Unstructured Data Analysis

Lecture 11: Wrap-up predictive model evaluation, classical classifiers

George Chen
Belated Questionnaire Results

• Lots of comments regarding the coding aspect of the course
  • Some want more explanation of the code
  • Some want code in lecture to be much closer to what you’re expected to write for code in HW

• Some comments on the course not being technical enough (e.g., not enough math, not enough details on models)

• Try using Kahoot (I’ve never heard of this before)
Questionnaire: HW1 time

Self-reported number of hours to complete HW1
Questionnaire: HW2 time

Self-reported number of hours to complete HW2
HW3 is out! It’s short.
Quiz Results

94-775 Quiz Score Histogram

Mean: 70.8, std dev: 18.5, median: 72
1 student did get a perfect score
Quiz Regrade

• First look over solutions very carefully

• If you think there is a genuine grading error, email me with:
  • An explanation of what was incorrectly graded
  • How many points are at stake

• We will regrade your exam and your score can go up, stay the same, or go down

• Due this Thursday 11:59pm Pittsburgh time
Other Announcements

- Projects: due next Friday (gasp)

  - This semester, because of the random CMU break days, the second half of my usual schedule got shifted…

  - We are *not* expecting as thorough/polished projects as in previous semesters, so don’t panic if your analysis ends up looking a bit “preliminary”

- Thursday lecture is remote

- I will hold OH specifically for 94-775 students 3pm-4pm after class on Thursday (to discuss course performance/projects/HW3/etc) — use my usual OH Zoom link
Wrap-up coverage of how to evaluate whether a predictive model is good & classical classifiers

In many datasets (especially small, structured ones), neural nets & deep learning could work poorly... in such cases, often decision-tree-based methods can work well

*random forests, gradient boosting* (e.g., XGBoost)

Start coverage of neural nets & deep learning
Decision Trees & Forests
Example Made-Up Data

Red: diabetic
Blue: not diabetic
Example Decision Tree

- **Age > 40?**
  - no
  - Age > 30?
    - no
    - **not diabetic**
    - yes
    - **diabetic**
  - yes
  - **weight > 200?**
    - no
    - **not diabetic**
    - yes
    - **diabetic**
Learning a Decision Tree

- Many ways: general approach actually looks a lot like divisive clustering but accounts for label information

- I’ll show one way (that nobody actually uses in practice) but it’s easy to explain
Learning a Decision Tree

1. Pick a random feature (either age or weight)

2. Find threshold for which red and blue are as “separate as possible” (on one side, mostly red; on other side, mostly blue)
Learning a Decision Tree

Within each side, recurse until a termination criterion is reached!

Example termination criteria: \( \geq 90\% \) points within region has same label, number of points within region is <5

Note: label within each region is majority vote
For a new person with feature vector (age, weight), easy to predict!
Decision Forest for Classification

- Typically, a decision tree is learned with randomness (e.g., we randomly chose which feature to threshold)
  → by re-running the same learning procedure, we can get different decision trees that make different predictions!
- For a more stable prediction, use many decision trees

Learn each tree separately using same training data

New test data point

Tree 1
  diabetic

Tree 2
  not diabetic

Tree 3
  diabetic

... 

Tree T
  diabetic

Final prediction: majority vote of the different trees’ predictions
**Decision Forest for Classification**

Randomly sample \( n \) points with replacement from \( n \) training data points.

New test data point.

- **Tree 1**: diabetic
- **Tree 2**: not diabetic
- **Tree 3**: diabetic
- \( \ldots \)
- **Tree \( T \)**: diabetic

**Question:** What happens if all the trees are the same?

*Adding randomness can make trees more different!*

- **Random Forest**: randomize training data used for each tree, randomly choose a few features to try to split on (and among these features, choose the best one to split on.)
Back to the demo
Another Way to Benchmark

• In the demo: we just saw that we can compare test set prediction accuracy across different algorithms and also look at confusion matrices

• For binary classification, we can do a more detailed analysis
Binary Classification: ROC Curves

For simplicity, think of the random forest for now

New test data point

Tree 1
- positive

Tree 2
- negative

Tree 3
- positive

…

Tree $T$
- positive

Final prediction: majority vote of the different trees’ predictions

$\geq 50\%$ of trees need to say positive for final prediction to be positive

We can vary this 50% threshold!
Binary Classification: ROC Curves

Error rates are computed on test data
A classifier with the green curve is better than the one with the blue curve.
It's possible that algorithms are better in different regimes.
Binary Classification: ROC Curves

The ideal curve (typically impossible to achieve)

Area under the curve (AUC) is a popular metric for comparing algorithms (higher is better)
Binary Classification: ROC Curves

What we just saw:

- For a classifier that we can set the threshold probability to different values, we can plot an ROC curve.
- True positive rate (TPR) and false positive rate (FPR) are evaluated on test data.

Other variants are possible:

- Plot precision vs recall instead of TPR vs FPR.
- Can actually plot ROC/precision-recall curves sweeping over hyperparameters aside from threshold probability!
- For ROC/precision-recall, rather than evaluating on test data, can evaluate on validation data during training to help choose hyperparameters.
Binary Classification: ROC Curves

Different hyperparameter settings (need not be only for threshold probability)

Can also be computed on validation data instead of test data!