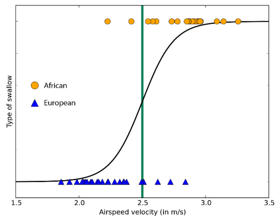
Feasibility

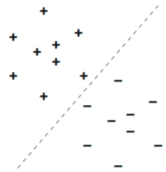
- What can your data support? (realistic!)
  - 10 subjects? 10k? 100k?
  - Powerful algorithms are usually data hungry
- What can you afford, computationally?
  - Not just one model
    - Tuning parameters, cross validation…
  - Implementation?
    - Laptop is easy but slow
    - HPC/AWS, fast, but hard

Choosing an algorithm

Linearity (or approach)

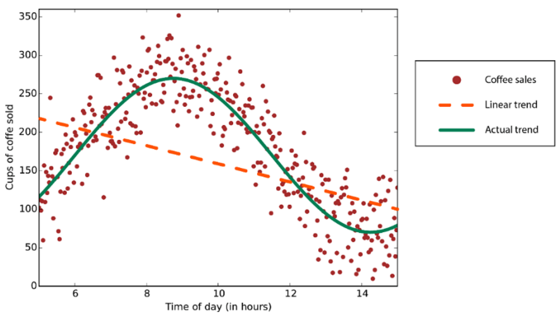
- Is the problem likely to have linear solution?
- Was your problem just *finding* the key variable?





Choosing an algorithm

- Did you just need more flexibility?



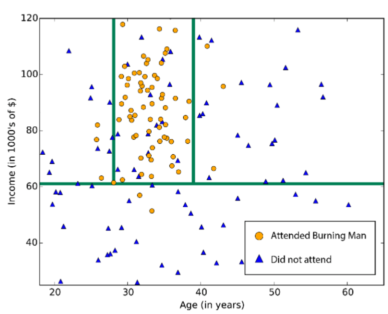
Data with a nonlinear trend *- using a linear regression method would generate much larger errors than necessary*

Despite their dangers, linear algorithms are very popular as a first line of attack. They tend to be algorithmically simple and fast to train.

Choosing an algorithm

Linearity (or approach)

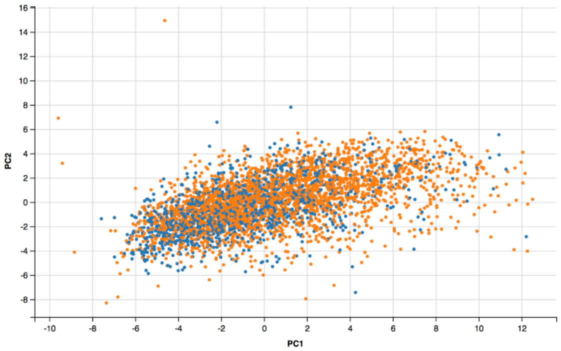
- How about a non-parametric approach?



Choosing an algorithm

Linearity (or approach)

- If only it was that simple…

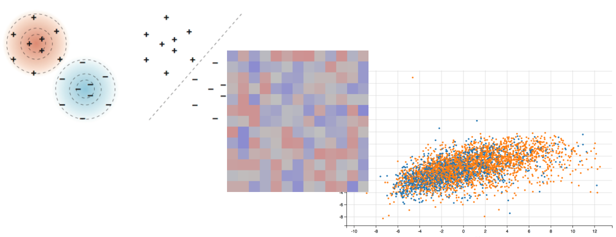


Choosing an algorithm

Inductive Bias: Algorithm selection *necessarily* introduces a bias

- We can only approximate a correct output
- Not necessarily a bad thing
    - That is how we learn!

      …but we want to be as less wrong as possible.



Inductive Bias (reading)

Algorithm	Inductive Bias
Candidate-Elimination	The target concept c is contained in the hypothesis space H.
Linear Regression	The relationship between the attributes x and the output y is linear. The goal is to minimize the sum of squared errors.
Decision Trees	Shorter trees are preferred over longer trees. Trees that place high information gain attributes close to the root are preferred over those that do not.
Neural Networks with Backpropagation	Smooth interpolation between data points.
K-Nearest Neighbors	The classification of an instance x will be most similar to the classification of other instances that are nearby in Euclidean distance.
Support Vector Machines	Distinct classes tend to be separated by wide margins.
Naive Bayes	Each input depends only on the output class or label; the inputs are independent from each other.

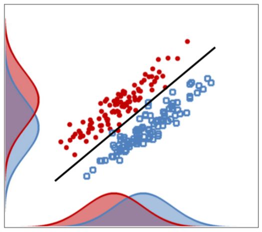
Interim summary

Working definition: using computer algorithms to identify patterns in data that help us **predict** things we care about

- Pragmatic approach, different to classic inference

Try them all, but think about:

- Accuracy
- Feasibility
- Inductive bias



What are the options?

ML framework called *caret*

- Collection of functions that streamline ML processes

Contains tools for

- Data splitting
- Pre-processing
- Feature selection
- Model tuning
- Variable importance estimation
- Model validation

- **Many** models already integrated into this framework:

Model
Boosted Classification Trees
Bagged AdaBoost
AdaBoost.M1
Adaptive Mixture Discriminant Analysis
Adaptive-Network-Based Fuzzy Inference System
Model Averaged Neural Network
Naive Bayes Classifier with Attribute Weighting
Tree Augmented Naive Bayes Classifier with Attribute Weighting
Bagged Model
Bagged MARS
Bagged MARS using gCV Pruning
Bagged Flexible Discriminant Analysis
Bagged FDA using gCV Pruning
Bayesian Additive Regression Trees
Bayesian Generalized Linear Model
Self-Organizing Map
Binary Discriminant Analysis
Boosted Tree
The Bayesian lasso
Bayesian Ridge Regression (Model Averaged)
Random Forest with Additional Feature Selection
Bayesian Ridge Regression
Bayesian Regularized Neural Networks
Boosted Linear Model
Boosted Smoothing Spline
Boosted Tree
C5.0
Cost-Sensitive C5.0
Single C5.0 Ruleset
Single C5.0 Tree
Conditional Inference Random Forest
CHi-squared Automated Interaction Detection
SIMCA
Conditional Inference Tree
Conditional Inference Tree
Cubist

http://topepo.github.io/caret/modelList.html

What are the options?

<u>High-level overview</u>

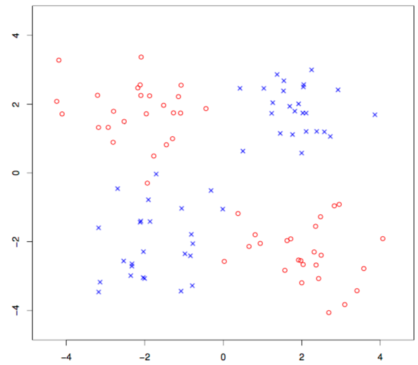
- *No stats*
- *No equations*
- *No jargon***
- *All "off the shelf"*

<u>Cover some popular algorithms</u>

- *k*-Nearest Neighbours
- Support vector machine
    - linear
    - non-linear (radial)
- Decision tree
- Penalized regression

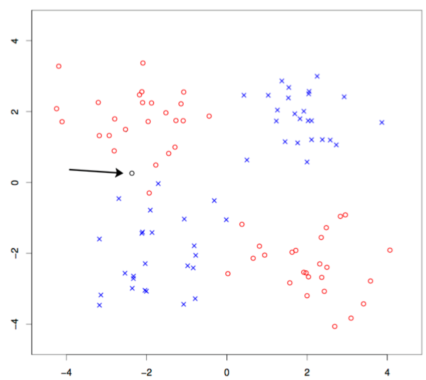
***call me out on it if I do!*

k-Nearest Neighbours



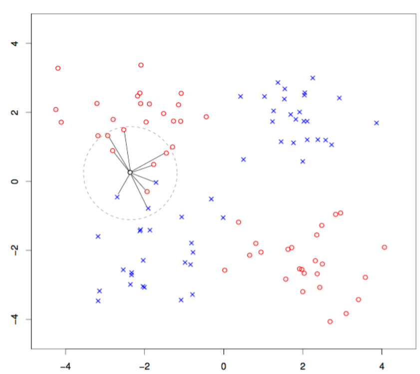
1 - Get data

k-Nearest Neighbours



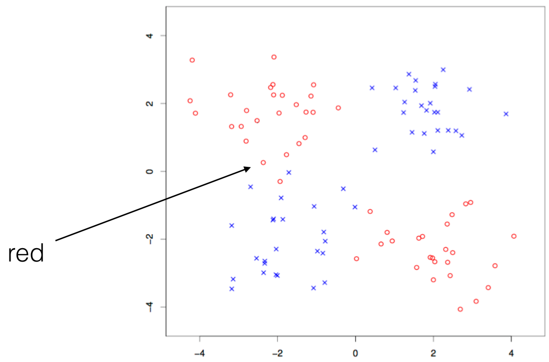
2 - bring a new observation

k-Nearest Neighbours



3 - find the k most similar observations to that one

k-Nearest Neighbours

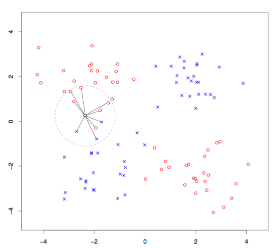


4 - prediction is the most common outcome in the circle

k-NN - finer details (optional)

Defining neighbours

- Which “distance measure”?
  - Pearson/spearman correlation
- How many neighbours?
  - Choose k through cross-validation
- Are all neighbours equal?
  - Weighted circle of neighbours

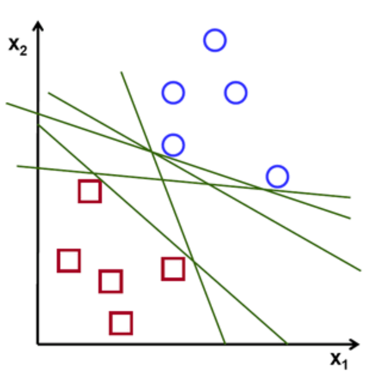


Dimensionality?

- PCA first, then k-NN

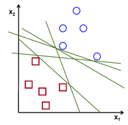
https://saravananthirumuruganathan.wordpress.com/2010/05/17/a-detailed-introduction-to-k-nearest-neighbor-knn-algorithm/

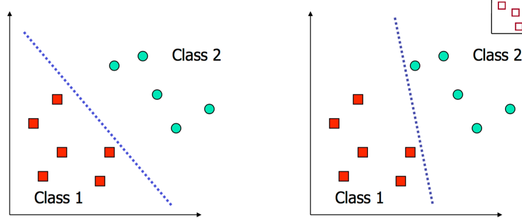
Support Vector Machine



Aim: find a boundary that (optimally) separates our classes

Support Vector Machine

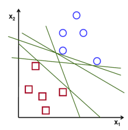


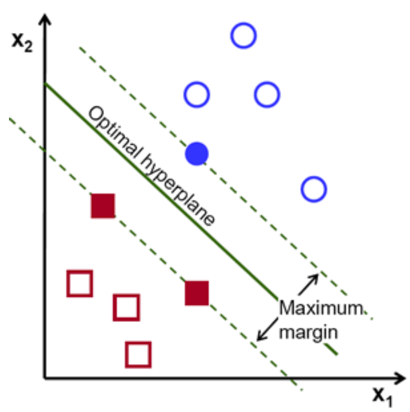


(Bad boundaries)

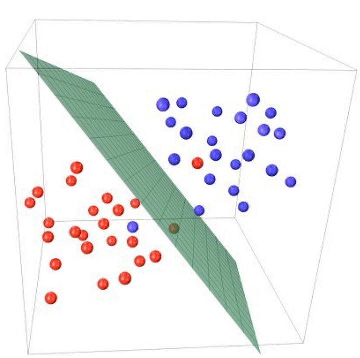
- Max-margin theory: boundary should be as far away from the data as possible

Support Vector Machine





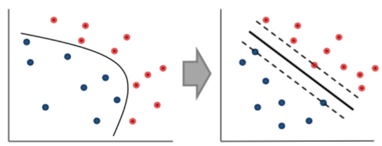
Support Vector Machine

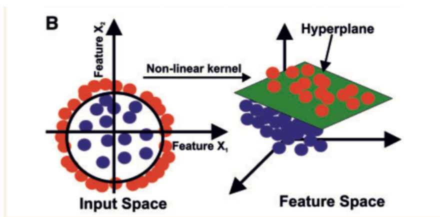


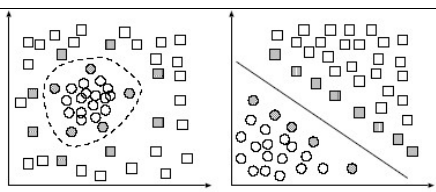
(hyperplane is not just in 2-3D)

Support Vector Machine

Nonlinear SVM

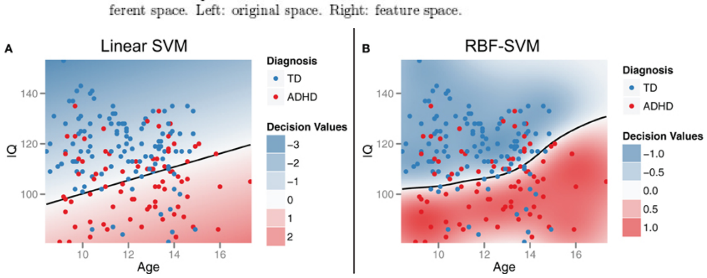




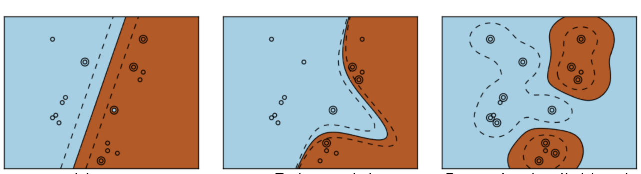


(a) Radial Basis Function (b) RBF mapping

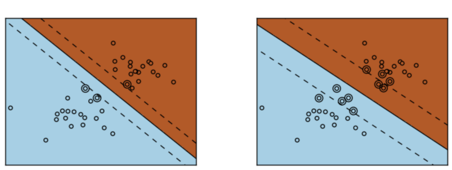
Separable classification with Radial Basis kernel functions in different space. Left: original space. Right: feature space.



SVM - finer details (optional)

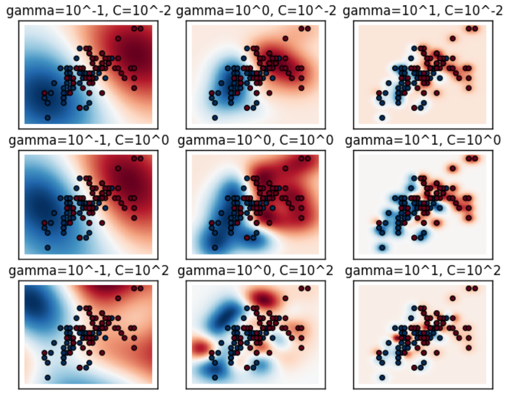


Linear Polynomial Gaussian/radial basis



Varying “cost”, *C*

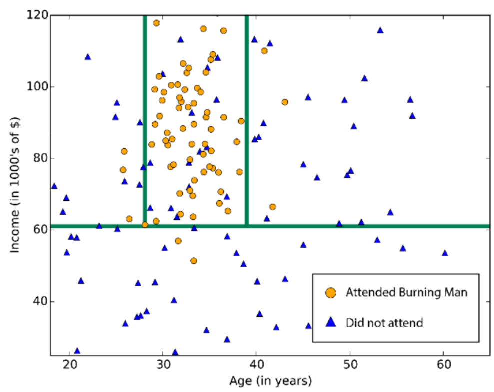
SVM - finer details (optional)



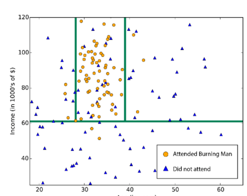
—-> See scikit learn SVM documentation for more information

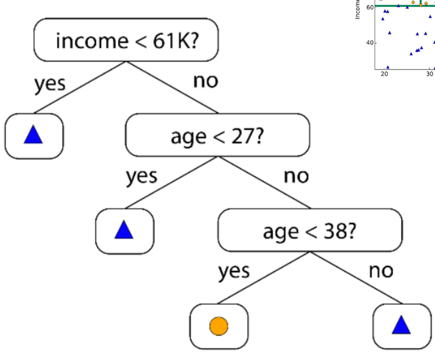
Decision trees

- Disclaimer: I think they are awesome



Decision trees





Decision trees - downsides

Not a one-size-fits-all

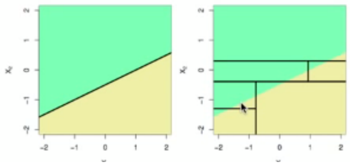
- can be clumsy

Finer details

- how do we choose the variable to split on?
- when do we stop splitting?
- what if the tree gets too large?
- define *accuracy*?

Low bias, but high variance

- Accurate, but can overfit



Trees vs linear models

Decision trees - advanced

What if we make loads of really small trees?

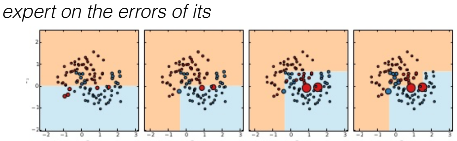
Bagging - “forests” of trees

- Take bootstrap samples (e.g. 1000)
- Fit a small tree (bad accuracy)
- Average predictions of all 1000 trees

NB: when these trees only use a few (randomly chosen) features, they are called a “random” forest

Boosting

- As above, but every time we fit a new tree, upweight the misclassified observations
- *“each tree is an expert on the errors of its predecessor”*



Penalized regression

Penalization, or regularization, is the imposing of constraints on a model to limit its complexity

Key concept:

- Just because our model is accurate, doesn’t mean it will fit new data
- Maybe a simpler, less accurate model would be better in future?

(Statistically:

- Inclusion of more terms in the model causes high variance in coefficient estimates
- Will re-visit when we talk about bias-variance tradeoff)

Penalized regression

Only include predictors if they are "worth" it

- i.e. they reduce RSS enough to justify inclusion
- two typical methods: LASSO and Ridge regression

L1-regularization:

- aka. LASSO regression
- minimize total absolute beta weight

$$\text{minimize} \quad SSE + \underbrace{\lambda}_{\text{parameter}} |\underbrace{\beta}_{\text{coefficients}}|$$

Penalized regression

L2-regularization:

- aka. Ridge regression
- minimize total squared beta weight

$$\text{minimize} \quad SSE + \underbrace{\lambda}_{\text{parameter}} |\underbrace{\beta^2}_{\text{coefficients}}|$$

Elastic net regression: blend of both penalties

Conclusions

- ML algorithms help us identify patterns in the data that predict outcomes we care about
- No one algorithm fits all, so try multiple. But consider:
    - Accuracy, feasibility, and the nature of your problem
- Starting points?
    - k-NN, SVM, Trees (and fancy trees, or penalized methods)
- This was just an intro!
- Try them out— best way to learn good parameters

BUT

- **ML is not a replacement for *lots* of good, useful data.**
- Fancy algorithms not a substitute for good predictors
    - Don't fall down the rabbit hole…..