User Review Analysis of Google Play Store Apps

Mir Tafseer Nayeem (mir.nayeem@uleth.edu)

Department of Mathematics and Computer Science, University of Lethbridge

Tool(s) Used ….

Natural Language
Analyses with NLTK

python™

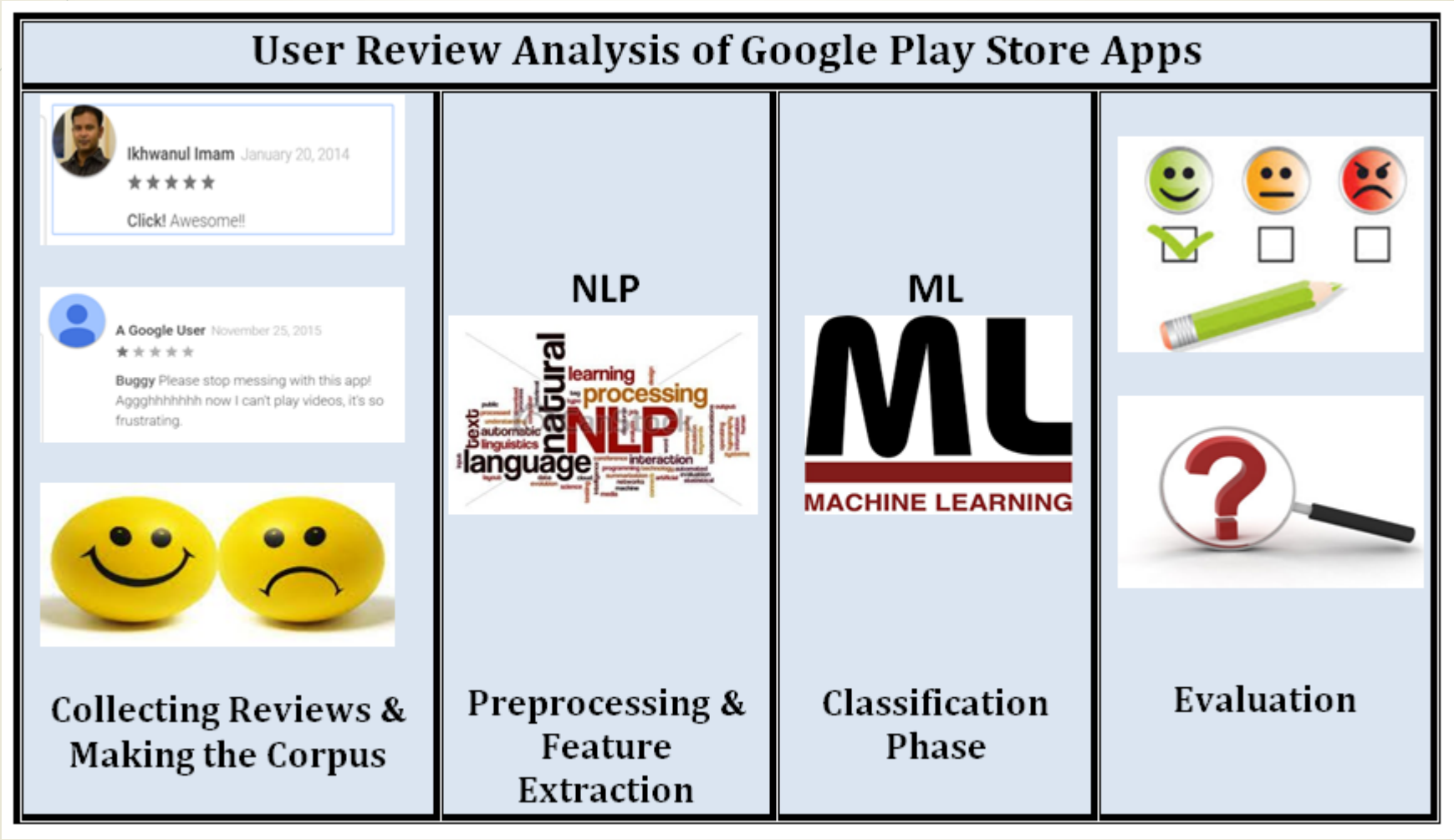
Outline

- **Motivation**
- Collecting Reviews & Making the Corpus
- Preprocessing and Feature Extraction
- Classification
- Evaluation
- Conclusion & Future Work

Motivation

- User review analysis more specifically **Sentiment Analysis** is becoming a popular area of research.
- App stores like Google Play [1] allow users to submit feedback for downloaded apps in form of **star ratings and text comments**.
- As of **February, 2015**, Google Play Store holds 1.4 million apps Android apps [2] both free and paid apps.
- It is very challenging for a potential user to read all of the comments one by one. For example, very popular apps such as **Facebook** get more than 4000 reviews per day.
- A textual review generally holds a mixed sentiment . I'll focus on 2 possible sentiment classifications of user reviews: **positive** and **negative.**

Review Analysis of App Store project Phases



Outline

- Motivation
- **Collecting Reviews & Making the Corpus**
- Preprocessing and Feature Extraction
- Classification
- Evaluation
- Conclusion & Future Work

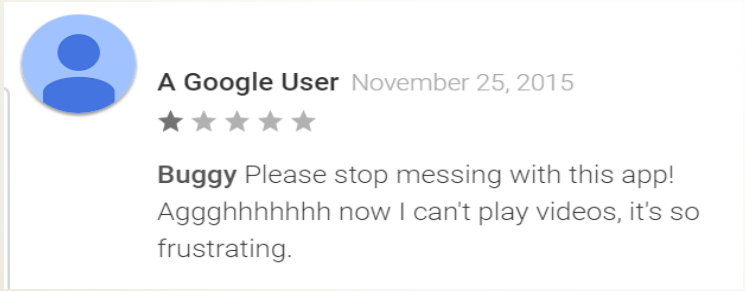
Collecting Reviews & Making the Corpus

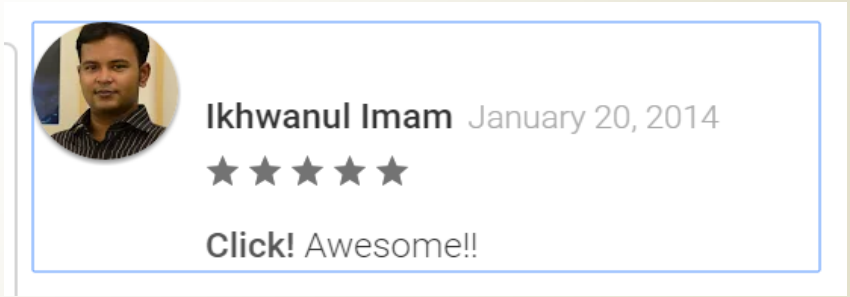
- NO Standard DATASET !!!!
- I wrote a crawler script to collect reviews from Google Play Store.
- I took the top apps id of 3000 Apps with their play store rankings between 1-10.
- I maintained the default sorting order i.e. most helpful reviews (The reviews and ratings marked helpful by others).
- **Limitations:**
  - Reviews were not annotated (positive/negative).
  - Many apps data was country specific (different languages).
  - Not all apps show up when querying for App IDs. For example, querying for “angry birds” App does not return results.
  - Only a maximum of 4500 reviews can be downloaded for any app.

Collecting Reviews & Making the Corpus (contd..)

- Need labeling for supervised classification.
- Studies highlighted Apps with better reviews with better star ratings[6].
- Assumptions :
  - 5 star rating reviews are labeled positive.
  - 1 and 2 star ratings are labeled as negative.
  - All the reviews which length is greater than 500 are ignored.
  - 3 star and 4 star ratings are ignored for better training data they usually contain both of the labels.

Review





Label

Outline

- Motivation
- Collecting Reviews & Making the Corpus
- **Preprocessing and Feature Extraction**
- Classification
- Evaluation
- Conclusion & Future Work

Preprocessing and Feature Extraction

Preprocessing

- **Removing the numbers**
- **Removing the Punctuations**: Punctuation marks (. , ; ( ) ? : etc) are removed.
- **Not Lowered Down**
  - All uppercase words like AWESOME, BEST, LOVE IT, RIP OFF are highly sentimental words. [decrease the accuracy by 2%.]
  - I only lowered down the first character if not all UP_CASE to reduce the overall feature size for instance “Loved it” and “I just loved it”.
- **No Spell Correction**
  - User reviews have typos and as well as contractions (U, coz, awsm , gr8 , Plz etc)
  - Spell checker algorithms converts some of the contractions into dictionary words and eventually reduces the performance by 1.3%.
- **Things Not done:**
  - Combination of punctuation marks represents the emoticons (like “:)” means ☺ and “:(” means ☹)
  - Repetitions (like soooooooo happyyyyyyyy.., greatttttt…. , looooved it , plzzzzz)

Preprocessing and Feature Extraction

Feature Extraction

- **Stopword removal:** 128 english stopwords (I , we , it , have , a , the , not , have , should , very , down , off etc) are **the neutral and common words** present in almost every sentences.
- **Lemmatization:** We use the Wordnet [9] lemmatizer from NLTK for grouping the different inflected forms of words syntactically different but semantically equal (like verbs “fixing”, “fixed”, and “fixes” are grouped into the term "fix".)
- **Unigram Features:** Taking every single word as features. Excluding the stopwords increase the accuracy by 2.73%.
- **Bigram Features:**
  - People normally use positives words in negative reviews such as **“not great”.**
  - Sequences of words that co-occur more often called collocations. For instance, “great app”.
  - Words such as **“rip off”, “slow down”, “very good”** those are highly sentimental bigrams but “not”, “off”, “down” and “very” etc are included in the stopwords list.
  - Including the stopwords for bigram finding will decrease the accuracy by 3%.

Outline

- Motivation
- Collecting Reviews & Making the Corpus
- Preprocessing and Feature Extraction
- **Classification**
- Evaluation
- Conclusion & Future Work

Classification

- The Binary classification labeled with either “pos” or “neg” for a single user review.
- The bag of words model is used.
- **Classifiers Used in the Project**
  - **Naive Bayes**
  - **Logistic regression** also known as Maximum Entropy
  - **Decision Tree**
- **Training Set vs Test Set**
  - The app reviews corpus has 2,861 positive files and 2,764 negative files.
  - 4-Fold Cross Validation method with 75:25 split ratio.
  - This gives us 4218 training instances and 1407 test instances.
- **Evaluation Measures**
  - **Accuracy**
  - **Precision**
  - **Recall**

Evaluation Results

- Naïve Bayes outperforms compare to other two models.
- Naïve Bayes and Logistic Regression are quite close to each other.
- Decision tree takes longer time in training as well perform worst.
- The combination of unigram and bigram performs better.

Feature Selection	Classification Models														
	Naïve Bayes					Logistic Regression					Decision Tree				
	A	P		R		A	P		R		A	P		R	
		PP	NP	PR	NR		PP	NP	PR	NR		PP	NP	PR	NR
Unigram	93.8	**90.2**	98.2	98.4	**89.0**	91.6	86.6	98.4	98.7	84.2	**87.2**	87.3	87.0	87.5	86.8
Bigram	96.3	**96.6**	96.1	96.2	**96.5**	96.1	95.8	96.4	96.6	95.6	86.3	86.2	86.9	87.3	86.4
Unigram + Bigram	**96.5**	96.1	96.9	97.0	95.9	96.0	94.7	97.6	97.7	94.3	87.4	88.0	86.8	87.1	87.6

Evaluation Results contd..

- Unigram model performs poorly compare to other two models because of independence of each words.

  (- + = -) (“not” + “good”) = bad (“-ve” word/feature)

  (- - = +) (“not” + “bad”) = good (“+ve” word/feature)

  (+ + = +) (“very” + “good”) = good**+** (“+ve” word/feature)

- Positive precision is higher in Bigram model and lower in unigram model.
- Lower precision means more false positives.
- This can only be occur when someone use a positive word in a negative review like the previous example.
- Lower negative recall when negative word is used in a positive review.

Choosing High Informative Features

- A high information feature is a word or group of words that is strongly biased towards a single classification label.
- Eliminating low information words from the training data can actually improve accuracy, precision, and recall.
- For example, the presence of the word **“AWESOME”** in an App review is a strong indicator that the review is **positive**.

High Ranked Features	Classification Models														
	Naïve Bayes					Logistic Regression					Decision Tree				
	A	P		R		A	P		R		A	P		R	
		PP	NP	PR	NR		PP	NP	PR	NR		PP	NP	PR	NR
10 Unigrams	**56.3**	53.8	100	100	11.1	60.9	56.6	99.3	99.8	20.7	**86.5**	86.4	86.6	87.2	85.8
100 Unigrams	92.8	88.4	98.5	98.7	86.6	**92.9**	88.6	98.5	98.7	86.8	87.7	88.6	86.7	87.0	88.4
1000 Unigrams	**95.8**	94.4	97.4	97.6	94.0	95.6	93.8	97.7	97.9	93.3	87.2	87.6	86.7	87.1	87.2
10000 Unigrams	95.6	93.7	97.8	98.0	93.1	95.1	92.6	98.1	98.3	91.8	87.4	87.5	87.2	87.7	87.1
15000 Unigrams	95.3	93.2	97.8	98.0	92.6	94.8	91.8	98.4	98.6	90.8	87.1	87.2	87.0	87.5	86.6
200 Bigrams	96.3	96.6	96.1	96.2	96.5	96.1	95.8	96.4	96.6	95.6	86.3	86.2	86.9	87.3	86.4

High Informative features of Classifiers

- Some top informative features **(unigram + bigram)** of the classifiers.
- The top features are different for classifiers.
- These classifiers can be combined to improve accuracy.

Features	Classification Models		
	Naïve Bayes	**Logistic Regression**	**Decision Tree**
Unigram	waste = True neg : 78.8	AWESOME = True pos : 63.9	**amazing = True pos**
	excellent = True pos : 77.9	**crap = True neg : 58.8**	terrible = True neg
	amazing = True pos : 76.3	BEST = True pos : 53.3	**crap = True neg**
Bigram	(u'highly', u'recommended') = True pos : 60.2	(u'very', u'useful') = True pos : 67.0	**(u'LOVE', u'IT') = True pos**
	(u'not', u'working') = True neg : 56.5	**(u'LOVE', u'IT') = True pos : 63.9**	(u'Not', u'happy') = True neg

Combining classifiers with voting

- Choose whichever label gets the most votes.
- Max vote classifier outperforms all the previous classifiers in terms of accuracy precision and recall in different combination of features.

High Ranked Features	Max Vote Classifier				
	A	P		R	
		PP	NP	PR	NR
10 Unigrams	60.9	56.6	99.3	99.8	20.6
100 Unigrams	92.8	88.5	98.5	98.7	86.8
1000 Unigrams	95.8	94.4	97.4	97.6	94.0
10000 Unigrams	95.7	93.8	97.8	98.0	93.3
15000 Unigrams	95.5	93.3	97.8	98.0	92.9
200 Bigrams	96.4	96.9	95.9	96.0	96.8

Feature Selection	Max Vote Classifier				
	A	P		R	
		PP	NP	PR	NR
Unigram	93.8	90.4	98.1	98.3	89.3
Bigram	96.5	96.9	96.0	96.1	96.8
Unigram + Bigram	96.7	96.4	96.9	97.0	96.2

Conclusion & Future Work

- I assumed all 5 star reviews are likely to be positive and all 1 or 2 star reviews are negative.
- Using this assumption I have labeled the corpus. The results somewhat justifies the assumption is correct.
- In future, I will try to device a mechanism to annotate the 3 or 4 star reviews in the corpus and evaluate the performance.
- Generating feature based summaries both for the users and the developers.
  - End-users can use these summaries to choose the apps with the best user experience according to specific features.
  - App developers can use these summaries to improve the quality, re-implement missing features, fixing bugs etc.

Reference

- **[1]** https://play.google.com/store/apps
- **[2]** http://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/
- **[3]** http://www.nltk.org/
- **[4]** Hu, M., & Liu, B. (2004a). Mining and summarizing customer reviews. In Proceedings of the 10th ACM SIGKDD international conference on knowledge discovery and data mining (pp. 168–177). Seattle, WA.
- **[5]** Hu, M., & Liu, B. (2004b). Mining opinion features in customer reviews. In Proceedings of the nineteenth national conference on artificial intelligence, sixteenth conference on innovative applications of artificial intelligence AAAI 2004 (pp. 755–760). San Jose.
- **[6]** H. Li, L. Zhang, L. Zhang, and J. Shen. A user satisfaction analysis approach for software evolution. In Progress in Informatics and Computing (PIC), 2010 IEEE International Conference on, volume 2, pages 1093–1097. IEEE, 2010.
- **[7]** A. Finkelstein, M. Harman, Y. Jia, W. Martin, F. Sarro, and Y. Zhang. App store analysis: Mining app stores for relationships between customer, business and technical characteristics. Reseach Note RN/14/10, UCL Department of Computer Science, 2014.
- **[8]** https://en.wikipedia.org/wiki/Stop_words
- **[9]** G. A. Miller. WordNet: a lexical database for English. Communications of the ACM, 38(11):39–41, 1995.
- **[10]** S. Bird, E. Klein, and E. Loper. Natural language processing with Python. O'reilly, 2009.
- **[11]**Intelligence, Artificial. "A modern approach." Russell and Norvig (2003).

Thank you, any question?