L90 Practical, Part I

Overview of Natural Language Processing

Simone Teufel

2016/10/28

The Task: Sentiment Classification

- Write Code that automatically classifies a given review of a movie downloaded from IMDB into Negative or Positive
- Obviously without looking at the Star Rating!
- Data: 1000 Pos + 1000 Neg example texts
- Use Virtual Machines (VM) on Mphil machines; submit code running on these.

Procedure/Timeline

- Today: How to write the baseline system(s)
- In Two Weeks (11/11):
  - How to write baseline report
  - Check on your baseline system
- In Four Weeks (25/11):
  - How to write an extension system (of your choice)
  - You submit draft baseline report (voluntary, non-assessed)
- 30/11: Receive feedback on your draft report
- Beginning of Lent term (around mid-January 2017): Submit your full report (baseline system + extension)

Symbolic System – First Baseline

- Use existing sentiment lexicon
- “Tokenize” the input texts
- Count and sum number of matches between text and lexicon (pos–neg)
- Variation of system – Use Magnitude to sum matches
- But: Does using magnitude produce better results?
- “Better” in science means: “statistically better”

Tokenisation

- Split on whitespace
- Split off punctuation (difficult cases?)
- Do something sensible with mid-word alphanumerics
  - eight-year-old-child
  - love/height relationship
  - B456F7-3
- Treat contractions (negation, modal verb) in a sensible way
  - What is sensible?
  - All strings representing the negation should look the same (why?)
  - All strings representing the base part of the word should look like their non-negated counterparts
  - How many cases do you have to treat? Listable?
  - 80–20 rule
- Genitives: Treat both plural and singular cases
- Don't spend more than 1 - 1.5 hours on this subtask
- Try to explain the **Principles** of this treatment in as few words as possible for the report.

Statistical Significance Testing

- Null Hypothesis: two result sets come from the same distribution
  - System 1 is (really) equally good as System 2.
- First, choose a significance level (p), e.g., $p = 0.01$.
- We then try to reject the null hypothesis with at least probability $1 - p$ (99% in this case)
- Rejecting the null hypothesis means showing that the observed result is very unlikely to have occurred by chance.
- If we manage to do so, we can report a statistically significant difference at $p = 0.01$.
- Now – a particularly nice and simple test (non-parametric and paired): the sign test

Sign Test

- The sign test uses a binary event model.
- Here, events correspond to documents (2000 events in our case)
- Events have binary outcomes:
  - Positive: System 1 beats System 2 on this document.
  - Negative: System 2 beats System 1 on this document.
  - (System 1 and System 2 could also have the identical result on this document – let's forget this for now.)
- Call the probability of a positive outcome q (here $q = 0.5$)
- Binary distribution allows us to calculate the probability that, say, at least 1247 out of 2000 such binary events are positive.

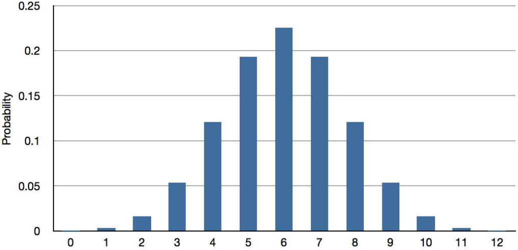
Binomial Distribution $B(N, q)$

- Probability of $X = x$ positive events out of N:

$$P_q(X = x|N) = \binom{x}{N} q^x (1-q)^{N-x}$$

- At least x positive events:

$$P_q(X \geq x|N) = 1 - \sum_{i \geq x} \binom{i}{N} q^i (1-q)^{N-i}$$



Binary Event Model and Statistical Tests

- (This is a breather slide; repetition of facts)
- If the probability of observing our events under Null Hypothesis is too small, we can safely reject the Null hypothesis.
- This means we have proven that System 1 and System 2 must be different.
- Hurray.
- And the calculated $P(X \geq x)$ directly gives us the significance level p we are after.
- Well, almost...

Two-Tailed vs. One-Tailed Tests

- So far we received $P(X \geq x)$ as answer to the question:
  - What is the probability of getting at least 10 positive out of 12 trials? [One-tailed test]
- But maybe the question should be
  - What is the probability of getting a result that is as extreme as the one I observed (or even more extreme)? [Two-tailed test]
- The answer to this question is $2P(X \geq x)$ (because $B(N, 0.5)$ is symmetric).
- Why is the second question better?
- Because the first question makes the assumption that there are always going to be more positive than negative events. This assumption does not hold in the general case.
- Therefore – use the two-tailed test.

Don’t Ignore Ties

- In NLP, we often have situations where there is a large number of ties.
- These don’t occur in biological situations such as growth, where measurements are naturally continuous.
- Standard textbooks often recommend to ignore ties.
- They don’t know about NLP, so we have to ignore them.
- When you count your x (number of positive outcomes), account for each tie by adding 0.5 events to the positive and 0.5 events to the negative side (round up at the end).
- (What does this do?)
- Keep using the two-tailed test.

Binomial and Normal Distributions

According to the Central Limit Theorem (CLT), we can use the normal distribution for large N to estimate our significance level p. If X is a random variable with distribution $B(N, q)$, then for sufficiently large N, the following random variable has a standard normal distribution N(0,1):

$$Z = \frac{X - \mu}{\sigma}$$

where

$\mu = Nq$ and $\sigma^2 = Nq(1-q)$

- What does “large enough” mean?
- $Nq \geq 5$ and $N(1-q) \geq 5$

Naive Bayes

- Naive Bayes Classifier:

$$\hat{c} = \underset{c \in C}{\operatorname{argmax}} P(c|\vec{f}) = \underset{c \in C}{\operatorname{argmax}} P(c) \prod_{i=1}^{n} P(f_i|c)$$

Feature vector $\vec{f}$; most probable class $\hat{c}$; C set of classes.

- Write code that calculates $P(f_i|c)$ for each f_i of $\vec{f}$, using only the Training Set.
- (Why not the Test Set too?)
- Then apply the classifier to the Test Set.
- We don't need to worry about calculating $P(c)$ – why not?
- When you design your data structures, please think about later parts of this practical where you will dynamically split data into Training and Test on the fly, and train on different sets each time (Cross-Validation)

Smoothing

- You will see a big improvement in NB Classification with Smoothing.
- Smoothing in general: Instead of

  $$\frac{f(f_i)}{N}$$

  Use

  $$\frac{f(f_i) + smoothing(f_i)}{N + \sum_{\omega in V} smoothing(\omega)}$$

- Laplace Smoothing: $smoothing(f_i)$ is a small positive constant.

N-Fold Cross-Validation: Splitting

- Split data into N splits
- Use different strategies for splitting:
  - Consecutive splitting:

    cv000–cv099 = Split 1
    cv100–cv199 = Split 2
    ...
  - Round-robin splitting (mod 10):

    cv000, cv010, cv020,... = Split 1
    cv001, cv011, cv021,... = Split 2
    ...
  - Random sampling/splitting: Not used here (but you may choose to split this way in an non-educational situation)

N-Fold Cross-Validation

- For each split X – use all others for training, test on split X only
- The final performance is the average of the performances for each fold/split
- Apply sign test across splitting methods – why does it make sense to do so? What does it tell us if we pass this test?
- BTW, the way we have chosen to distribute data among splits is called stratified cross-validation.

Relationship to Pang et al. (2002)

- Your rough target for the baseline is a simplified implementation of Pang et al (2002).
- But: you are doing some extra things Pang et al didn't do (e.g., lexicon lookup)
- And Pang is doing much more than you do for the baseline:
    - Stemming
    - POS-tagging
    - Ngrams as features
    - Support Vector Machines (SVM) Classifier
- You **can** do these things for the baseline, but your are not expected to.
- Some of these things **could** be included in your extension system.