Decisions, Trees, Forests (Part 2, Forests)

Daniel Lawson University of Bristol

Lecture 06.1.2 (v1.0.1)

Signposting

- Lecture 6.1 is split into two parts:
  - 6.1.1 Trees
  - 6.1.2 Forests
- This is 6.1.2
- The Workshop 6.2 is intimately linked.

Random Forest

- A **random forest** is a set of **decision trees** that are combined together to perform classification.
- For each of T trees, the following steps are run:
  - Choose which variables to include:
    - Choose m_f **random features**. A typical choice is $m_f = \sqrt{m}$ where m is the number of features.
    - Analogous to **bagging for features** (downsampling without replacement in this case)
  - Learn a tree classifier independently via some standard Tree learning algorithm:
  - For each feature, for each leaf, find the split that maximises a score function, e.g.:
    - CART (Classification and Regression Trees) uses Gini Index as metric.
    - ID3 (Iterative Dichotomiser 3) uses Entropy function and Information gain as metrics.
  - Choose the feature that maximises the score

Random Forest outputs

- The Random Forest **combines decision trees** into a classification by:
  - Weighting each tree according to its performance
  - Report the weighted vote
- It is also possible to extract feature importance:
  - The **importance** of features is measured by how much each decreases the score, averaged over all trees
  - Features that are never used will get a score of 0
  - Features that are important in every tree in which they appear will get a high score
  - Features that are correlated will often split their importance

Random Forest vs boosted decision tree

- Gradient Boosting Machine (GBM) is the go-to boosted decision tree
- GBM and RF differ in the way the trees are built, the order, and the way the results are combined
- RF can be trivially paralellized
- GBMs seem to outperform RFs under competition conditions, but do worse when their parameters are untuned[1]

[1] http://fastml.com/what-is-better-gradient-boosted-trees-or-random-forest/

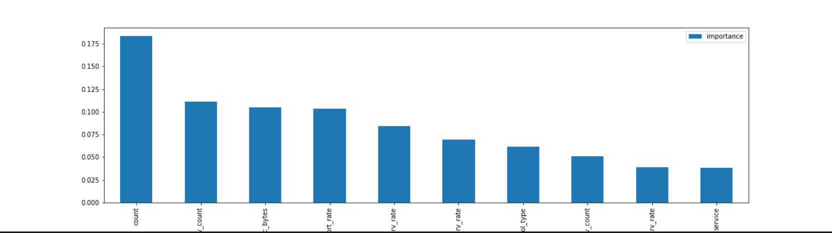
Random Forest algorithm

```
from sklearn.ensemble import RandomForestClassifier
clf= RandomForestClassifier(n_jobs=-1,
    random_state=3, n_estimators=102)
trained_model= clf.fit(X_train, y_train)
clf_score=trained_model.score(X_train, y_train)
y_pred = clf.predict(X_test)
```

Random Forest Feature Importance

```
feature_importances = pd.DataFrame(clf.feature_importances_,
    index = X_train.columns,
    columns=['importance']).sort_values('importance',
    ascending=False)

feature_importances.nlargest(10,
    columns=['importance']).plot(kind='bar',figsize=(18, 5))
```

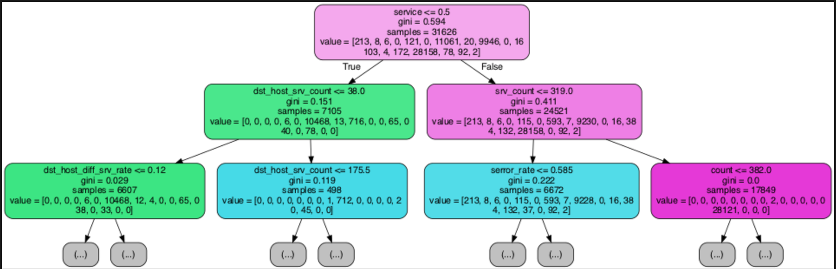
Random Forest Feature Importance



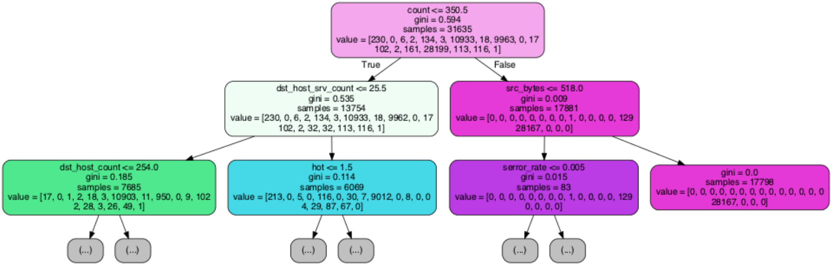
Random Forest Extract single trees

```
estimator5 = clf.estimators_[5]

dot_data = StringIO()
export_graphviz(estimator5, out_file=dot_data,max_depth=2,
                feature_names=X_train.columns.values,
                filled=True, rounded=True)

graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

Random Forest Feature Trees



Random Forest Feature Trees



Final thoughts

- Random Forests are typically better than bagged decision trees
- There are theoretical examples where either dominates
- Boosting changes things but isn't a magic bullet
- Usually worth being open minded; the differences could be seen as tuning parameters of a more general algorithm

Reflection

- Why are Random Forests considered to be important?
- What variations on Random Forests can you think of? Under what circumstances would you expect them to work?
- What does a Random Forest decision boundary look like? How dependent are they on the specific choice of features?
- By the end of the course, you should:
  - Know what a decision tree is, and be able to implement the basic algorithm
  - Know what a Random Forest is, and understand its advantages and disadvantages
  - Be able to use pre-existing implementations
  - Be able to interpret their output appropriately

Signposting

- In the practical we'll implement these models in R and Python; compare implementations, and to previous results.
- Next semester we'll start with the "other" LDA (Latent Dirichlet Allocation), Topic Modelling, and Modelling Documents.
- **References**:
  - Chapter 15 of The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Friedman, Hastie and Tibshirani).
  - Implement a Random Forest From Scratch in Python
  - A Gentle Introduction to Random Forests at CitizenNet
  - DataDive on Selecting good features
  - Cosma Shalizi on Regression Trees
  - Gilles Louppe PhD Thesis: Understanding Random Forests