Predicting YouTube Comedy Slam Winners

STAT 222 Class 2013

Department of Statistics
University of California, Berkeley

7 May 2013

YouTube Comedy Slam Data

- from UCI ML Repository; donated by Google employee
- a trial is an ordered pair of video IDs (v_i, v_j)
- data: ordered pair + "left" or "right" found funnier by viewer
- order of the pair in each trial was random
- repository has training data and test data
- YouTube has metadata about videos
- no data about viewers

Goal

Predict which video in a pair will be funnier, from metadata

(Is it OK to use the video IDs in the prediction?)

Tools

- Github
- IPython notebook
- numpy, scipy, matplotlib, nltk, scikit-learn

Descriptive statistics of training data

- 912,969 records
- 18,474 distinct video IDs
- 267,211 distinct video ID pairs
- 359,874 distinct ordered pairs of video IDs
- right video won 51.77% of the time. *P*-value: nil
- judgments often discordant:
  e.g., '-iuk0PbfaHY wDx28Y2RcCI' left and right each "funnier" 119 times
- accuracy of ideal classifier for training data: 73.449% (includes videos with no comments)
- for individual videos, directed graph of "funnier than"
- can summarize that graph by PageRank

PageRank

- Assign each video ID (node) in the graph a numerical value between 0 to 1, known as its *PageRank*.

At $t = 0$, $PR(p_i; 0) = \frac{1}{N}$, N is the total number of nodes.
$PR(p_i; t+1) = \frac{1-d}{N} + d \sum_{p_j \in M(p_i)} \frac{PR(p_j; t)}{L(p_j)}$
Algorithm ends when $\lvert PR(t+1) - PR(t) \rvert \leq \epsilon$
d is a damping factor, default 0.85 in scikit-learn.

Acquiring metadata

- queried YouTube with video IDs using Google APIs w/i Python
- speedbumps: had to throttle the requests
- stored "snapshot" data as pickled Python dict

Prediction using metadata: Feature selection

- used mutual information to screen potential features
- quantitative metadata (e.g., #views, rating, #raters) unhelpful
- comments more promising

Issues with comments

- Data Encoding: encoded the words to ASCII format and omit the garbled words
- Inconsistent Spellings
  - Correct Spelling: tried Norvig's spelling corrector, ported to AWS. (e.g.: "speling" → "spelling")
    drawback: not accurate on this corpus
    (e.g.: "youtube" → "couture", "lol" → "ll", "haha" → "hata")
  - Text-speak: used RegExp to standardize words

Patterns	As
lol, lolll, llol, lolloloII	lol
ha, hahahh, ahaha, jhajha	ha

  - Stemming: used Porter Stemmer from NLTK package
- Addressing Emoticons: used RegExps to replace happy faces (e.g., ":-]") with "happyface" before stripping other punctuation

Bags of Bags of Words

- ordered pair of videos (v_1, v_2) reduced to two "bags of bags of words" each comment is a bag, each video has a bag of bags
- features derived from bags of bags:
  - presence of a word in any comment
  - frequency of comments with a given word
  - relative frequency of comments with a given word
  - logOdds of frequencies and relative frequencies
  - etc.
- another derived feature: PageRank predicted by linear regression using (among other things) arctan(difference in LOL counts)

Models

Logistic regression with Log Bayes Factor

Binary Output (Left v.s. Right) as response variable
log bayes factor of the two ordered ids and a constant term as features

Logistic regression Wenchang

Explain

CART

Explain

Page Rank

Explain

Features for Logistic Regression 1

Log Bayes Factors

1. determine whether video v_i won more than it lost, or vice versa.
   if v_i won more than lost, it's a "winner"
   if v_i lost more than won, it's a"loser"

2. for each word w, find:
   $\mathbb{P}_1(w) =$ percentage of winner comments that contain w
   $\mathbb{P}_0(w) =$ percentage of loser comments that contain w
   (pad to avoid zeros)
   (Only used comments of "significant" winners/losers)

3. derive new feature:
   $logOdds(v) = \sum_w m_w \log(\frac{\mathbb{P}_1(w)}{\mathbb{P}_0(w)}) + (n - m_w)\log(\frac{1-\mathbb{P}_1(w)}{1-\mathbb{P}_0(w)})$

   n: number of comments on video v

   m_w: number of comments on video v containing w.

Features for Logistic Regression 2

Wenchang please fill in this slide.

Performance on the training set

Common criterion to measure performances

score$= \frac{NumberofCorrectPrediction}{NumberofPredictablePairs}$

- Logistic with bayes odds: 53.3%

The Test data

- 225,593 records
- 75,447 distinct ordered pairs of videos

data?

- Accuracy of the ideal classifier: 0.6946 (excluding pairs with no comments)
- right video won: 51.62% of the time

Performance on test data

- Bayes classifier from training data applied to test data
- logistic regression
- CART

Conclusions

- Quantitative metadata: not informative
- Comments:
  - Logistic Regression with Bayes Classifier
  - Logistic Regression with LogOdds
  - CART
  - PageRank
- Results
- Goal achieved?

Improvements

- Make better use of GitHub
- Having data that are of higher quality
- Incorporate personal preference in the dataset

Acknowledgement

Thanks to Philip, David and Aaron
for the valuable help!