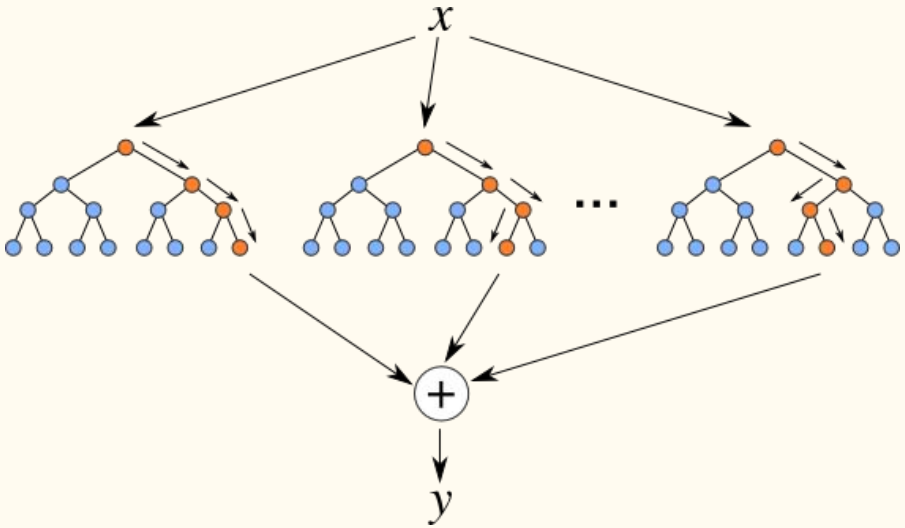
import ensemble_factory as ef

HackMIT 2019: DevTools Hack

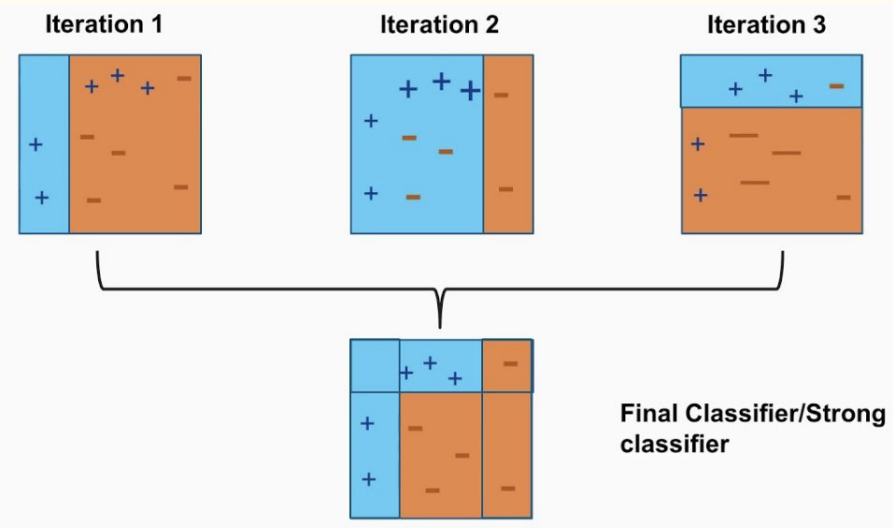
Team: Josh Gruenstein, Lior Hirschfield, Aaditya Singh, Albert Yue

Ensemble Learning

- Combining multiple weak learners to produce a strong learner
- Classical examples:



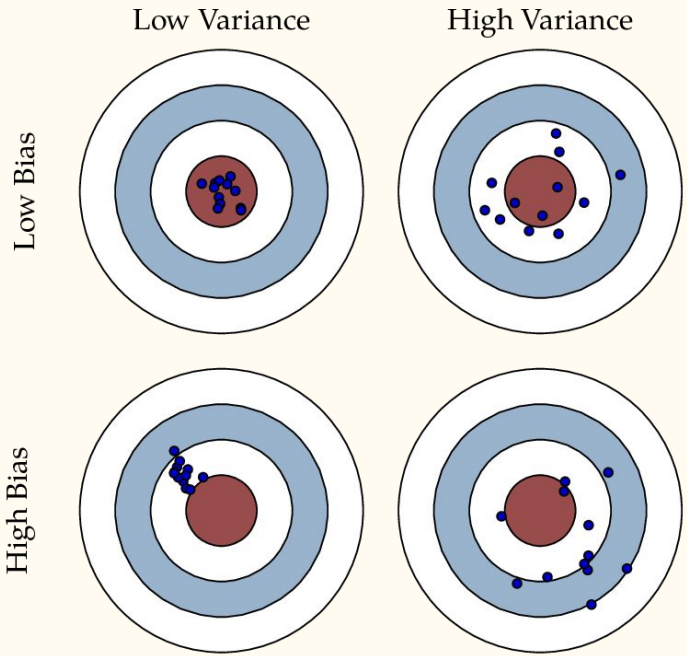
Random Forests (Bagging)



AdaBoost (Boosting)

Why ensembling?

- Many ways to skin a cat in Data Science
- Always trying to optimize bias-variance tradeoff
- Top-5 finalists for basically all recent Kaggle competitions use two or three methods, and ensemble them



Heterogeneous Compositional Ensembles

- Current ensemble methods are primarily homogeneous (composed of the same base model), only heterogenous at the top level if at all
- Libraries that support ensemble learning are extremely limited in this sense and do not allow ensembles of ensembles

- Goal: Build a library that can compositionally create and learn ensembles of different learners (including ensembles of ensembles, etc.)

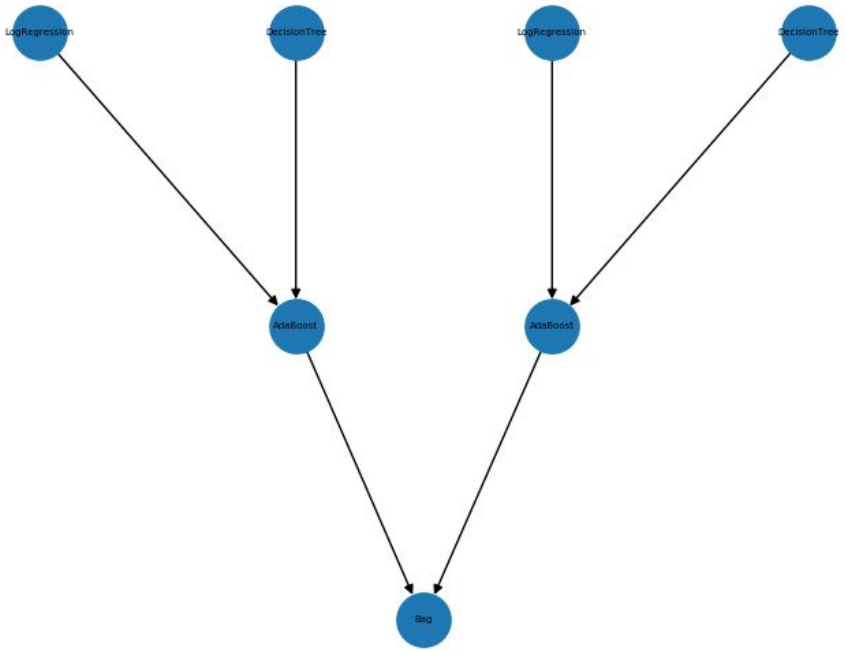
Ensemble Factory

- A library for compositional ensemble model construction and learning
- Supports 3 ensembling operators:
  - Bagging: Average submodels trained on subsets of the data
  - Gradient Boosting: Train submodels on residualized outputs
  - AdaBoost: Reweight misclassified training samples
  - Stacking: Combine submodel outputs (as defined by a stack model) to achieve desired output
- Supports standard base models for classification and regression:
  - SVMs
  - Logistic Regression
  - Decision Trees
  - Linear Regression
  - Multi-Layer Perceptrons
  - and more!
- Offers graph visualization tools to better understand model performance

```
import ensemble_factor as ef

classifier = ef.logistic_regression()
boosted_classifier = ef.adaboost(classifier, ef.decision_tree_classifier())

bagged_boosters = ef.bag(0.7, boosted_classifier, boosted_classifier.copy())
```



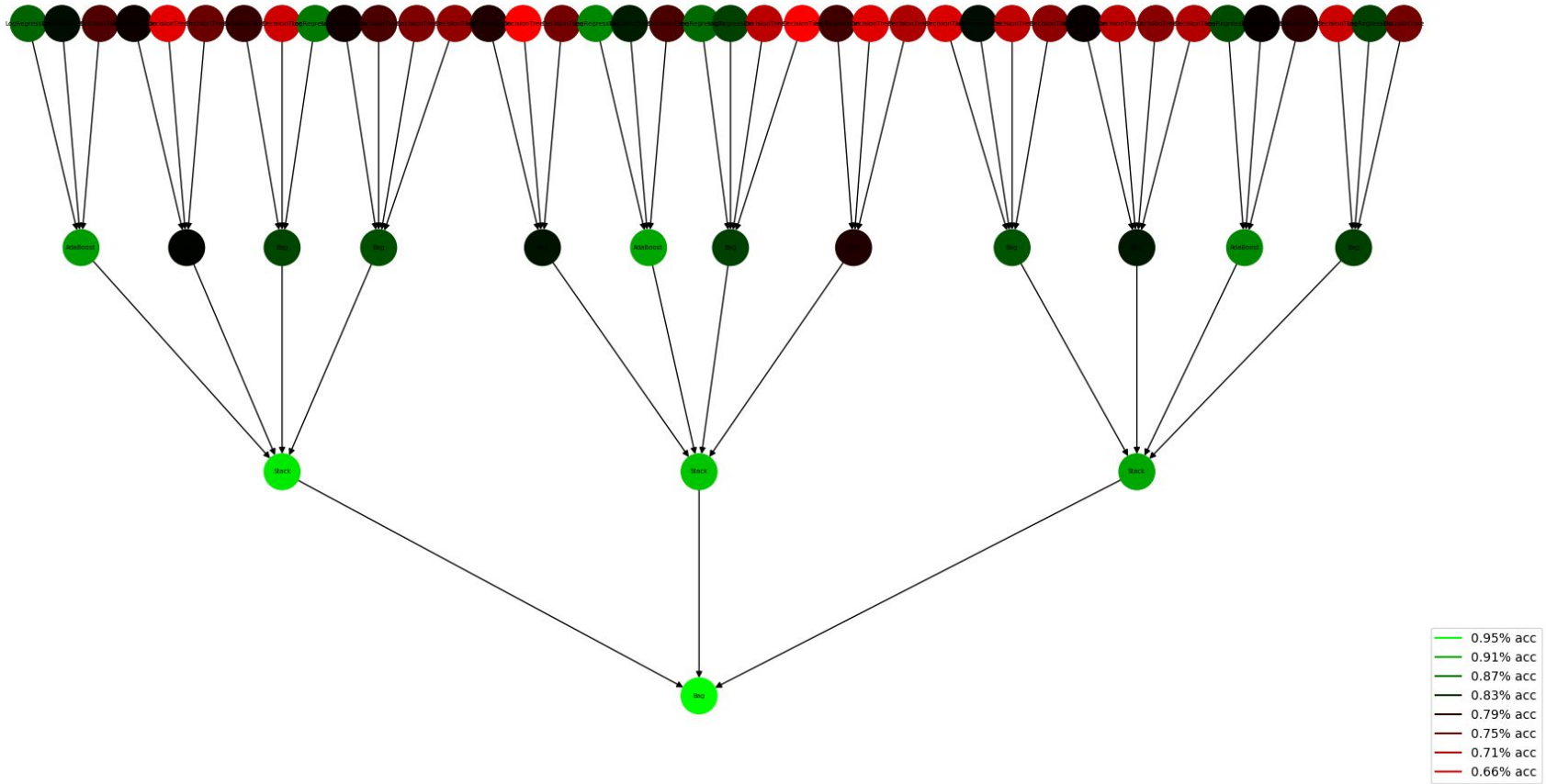
AutoML

- Dynamically learn architectures for ensemble trees that best fit the data
- Motivated from Google's Neural Architecture Search (NAS)
- Compositional structure and library's ease of use allows for easy coding of complex evolutionary learners

Performance Benchmarks

- Consistently outperform random forests (and other common baselines) on MNIST with much smaller ensembles by >= 2% test accuracy
- Succeeds in assembling, discovering, and training novel ensemble architectures

- Next steps
  - Compositional ensemble learning of neural networks and regression problems (supported but not tested due to training time limitations)
  - Experimenting with more complex stacking models (supported but not tested due to training time limitations)


0.95% acc
0.91% acc
0.87% acc
0.83% acc
0.79% acc
0.75% acc
0.71% acc
0.66% acc

Conclusions

- Performance of learned compositional ensembles indicates a need for research in this area

- Our library is an enabler to such research in compositional ensemble learning, just as tensorflow is a huge enabler for neural network training and neural architecture search

- Simple API offers an easy way to get started