BigSE: Lessons Learned from Validating Industrial Text Mining

Rahul Krishna, Zhe Yu, Amritanshu Agrawal, NC State
Manuel Dominguez, David Wolf, LexisNexis
May 16, 2016

2nd International Workshop on BIG Data
Software Engineering: BIGDSE ’16
BigDSE’16: CFP

– Big Data for run-time monitoring and adaptation of software systems
– Big Data for software quality assurance and diagnosis
– Software architectures and languages for Big Data
– Quality and cost-benefit of Big Data software
– Curriculum for Big Data

Only paper in this workshop on this topic
Results (text mining, legal doc)

Surprising success

• In a 3 month time period
  – 3 first year grad students
  – With no prior data mining experience

• Find options for text miners that were
  – Better than those used in LN products
  – Now shipping in LN products
    • Specifically, SMOTE

Some surprises

• Mixed results with “standard” text mining method

• Many methods, widely used, not found useful here.

• Value of turning loose an army of students
What we found...

- Tokenization: stemming and stopwords removal
- Featurization: term frequency
- Dimensionality Reduction: feature hashing
- Normalization: L2 normalization on rows
- Data Balancing: SMOTE
- Classification: linear SVM

No presumption of generality

Need cost effective ways to explore your own data. Grad student teams?
Data

StackExchange Data:
• Tag-level data
  – predict tag within StackExchange sites
• Site-level data
  – predict site to which a certain post belongs to
  – generated by the “mock” data generator

Pre-process to LN “usual problem”: 
• 3% target class
• 97% other
• Processed as a binary classification problem
Standard Experiment Procedure

• Greedy Local Search
  – one step away from current best practice
  – if better, lock it as new best practice
  – iterate
5*5-way cross val
5 times, shuffle data, divide into 5 bins
Train on bin\[i\], Test on Data - test

Collect F-score
A combination of precision and recall:
  Precision: save review efforts
  Recall: retrieve as many as possible

(median = 50th percentile;
IQR = (75-25)th percentile)
Mind the gap:
If no gap between treatments, then pick treatment that runs fastest
Use of CPU farms

- CPU bottleneck
  - Each graph: 4 days (on single machine)
  - Given learning curve of students,
  - Run many many times

- On NC state’s high performance cluster:
  - 1 hour
  - Much better for learning curve of newbies
Example: small gap Featurization

- Term Frequency
- Tf*idf

Following on from this study, LN continues to use term frequency (since term frequency is 1 pass, not the 2 required for tf*idf)
Example, large gaps: Data Balancing

- **No Balancing**
- **SMOTE** (Synthetic Minority Over-sampling Technique)

Following on from this study, LN now SMOTEing
Other results

- Tokenization: stemming & stop word removal, shingling, just bag of words
  - No significant improvement over just bag of words
- Dimensionality Reduction: feature selection by tf*idf weight, feature hashing
  - No observed improvement over feature hashing
- Classification:
  - SVM beats NB, CART
  - SVM (linear) beats polynomial, sigmoid, RBD
- Normalization:
  - Standard method (L2 normalization on rows) wins
Challenges

• Generality: no presumption.
  – But, we did find that certain standard methods did not work well here (tf*idf)
  – And that’s the business case for recruiting an army of grad students to try various options

• Local Optimum
  – All this done very fast: find one improvement, lock it in, and repeat
  – Can we do better? Auto-tuning?
    E.g. IST journal’16 https://goo.gl/5w5GmM

• Scalability:
  – now exploring active learning and MT for crowd-based classification
THANK YOU!
Back up slides
Dimensionality Reduction

- Feature Hashing
- Feature selection by tf*idf weight
Dimensionality Reduction

- **Bag of words**
- **Stemming and stopwords removal**
- **Bigram shingling**
- **Trigram shingling**
Normalization

- No normalization
- L2 normalization on rows
- Normalization on columns
Classifier

- Naive Bayes (multinomial)
- Decision Tree (CART)
- SVM
  - Linear kernel
  - Polynomial kernel
  - Sigmoid kernel
  - RBF kernel
THANK YOU!