Utilizing Crowd Intelligence for Online Detection of Emotional Distress

Master's Thesis Presentation

Siddhant Goel
Advisor: Han Xiao
Supervisor: Prof. Dr. Claudia Eckert

Chair for IT Security
Technische Universität München

March 13, 2013

Outline

- Introduction
    - Backdrop
    - Motivation
    - Problem Definition
- Theoretical Background
    - Machine Learning and Text Classification
    - Support Vector Machines
    - Ensemble Learning methods
- Experimental Results
    - Experiments
    - Results
- Conclusion and Future Work
- Q/A

Introduction

Backdrop

- Millions of people die every year because of suicide
- Most people are between 15 to 29 years old
- Rise of social media - Twitter, Facebook, Reddit, Wordpress
- Sections of interest on Reddit - "/r/happy" [1] and "/r/suicidewatch" [2]
- Sections of interest on Twitter - the entire website
- People want to post their inner feelings on the web
- **Direct phrases** - *"thoughts of suicide make me happy"*, *"I have a rope around my neck"*
- **Indirect phrases** - *"I don't know anything anymore"*, *"Need someone to talk to"*

[1] `http://www.reddit.com/r/happy`

[2] `http://www.reddit.com/r/suicidewatch`

Backdrop

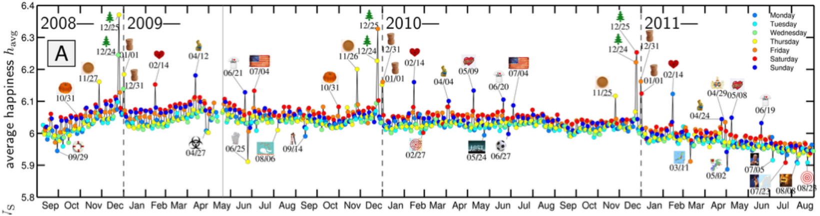
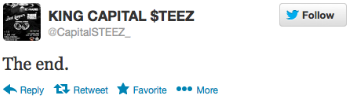


Figure : Happiness on Twitter as a function of time

- Study conducted in 2011
- 46 billion words collected over 33 months
- Negativity on Twitter has been on the rise
- Words include *death*, *hate*, and even *suicide*

Motivation



KING CAPITAL $TEEZ

@CapitalSTEEZ_

The end.

Figure : Last tweet of Twitter user "@CapitalSTEEZ_"[3]

- Some accounts have lots of followers, some don't
- Lives can be saved if there is a surveillance system of suicide
- Public sentiment information available on the web + No analysis possible = Disconnect

[3] http://twitter.com/CapitalSteez_

Problem Definition

- Evaluate machine learning algorithms that can be used for identifying depressed emotions in pieces of text
- Build a web based system that can
  - tap into crowd intelligence to incrementally improve the classifiers
  - detect content on the web that indicates that its author may be depressed or suicidal

Theoretical Background

Machine Learning

- Algorithms that can learn from data
- Construct a model from a given dataset, and then perform the required task on another dataset
- **Supervised learning** - Train the models on the training data, and predict on the test data
- **Unsupervised learning** - No distinction between training and test data

Text Classification

- Subset of machine learning algorithms (we focus on supervised text classification)
- Given some pieces of text, put unseen pieces of text into two or more categories
- Dataset $\{(\mathbf{x_n}, y_n)\}_{n=1}^{N}$ containing N instances
- Each instance $(\mathbf{x_n}, y_n)$ is of the form $[(x_{n,1}, x_{n,2}, ..., x_{n,D}), y_n]$
- Supervised learning - calculate y_n of test data given information about y_n from training data
- Unsupervised learning - calculate y_n given only information about $\mathbf{x_n}$

Support Vector Machines

- Fairly popular class of algorithms used for binary classification
- Given training data in some D dimensional space, find a decision boundary (hyperplane) that separates the two classes
- Maximize the distance of the boundary from any data point
- Decision function depends on a (usually small) subset of points called support vectors
- Kernel (linear/polynomial/RBF/Sigmoid) functions used to calculate distance between two points

Linear kernel SVM

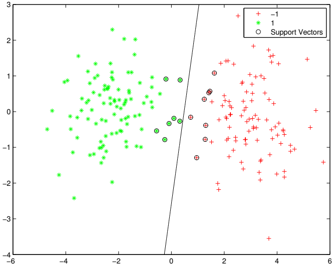


Figure : Binary classification on a dataset using a linear kernel SVM

Kernel functions

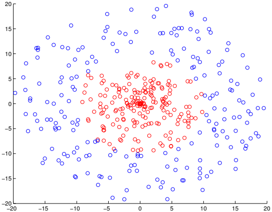


Figure : Dataset in 2D (cannot be classified using a linear kernel SVM)

Kernel functions

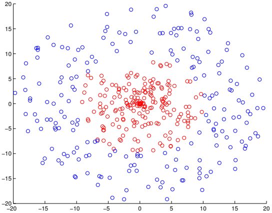


Figure : Dataset in 2D (cannot be classified using a linear kernel SVM)

$$\xrightarrow{x_3=\sqrt{x_1^2+x_2^2}}$$

Kernel functions

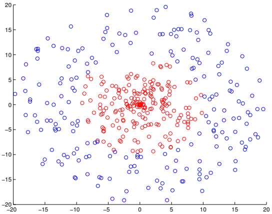


Figure : Dataset in 2D (cannot be classified using a linear kernel SVM)

$$\xrightarrow{x_3=\sqrt{x_1^2+x_2^2}}$$

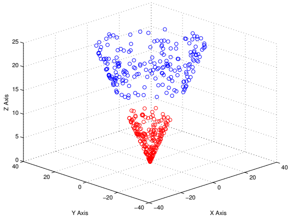


Figure : Dataset transformed to 3D

Ensemble Learning

- Class of machine learning methods that combine models to obtain better predictions
- Performance not guaranteed to be better than constituent classifiers
- Ensemble methods still usually outperform individual classifiers
- Soft requirement - underlying models should be diverse
- Various strategies to combine models - *select best*, *voting (bagging)*, *boosting*, *stacking*

Bagging

- Combine M classifiers to form a single classifier
- To predict, obtain predictions from all constituent classifiers, and take majority vote
- Requirement - classifiers should change for even small changes in underlying classifiers
- Two main approaches for training individual classifiers
  - Sample split - different samples for different classifiers
  - Feature split - different features for different classifiers
- Final prediction $= \mathrm{sign}(\sum_{m=1}^{M} y_m(\mathbf{x_n}))$

Boosting

- Assign each sample a weight value (same for all samples in the beginning)
- Train M classifiers successively
- Each classifier focuses more on samples that the previous classifier classified incorrectly
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)
- Final prediction $= \mathrm{sign}(\sum_{m=1}^{M} \alpha_m y_m(\mathbf{x_n}))$

Stacking

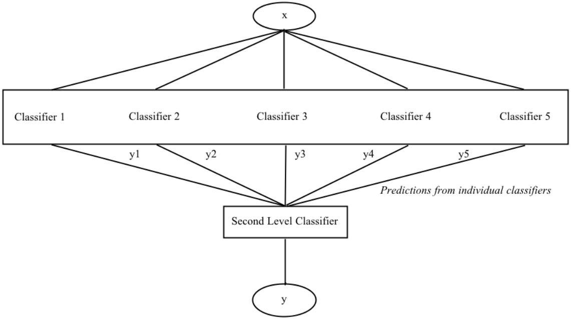


Figure : Prediction using a stacking ensemble

- Outputs from first layer form the input for second layer
- Layer-1 classifiers can be trained using bootstrapping or selecting random features

Experimental Results

Experiments

Evaluation of algorithms

Datasets

- Evaluation
  - Comments dataset from a competition on Kaggle
  - 6182 comments, each having a binary label
  - Label depends on whether or not a particular comment insults another user
- Web based system
  - Training data - Reddit
  - Fetch posts from "/r/happy" [4] and "/r/suicidewatch" [5]
  - Prediction data - Twitter
  - Gather tweets from the public streaming API [6]

[4] http://www.reddit.com/r/happy

[5] http://www.reddit.com/r/suicidewatch

[6] https://dev.twitter.com/docs/streaming-apis/streams/public

Web based system

Data consolidation frequencies

Task	**Frequency**
Fetch 1000 posts from Reddit	24 hours
Fetch 100 tweets from Twitter	3 hours
Re-assign labels to previous tweets and update statistics	24 hours

Evaluation of algorithms

Approach

- Implement all the models in MATLAB
- Extract n-grams (size upto 2) out of the Kaggle dataset
- Use tf-idf information as feature values
- Input matrix - 6182 rows and 23175 columns
- **SVM**, **Bagging**, **Boosting**, **Stacking** - obtain accuracy (10-fold cross validation) against number of samples
- **SVM** - Growth of number of support vectors with the number of samples
- **Bagging** - Accuracy against number of underlying models
- Start with 100 samples, and continue adding 100 samples in each iteration until no more samples are left

Results

Evaluation of algorithms

Support Vector Machines

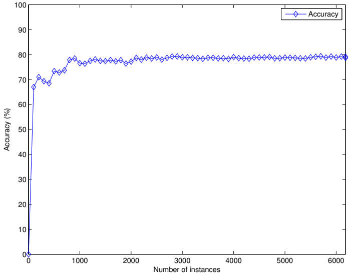


Figure : Accuracy v/s Number of instances

- Linear kernel accuracy - 79.02%
- Polynomial/RBF/Sigmoid kernel accuracy - 34.39%
- Having a large number of features implies less need for a kernel function

Evaluation of algorithms

Support Vector Machines

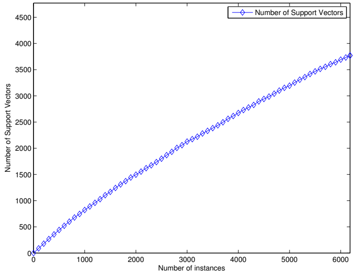


Figure : Number of support vectors v/s Number of instances

- 100 samples - 90% support vectors
- 6182 samples - 60% support vectors
- Less support vectors implies that the classification is easy

Evaluation of algorithms

Bagging

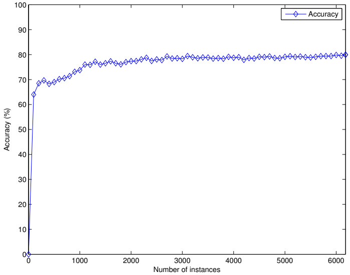


Figure : Accuracy v/s Number of instances

- 9 linear kernel SVMs underneath
- Performance ultimately governed by how SVMs perform
- Average accuracy on all 6182 samples = 79.65%

Evaluation of algorithms

Bagging

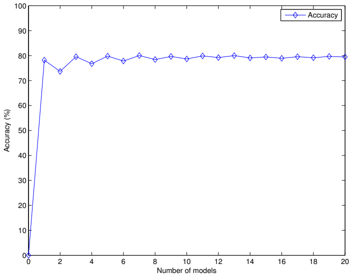


Figure : Accuracy v/s Number of models

- Accuracy “stabilizes” slowly with the number of models
- Each model is trained on a random subset of samples
- Increase in number of models *implies* Subsets overlap *implies* Performance stabilizes

Evaluation of algorithms

Boosting

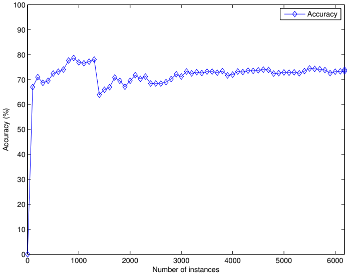


Figure : Accuracy v/s Number of instances

- 9 linear kernel SVMs trained successively on each iteration
- Average accuracy = 72.84%

Evaluation of algorithms

Stacking

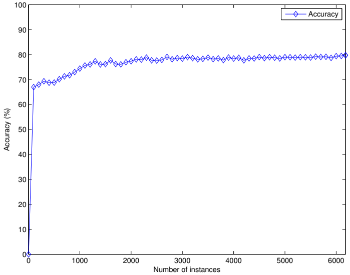


Figure : Accuracy v/s Number of instances

- 9 linear kernel SVMs at first level
- Second level SVM trained using a linear kernel as well
- Average accuracy = 79.48%

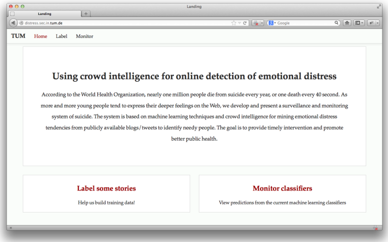
Web based system

Approach

- Implement all the models in Python, and web interface in Django
- No training data available => build our own
- Training data
  - Consolidated from Reddit
  - "/r/happy" - users posts their happy moments
  - "/r/suicidewatch" - users post when they want to commit suicide
  - Pull 500 stories from each, every day
- Prediction data
  - Should be the general sentiment of the overall public
  - Hence, pull 100 tweets every 3 hours from Twitter

Web based system

Architecture



Using crowd intelligence for online detection of emotional distress

According to the World Health Organization, nearly one million people die from suicide every year, or one death every 40 second. As more and more young people tend to express their deeper feelings on the Web, we develop and present a surveillance and monitoring system of suicide. The system is based on machine learning techniques and crowd intelligence for mining emotional distress tendencies from publicly available blogs/tweets to identify needy people. The goal is to provide timely intervention and promote better public health.

Label some stories

Help us build training data!

Monitor classifiers

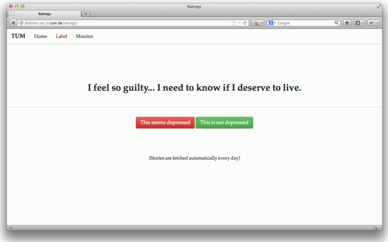
View predictions from the current machine learning classifiers

Figure : Landing page

- Two sections - *Ratings* and *Monitoring*
- *Ratings* - used to build the training data
- *Monitoring* - used to monitor the status of the models and check distressed tweets

Web based system

Ratings module



I feel so guilty... I need to know if I deserve to live.

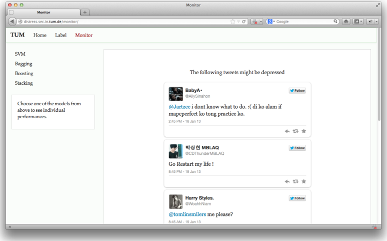
(Stories are fetched automatically every day)

Figure : Landing page of the Ratings module

- Taps into crowd intelligence
- Automated scripts fetch 1000 posts from Reddit every day
- Displays the first unlabelled post
- Users can then assign labels to stories
- This builds the training data

Web based system

Monitoring module



Choose one of the models from above to see individual performances.

The following tweets might be depressed

BabyA•
@AllySinahon

@Jartzee i dont know what to do. :(di ko alam if mapeperfect ko tong practice ko.

2:45 PM - 18 Jan 13

박상현 MBLAQ
@CDThunderMBLAQ

Go Restart my life !

8:45 PM - 18 Jan 13

Harry Styles.
@WoahhNiam

@tomlinsmilers me please?

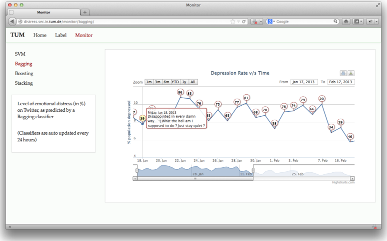
8:45 AM - 19 Jan 13

Figure : Landing page of the Monitoring module

- Displays the top few tweets that were classified as depressed by all the classifiers
- Links for checking the status of individual classifiers

Web based system

Monitoring module



Level of emotional distress (in %) on Twitter, as predicted by a Bagging classifier

(Classifiers are auto updated every 24 hours)

Friday, Jan 18, 2013
Disappointed in every damn way... :(What the hell am I supposed to do ? Just stay quiet ?

Figure : Statistics of the bagging classifier

For each classifier -

- displays the overall level of distress (in %) on Twitter
- each bubble displays
  - the number of stories that were found depressed on that particular day
  - the first tweet from that day which was depressed (no confidence values, yet)

Conclusion and Future Work

Conclusion

- An evaluation of Support Vector Machines and Ensemble Learning methods (Bagging/Boosting/Stacking) in the domain of text classification
- Bagging outperformed Stacking outperformed SVM outperformed Boosting
- A web based system that can detect emotional distress on Twitter
- No labels implies qualitative evaluation is difficult except observation
- Observed results seem to be reasonable

Future Work

- Fetch more tweets
- Increase the crowd intelligence involved
- Relabelling process (decreases wastage of resources)
- Select best performing model
- Store confidence values

Thank you!

Questions?