3011979 Practical Python for Data Sciences and Machine Learning

L6: Machine learning pipeline

Feb 18th, 2022

Sira Sriswasdi, Ph.D.
Research Affairs, Faculty of Medicine
Chulalongkorn University

We seek understanding of mechanisms

Trajectory of the ball

Newton's laws of motion

Initial force and angle

Gravity

$$y = f(x)$$

Drug response in patient

Drug chemistry

Human biology

Clinical data

Genomics data

And a lot more…

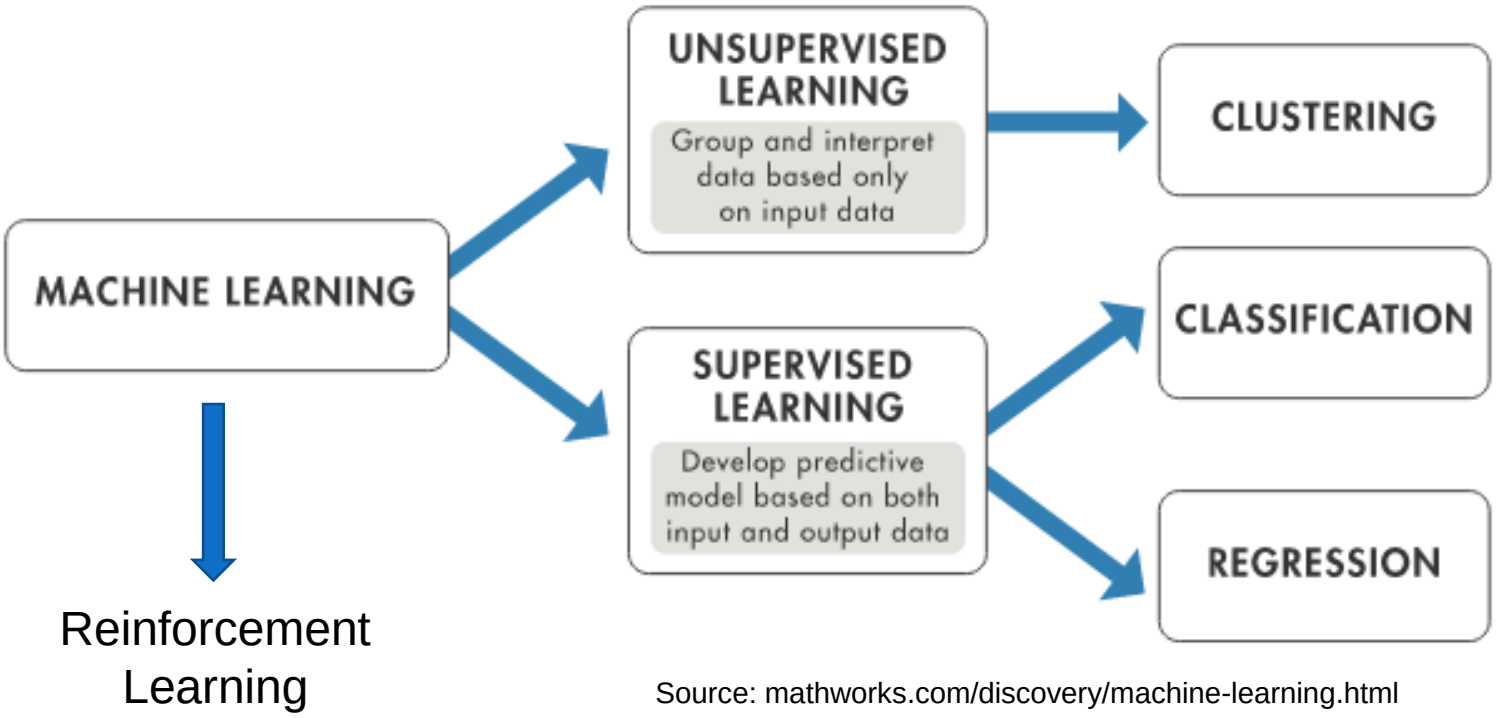
Statistics vs machine learning

$$y = f(x)$$

- **Statistics (and math modeling)** uses the whole (x, y) to learn about f
  - Constrained by our knowledge of the mechanisms
  - First order approximation = linear model

- **Machine learning** uses some of (x, y) to optimize the best f and then test it on the unseen data
  - Model can be more complex than the actual mechanisms
  - Even noises can be fitted almost perfectly
  - Multiple solutions exist!

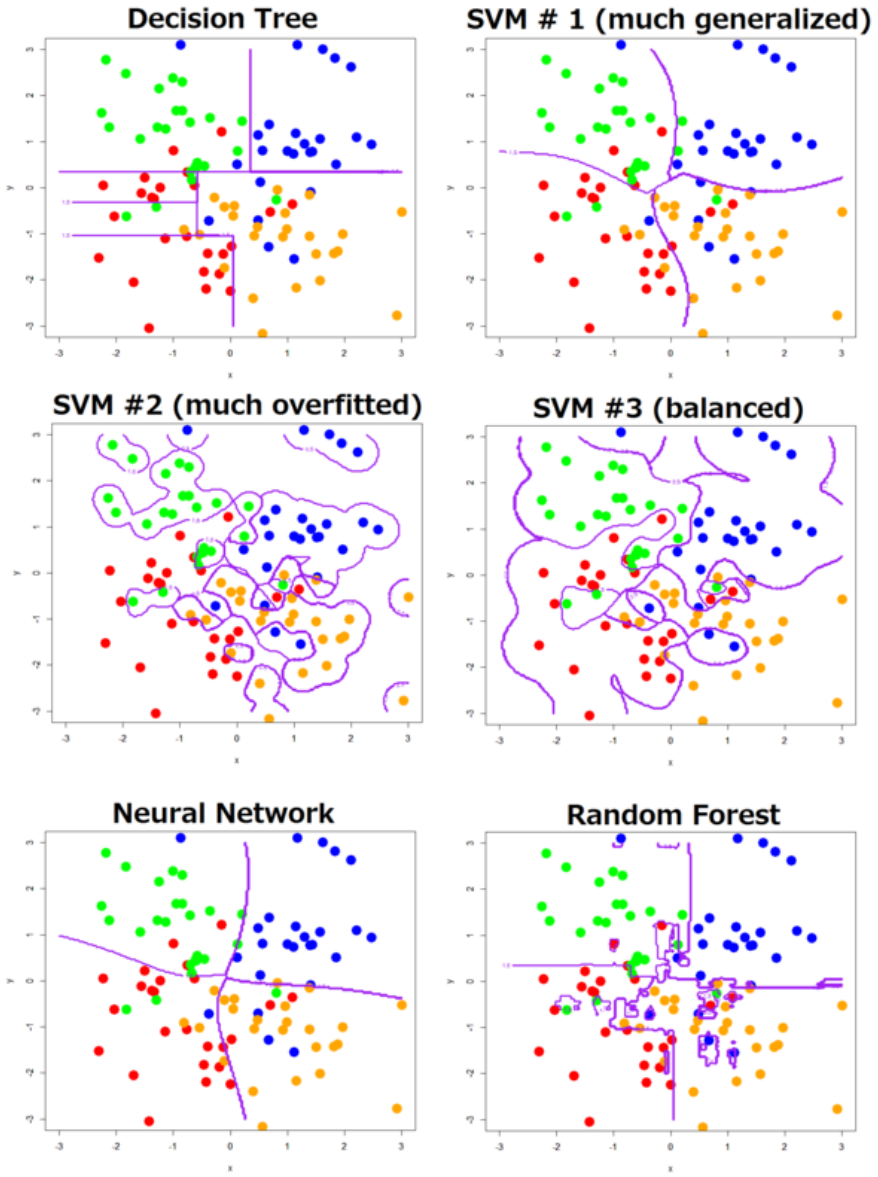
Supervised learning



Source: mathworks.com/discovery/machine-learning.html

- Classification = predict class (e.g., yes/no, subtype A/B/C)
- Regression = predict real number (e.g., p(sick), drug effect)

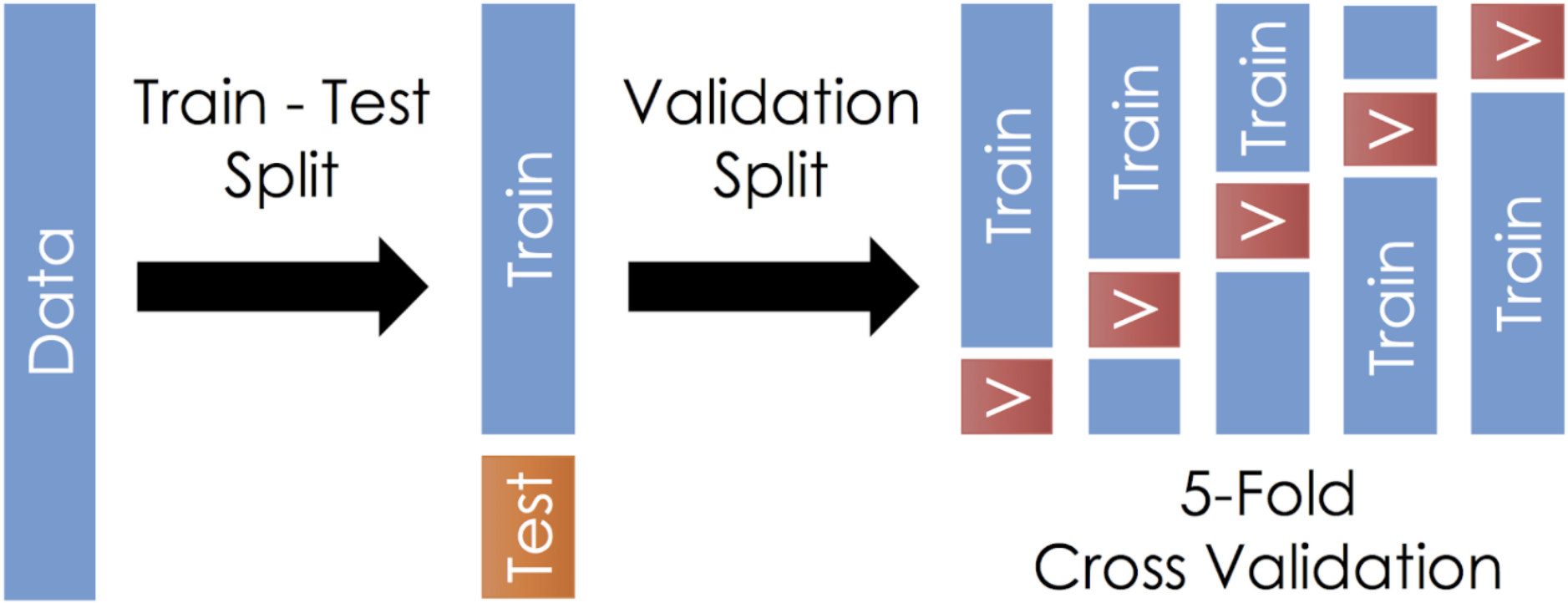
Complex model **tends to** overfits



- Most of what we do in machine learning is to <u>prevent overfitting</u>
- Use of validation set
- Remove noisy features
- Regularize model complexity

Validation

Validation



Source: medium.com

- **Training set** is used to optimize model f
  - Select the model that performs best on the **validation set(s)**
  - Cross-validation lessens the chance that the model overfit
- Then check the model's performance on the **test set**

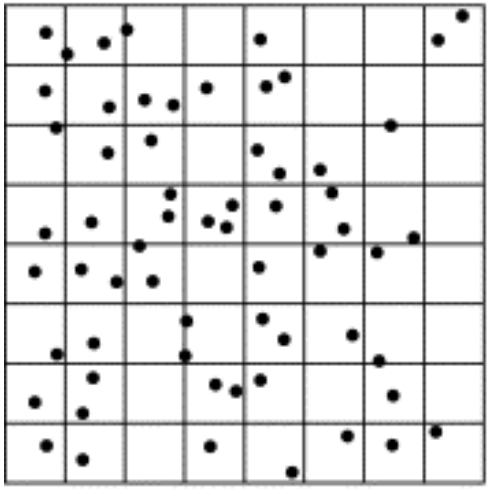
Expected model behavior

- Performance on **training set** is near perfect
- Performance on **validation set** is lower
  - But this is still an over-estimation of the model's performance
- Performance on **test set** is most representative

- **Key assumption**: all sets are representative of the distribution of input data

- In practice, you may find that performance on the validation set is lower than on the test set
  - It's about how the data were split

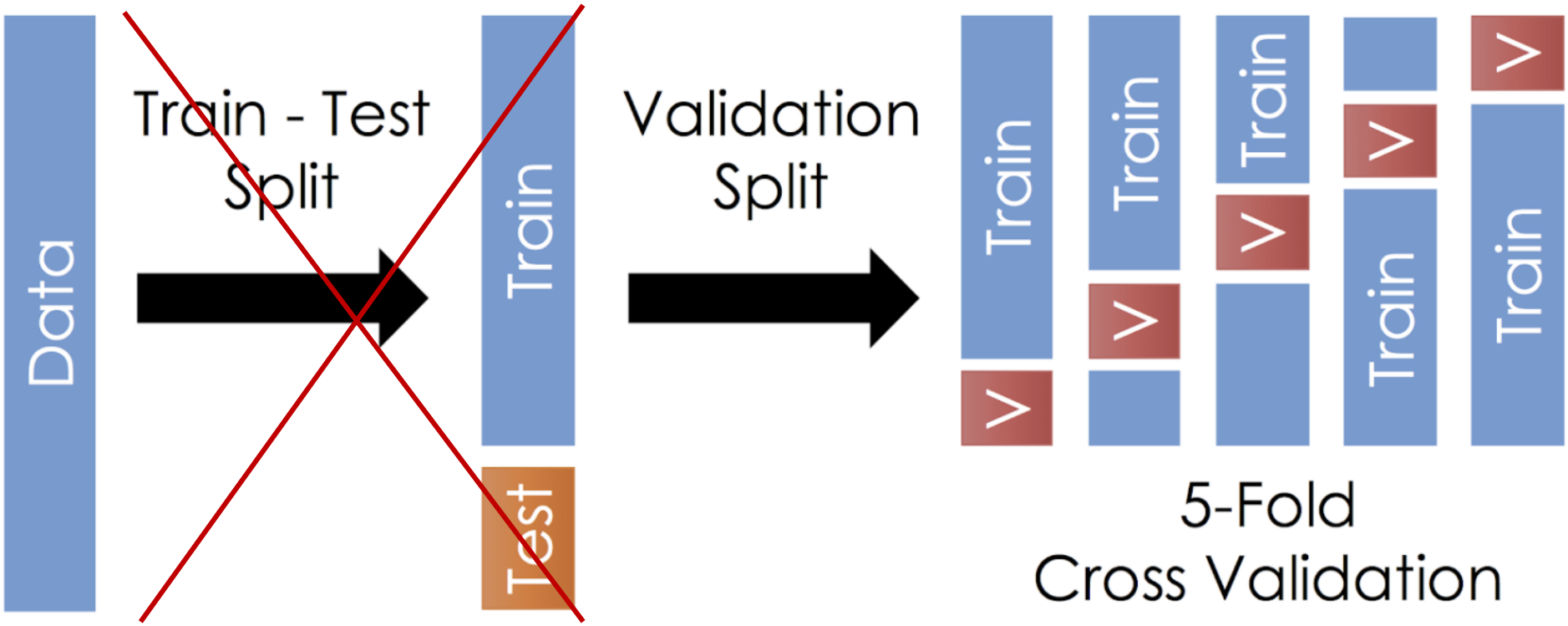
Dealing with small dataset



Some parts of data distribution are unobserved

- Small dataset may not reflect the whole data distribution
  - Further amplified by data splitting
- Which set would you assign less samples to?
  - Small test set = cannot trust the estimated performance
  - Small validation set = cannot reliably identify the best model
  - Small training set = the model does not fit some part of the distribution

It is ok to not have the (external) test set



- With small dataset, train-test split is appropriate
  - Splitting reduces the amount of data for training and tuning
  - Small test set cannot represent the true data distribution anyway
- Can only propose the method, not the trained model

Validation schemes

- ***k*-fold** cross-validation
  - Split data into ***k*** equal-sized partitions
  - In each round, train models on ***k* – 1** partitions and evaluate the performance on the last partition
  - ***k*** models are trained and evaluated

- **Leave-one-out** (LOO) cross-validation
  - ***k*-fold** cross-validation with ***k*** = number of samples – 1
  - **number of samples – 1** models are trained and evaluated
  - The performance is either 0 or 1 (for classification)

- **Bootstrapping**
  - **Randomly split x%** of the data as training set
  - Repeat the process multiple times

Feature selection

Uninformative features can be detrimental

- Uninformative features can correlate with the outcome by chance and fool the model
  - Often the case for small dataset
  - Complex model can always discover a combination of noisy features that fit the data well
- Some models have built-in ability to avoid uninformative features, but many do not
  - Linear models are forced to consider all features
    - Alleviated by L1-regularization (LASSO)
  - Random forest can handle this well
  - Neural network suffers the most from this problem

Univariate feature selection

- A preliminary filter using univariate technique can be helpful
  - Correlation between an input feature and an output (regression)
  - Test of mean or ANOVA between input feature means across different output classes (classification)

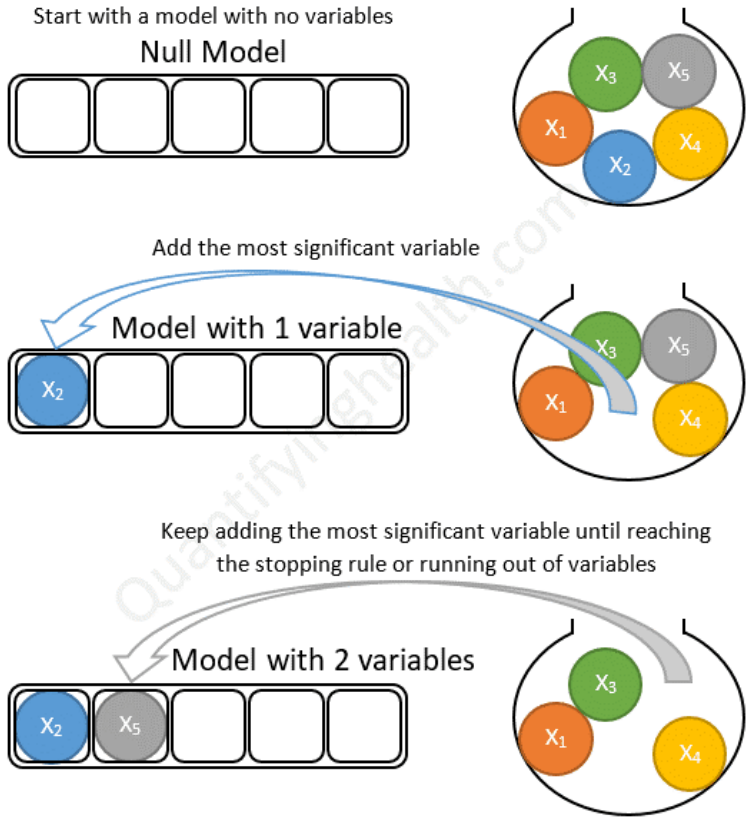
- Some features might be predictive only on a subpopulation and do not exhibit high correlation on the whole dataset
  - Do not be too aggressive on the univariate step

Sequential feature selection schemes

Forward Selection

Add one good feature at a time

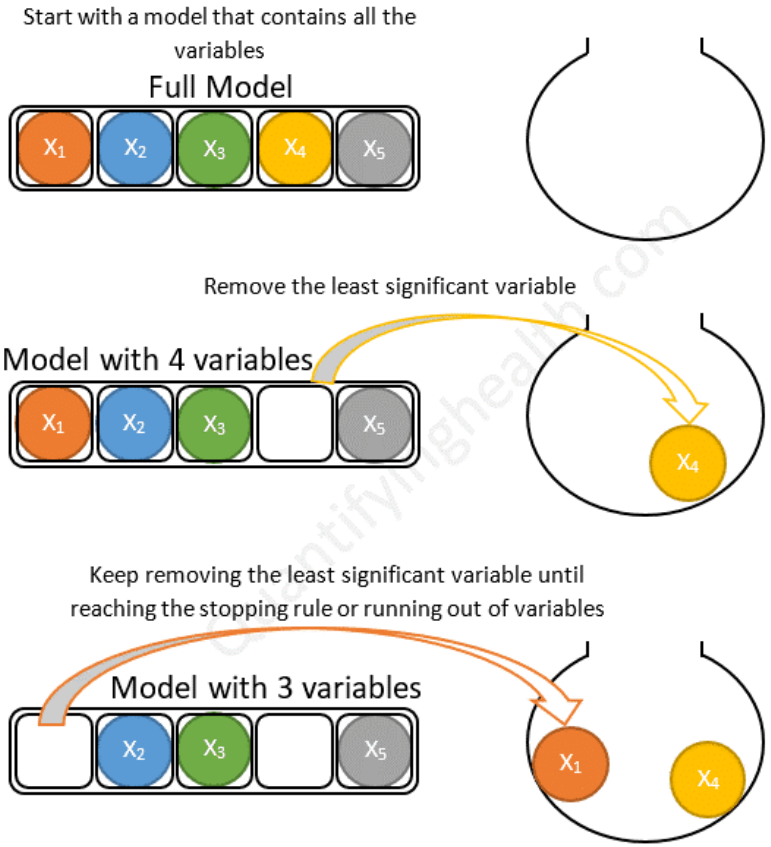
Forward stepwise selection example with 5 variables:



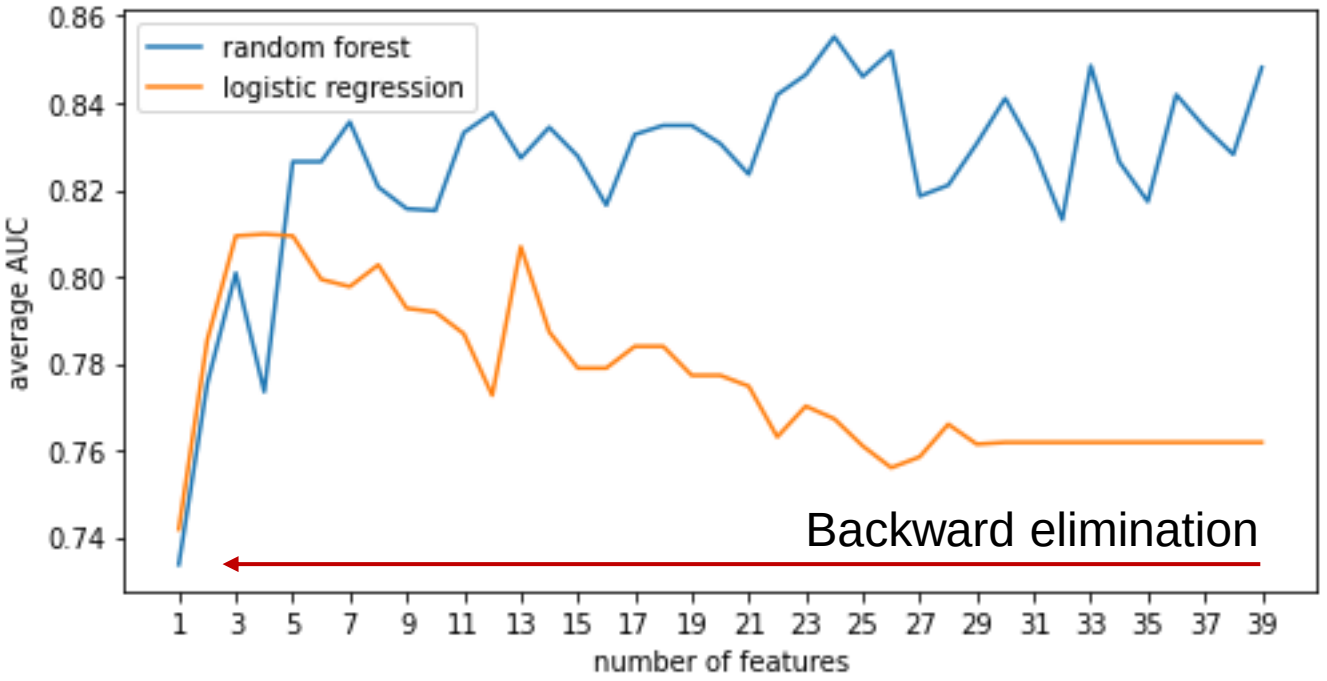
Backward Elimination

Remove one poor feature at a time

Backward stepwise selection example with 5 variables:

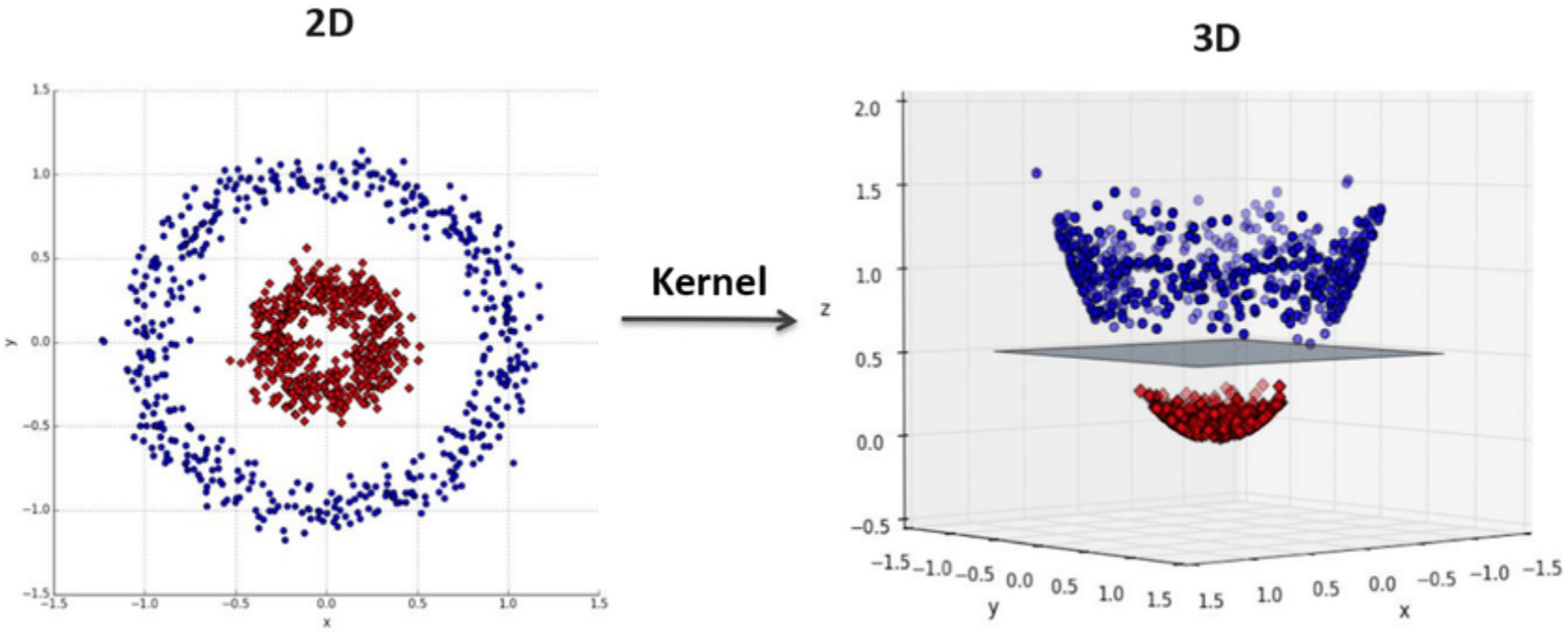


Impact of feature selection



- Random forest does not need help select feature
  - But doing so can reduce the burden of future data collection
- Logistic regression benefits a lot from feature selection
- There are ~5-7 essential features that cannot be removed

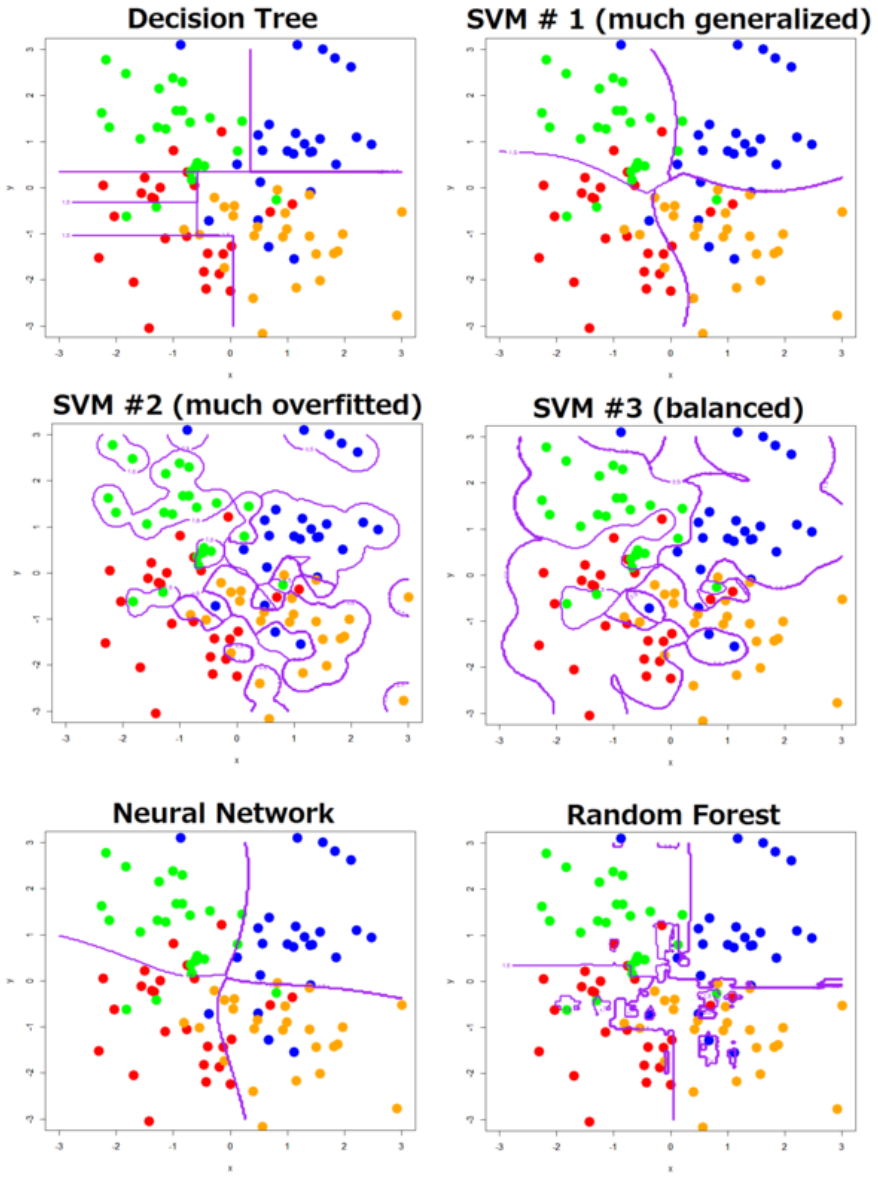
Feature engineering



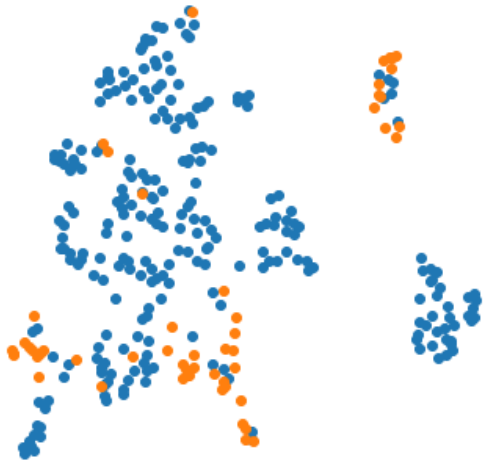
- Transform (x, y) to (x, y, $x^2 + y^2$)
  - Blue data points are further away from origin → larger $x^2 + y^2$
  - Separating hyperplane is a linear combination of x, y, and $x^2 + y^2$ which is nonlinear with respect to x and y

Model regularization

Simple model can perform better



- Similar to nested model testing in statistics
- Develop a simple model as **baseline**
  - Linear / logistic regression
- Visualize data distribution



Coefficient regularization

$$\text{Ridge Loss} = C \cdot \sum_i \left(y_i - [b_0 + b_1 x_1^{(i)} + \cdots + b_n x_n^{(i)}] \right)^2 + \sum_i {b_i}^2$$

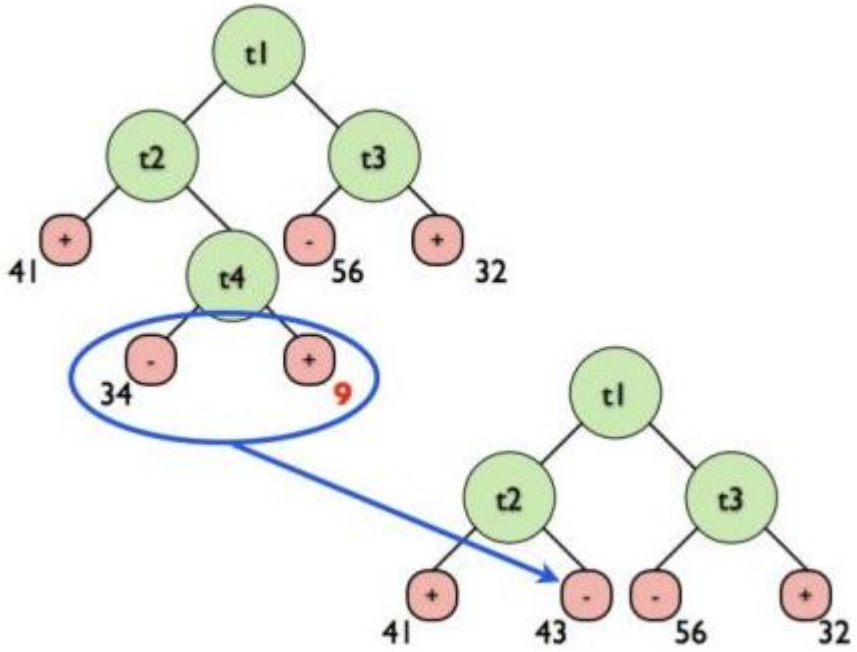
$$\text{LASSO Loss} = C \cdot \sum_i \left(y_i - [b_0 + b_1 x_1^{(i)} + \cdots + b_n x_n^{(i)}] \right)^2 + \sum_i |b_i|$$

- In addition to minimizing prediction error, we can force the model to consider other objectives
  - Make the magnitude of $b = (b_0, b_1, \ldots, b_n)$ small
  - Assign similar b_i's to clinical features from the same group

- **Small coefficient** means that a unit change in input feature will lead to **small change** in prediction output
  - Model becomes more robust to input noises

- Works with neural network as well

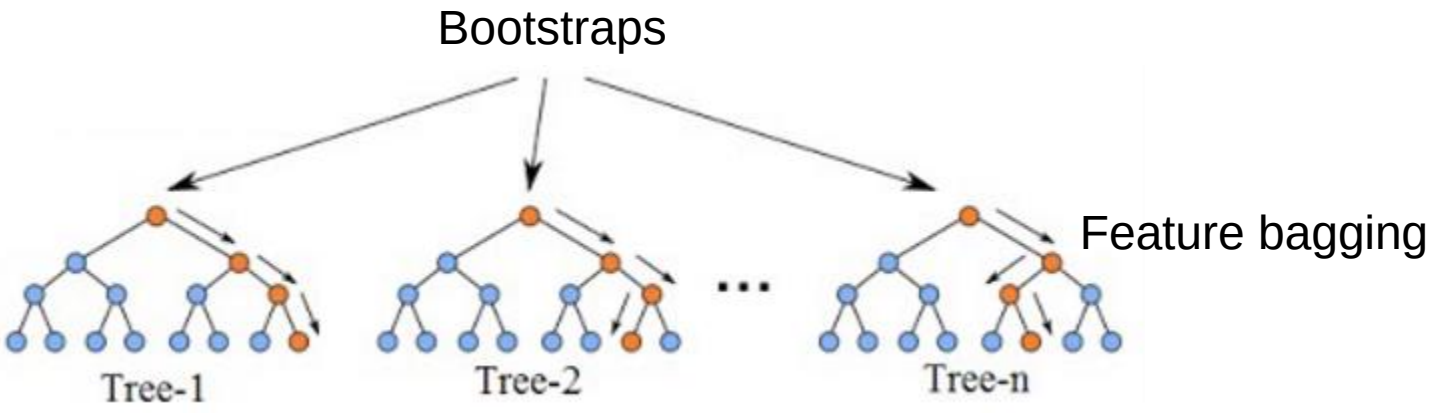
Tree pruning



Patel, N. and Upadhyay, S. “Study of Various Decision Tree Pruning Methods with their Empirical Comparison in WEKA” IJCA 2012

- Tree model’s complexity = number of decisions
  - $R_\alpha(\text{Tree}) = \text{Performance}(\text{Tree}) + \alpha \cdot \text{Number of leaves}$
  - α indicates the penalty on tree complexity

Ensemble approach



- Bootstrap bagging to generate multiple training sets
  - One model for each split
  - Aggregate prediction with a vote or average
- Ensemble can be formed between different model families
  - Logistic regression + decision tree

Any question?