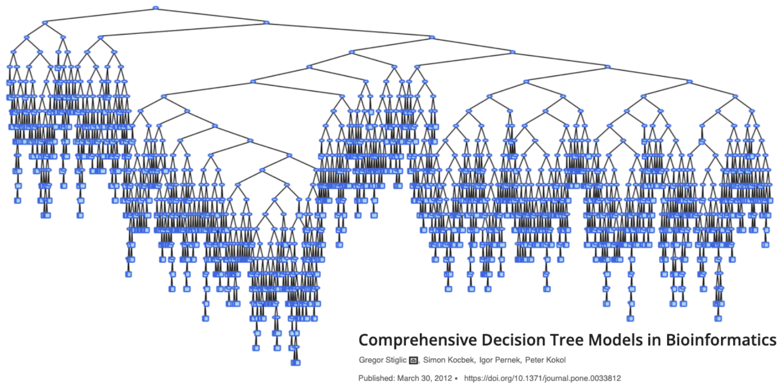
CSC 461: Machine Learning

Fall 2024

Bagging

Prof. Marco Alvarez, Computer Science
University of Rhode Island

Decision tree models

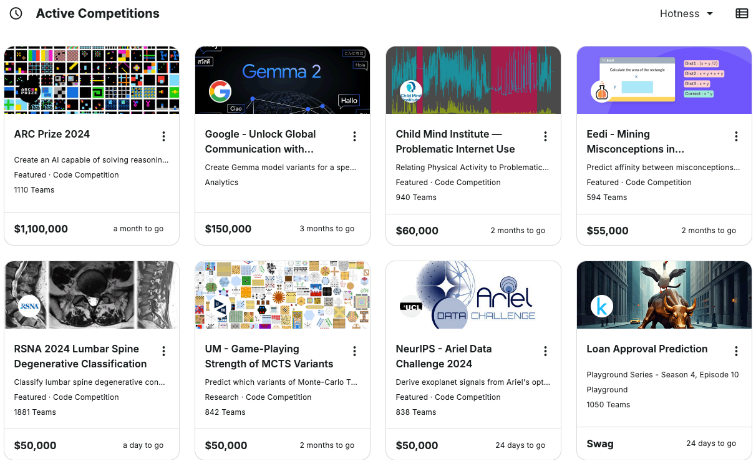


Complicated decision boundaries → Overfitting

Ensemble methods

- Decision tree challenges
  - overfitting: capturing noise in the training data
  - instability: small changes in data can lead to significantly different tree structures
- Ensembles
  - combining multiple models to improve predictive performance
  - key idea: harness the "wisdom of the crowd" in machine learning
- Main types
  - bagging: train models independently (in parallel) on different subsets of the data
  - boosting: train weak models sequentially, focusing on instances that were misclassified by previous models
  - stacking: train multiple base models and then use a meta-model to combine their predictions

Kaggle competitions

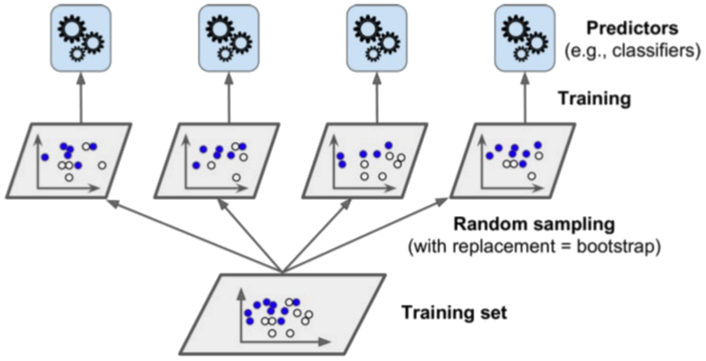


Ensemble methods have been successfully used in several Kaggle competitions

Bagging

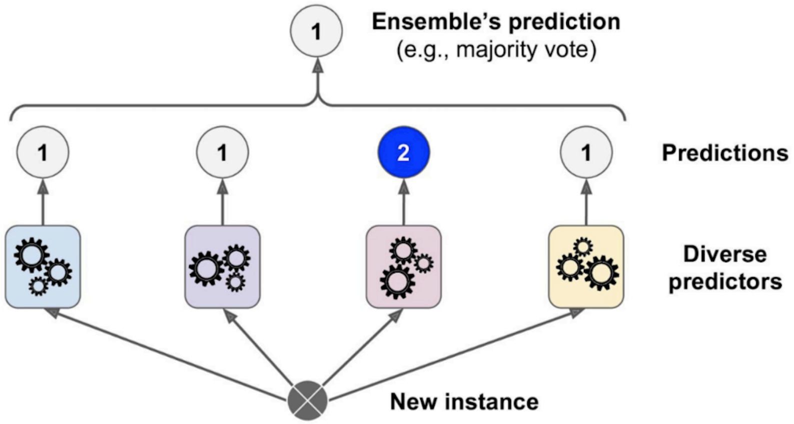
- Bagging predictors
  - proposed by Leo Bremen in 1996
  - train multiple models on random subsets of the data
    - aims to reduce variance and overfitting
  - three basic steps:
    - bootstrapping: sampling technique to generates different subsets of the training data
    - parallel training: bootstrap samples are trained independently using weak or base learners
    - aggregation: depending on the task (that is, regression or classification), an average or a majority of the predictions are taken to compute a more accurate estimate
- Bootstrapping
  - given a dataset $\mathscr{D}$ with n examples
  - generate m datasets $\mathscr{D}_1, \mathscr{D}_2, \ldots \mathscr{D}_m$
    - each $\mathscr{D}_i$ containing n instances sampled **with replacement** from $\mathscr{D}$
    - some elements will appear multiple times in $\mathscr{D}_i$, some elements may not appear at all

Bootstrapping



https://www.bpesquet.fr/mlhandbook/algorithms/decision_trees_and_random_forests.html

Inference



https://www.bpesquet.fr/mlhandbook/algorithms/decision_trees_and_random_forests.html

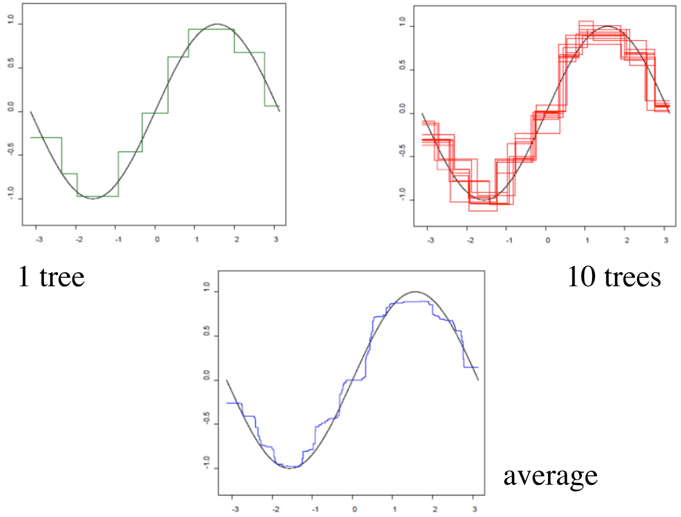
Exercise

- Write a script that generates a random sequence of N elements and creates M bootstrap samples from that sequence
  - can use `random.randint` and `random.choices`

Random forests

- Combining tree predictors
  - introduced by Leo Bremen in 2001
  - introduces random feature selection
- Algorithm
  - create m bootstrap samples from the original data
  - for each sample, grow a decision tree
    - at each node, randomly select k features and choose the best split among these k features
  - aggregate predictions
    - majority vote for classification, average for regression

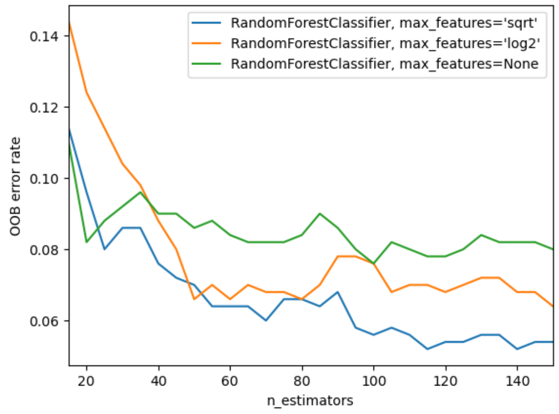
Regression example



Random forests

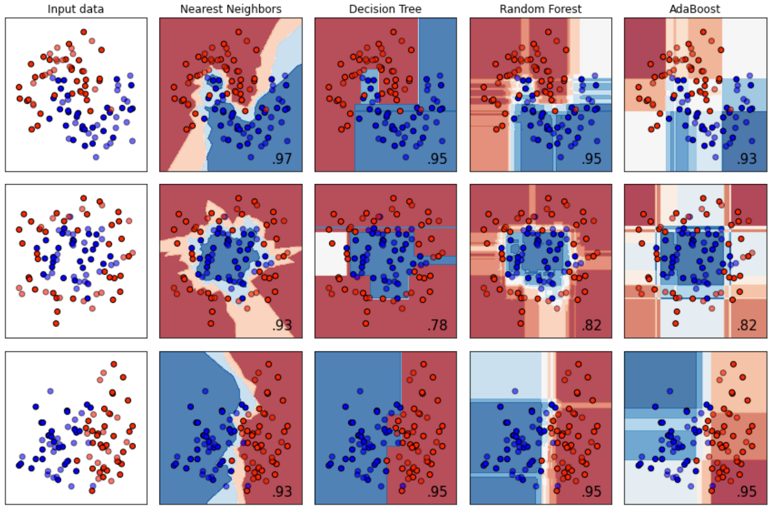
- Feature importance
  - measure of how much each feature contributes to the overall prediction accuracy
  - e.g., mean decrease in impurity, mean decrease in accuracy
- Out-of-Bag (OOB) error estimation
  - each bootstrap sample leaves out approximately 37% of instances (OOB)
  - these OOB examples can be used for built-in cross-validation
- Typical hyperparameters
  - number of trees
  - number of features to consider for best split
  - maximum tree depth
  - minimum samples per leaf

Tracking OOB



OOB error can be measured at the addition of each new tree during training. The plot allows a ML practitioner to approximate a suitable value of n_estimators

Comparing classifiers



Final remarks

- Random forests are powerful ensemble methods
  - balance between performance, robustness, and ease of use
  - robust to outliers and non-linear data
  - efficiency via parallelism
- Limitations and considerations
  - computational intensity for large datasets or many trees
  - may struggle with very high-dimensional, sparse data
  - lack of interpretability compared to single decision trees
  - extreme gradient boosting (XGBoost) often outperforms standard random forests