Random Forests

Ensembles of Trees!

What is a decision tree?

- A decision tree is a classifier that uses binary rules in sequence to make a classification
- Powerful for the same reasons as binary search: Can cut solution space in half once per decision
- “Stumps” refer to single-rule trees

Why use decision trees?

- Works on continuous and binary classification tasks
- Invariant under scaling and various other transformations
- Robust to irrelevant features
- Very easy to inspect and reason about decision tree models
- Low bias and high variance

How do I build a decision tree?

- Most popular/widely available is C4.5, which is seen as an improvement on ID3
- Uses *information gain* as a metric for splitting the search space by a given attribute.
- Basic algorithm is:
  - Check for a base case: all examples are of one class, or there's an unencountered class in the example set, or there are no attributes with information gain.
  - Choose the attribute with the highest information gain in the search space, split the tree on that node
  - Recur over sublists formed by the new splits in the tree.

Ensemble methods

- Bootstrap aggregating, or “Bagging”: train many models on random subsets of the data and average their predictions (continuous values) or using the majority vote (binary classification).
- Feature Bagging: Use only a random subset of the features in training
- Boosting: Weigh different classifiers by how well they do on examples the other classifiers do poorly on.

Random Forests

- Random forests: Use bagging and feature bagging. Both seem to always reduce variance and increase accuracy
- Boosting, notably, does extremely well on some data sets to reduce variance, but can actually make variance larger in data sets with a lot of noise. Might be a good solution to some problems but not as commonly used.

Theory/Further reading

- Information gain, described by the Kullback-Leibler Divergence formula, is seemingly the mathematical basis of these trees, I'd like to read more about it.
- Random forests seem to do better than boosted trees, but there's a recent paper that suggests that boosting random forests might help efficiency (Comparable performance with less trees)

Applications

Trees in general and random forests are extremely popular, so applications are varied. Common uses are in

- Natural language processing
- Map classification
- Facial recognition

Implementations

- R package “randomForest” http://cran.r-project.org/web/packages/randomForest/randomForest.pdf
- scikit-learn “RandomForestClassifier” http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Conclusions

- Decision trees are a good way to have a model that both works well on a lot of kinds of problems but is nonetheless a lot more transparent than, say, a neural network.
- Ensemble methods are really generalizable and cool, boosting and bagging both seem to be broadly used on lots of kinds of classifiers, usually improving them

References

- Wikipedia pages:
  - Random Forests
  - Decision Tree Learning
  - Entropy (information theory)
  - Kullback-Leibler Divergence
  - Boosting (machine learning)
  - Bootstrap aggregating

References

- Links to papers
  - http://link.springer.com/article/10.1023/A:1007607513941
  - http://www.security.iitk.ac.in/contents/publications/mtech/VidyutGhosal.pdf
  - http://qwone.com/~jason/writing/stumps.pdf
  - http://jessica2.msri.org/attachments/10778/10778-boost.pdf
  - http://www.vision.cs.chubu.ac.jp/MPRG/C_group/C058_mishina2014.pdf

Credit where credit's due

This presentation thrown together by Alex Kelly for the PDX Data Science meetup

Thanks H Forrest Alexander for making the outline and initial slides and finding most of these papers.