3SAT- A Semi-Supervised Learning approach to Sentiment Analysis and Classification of Tweets

Arpit Sharma[1], **Sarah Riaz**[2]
University of Southern California
Los Angeles, CA 90007
arpitsha@usc.edu, sriaz@usc.edu

Abstract

Twitter is a microblogging platform in which users can post status messages, called tweets, to their friends. Having provided an enormous dataset for the Sentiments, a thorough analysis of these tweets helps gain an insight to users opinion. Classification of such a massive dataset can be performed via supervised learning. To build supervised learning models, classification algorithms require a set of representative labeled data. However, labeled data are usually difficult and expensive to obtain, which motivates the interest in semi-supervised learning. This brand of learning utilizes unlabeled data to complement the information obtained from the labeled data during the training phase; thus, such an approach can be crucial in applications that include sentiment analysis, where there is prodigious amount of unlabeled data at hand. Although an appealing research topic now, this concept is relatively new. In this project, we aim to develop a system to detect the sentiment from tweets. There are several approaches that can be employed in semi-supervised learning, such as wrapper based, topic based and graph based. We look to focus on self-training which is one of wrapper based technique.

1 Introduction

Microblogging websites have evolved to become a source of varied kind of information[1]. This is due to nature of micro-blogs on which people post real time messages about their opinions on a variety of topics, discuss current issues, complain, and express positive sentiment for products they use in daily life. The main idea is to build technology to detect and summarize an overall sentiment.

3SAT is a classifier that performs sentiment analysis and classification of tweets using semi-supervised learning. The crux of the application is to classify the tweets into three classes, namely *Positive, Negative or Neutral*. The tweets may be pre-processed for sentiment-aware tokenization, spell-correction, word normalization, word segmentation (for splitting hashtags) and word annotation. This pre-processed text can act as a foil on which several methods can be applied to perform the objective task[2].

The emphasis lies in using wrapper-based methods such as self-training. In self-training we seek to experiment with and without replacement of data-points. SVMs of the Scikit learn can be used to classify the data, we seek to tune the hyperparameters such as C-parameter, kernel, gamma and probability.

Accordingly, we can focus on three crucial aspects:

- Decision function
- Prediction probability
- Prediction accuracy

2 Analysis

Sentiment analysis of data is one of the biggest challenges of this time. Even tech giants like Facebook are struggling to tackle this problem. Recently, Congress asked Mark Zuckerberg about how Facebook is eliminating hate speech and hate figures from their platform because currently, they dont have any efficacious sentiment analysis algorithm for it which can solve this problem. Sentiment analysis is also helpful in better targeted marketing, brand-reputation protection, faster detection of opportunities and threats, and increasing profit. Another tech giant is Twitter. Twitter is a microblogging platform in which users can post status messages, called tweets, to their friends. It has provided an enormous dataset of the so-called sentiments, whose classification can take place through supervised learning. To build supervised learning models, classification algorithms require a set of representative labeled data. However, labeled data are usually difficult and expensive to obtain, which motivates the interest in semi-supervised learning. This type of learning uses unlabeled data to complement the information provided by the labeled data in the training process; therefore, it is particularly useful in applications including tweet sentiment analysis, where a huge quantity of unlabeled data is accessible. Semi-supervised learning for tweet sentiment analysis, although appealing, is relatively new. We provide a comprehensive survey of semi-supervised approaches applied to tweet classification. A comparative study of algorithms based on self-training, co-training, topic modeling, and distant supervision highlights their biases and sheds light on aspects that the practitioner should consider in real-world applications. Analyzing sentiment of tweets is important as it helps to determine the users opinion. Knowing peoples opinion is crucial for several purposes starting from gathering knowledge about customer base, e-governance, campaigns and many more. In this project, we aim to develop a system to detect the sentiment from tweets.

3 Previous Work

3.1 Go, Bhayani and Huang (2009)

They classify Tweets for a query term into negative or positive sentiment. They collect training dataset automatically from Twitter. To collect positive and negative tweets, they query twitter for happy and sad emoticons.

- Happy emoticons are different versions of smiling face, like ':)', ':-)', ':)', ':D', '=)' etc.
- Sad emoticons include frowns, like ':(', ':-(', ':(' and so on.

They try various features unigrams, bigrams and Part-of-Speech and train their classifier on various machine learning algorithms Naive Bayes, Maximum Entropy and Scalable Vector Machines and compare it against a baseline classifier by counting the number of positive and negative words from a publicly available corpus. They report that bigrams alone and Part-of-Speech Tagging are not helpful and that Naive Bayes Classifier gives the best results.

3.2 Pak and Paroubek (2010)

They identify that use of informal and creative language make sentiment analysis of tweets a rather different task . They leverage previous work done in hashtags and sentiment analysis to build their classifier. They use Edinburgh Twitter corpus to find out most frequent hashtags. They manually classify these hashtags and use them to in turn classify the tweets. Apart from using n-grams and Part-of-Speech features, they also build a feature set from already existing MPQA subjectivity lexicon and Internet Lingo Dictionary. They report that the best results are seen with n-gram features with lexicon features, while using Part-of-Speech features causes a drop in accuracy.

3.3 Koulompis, Wilson and Moore (2011)

They investigated the utility of linguistic features for detecting the sentiment of Twitter messages. They evaluated the usefulness of existing lexical resources as well as features that capture information about the informal and creative language used in microblogging. They took a supervised approach to the problem, but leverage existing hashtags in the Twitter data for building training data.

4 3SAT

4.1 Preprocessing

The first step in this project was to preprocess tweets so that we can get features using this preprocessed data for training our models and then, using the best model for predictions. Preprocessing of data is essential to retrieve meaningful information from these tweets. The following preprocessing is done on the tweets in our dataset:

- **Replacement of Links and Twitter Handles:** First, we have replaced the URLs and Twitter handles in each tweet with placeholder text *'url'* and *'@user'* respectively because they dont play any significant role in determining the sentiments expressed in a tweet. In other words, we can easily ignore URLs and Twitter handles while analyzing the sentiment of a tweet.
- **Replacement of Slangs:** Slangs are informal nonstandard vocabulary composed typically of coinages, arbitrarily changed words, and extravagant, forced, or facetious figures of speech; and are more common in speech than in writing but are commonly used in social media posts. Therefore, we have replaced slangs in the tweets with proper standard words so that they can be meaningful information from them.
- **Negating Words:** There are words that are either positive or negative when considered in isolation, but when they are considered in the context they are used, their meaning is negated. For example, if not is used in a sentence, then the meaning of negative words becomes positive and vice versa. Hence, we put *'_neg'* at the end of the words whose meaning is negated because this gives us significant information about the sentiment of a tweet and is used while calculating the score of the tweets in out dataset.
- **Removing Stop Words:** Stop words, such as 'and', 'the', 'a', 'an' and similar words, are natural language words which have very little meaning and are irrelevant when analyzing the sentiment of a tweet. Therefore, we have removed these useless words while preprocessing our tweets.
- **POS Tagging:** Part of speech tagging is done on each word of the tweet because knowing which part of speech a word belongs to gives us significant information about the sentiment of a tweet. For example, a noun is not as significant as an adjective for sentiment analysis.
- **Word Frequency:** After performing the above-mentioned steps, the frequency of each word is calculated in a tweet because higher frequency of either negative or positive words play an important role in determining score of a tweet and hence, its sentiment.

4.2 Vectorization of Features:

After performing the above-mentioned steps, twelve different scores are calculated for each tweet. These scores are then used as predictors and our models are fit and tested based on these features. These twelve scores are: Afinn Score 96, Afinn Score 111, Lexicon Score, Senti 140 Score, NRC Score, Binliu Score, Sentiword Score, Writing Style, Emoticon Score, Unicode Emoticon Score, Unigram Score and Post Unigram Score. These scores are calculated by considering the affinity, frequency, sentiment of words, emoticons and hashtags. It is noteworthy here that the positive words we negated in the preprocessing contribute in score of negative words and vice versa.

4.3 Assigning Class Weights:

We have assigned class weight as follows:

- Class weight(Positive) = No. of Neutral Tweets / No. of Negative Tweets
- Class weight(Negative) = No. of Neutral Tweets / No. of Positive Tweets

This means that there is an inverse relation between them. When the number of positive tweets increase, a smaller class weight is given to negative class and vice versa.

4.4 Scaling and Normalization of data:

To standardize the range of our features, we have normalized the data so that our objective functions can perform properly. Standardizing the features so that they are centered around zero with a standard deviation of one is important because we are comparing different scores.

4.5 Training Strategy

To implement what we have previously discussed, we will split the Train data into two sets. Set one, the set of data with the labeled data was used to train a supervised SVM to fit an initial model on the data. The second set of data was unlabeled. Post fitting the model a decision function was used to determine the closest twenty support vectors near the decision boundary. The model was used to predict the labels of these support vectors, post-prediction these data-points were also added to the training set and removed from the unlabeled training buffer[3]. We triggered the *refit* hyperparameter to fit this new model on the training data after each iteration. This process was performed until the unlabeled training buffer was empty. This iterative improvement on the previously fit model can be significant when one has a huge dataset, as in our scenario.

4.6 Model Selection

We tried several models to fit as our base model for the initial run, namely Logistic Regression, Support Vector classifier with linear and radial kernel, Decision Tree classifier, Nave Bayes, Stochastic Gradient Descent(SGD.

- **Logistic Regression:** The binary logistic model is used to estimate the probability of a binary response based on one or more predictor (or independent) variables (features). It allows one to say that the presence of a risk factor increases the odds of a given outcome by a specific factor. The model is a direct probability model and not a classifier. The Logistic function is given by:

  $$p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

  Note that $p(x)$ is interpreted as the probability of the dependent variable equaling a "success" or "case" rather than a failure or non-case. It's clear that the response variables Y_i are not identically distributed: $P(Y_i = 1 \mid X)$ differs from one data point X_i to another, though they are independent given design matrix X and shared parameters β[4].

- **Support Vector Classifier:** Formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high-or infinite-dimensional space, which can be used for classification, regression, or other tasks like outliers detection. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training-data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier[5].

  The original maximum-margin hyperplane algorithm proposed by Vapnik in 1963 constructed a linear classifier. However, in 1992, Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik suggested a way to create nonlinear classifiers by applying the kernel trick (originally proposed by Aizerman et al.) to maximum-margin hyperplanes[6]. The resulting algorithm is formally similar, except that every dot product is replaced by a nonlinear kernel function. This allows the algorithm to fit the maximum-margin hyperplane in a transformed feature space. The transformation may be nonlinear and the transformed space high dimensional; although the classifier is a hyperplane in the transformed feature space, it may be nonlinear in the original input space.

  Several kernels like polynomial (homogeneous and inhomogeneous), hyperbolic tangent and Gaussian radial kernel exist. We have used the Gaussian kernel in this project.

- **Decision Tree Classifier** Decision tree learning is a method commonly used in data mining. The goal is to create a model that predicts the value of a target variable based on several input variables[7]. A decision tree is a simple representation for classifying examples. A decision tree or a classification tree is a tree in which each internal (non-leaf) node is labeled with an input feature. The arcs coming from a node labeled with an input feature

are labeled with each of the possible values of the target or output feature or the arc leads to a subordinate decision node on a different input feature. Each leaf of the tree is labeled with a class or a probability distribution over the classes.

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features[8].

They are able to handle both numerical and categorical data. Other techniques are usually specialized in analyzing datasets that have only one type of variable.

- **Nave Bayes** Naive Bayes classifiers are highly scalable, requiring a number of parameters linear in the number of variables (features/predictors) in a learning problem. Maximum-likelihood training can be done by evaluating a closed-form expression, which takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers. Given a way to train a Naive Bayes classifier from labeled data, it's possible to construct a semi-supervised training algorithm that can learn from a combination of labeled and unlabeled data by running the supervised learning algorithm in a loop:

  Given a collection, $D = L \uplus U$ of labeled samples L and unlabeled samples U, start by training a naive Bayes classifier on L. Until convergence, do:

  – Predict class probabilities
  $$P(C \mid x) \forall x \in D$$
  – Re-train the model based on the probabilities (not the labels) predicted in the previous step.

  Convergence is determined based on improvement to the model likelihood
  $$P(D \mid \theta),$$
  where θ denotes the parameters of the naive Bayes model.citeninth. This training algorithm is an instance of the more general expectationmaximization algorithm (EM): the prediction step inside the loop is the E-step of EM, while the re-training of naive Bayes is the M-step. The algorithm is formally justified by the assumption that the data are generated by a mixture model, and the components of this mixture model are exactly the classes of the classification problem.

- **Stochastic Gradient Descent Classifier** The standard gradient descent algorithm updates the parameters θ of the objective function $J(\theta)$ as,
  $$\theta = \theta - \alpha \bigtriangledown_\theta E[J(\theta)]$$
  .
  where the expectation in the above equation is approximated by evaluating the cost and gradient over the full training set. Stochastic Gradient Descent (SGD) simply does away with the expectation in the update and computes the gradient of the parameters using only a single or a few training examples. The new update is given by,
  $$\theta = \theta - \alpha \bigtriangledown_\theta J(\theta; x^{(i)}, y^{(i)})$$
  with where $(x^{(i)}, y^{(i)})$ is a pair from the training set. In SGD the learning rate alpha is typically much smaller than a corresponding learning rate in batch gradient descent because there is much more variance in the update[10].

4.7 Predictions:

After fitting different models, we have calculated their accuracy, F1-scores and error rates of the predictions made by these models and compared them.

5 Experiments and Observations

5.1 Experimentation with ratio of labeled and unlabeled data

We experimented with different split ratios of the labeled and unlabeled data in the train set.

To begin with we ran a vanilla supervised SVC treating the entire data-set as labeled and observed the F-1 score, illustrated by Figure 1.

Also, Ratios of *1:1, 2:3, 3:7, 1:4, 1:9 and 1:99* were executed. We then observed the F-1 Score of the initial supervised run and then the highest F-1 score obtained after employing 3SAT algorithm on the data-set. A comparison of these two scores was drawn up and noted. We observed a noteworthy improvement from the initial run score to the 3SAT one when the ratio of split was skewed in favor of the unlabeled data, i.e. when there was significantly more unlabeled data as compared to labeled data-points[11]. There was no significant improvement when the ratio between the two was even.

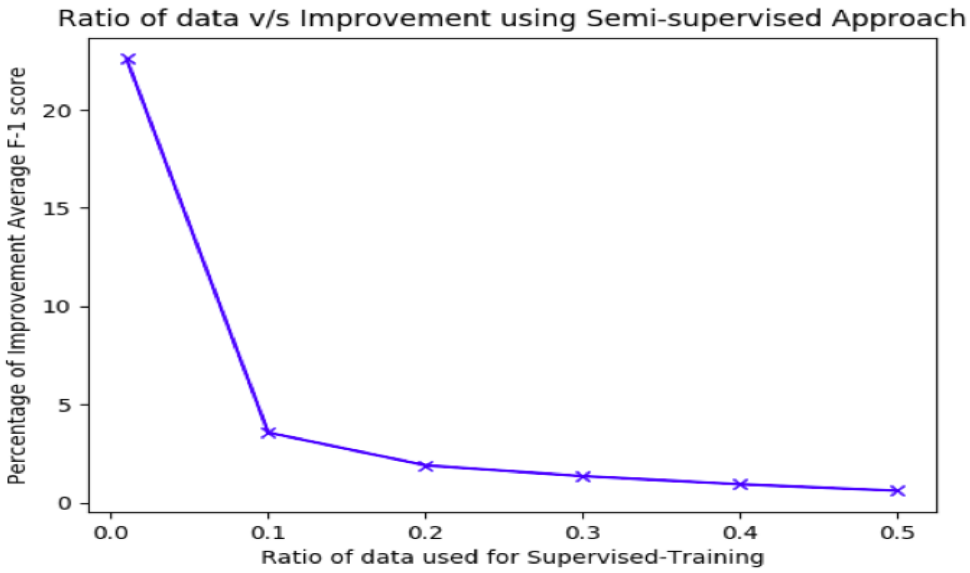

Figure 1: Relaxation in improvement from supervised to 3SAT approach across various split-points in the training data

This was expected as the model had not seen a lot of data when the ratio was say, 1:9 or 1:99 thus, the initial model was well below par, as shown in Figure 2. But through the execution of 3SAT it was able to learn and train on more diverse and extensive data-points, eventually resulting in a prodigious amp up in the performance of the model. Also, this exercise helped us in determining the accuracy and F-1 score for a range of splits in the data. This was vital in determining the optimum split of the data that would result in the best possible performance. We observed that the accuracy peaked at the ratio *9:11*, implying about *45%* data as labeled and *55%* of it as unlabeled. This is very comparable to a real-world scenario where there is not abundance of labeled data.

5.2 Exploration of different underlying models

As discussed previously, we tried out a range of different models as our initial run champion, illustrated by Figure 3. We used a 10-fold cross-validation to determine and optimize the best penalty parameter for the penalty term and the width of the decision boundary[12].

- The logistic regression model performed poorly when compared to others. This was anticipated, as a logit model yields a linear decision boundary in the actual feature space. However, the data at hand is not linearly separable in the original feature space. This is clearly explained by the performance of a Support Vector classifier with a linear kernel. Although, this model also linearly separates data but does so by expanding the dimensions using the kernel trick. This explains why SVC with a linear kernel outperforms the logistic model.
- Even though Nave Bayes classifier was able to achieve comparable accuracy to that of SVC with linear and radial kernels it had a poor precision statistic which yielded below average

Percentage of Semeval data used as initial label data	Initial-run F score of the supervised SVC	Highest F score obtain out of first 10 3SAT iterations	Relative improvement after use of 3SAT
0.5	0.5023	0.5054	0.62%
0.4	0.5047	0.5095	0.95%
0.2	0.466	0.4749	1.91%
0.1	0.4327	0.4482	3.58%
0.01	0.386	0.4732	22.59%

Figure 2: Various split-points in the training data

F-1 scores for the model. This could be due to the skewness of data towards the negative tweets resulting in a poor precision statistic of the positive tweets and eventually poor F-1 score.

- The decision tree classifier was as close to the SVC as one could get in terms of the accuracy and F-1 score. However, it took 4 times the execution time of SVM. We allowed the classifier to go as deep as possible by leaving the *max-depth* hyper-parameter None. Such a slow convergence on comparable systems was the reason we still believe SVCs do the job much more efficiently and quickly.

- The stochastic gradient descent(SGD) is essentially an SVM with some learning rate. The execution of this stochastic model results in results very much alike to those of SVC. It is noteworthy to mention that this model took a very long time to converge. We ran five epochs of this model without triggering the random state hyper-parameter. In two of those epochs the model was unable to attain global minima, a consequence of its optimization drawback.

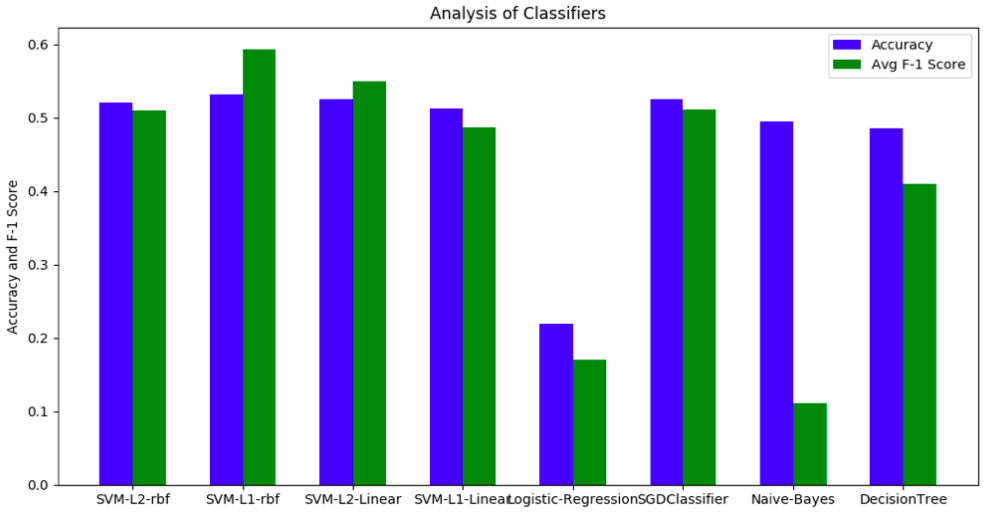

Figure 3: Comparison of all the underlying models

The lack of linear separability in data in the original feature space and the incompetence of logistic regression to mimic the kernel trick of SVC in any manner points to the fact that SVC is a much more sensible choice of classifier in this environment.

Considering the heavy computational overhead in the cases of decision tree and SGD classifier, it is safe to interpret that SVC performs more rapidly in a similar environment.

5.3 Supervised Learning vs the 3SAT approach

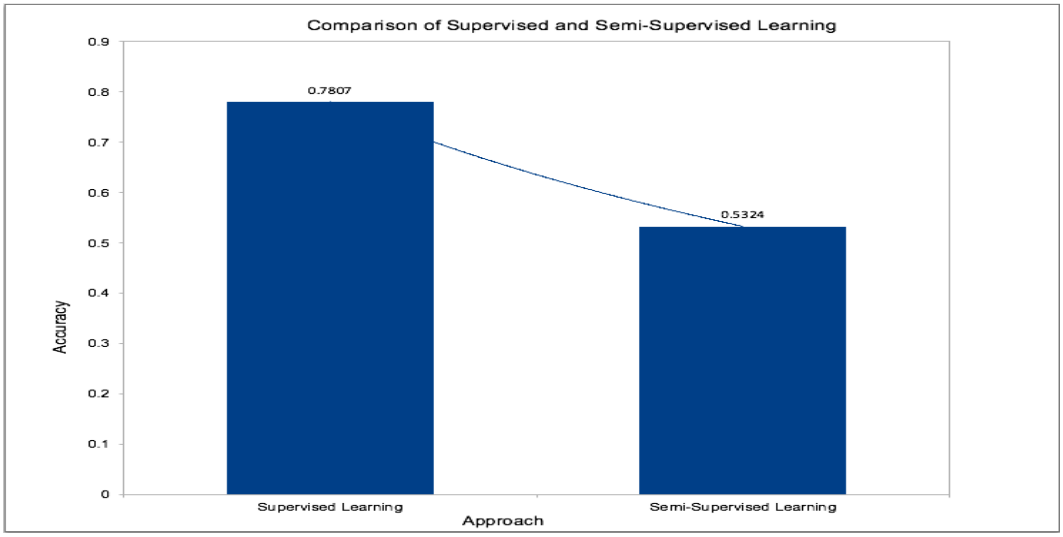

Figure 4: Performance Comparison of Supervised learning v/s 3SAT

We decided to draw a comparison of the best accuracy of 3SAT, shown in Figure 4, with the accuracy of a supervised execution. We observed that the supervised learning out-performed the 3SAT. However, reiterating to the lack of abundance of labeled data in a real-life scenario, 3SAT achieving such accuracy with a lot fewer labels is an ode to the beauty of the concept that is semi-supervised learning.

6 Conclusions

At the end of the day, some machine learning projects succeed and some fail. What makes the difference is how easily the features are determined and used. If one has many independent features that each correlate well with the class, learning is easy. On the other hand, if the class is a very complex function of the features, one may not be able to learn it. Often, the raw data is not in a form that is amenable to learning, but one can construct features from it that are. This is typically where most of the effort in a machine learning project goes. It is often also one of the most interesting parts, where intuition, creativity and *'black art'* are as important as the technical stuff. Our case was no different as we had spent a lot of time brainstorming several ways of extracting complex features from a barely simple looking sentence. The accuracy and results we obtained are a tribute to the pre-processing performed which laid the foundation for effective training of the model[13].

On performing this vigorous classification experiment, we have concluded that for the semeval data-set of tweets a Support vector classifier with radial kernel is the best underlying model for our proposed 3SAT approach[14].

As we know, there is no free lunch in machine learning tasks we had to extensively test several models for the best possible performance. We deemed Logistic Regression and Nave Bayes to be unfit approaches for the current data-set for the respective reasons, lack of linear separability in the data within the original feature space and skewed structure of the data towards negative tweets. Thus, we conclude that the Nave Bayes model does not correspond to a robust approach for the current scenario. Also, Logistic Regression is incompetent in performance due to the decision boundary

being non-linear. The Decision Tree and the SGD classifiers yielded good F-1 scores and accuracy statistic indicating their strong robustness to the data. However, the convergence was extremely slow when compared to an SVC and the trade-off in accuracy was not significant enough to use them as the underlying model; In fact the SVC yet performed better in terms of both accuracy and robustness.

For these reasons, it is safe to say that for sentiment analysis of tweets from the semeval dataset a *Support Vector Classifier* is the most scalable, robust and accurate model that can be used. We regard this as our major discovery in the process of classification.

We can also infer that a split at **45%** labeled and **55%** unlabeled data yields the best accuracy in our 3SAT methodology. This provides a sweet-spot for the split of the training data and is very comparable to a real-life scenario, where there is a plethora of unlabeled data but a scarcity of its labeled counterpart.

Our major takeaways from this project were, the time we spent in performing thorough research on semi-supervised learning methodologies, the extensive testing of several models we tested in order to determine the best possible model for our 3SAT. We gained invaluable hands-on experience in the domain of Natural language processing as well as Semi-supervised learning via this project. We discovered it's possible to quickly and automatically produce models that can analyze bigger, more complex data and deliver faster, more accurate results even on a very large scale. And by building precise models, an organization might have a better chance of identifying profitable opportunities or avoiding unknown risks.

References

[1] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of twitter data. *Proceedings of the Workshop on Languages in Social Media (LSM11). Association for Computational Linguistics*, Stroudsburg, PA, 3038.

[2] Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. DataStories at SemEval-2017 Task 4: Deep LSTM with Attention for Message-level and Topic-based Sentiment Analysis. *Proceedings of the 11th International Workshop on Semantic Evaluation.* Vancouver, Canada, SemEval 17, pages 746 753.

[3] Wesley Baugh. 2013. bwbaugh: Hierarchical sentiment analysis with partial self-training. *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013). Association for Computational Linguistics,* Atlanta, GA, 539542.

[4] Menard, Scott (2002). Applied Logistic Regression Analysis. *SAGE*. p. 91.

[5] Simon Tong, Daphne Koller: Support Vector Machine Active Learning with Applications to Text Classification. *Journal of Machine Learning Research (2001)* 45-66.

[6] Boser, Bernhard E.; Guyon, Isabelle M.; Vapnik, Vladimir N. (1992). "A training algorithm for optimal margin classifiers". *Proceedings of the fifth annual workshop on Computational learning theory* COLT '92. p. 144.

[7] Rokach, Lior; Maimon, O. (2008). Data mining with decision trees: theory and applications. *World Scientific Pub Co Inc.* ISBN 978-9812771711.

[8] Ben-Gal I. Dana A., Shkolnik N. and Singer (2014). "Efficient Construction of Decision Trees by the Dual Information Distance Method". *Quality Technology & Quantitative Management (QTQM)*, 11(1), 133-147.

[9] Rennie, J, Shih, L, Teevan J, Karger D. (2003). Tackling the poor assumptions of Naive Bayes classifiers, *ICML.*

[10] UFLDL.stanford.edu/tutorial/supervised/OptimizationStochasticGradientDescent/

[11] Blum, A. and Chawla, S. (2001). Learning from Labeled and Unlabeled Data using Graph Mincuts. *Proceedings of the Eighteenth International Conference on Machine Learning*, pp. 19-26, San Francisco, CA.

[12] Towardsdatascience.com/12-useful-things-to-know-about-machine-learning-487d3104e28

[13] Statweb.stanford.edu/ tibs/sta306bfiles/cvwrong.pdf

[14] docs.google.com/document/d/1DKNBO-1tNlBJzWs2HzH-UKUhxB8SLR6TQSEX6fyhQ5w