Random Forests

Improving Feature Selection

Ronan Perry
Department of Biomedical Engineering
Johns Hopkins University

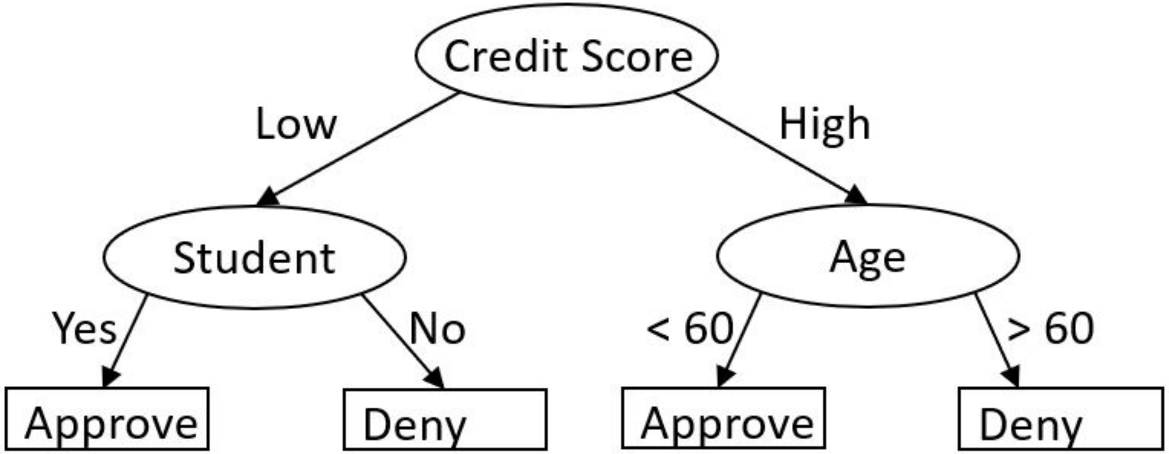
Overview

- A motivating problem
- Random Forests as a solution
- How decisions are learned
- How we can improve learning
- Why use Random Forests

Question: Should this loan request be approved?

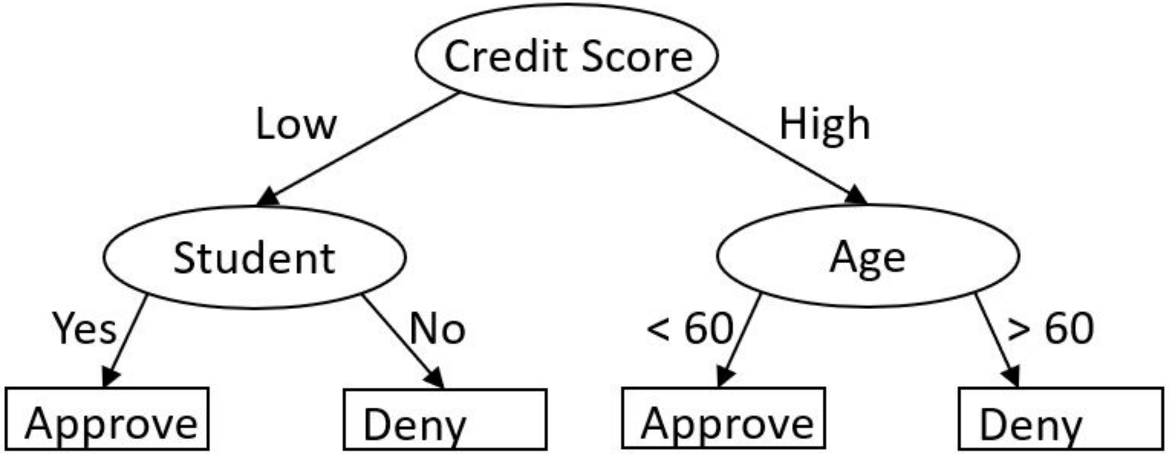
Question: Should this loan request be approved?

Possible Answer: Learn a Decision Tree



Question: Should this loan request be approved?

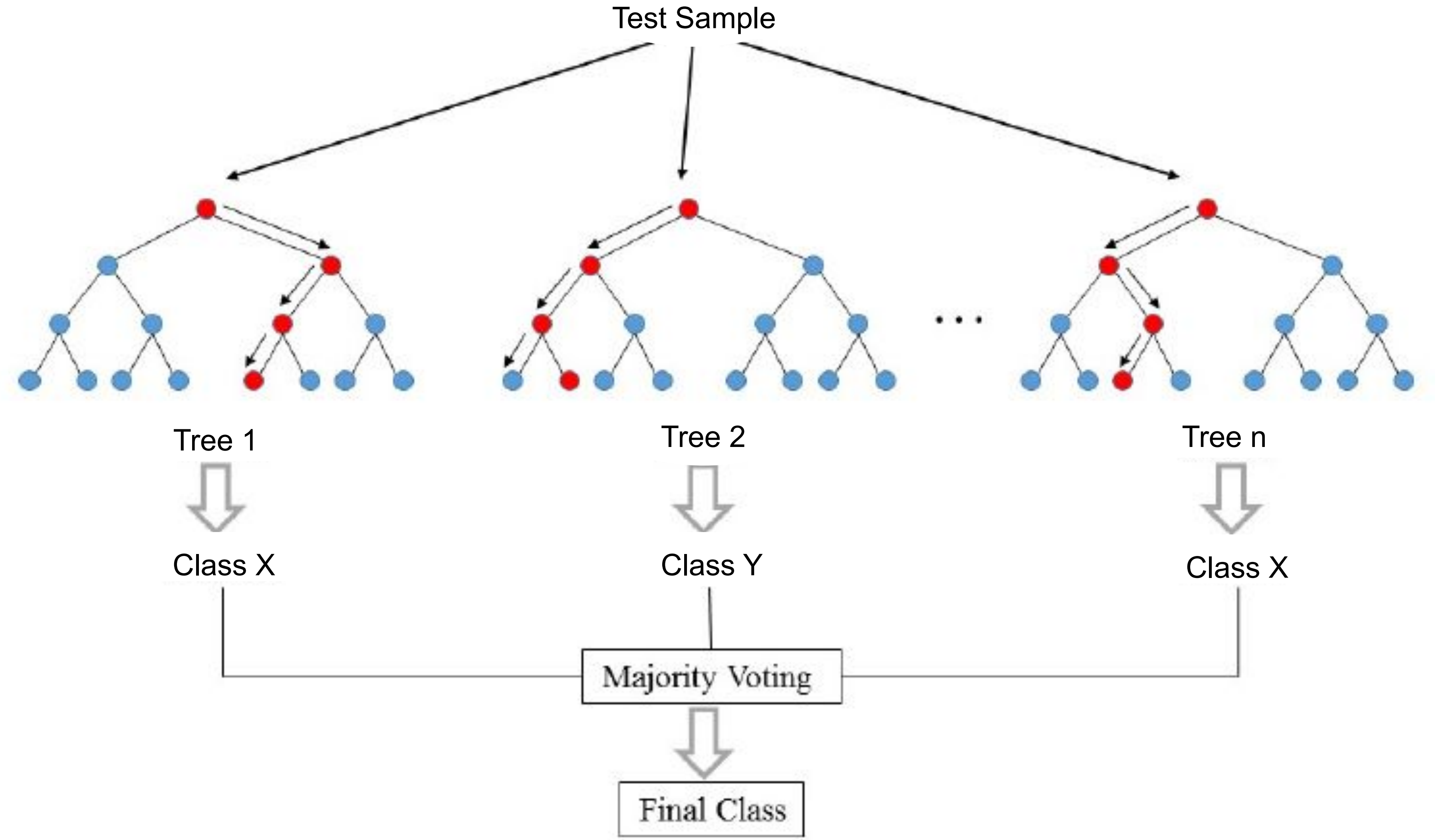
Possible Answer: Learn a Decision Tree



Problem: High variance

Random Forests

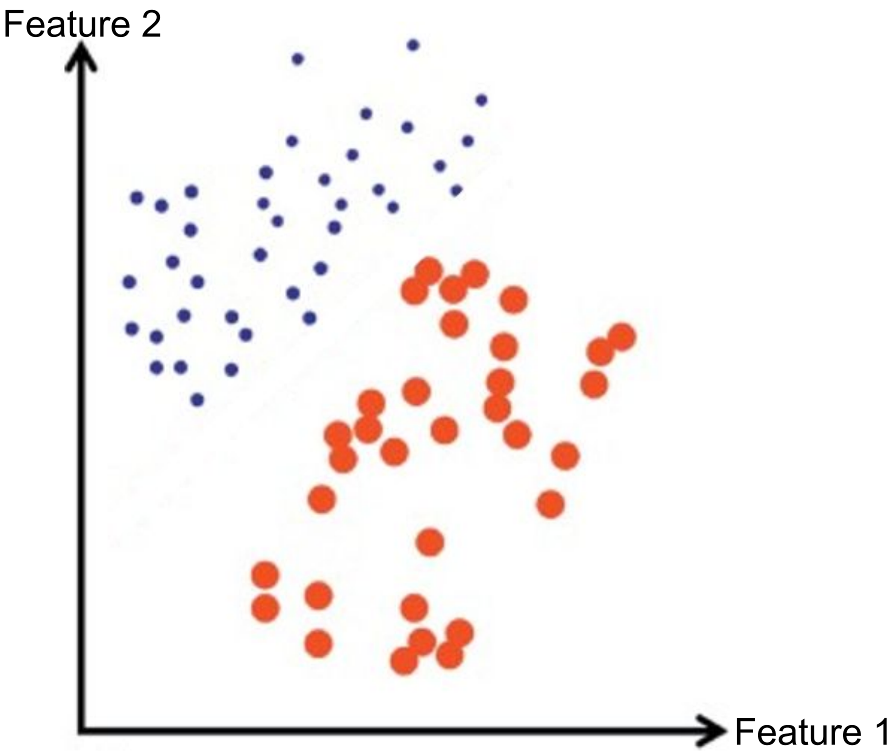
An **ensemble** of individual decision trees, each learned from a subset of the data, whose individual decisions are joined to make one final decision.


Test Sample
Tree 1
Tree 2
Tree n
Class X
Class Y
Class X
Majority Voting
Final Class

How are Decisions Learned?

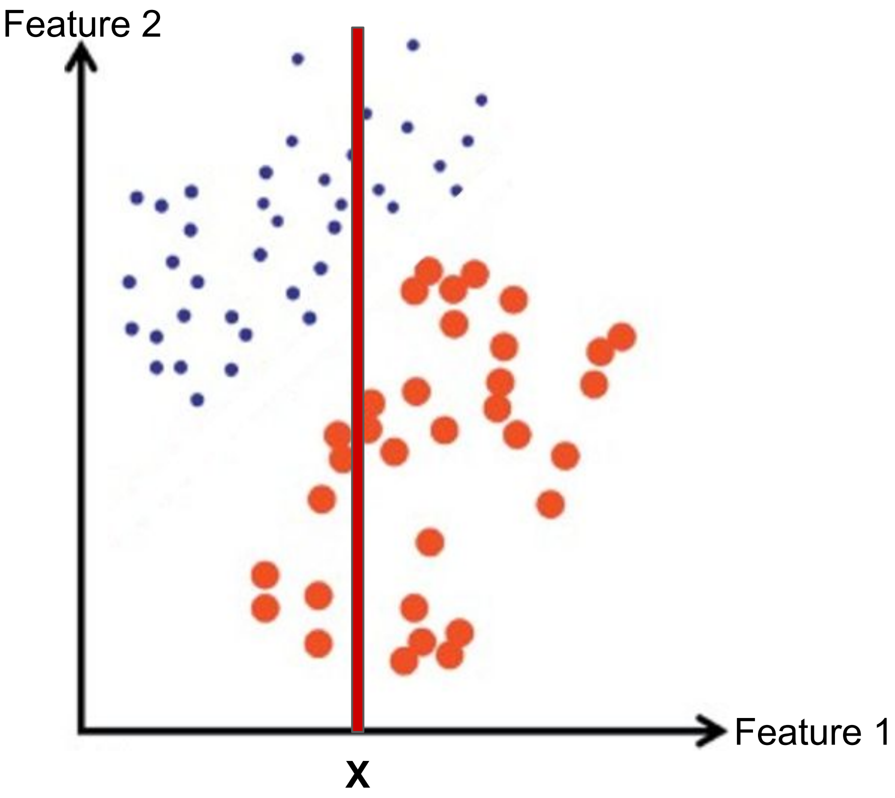
Axis-Aligned Splits

- At each split node in a tree:
  - Select a **single** feature (i.e. age)
  - Select a threshold (i.e. 60)



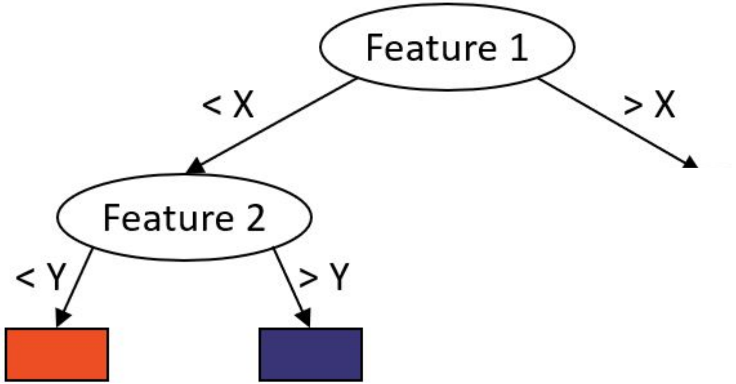
Axis-Aligned Splits

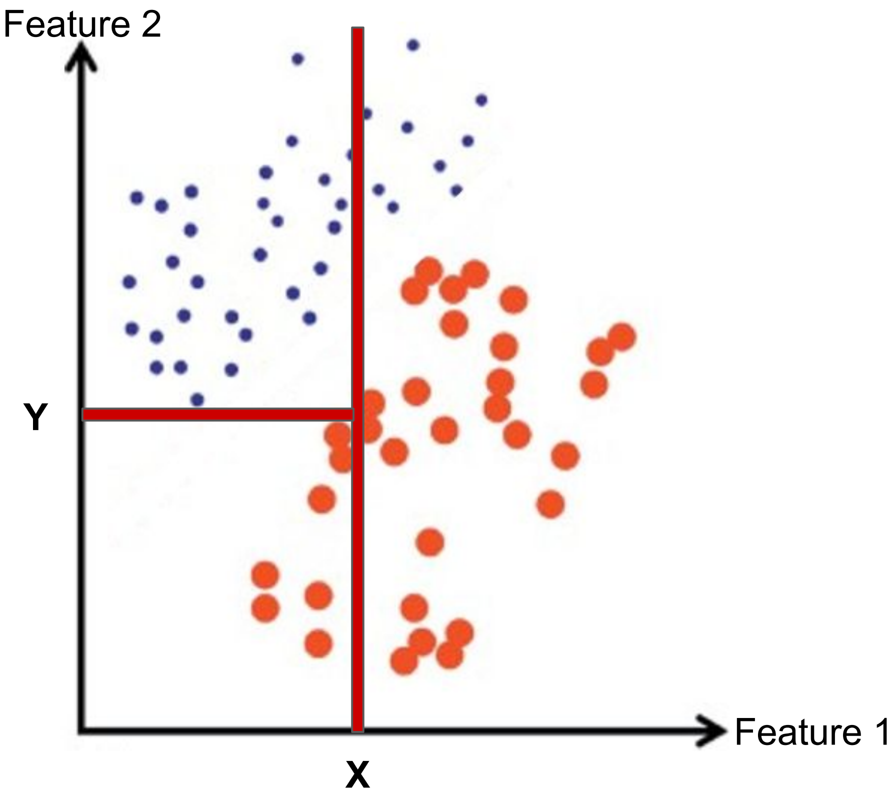
- At each split node in a tree:
  - Select a **single** feature (i.e. age)
  - Select a threshold (i.e. 60)



Axis-Aligned Splits

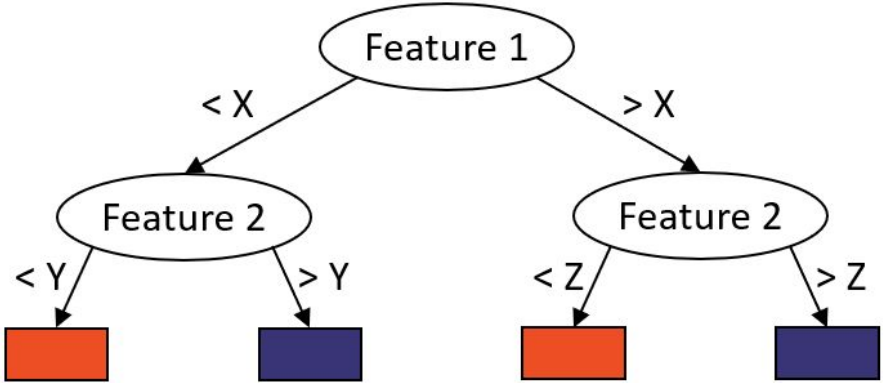
- At each split node in a tree:
  - Select a **single** feature (i.e. age)
  - Select a threshold (i.e. 60)

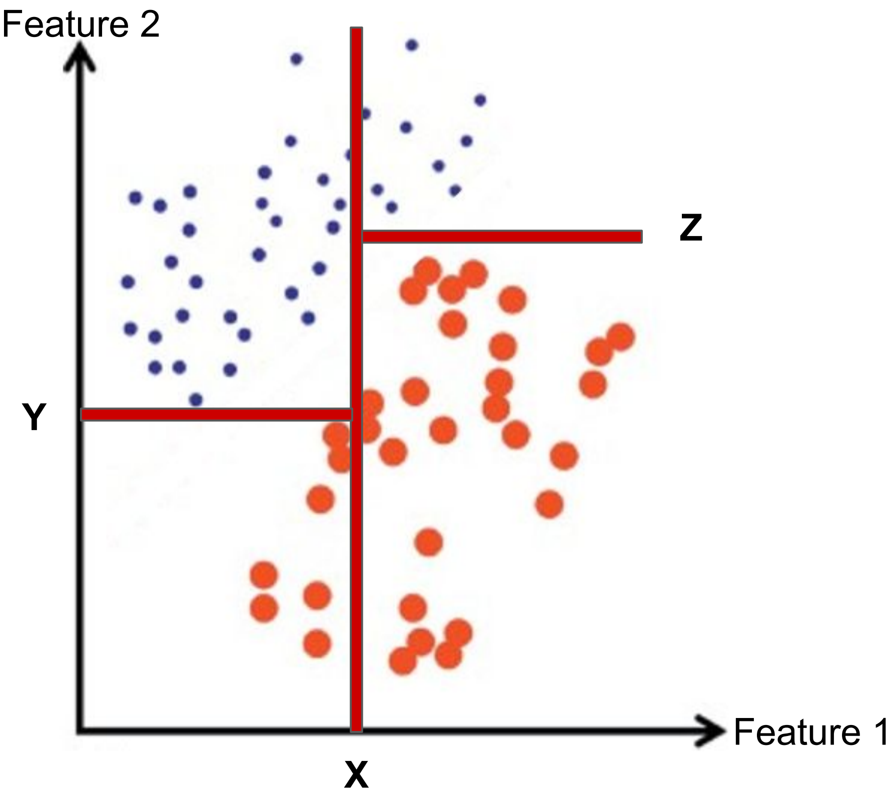




Axis-Aligned Splits

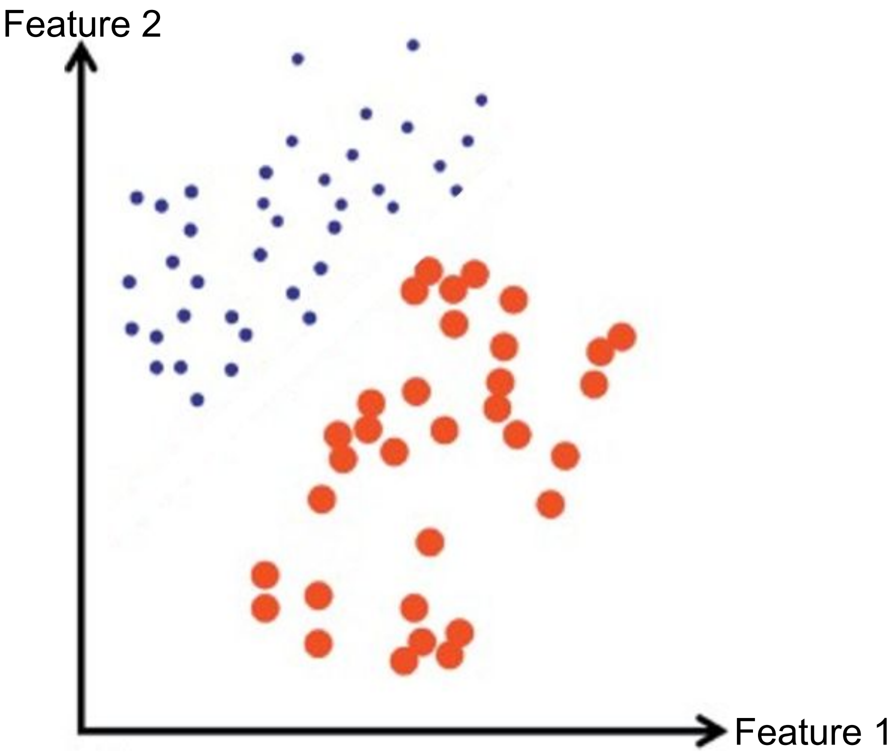
- At each split node in a tree:
  - Select a **single** feature (i.e. age)
  - Select a threshold (i.e. 60)





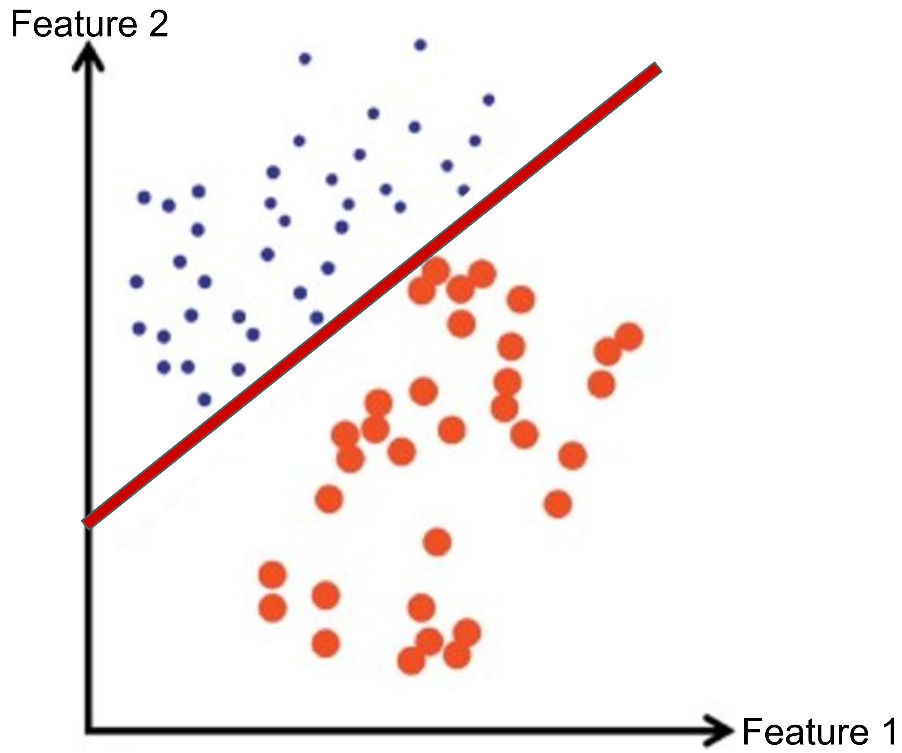
Axis-Aligned Alternative

- Oblique (angled) splits
  - Select a **combination** of features
  - Select a threshold



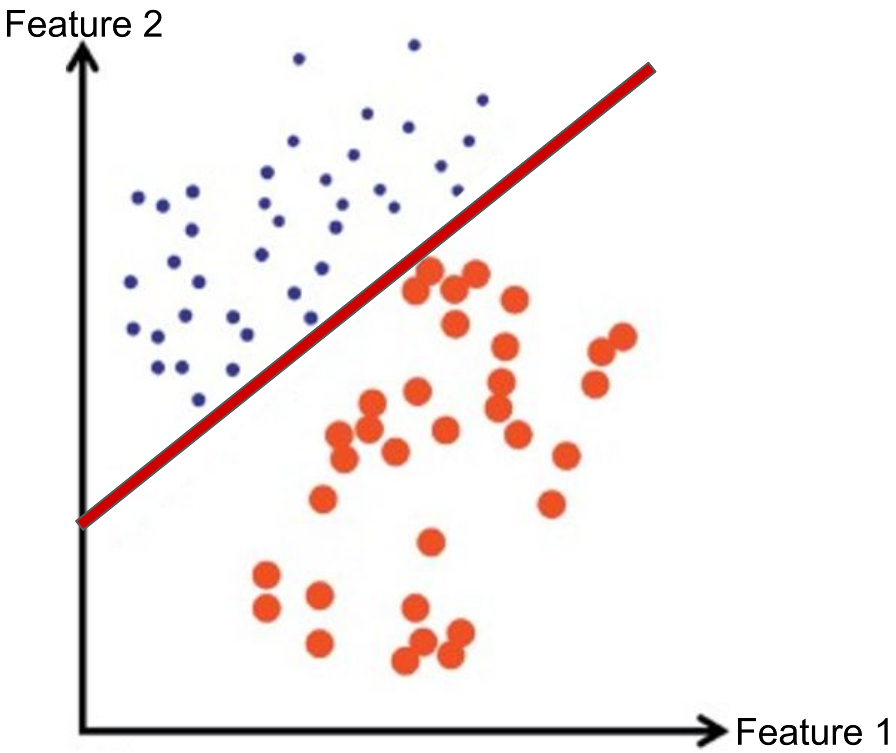
Axis-Aligned Alternative

- Oblique (angled) splits
  - Select a **combination** of features
  - Select a threshold



Axis-Aligned Alternative

- Oblique (angled) splits
  - Select a **combination** of features
  - Select a threshold
- Benefits:
  - Can identify more complex relationships
- Problem:
  - Can be computationally slow

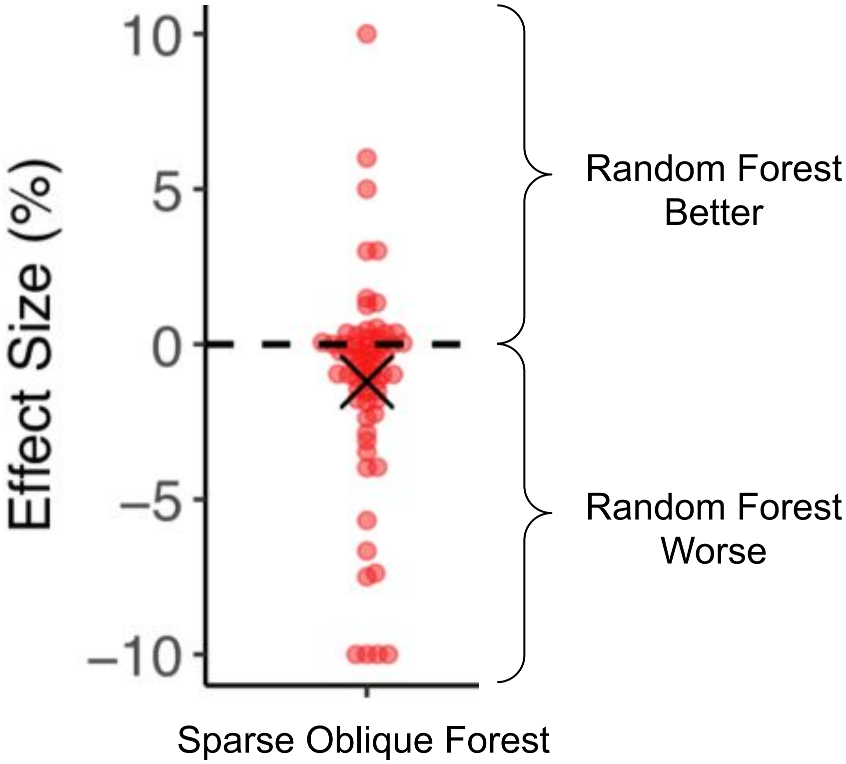


Sparse Oblique Splits

- A **sparse combination** of features

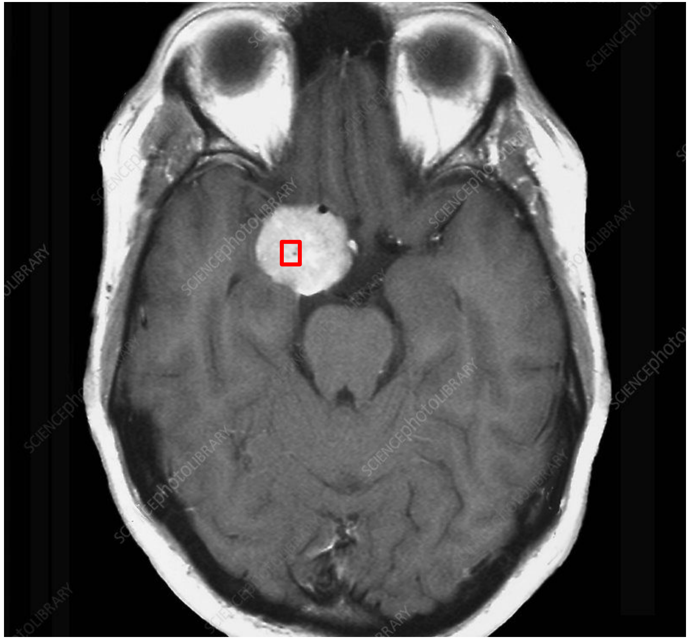
Sparse Oblique Splits

- A **sparse combination** of features
- Benefits
  - Increased signal to noise ratio
  - Faster computation
  - Improved accuracy in practice



Data with Feature Structure

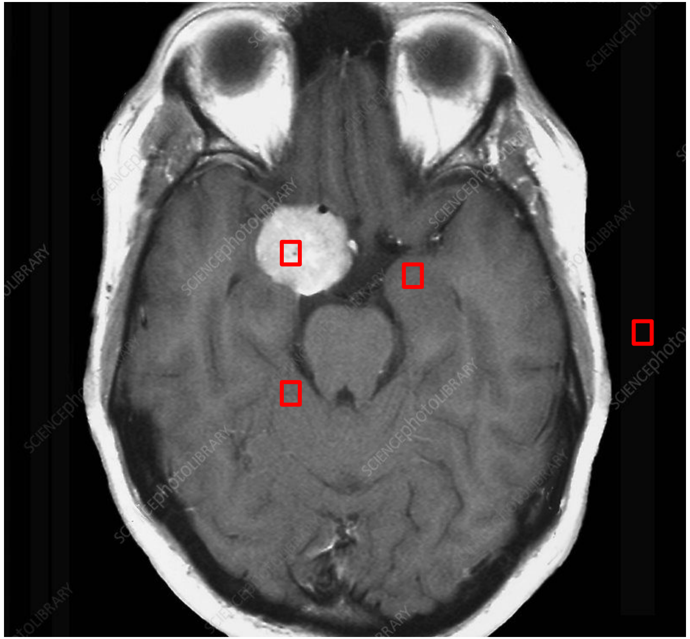
- In some data, **feature indices** matter
  - i.e. Images, time series, networks, etc.
- **Problem**: Random forests don't care



Random Forest Feature

Data with Feature Structure

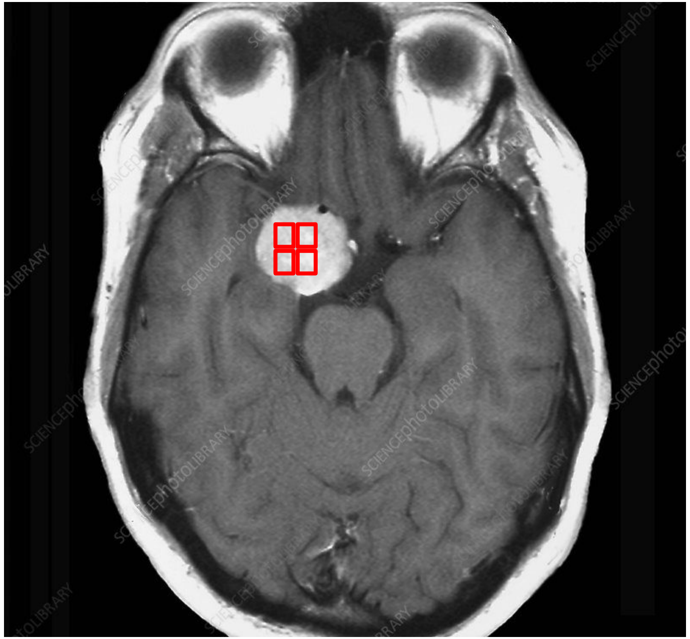
- In some data, **feature indices** matter
  - i.e. Images, time series, networks, etc.
- **Problem**: Random forests don't care



Sparse Forest Features

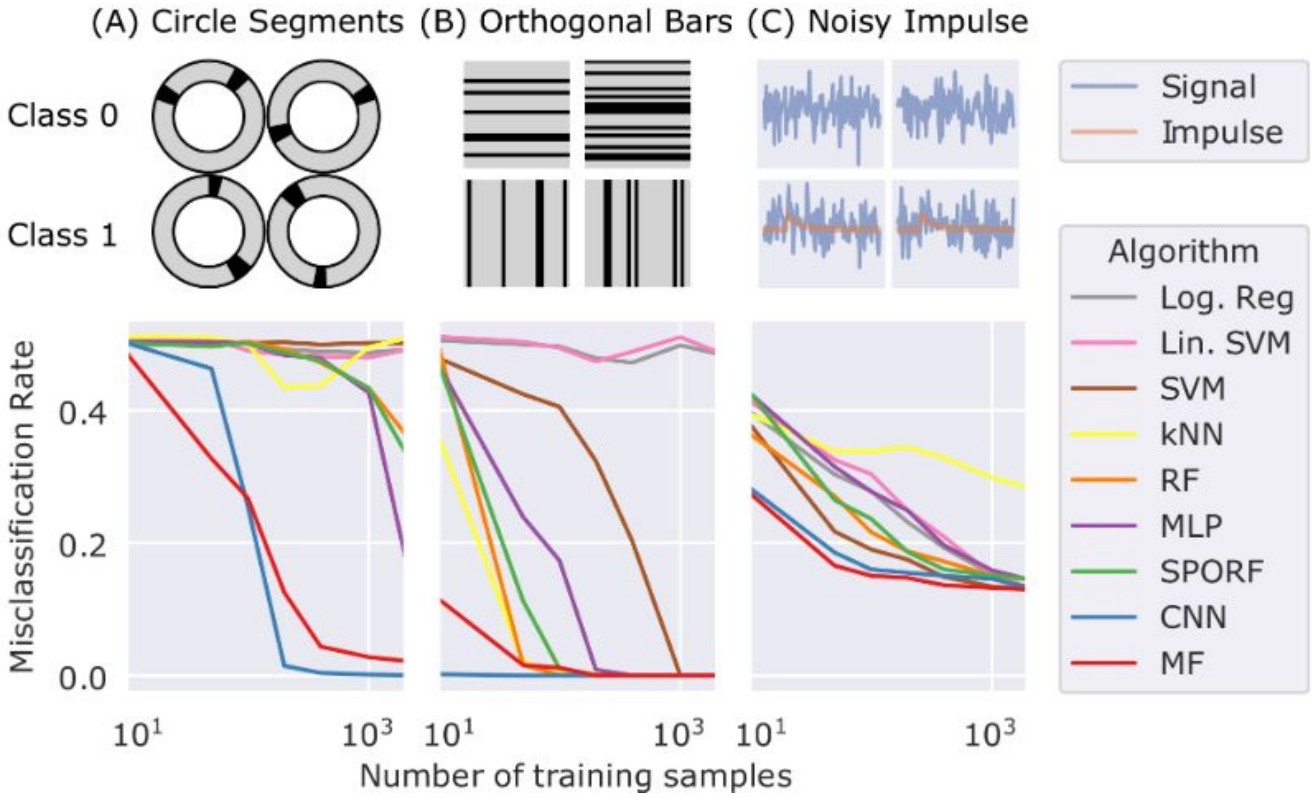
Data with Feature Structure

- In some data, **feature indices** matter
  - i.e. Images, time series, networks, etc.
- **Problem**: Random forests don't care



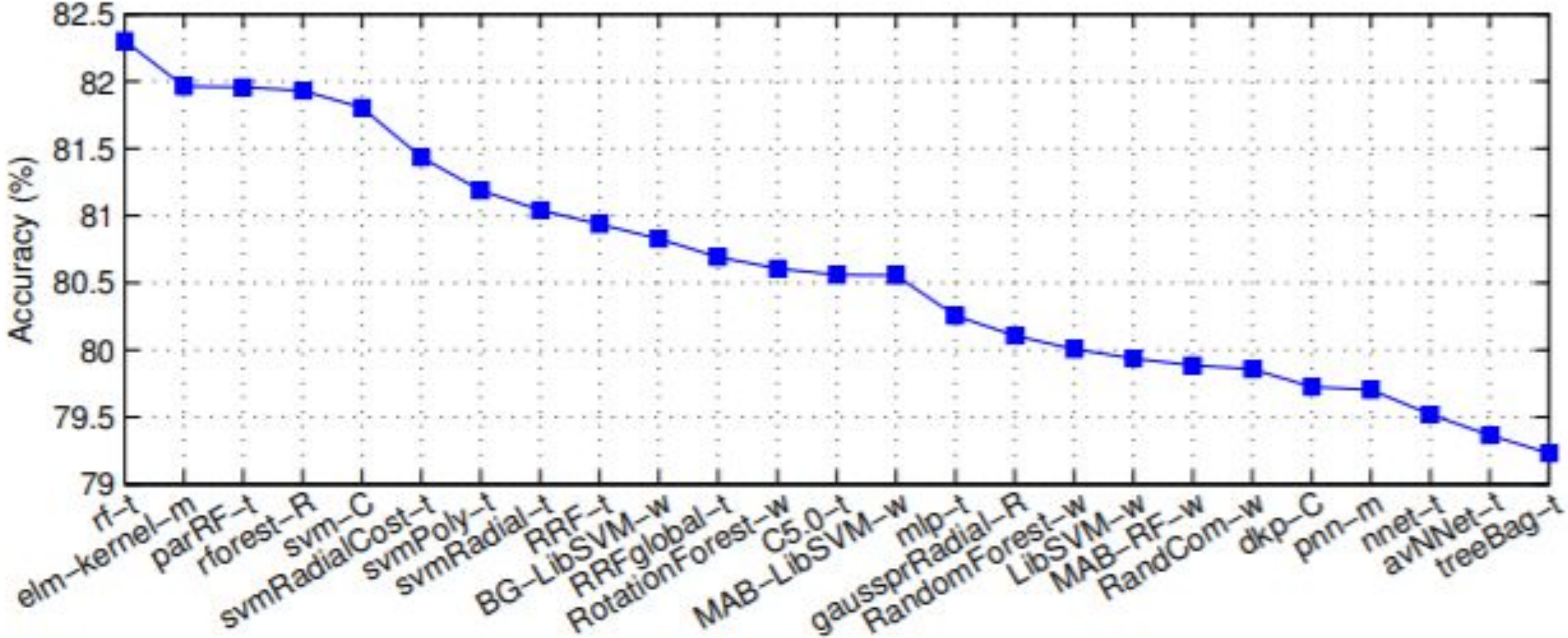
Structured Forest Features

Structured Splits



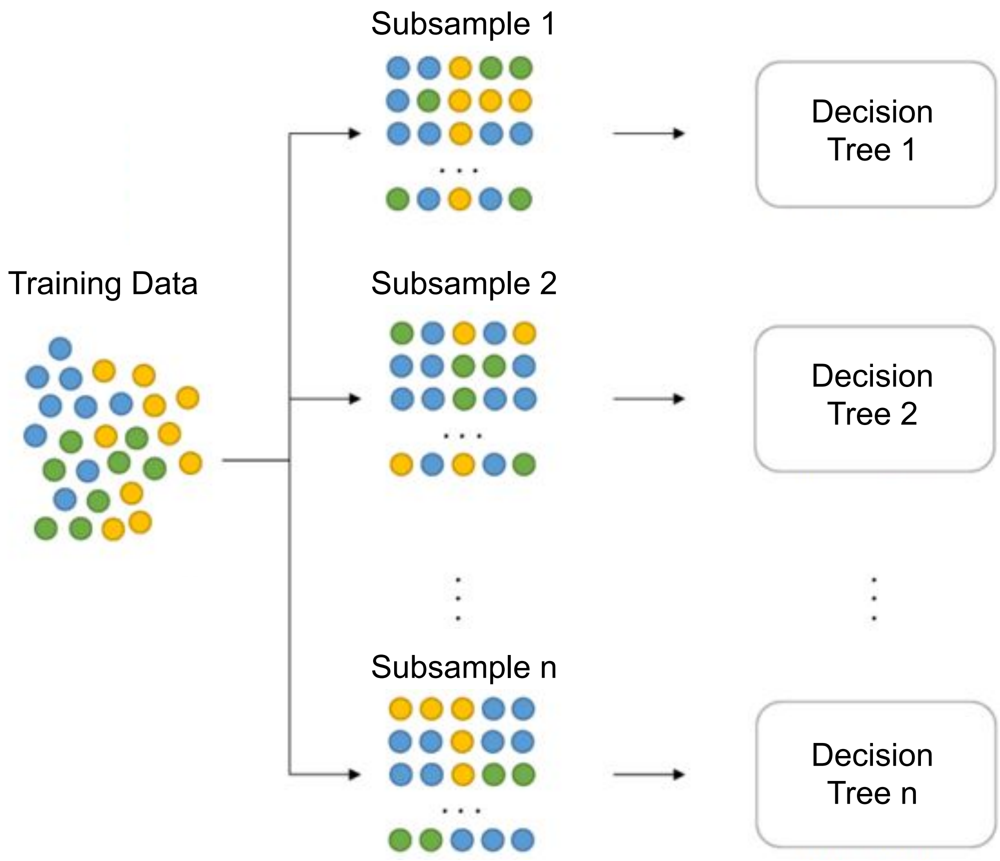
Why Random Forests?

Best average accuracy across hundreds of data sets



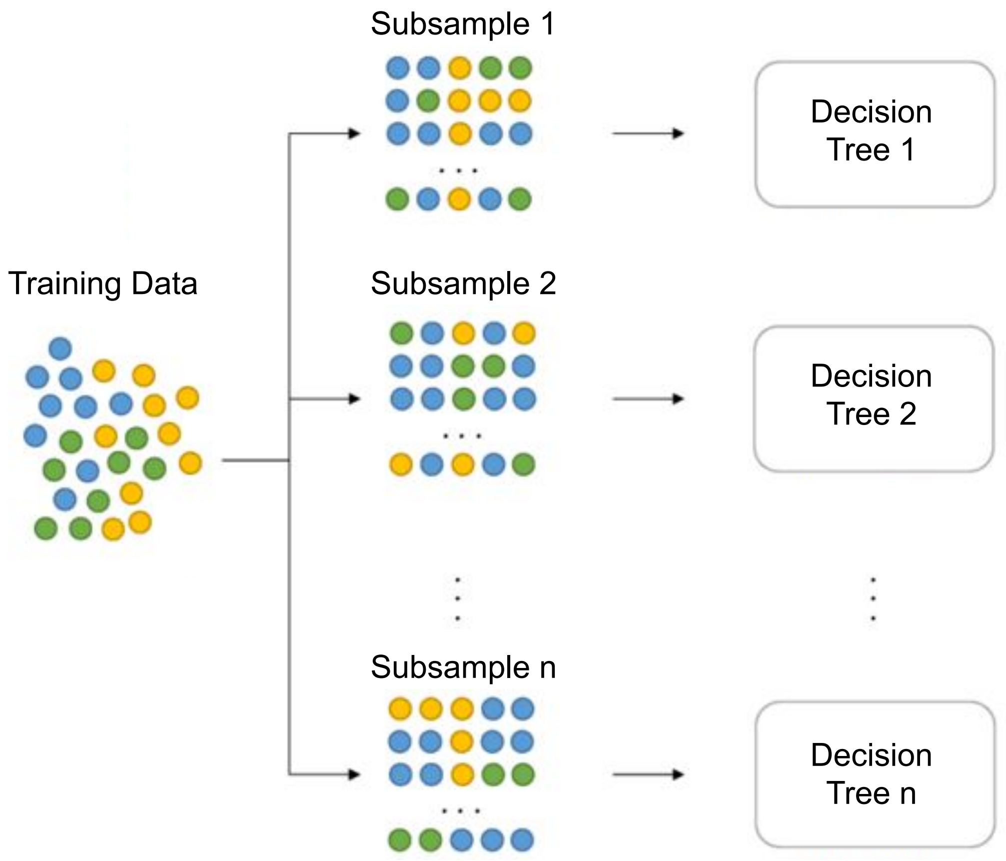
Source: Fernández-Delgado (2014)

Bagging (subsampling)



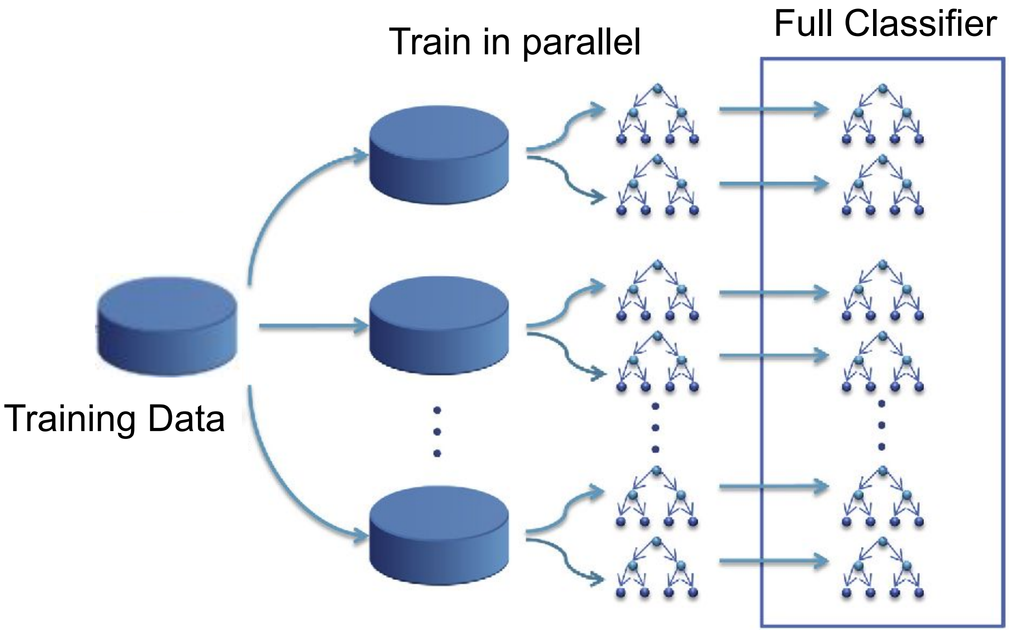
Bagging (subsampling)

- Variance reduction
- Robustness to outliers
- No need for a test data set (out-of-bag error estimates)

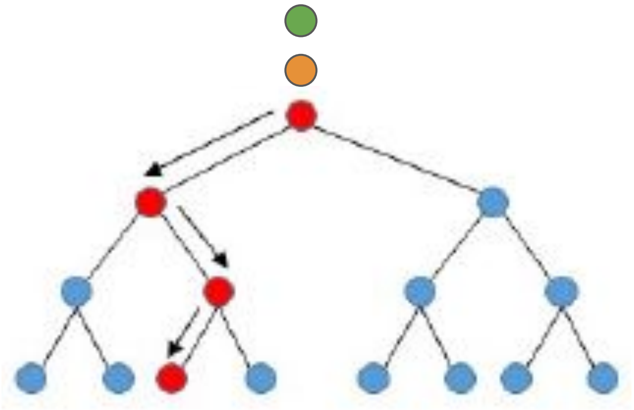


Highly Parallelizable

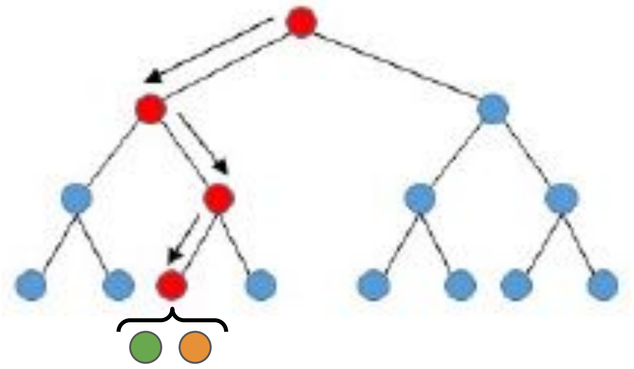
Trees are trained **independently** of one another



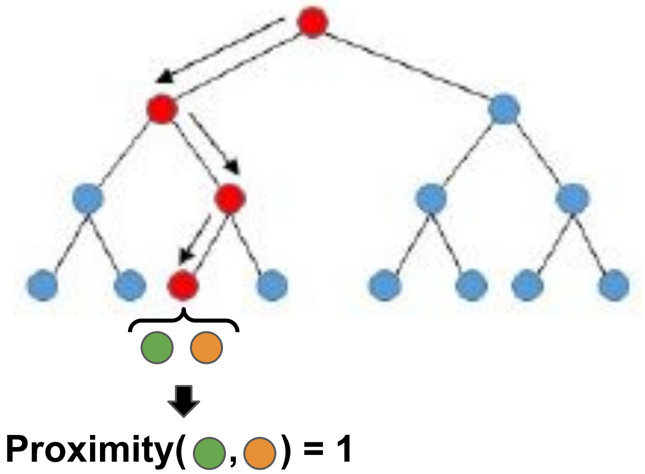
Yields a distance metric



Yields a distance metric

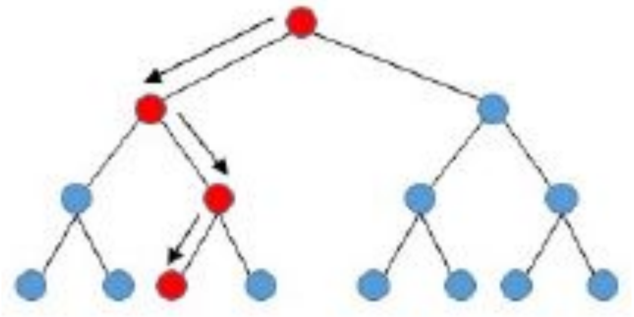


Yields a distance metric



Yields a distance metric

- Applications
  - Missing data imputation
  - Outlier detection
  - Low-dimensional representation



Conclusion

- Random forests are a well-performing algorithm
- Many possible learning modifications exist
- They are flexible in their uses

Acknowledgements

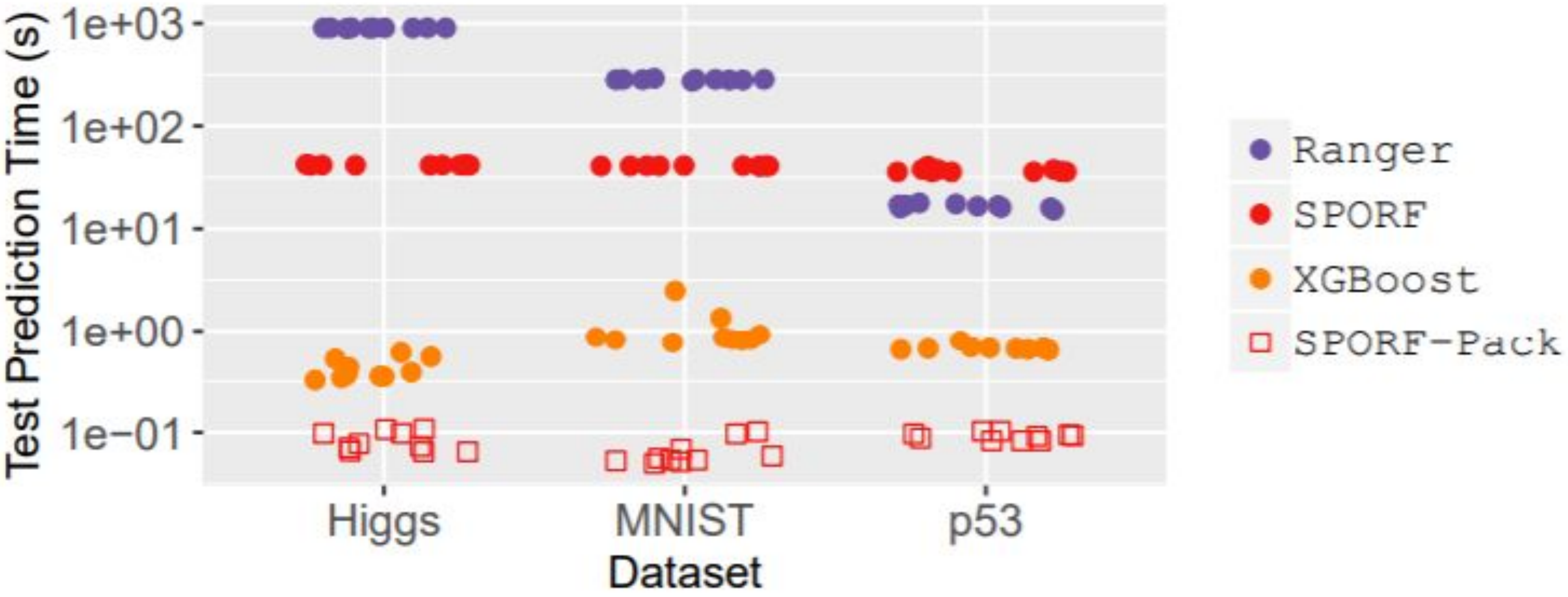
NEURODATA

References

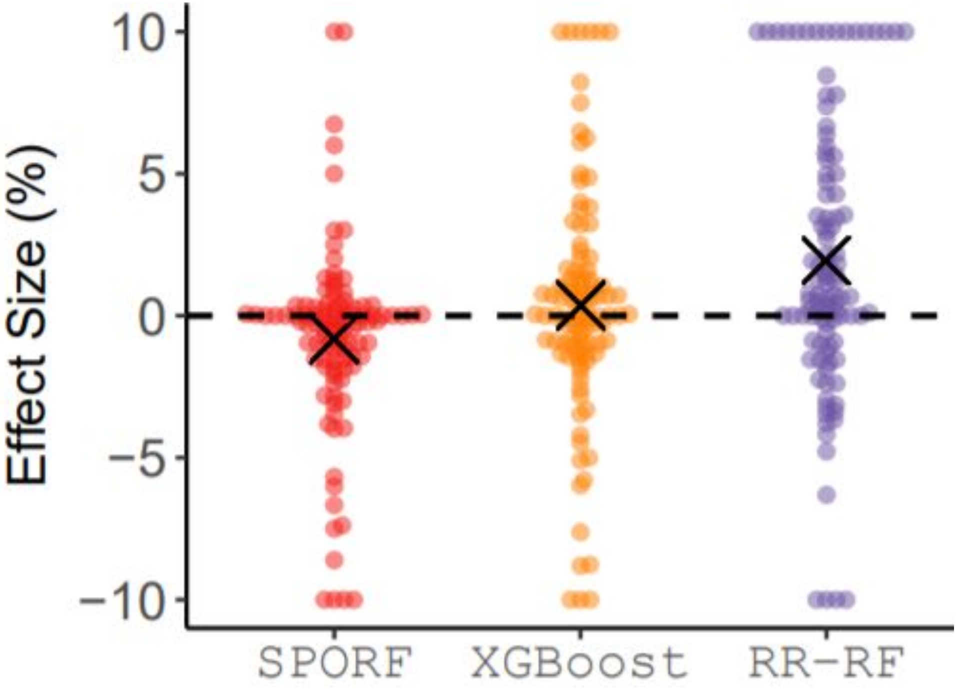
- Random Forests (Leo Breiman, Adele Cutler)
- Sparse Projection Oblique Randomer Forests (Tomita 2019)
- Manifold Forests: Closing the Gap on Neural Networks (Perry 2019)
- Do we Need Hundreds of Classifiers to Solve Real World Classification Problems? (Fernández-Delgado 2014)

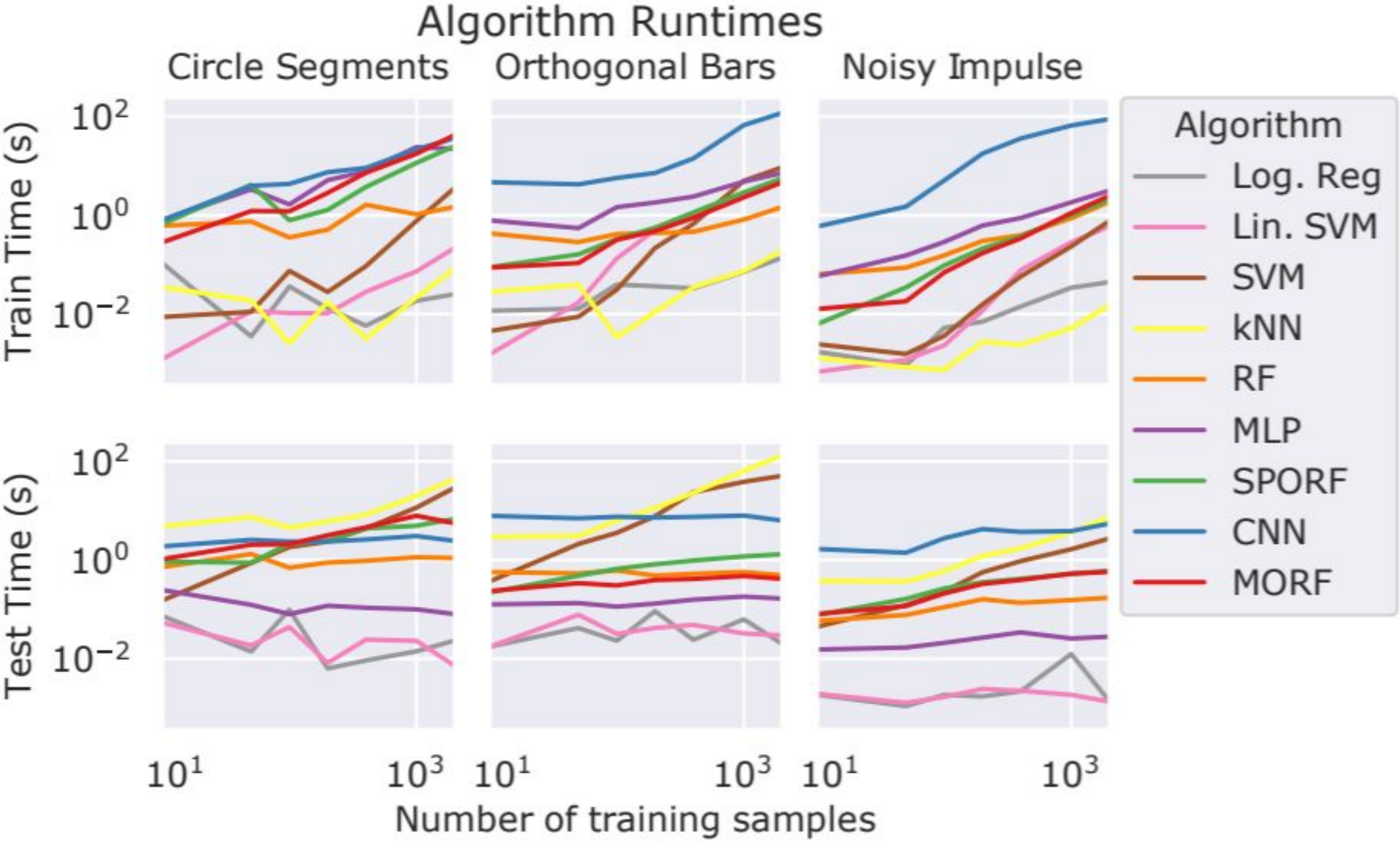
Extra Slides

Forest Packing



Comparison to other Algorithms





Algorithm Runtimes
Circle Segments
Orthogonal Bars
Noisy Impulse
Train Time (s)
Test Time (s)
10^2
10^0
10^{-2}
10^1
10^3
Number of training samples
Algorithm
Log. Reg
Lin. SVM
SVM
kNN
RF
MLP
SPORF
CNN
MORF

Single Feature Importance

- Select a feature
- Permute values in each sample at that feature
- Evaluate forest
- Evaluate difference in accuracy

Gini Importance

Change in information

- Probability of class *k* in a partition $\hat{p}_k = \frac{1}{|S|} \sum_{y_i \in S} \mathbb{I}[y_i = k]$

- Information in the partition *S*

$$I(S) = \sum_{k=1}^{K} \hat{p}_k (1 - \hat{p}_k)$$

- Maximum purity of a split

$$\theta^* = \underset{\theta}{\operatorname{argmax}} |S| I(S) - |S_\theta^L| I(S_\theta^L) - |S_\theta^R| I(S_\theta^R).$$