

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 14, 2013

Outline

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

Outline

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

Outline

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

Introduction

Backdrop

Depression and Suicide

- Nearly one million people die every year because of suicide
- Most people are between 15 to 29 years old

Social Media

- Rise of Twitter, Facebook, Reddit, Wordpress
- Sections of interest
 - Reddit - “/r/happy” ^a and “/r/suicidewatch” ^b
 - Twitter - the entire website

^a<http://www.reddit.com/r/happy>

^b<http://www.reddit.com/r/suicidewatch>

Backdrop

Depression and Suicide

- Nearly one million people die every year because of suicide
- Most people are between 15 to 29 years old

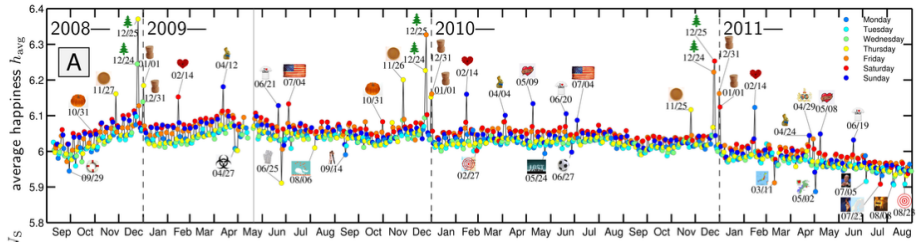
Social Media

- Rise of Twitter, Facebook, Reddit, Wordpress
- Sections of interest
 - Reddit - “/r/happy” ^a and “/r/suicidewatch” ^b
 - Twitter - the entire website

^a<http://www.reddit.com/r/happy>

^b<http://www.reddit.com/r/suicidewatch>

Backdrop



- 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.



Reply



Retweet



Favorite



More

- **Direct** - “thoughts of *suicide* make me happy”, “I have a *rope around my neck*”
- **Indirect** - “I *don't know* anything anymore”, “*Need someone* to talk to”
- 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

Motivation



KING CAPITAL \$TEEZ

@CapitalSTEEZ_

Follow

The end.



Reply



Retweet



Favorite



More

- **Direct** - *“thoughts of **suicide** make me happy”, “I have a **rope around my neck**”*
- **Indirect** - *“I **don't know** anything anymore”, “**Need someone** to talk to”*
- 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

Problem Definition

Experiments

Evaluate machine learning algorithms that can be used for identifying depressed emotions in pieces of text

System

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

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

Formal definition

Given a dataset $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ containing N instances, where each instance (\mathbf{x}_n, y_n) is of the form $[(x_{n,1}, x_{n,2}, \dots, x_{n,D}), y_n]$, calculate the y_n values.

- Given some pieces of text, put unseen pieces of text into two or more categories
- **Supervised** - calculate y_n of test data given information about y_n from training data
- **Unsupervised** - calculate y_n given only information about \mathbf{x}_n

Text Classification

Formal definition

Given a dataset $\{(\mathbf{x}_n, y_n)\}_{n=1}^N$ containing N instances, where each instance (\mathbf{x}_n, y_n) is of the form $[(x_{n,1}, x_{n,2}, \dots, x_{n,D}), y_n]$, calculate the y_n values.

- Given some pieces of text, put unseen pieces of text into two or more categories
- **Supervised** - calculate y_n of test data given information about y_n from training data
- **Unsupervised** - calculate y_n given only information about \mathbf{x}_n

Document Representation

Text Corpus

“I am happy today”

and

“I am not happy
today, but I was
happy yesterday”

Document Representation

Text Corpus

"I am happy today"

and

→

"I am not happy
today, but I was
happy yesterday"

Document Representation

Text Corpus

"I am happy today"

and

"I am not happy
today, but I was
happy yesterday"

→

Token dictionary

"I": 1,

"am": 2,

"happy": 3,

"today": 4,

"not": 5,

"but": 6,

"was": 7,

"yesterday": 8

Document Representation

Text Corpus

"I am happy today"

and

"I am not happy
today, but I was
happy yesterday"

→

Token dictionary

"I": 1,

"am": 2,

"happy": 3,

"today": 4,

"not": 5,

"but": 6,

"was": 7,

"yesterday": 8

→

Document Representation



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
- Hyperplane ($\mathbf{w} \cdot \mathbf{x} - b = 0$) should have maximum distance from any data point
- Solution for linear classifiers: $\mathbf{w} = \sum_{i=1}^S \alpha_i \mathbf{x}_i$
- Replace $\mathbf{x}_i \cdot \mathbf{x}$ with $k(\mathbf{x}_i, \mathbf{x}) \implies$ represents the dot product of two vectors in higher dimensions (kernel function)

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
- Hyperplane ($\mathbf{w} \cdot \mathbf{x} - b = 0$) should have maximum distance from any data point
- Solution for linear classifiers: $\mathbf{w} = \sum_{i=1}^S \alpha_i \mathbf{x}_i$
- Replace $\mathbf{x}_i \cdot \mathbf{x}$ with $k(\mathbf{x}_i, \mathbf{x}) \implies$ represents the dot product of two vectors in higher dimensions (kernel function)

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
- Hyperplane ($\mathbf{w} \cdot \mathbf{x} - b = 0$) should have maximum distance from any data point
- Solution for linear classifiers: $\mathbf{w} = \sum_{i=1}^S \alpha_i \mathbf{x}_i$
- Replace $\mathbf{x}_i \cdot \mathbf{x}$ with $k(\mathbf{x}_i, \mathbf{x}) \implies$ represents the dot product of two vectors in higher dimensions (kernel function)

Linear kernel SVM

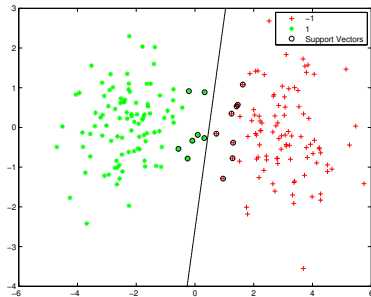


Figure : Binary classification on a dataset using a linear kernel SVM

Kernel functions

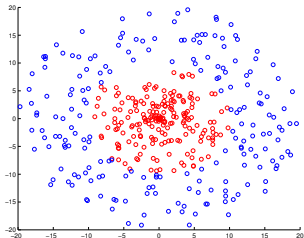
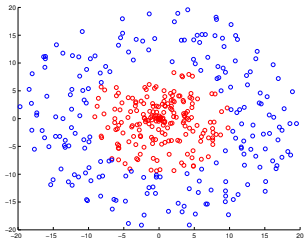


Figure : Dataset in 2D (cannot be classified using a linear kernel SVM)

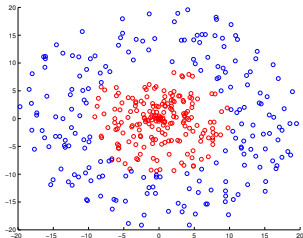
Kernel functions



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

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

Kernel functions



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

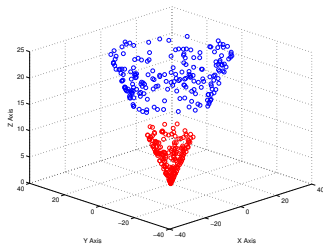


Figure : Dataset in 2D (cannot be classified using a linear kernel SVM)

Figure : Dataset transformed to 3D

Ensemble Learning

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

Ensemble Learning

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

Bagging

- Obtain predictions from all constituent classifiers, and take a majority vote

- Final prediction = $\text{sign}(\sum_{m=1}^M y_m(\mathbf{x}_n))$

Bagging

- Obtain predictions from all constituent classifiers, and take a majority vote

- Final prediction = $\text{sign}\left(\sum_{m=1}^M y_m(\mathbf{x}_n)\right)$

$$\begin{pmatrix} x_{1,1} & x_{1,2} & x_{1,3} & \dots & x_{1,D} \\ x_{2,1} & x_{2,2} & x_{2,3} & \dots & x_{2,D} \\ x_{3,1} & x_{3,2} & x_{3,3} & \dots & x_{3,D} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_{N,1} & x_{N,2} & x_{N,3} & \dots & x_{N,D} \end{pmatrix}$$

Sample split

Bagging

- Obtain predictions from all constituent classifiers, and take a majority vote

- Final prediction = $\text{sign}\left(\sum_{m=1}^M y_m(\mathbf{x}_n)\right)$

$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	\dots	$x_{1,D}$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	\dots	$x_{2,D}$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	\dots	$x_{3,D}$
\vdots	\vdots	\vdots	\vdots	\vdots
$x_{N,1}$	$x_{N,2}$	$x_{N,3}$	\dots	$x_{N,D}$

Sample split

$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	\dots	$x_{1,D}$
$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	\dots	$x_{2,D}$
$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	\dots	$x_{3,D}$
\vdots	\vdots	\vdots	\vdots	\vdots
$x_{N,1}$	$x_{N,2}$	$x_{N,3}$	\dots	$x_{N,D}$

Feature split

Boosting

- Assign each sample a weight value (same for all samples in the beginning), and train M classifiers successively
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)

- Final prediction = $\text{sign}(\sum_{m=1}^M \alpha_m y_m(\mathbf{x}_n))$

Boosting

- Assign each sample a weight value (same for all samples in the beginning), and train M classifiers successively
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)

- Final prediction = $\text{sign}(\sum_{m=1}^M \alpha_m y_m(\mathbf{x}_n))$

\mathbf{x}_1
\mathbf{x}_2
\mathbf{x}_3
\mathbf{x}_4
\mathbf{x}_5

Boosting

- Assign each sample a weight value (same for all samples in the beginning), and train M classifiers successively
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)

- Final prediction = $\text{sign}(\sum_{m=1}^M \alpha_m y_m(\mathbf{x}_n))$

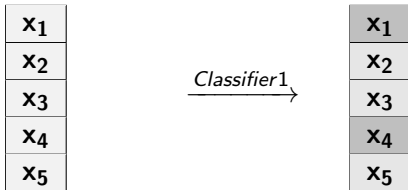
\mathbf{x}_1
\mathbf{x}_2
\mathbf{x}_3
\mathbf{x}_4
\mathbf{x}_5

$\xrightarrow{\text{Classifier1}}$

Boosting

- Assign each sample a weight value (same for all samples in the beginning), and train M classifiers successively
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)

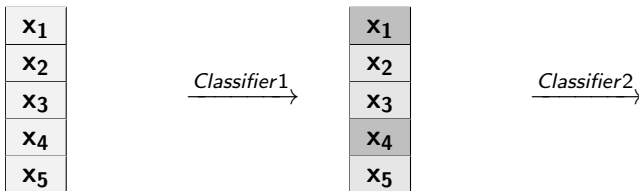
- Final prediction = $\text{sign}(\sum_{m=1}^M \alpha_m y_m(\mathbf{x}_n))$



Boosting

- Assign each sample a weight value (same for all samples in the beginning), and train M classifiers successively
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)

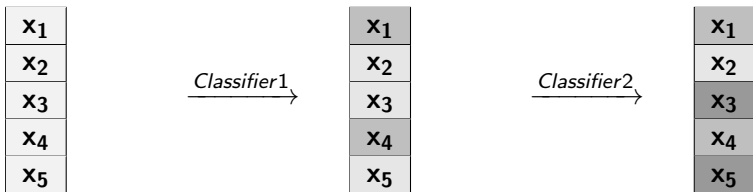
- Final prediction = $\text{sign}(\sum_{m=1}^M \alpha_m y_m(\mathbf{x}_n))$



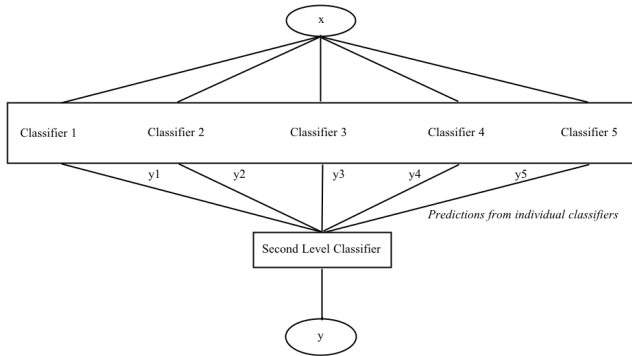
Boosting

- Assign each sample a weight value (same for all samples in the beginning), and train M classifiers successively
- For each classifier, calculate ϵ (measure of error) and α (decreases with ϵ)

- Final prediction = $\text{sign}(\sum_{m=1}^M \alpha_m y_m(\mathbf{x}_n))$



Stacking



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

Experiments

Experiments

Dataset

- List of 6182 comments from the internet - Kaggle ¹
- label | timestamp | comment
- Examples
 - 1 - How arrogant you are
 - 1 - you are human garbage
 - 0 - i really don't understand your point. It seems you are mixing apples and oranges.
 - 0 - you may be right

¹<http://www.kaggle.com/c/detecting-insults-in-social-commentary>

Experiments

Approach

- Extract n-grams upto size 2 and use tf-idf information as feature values
- Input matrix - 6182 rows and 23175 columns
- Implement all models in MATLAB
- Start with 100 samples, and continue adding 100 samples in each iteration until no more samples are left

Experiments

Approach

- Extract n-grams upto size 2 and use tf-idf information as feature values
- Input matrix - 6182 rows and 23175 columns
- Implement all models in MATLAB
- Start with 100 samples, and continue adding 100 samples in each iteration until no more samples are left

Experiments

Approach

- Extract n-grams upto size 2 and use tf-idf information as feature values
- Input matrix - 6182 rows and 23175 columns
- Implement all models in MATLAB
- Start with 100 samples, and continue adding 100 samples in each iteration until no more samples are left

Name	Accuracy	Support Vector Count	Model Count
SVM	✓	✓	✗
Bagging	✓	✗	✓
Boosting	✓	✗	✗
Stacking	✓	✗	✗

Experiments

Support Vector Machines

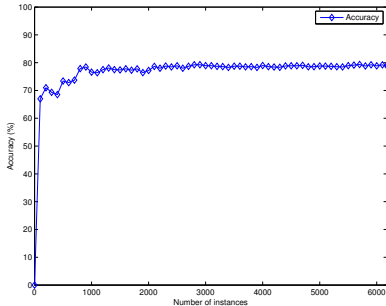


Figure : Accuracy

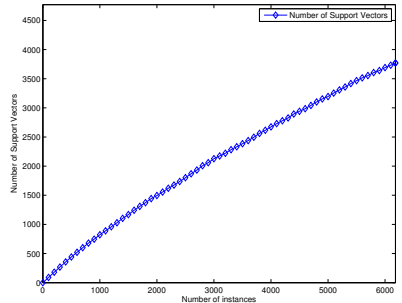


Figure : Support Vector count

79.02% for linear kernel, and 34.39% for Polynomial/RBF/Sigmoid kernels

Number of support vectors decreases from 90% to 60%

Experiments

Bagging

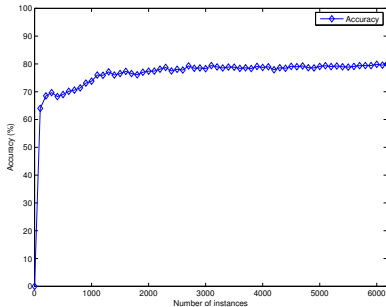


Figure : Accuracy v/s Number of instances

79.65% (9 linear kernel SVMs)

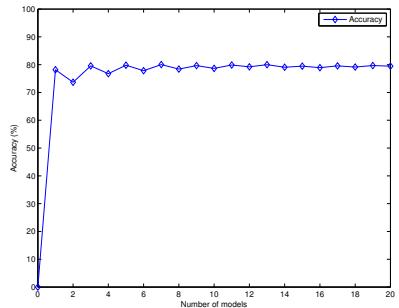
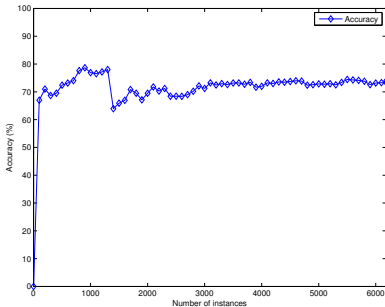


Figure : Accuracy v/s Number of models

Models increase \rightarrow Subsets overlap \rightarrow
Accuracy Stabilizes

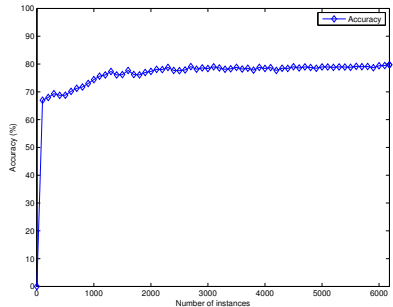
Experiments

Boosting



Average accuracy of 72.84%

Stacking



Average accuracy of 79.48%

All using linear kernel SVMs

System

Dataset

- **Training data** - Reddit
- Fetch posts from “/r/happy”² and “/r/suicidewatch”³
- **Prediction data** - Twitter
- Gather tweets from the public streaming API⁴

²<http://www.reddit.com/r/happy>

³<http://www.reddit.com/r/suicidewatch>

⁴<https://dev.twitter.com/docs/streaming-apis/streams/public>

Dataset

- **Training data** - Reddit
- Fetch posts from “/r/happy”² and “/r/suicidewatch”³
- **Prediction data** - Twitter
- Gather tweets from the public streaming API⁴

²<http://www.reddit.com/r/happy>

³<http://www.reddit.com/r/suicidewatch>

⁴<https://dev.twitter.com/docs/streaming-apis/streams/public>

Dataset

- **Training data** - Reddit
- Fetch posts from “/r/happy”² and “/r/suicidewatch”³
- **Prediction data** - Twitter
- Gather tweets from the public streaming API⁴

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

²<http://www.reddit.com/r/happy>

³<http://www.reddit.com/r/suicidewatch>

⁴<https://dev.twitter.com/docs/streaming-apis/streams/public>

Approach

- Implement all classifiers in Python, and web interface in Django
- No training data available → build our own
- Training data (Reddit)
 - “/r/happy” - people posts their happy moments
 - “/r/suicidewatch” - people post when they want to commit suicide
 - labels assigned by users of our system
- Prediction data (Twitter)
 - General sentiment of the overall public
 - Pull 100 tweets every 3 hours from Twitter

Approach

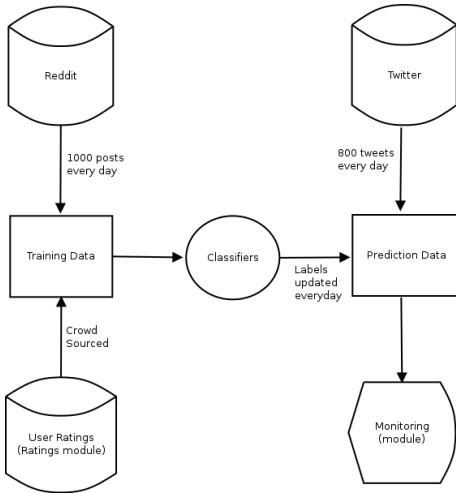
- Implement all classifiers in Python, and web interface in Django
- No training data available → build our own
- Training data (Reddit)
 - “/r/happy” - people posts their happy moments
 - “/r/suicidewatch” - people post when they want to commit suicide
 - labels assigned by users of our system
- Prediction data (Twitter)
 - General sentiment of the overall public
 - Pull 100 tweets every 3 hours from Twitter

Approach

- Implement all classifiers in Python, and web interface in Django
- No training data available → build our own
- Training data (Reddit)
 - “/r/happy” - people posts their happy moments
 - “/r/suicidewatch” - people post when they want to commit suicide
 - labels assigned by users of our system
- Prediction data (Twitter)
 - General sentiment of the overall public
 - Pull 100 tweets every 3 hours from Twitter

System

Architecture



- **Ratings** - allows users to assign labels to stories (crowd intelligence), building the training data
- **Monitoring** - displays predictions of classifiers in the form of depressed tweets and individual statuses of classifiers

Demo

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

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

Future Work

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

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?