Abstract English

Background
As Twitter has become a global microblogging site, it's influence in the stock market has become significant. This makes tweets an interesting medium for gathering sentiment. A sentiment that might influence trends in the stock market.

Motivation
If Twitter can be used to predict trends in the stock market the casual investor would gain an advantage over the day-trader or the modern trading algorithms.

Another interesting aspect is the role of Twitter in sentiment analysis. And how Twitters role as a data source influences trends in the stock market.

Data and Experiments
Twitter is used as the data source. It provides easy access, lots of data, and many possibilities to use available metadata. To find the sentiment of a tweet we use bag of words, SVM, and Naive Bayes. For the finance part and comparison of trends we use stock data from Oslo stock exchange. We use moving average(MA) and average directional index(ADX) as trend indicators. Trend comparison is based on MA and ADX. We calculate MA and ADX with finance data and sentiment data based on tweets. Then we compare the graphs.

Findings
We explore the usage of lists of words, dictionaries, in sentiment analysis. And we look at data retrieval from Twitter and the trend we can create from it. To a varying degree we get positive results with the dictionaries, while the trend aggregation lacks the finesse and results it should have had.

Conclusion
Sentiment classification of tweets worked with both methods. We managed to aggregate a trend based on sentiment. But the comparison with the finance trend did not work out as hoped. No similarities was found between the sentiment trend and the finance trend.

Metadata

Metadata?
github repository for code an thesis.
word count for fun.

Keywords
Twitter Finance Trading Social media Sentiment analysis Moving average NLP Classification Trend detection/analysis

Word use
Monogram is in this thesis consistently used instead of *Unigram*[1].

[1] Wikipedia: https://en.wikipedia.org/wiki/N-gram

Acknowledgements

TODO Arivd + Pinar
For supervision and scientific input, gratitude is extended towards:

Supervisor Prof. Pinar...
and
co Supervisor Prof Arvid H.

Thanks are also extended to the people that proof read the thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Context

This thesis consist of three parts. First the Twitter aspect. Where we use Twitter as a data source for sentiment analysis. Second, the sentiment analysis itself. Creation of dictionaries and how we determine the sentiment of a tweet. Third, the finance part, where we look at stock market prediction[1], and technical analysis.

It is well known that people are affected by sentiment. Sentiment analysis in finance is not a new concept, but it is has not been explored very much yet. We want to see if there are ways we can predict trends with the help of sentiment analysis. We generate trends based on sentiment, and stock exchange data.

Twitter is a great source of data, and a good place to express sentiment. So we chose Twitter as our primary source of sentiment. To use the data in some smart ways we compile dictionaries from tweets. And use dictionaries for sentiment classification.

Our greatest problem is creating a sentiment trend graph based on data from tweets.

1.2 Motivation

We want to find how sentiment can be extracted from Twitter and how it effects finance trends. We are motivated by the fact that social media has become a

[1] Wikipedia: http://en.wikipedia.org/wiki/stock_market_prediction

big part of peoples lives.

Why is this work done? We believe that microblogging and social media are great places to find and observe trends, and we would like to explore that. Twitter as a social media platform is a huge place for companies to post updates and provide customer support, so it is a natural place to gather data for sentiment analysis.

We would hopefully find relations between Twitter content and financial trends, and use that to prove increased revenue in trade. Our work is relevant for further endeavors about Twitter, sentiment and it's relation to finance.

1.3 Research questions

1.3.1 How can we determine the sentiment of a tweet?

The factors we look at are how, if possible, can we extract knowledge from tweets and then use that to find the sentiment of said tweet. The usefulness of a tweet as a source for sentiment is in question. Which parts of a tweet are useful in sentiment analysis, which methods can we use to classify a tweet, and which method is best?

1.3.2 How can Twitter be used to aggregate trends?

As trends are interesting and useful we aim to create a trend based on data from Twitter. We ask ourselves if Twitter as a microblogging site can be used as a data source in trend aggregation. The credibility of a tweet is also an aspect we would like to investigate. To create a trend we need to find which parts of Twitter we can use. Combining these, hopefully, we can create trends.

1.3.3 How does trends from Twitter compare to technical analysis in the stock market?

In finance, the moving average is a method in technical analysis to indicate trends. Can methods of technical analysis or other trend defining qualities of financial data be used to predict trends? Can this and other methods of technical analysis is used to predict trends. We will look at possible applications for sentiment in the stock market.

Twitter has data such as the amount of tweets posted today, the location where tweets are posted from, and which users has posted. Which Twitter

sources are most suitable for predicting the stock market trend? Aggregated, can these data represent a trend? Previously researchers have managed to predict direction of the market the next few days based on the volume of tweets[Doukas et al., January 10, 2010].

Ultimately we will compare trends in technical analysis with trends based on sentiment from twitter.

1.4 Methods

In this thesis we have two main ways of classifying the sentiment of tweets. Word counting and training a classifier. The word counting is the known method 'bag of words'[2], and the training of classifiers use support vector machine, and Naive Bayes. The trained classifiers provide insight and comparison of which classifying method is best.

Moving average[3] and average directional index[4] are used to plot trend graphs in finance. Based on the moving average and the average directional index we create our own adaption to create sentiment trends. The trend graphs can be seen in section 6.6, on page 52. Concepts and explanations of how the trend works can be found in chapter 5.

1.5 Thesis Outline

The nine chapters of this thesis all describe different aspects of the work done.

Introduction, chapter 1, introduces the context and describes what we aim to do, and our goals.

Background and Previous work, chapter 2, presents other research in this area and relevant background knowledge.

Data, retrieval and structure, chapter 3, describes the data sources, how we use them, and the structure of the data we use.

Sentiment Classification, chapter 4, describes the methods we use to determine sentiment of tweets.

Trending, chapter 5, considers the different aspects of a trend, on twitter and in finance. How trends are aggregated and how they are used is also covered.

[2]Wikipedia: https://en.wikipedia.org/wiki/Bag_of_words_model
[3]Wikipedia: https://en.wikipedia.org/wiki/Moving_average
[4]StockCharts.com http://stockcharts.com/school/doku.php?id=chart_school:technical_indicators:average_directional_index_adx

Experiments, chapter 6, contains the specifics of how we use our methods, and how we test the research questions.

Results and Discussion, chapter 7, presents what we find with our experiments, and discuss the results we get.

Conclusion, chapter 8, concludes this thesis and highlights the findings.

Future Work, chapter 9, covers the next logical steps in sentiment analysis with Twitter, and trends.

Also noteworthy is appendix A, where we have a detailed description of the code.

Chapter 2

Background and Previous Work

Previous work and the background information needed for this thesis is described in this chapter. We look at how microblogging activity can be related to finance and stock market prediction Bollen et al. [2011], and how sentiment can be used to predict stock markets.

This chapter contains related research, context knowledge and introduction to the subjects discussed in this thesis. Twitter as a microblogging platform is introduced in section 2.1. Section 2.2, describes sentiment analysis and what it can be used for. Finance and trading is described in section 2.3. In the last section, 2.4, of this chapter trends are introduces.

2.1 Twitter

Twitter is a social informations network. It's a real-time service for sharing and gathering small messages. These messages can represent everything from a persons opinion of ice cream, to the latest changes in the financial market or pictures from a Mars rover.

At the core of Twitter[1] you have the Tweet. The Tweet is a 140 character message. Tweets lets you communicate with other users, share photos and post all kinds of information. The small size of the tweets are not a hindrance for

[1] About Twitter: https://twitter.com/about

the flow of information.

The fast growing messaging service handles 1.6 billion search queries every day. In 2012 the 500 million users would generate 3.2 queries each, every day. 340 million tweets were posted every day. [2]

Most medium and large companies have a presence on Twitter today. Posts can contain any type of information, from promotional content to service status to financial reports. [Jubbega, 2011, p8] says that 77 of the Fortune 100 companies have a Twitter account.

Companies use Twitter to communicate with customers. Customers can post questions and feedback, while companies posts answers and information. Questions can be asked with a specific hashtag(#). Or with an at(@) sign to target a specific user. This makes it easy to filter the messages, and therefore easier to get in contact with the customer. The company Best Buy demonstrated the successfulness of twitter in customer relations by answering questions with a specific hashtag. In 2009 they had answered nearly 20 thousand questions using twitter. [Li and Li, 2013, p1] Market Intelligence is also a major aspect of the microblogging sphere.

Twitter represents one of the largest and most dynamic datasets of user generated content. Along with Facebook Twitter data is in real time. This has major implications for anyone who are interested in sentiment, public opinion or customer interaction. [Speriosu et al., 2011]

A typical tweet contains about 11 words and provides an opinion or state of mind or a piece of information. Tweets can contain hashtags: '#something', Twitter handles: '@username', or other adaptations of prefixes such as '$STO' which represents a stock. Different prefixes or tags ($, #, @) can easily distinguish content of a tweet. This also makes it easier to search and classify the content of tweets. Figure:2.1 and figure:2.2 are examples of tweets as shown on Twitter.

The retrieval of tweets seems like a challenge. But Twitter has made this easy by providing an API[3]. With the API you can write tweets and update the status of a user. But the best part of the API is that it provides search capabilities. To get a certain subset of all tweets, we can use the search function and view only the tweets we want.

On the front page of Twitter we have the search function at the top right of the page. The search provides the ability to specify which types of tweets you want. And gives you the opportunity to find the information you are looking

[2] Wikipedia: http://en.wikipedia.org/wiki/Twitter
[3] API: Application programming interface

for.

Figure 2.1: Typical tweet from Twitter.

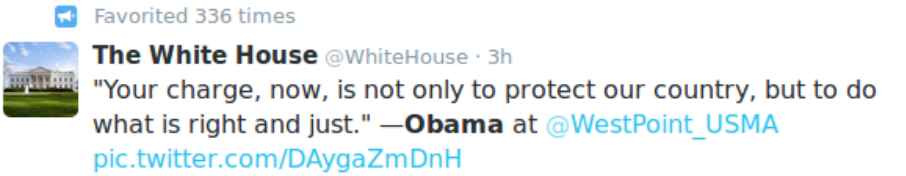

Figure 2.2: Typical tweet from Twitter.

2.2 Sentiment

Opinion mining on the web is not new. In recent years it has become attractive to traders. The use of Twitter and social media is increasing. Both in business and with private users. This means a surplus of raw data with easy access. Companies all over the world has started to use social networks to their benefit. The use of information from social media has become part of the trend, although there are some drawbacks and shortcomings. Noise and garbage is one of them. Even if you're right 80% of the time, the last 20% can prove devastating. [Stevenson, 2012]

Sentiment broadly refers to a persons state of mind. Based on the opinion the person will do optimistic or pessimistic choices. A positive mindset leads to optimistic judgements of future events, and a negative state of mind leads to pessimistic choices. [Doukas et al., January 10, 2010, p4]

The users may have different roles and intentions in different communities in the microblogging sphere, [Java et al., 2007]. A users intentions and its reasons for participation might be a factor in the sentiment analysis.

2.2.1 What is Sentiment Analysis

There are two main categories of approaches to sentiment detection. The first is to use a classifier. The classifier can use methods such as Naive Bayes, maximum entropy or support vector machine [Li and Li, 2013]. These classifiers are typical examples where it would be natural to use machine learning on evolutionary algorithms to increase the classification correctness over time. The second is the use of linguistic resources, such as corpora of negative and positive words. The developed linguistic resources are used to classify the sentiment of text, [Li and Li, 2013].

Li and Li has created a framework for sentiment analysis. The system consists of four main steps and is tested with experiments on twitter. First they do topic detection, identifying and extracting the topics mentioned in the tweet. Secondly opinions are classified. The polarity, how positive a tweets is, of the opinion is decided and the users impression is captured. Third, credibility is assessed. This creates a better summarization of the expresser's credibility. Fourth, step one, two, and three are aggregated to reflect the true opinion and point of view. Combining the first three steps, in the fourth, results in a truer reflection of the expresser's opinion. [Li and Li, 2013]

One way of classifying tweets is to use predefined lexicons of positive and negative words. Consumer confidence and fluctuations of voting polls can be tracked this way [Connor et al., 2010].

The work of [Diakopoulos and Shamma, 2010] describes a methodology for better understanding of temporal dynamics of sentiment. The system uses visual representation to achieve this. This is investigated in the reaction to debate video. Further [Diakopoulos and Shamma, 2010] detects sentiment pulse and controversial topics with the help of visualisation and metrics. [Diakopoulos and Shamma, 2010] used crowdsourcing[4] to classify batches of tweets. This was accomplished with Amazon Mechanical Turk, a crowdsourcing site[5].

[Barbosa and Feng, 2010] explores the problem of noise in biased and noisy data. They focus on noisy labels and add features to the tweets to increase the classification properties of the tweets. Objective tweets have very little sentiment or no sentiment at all. To improve classification, tweets are classified as objective of subjective. Then the subjective tweets, that have a sentiment are classified as positive of negative.

[4] Crowdsourcing is the practice of obtaining needed services, ideas, or content by soliciting contributions from a large group of people, and especially from an online community, rather than from traditional employees or suppliers. http://en.wikipedia.org/wiki/Crowdsourcing

[5] Amazon Mechanical Turk (AMT): https://www.mturk.com/mturk/

Classification of tweets can be generalised by using features. Features are small elements of a tweet, such as monograms[6], bigrams, or part-of-speech tags. Bigrams are words consisting of two words, monograms are words consisting of one word. An abstract representation of a tweet would be beneficial to the classification. In this abstract representation [Barbosa and Feng, 2010] propose to use characteristics about how tweets are written and meta-information about the words in tweets, meta-features and tweet syntax features that can further improve classification. Meta-features are information about the tweet, such as location, language, and number of retweets. Retweets are tweets that are posted again by other users. The tweet syntax features are things such as hashtags, retweet, reply, links, punctuation and emoticons [Barbosa and Feng, 2010].

Challenges with sentiment and Twitter is described in another approach by [Becker et al., 2013]. They explore techniques for contextual polarity disambiguation and message polarity classification. Constrained and supervised learning is used to create models for classification. They describe a system that solves these tasks with the help of polarity lexicons and dependency parsers. Expanded vocabulary is one of the main aspects of their success, as they say in their findings: "We hypothesize this performance is largely due to the expanded vocabulary obtained via unlabeled data and the richer syntactic context captured with dependency path representations." [Becker et al., 2013]

In contrast to [Becker et al., 2013], [Speriosu et al., 2011] has used distant supervision and labeled propagation on a graph based data structure. The data structure represents users with tweets as nodes. And tweets with bigrams, unigrams, hashtags, etc as subnodes of the tweets. A label propagation approach rivals a model supervised with in-domain annotated tweets, and outperforms the noisily supervised classifier, and a lexicon-based polarity ratio classifier. [Speriosu et al., 2011]

2.2.2 Sentiment Analysis in Finance

[Brown and Cliff, 2004, p2] writes the following on over-reaction of investors: "*He(Siegel (1992)) concludes that shifts in investor sentiment are correlated with market returns around the crash. Intuitively, sentiment represents the expectations of market participants relative to a norm: a bullish (bearish) investor expects returns to be above (below) average, whatever "average" may be.*". In the light of recent changes in the financial world, and the use of sentiment from social media, the notion that opinions and sentiment of investors and market

[6]Monogram, same as Unigram

actors affect the market is not a new observation.

Use of sentiment can potentially predict changes and trends in the market. Bad news in an optimistic period creates cognitive dissonance in the small investors. This impacts the market by slowing down the selling rate of loosing stocks, [Doukas et al., January 10, 2010, p29]. Further we can see that optimistic sentiment has a 2% monthly average return. While the investor sentiment is pessimistic we see a drastic reduction in returns. Down to 0.34%,[Doukas et al., January 10, 2010, p5]. After optimistic periods it is indicated that the monthly return is reduced to -0.49%. On the contrary there is no equivalent change after a pessimistic period, [Doukas et al., January 10, 2010, p6-7]. Momentum profits are only significant when the sentiment is optimistic, [Doukas et al., January 10, 2010, p29].

Hope and fear is used by [Zhang et al., 2011] to decide the movement of the market. The sentiment is aggregated to be hopeful or fearful. This focuses on positivity and negativity of the sentiment of that particular day. The daily sentiment is then compared to the market indicators of the same day to create a prediction of the market. [Zhang et al., 2011] finds that calm times give little hope or other emotions. Little turmoil results in few fluctuations in the market, and opposite, lots of emotions(hope, worry, fear), gives speed to the market.

[Brown and Cliff, 2004, p3] indicates that the sentiment does not cause subsequent market returns. For a short-term market timing this is bad news. However with the changes in social media over the last decade how is the situation today? With the microblogging sphere of today we can easily see the correlation of sentiment and the market indicators, [Jubbega, 2011]. But does the sentiment cause changes in the market-return? [Brown and Cliff, 2004, p3] also says that optimism is associated with overvaluation and subsequent low returns.

[Brown and Cliff, 2004, p] concludes that 'aggregated sentiment measures' has strong co-movement with changes in the market. He also indicates that sentiment doesn't appear to be a good trading strategy. This, in the view of [Zhang et al., 2011], indicates a leap in sentiment research and what is possible with the microblogging today.

2.3 Finance and Trading

The management of assets or liabilities and the management of funds over a period of time is called *Finance*. In finance the valuation of assets are time dependant. The value of an asses is always changing and might not have the

same value five minutes from now, as it does now. Assets are priced based on expected returns and risk level. The three sub categories of finance are: personal, corporate and public[7]. These categories describes very different parts of the financial world.

Trading is the action of buying or selling financial instruments. Financial instruments can be stocks, bonds, derivatives or commodities[8]. Trades take place in markets, stock markets, derivatives markets or commodity markets.

A new aspect to trading in the last decade has been online trading[9]. Speed, ease of use, and low costs made the online brokers popular. Many brokers provide platforms for trade and analysis to select potential investments.

The tools provided are often some form of technical analysis[10]. Technical analysis is the study of past market data. Mostly volume and price. The purpose of technical analysis is to forecast the direction of prices.

Among tools and techniques in technical analysis are charts, market indicators, and relative strength index. It is also quite common to combine techniques to acquire better predictions. When looking at put/call ratios short interest, implied volatility, and bull/bear indicators of sentiment is important to take into consideration.

Technical analysis focus at numeric values, such as volume and price, where fundamental analysis[11] analyse company health, financial statements, production rates, earnings, management, and competitive advantages. Day traders prefer technical analysis over financial analysis.

Two principles of technical analysis are: 'History repeats itself', and 'Prices move in trends'. 'History repeats itself' refers to the belief that traders will do the same actions again and again. Technicians believe that the repeated behaviour can be recognized as a pattern and be observed on a chart. 'Prices move in trends' is the belief that the price of a commodity will move directionally over a period of time. Relative highs and relative lows are indicators of a trend. Consecutive lower highs indicates a downward trend.

An interesting aspect of trading and finance is the behavioral one[12]. Behavioral finance is the field of research that study the effects of cognitive, social, emotional and psychological factors of economic decisions. It also includes the consequences of resource allocation, market price and returns.

[7] Wikipedia: http://en.wikipedia.org/wiki/Finance
[8] Wikipedia: http://en.wikipedia.org/wiki/Trader_(finance)
[9] Wikipedia: https://en.wikipedia.org/wiki/Online_trading
[10] Wikipedia: http://en.wikipedia.org/wiki/Technical_analysis
[11] Wikipedia: http://en.wikipedia.org/wiki/Fundamental_analysis
[12] Wikipedia: https://en.wikipedia.org/wiki/Behavioral_finance

2.4 The Trend

The trend is the general opinion of the masses. As defined by the Free Dictionary: "The direction and momentum of a market, price, economy, or other measure. For example, if the price of a security is going mainly downward with only a few gains, it is said to be on a downward trend. Identifying and predicting trends is important for finding the right moment to buy or sell securities. Trends are especially important in technical analysis, which recommends buying at the bottom of a downward trend and selling at the top of an upward trend."[13]

Trends work in much the same way as opinions. People are affected by their environment all the time, and are often influenced by trends and opinions. When people are affected by trends they start to move in the same direction as others. The first group of people that move in the same direction are called trend setters. They are the people that show others how the trend works and what this trend is about.

On Twitter we have lots of subcultures that all express themselves on their specific topic. It can be technology, art, finance, or fashion among others. In the sense of Twitter we can look at the content of messages and see if we can find common topics that people talk about. This is the topic of a subculture or a subspace of twitter. To get the trend we have to look at the content of the messages in a subspace, given that a trend is the combined general opinion of a group. We can analyse the group and see if we can find certain topics or areas of interest that aggregates to a trend.

Stock market prediction[14] is the act of trying to predict or determine the future value of a stock. A way to do this is to look for trends in the data. The trend is a tendency of movement in a particular direction for a financial market.

There are three categories of trends in finance. Primary, secondary and secular trends. Where primary trends have a medium time frame, secondary trends have a short time frame, and secular trends has long time frames[15]. Bull market and bear market are concepts that, respectively, describe upward and downward market trends. Trends are often found by using technical analysis.

Secular trends are trends that last between 5 and 25 years. A secular bull market consists of many large bull markets and many small bear markets. Primary trends last a year or more. We can also observe market tops and bottoms here. These are trend reversal points. Secondary trends has a duration of a

[13] Dictionary description of trend: http://financial-dictionary.thefreedictionary.com/Trend

[14] Wikipedia: http://en.wikipedia.org/wiki/Stock_market_prediction

[15] Wikipedia: http://en.wikipedia.org/wiki/Market_trend

few weeks or months. The secondary market trend is change in price direction within a primary trend. The small changes are often called market corrections. The short term correction is often between 5 and 20 percent.

When looking for usage of Twitter trends, we find little to confirm previous research in this area.

Chapter 3

Data, retrieval and structure

This chapter describe the structure, the characteristics, the metadata and usage of the data used in this thesis.

Twitter and tweets, section 3.1, describe how we use tweets, what tweets to use and how we acquire them.

We have a section describing the dictionaries, section 3.2. How we compile dictionaries from tweets and how we use them are covered there. Shortcomings and the possible improvements are also discussed briefly.

And last we have a section that describes the financial data we use, section 3.3. Where we get them, the structure and how we use the data is covered here.

3.1 Tweets

A tweet is a massage posted on twitter. The message in many ways resemble the well known SMS[1].

Tweets are posted to the users profile. When a user post an existing tweet again, it is referred to as a retweet.

All users can follow other users on Twitter. Tweets from users you follow will appear in your stream of tweets on Twitter. The stream is the collection of tweets from users you are following.

[1] Wikipedia on Short Messaging Service: https://en.wikipedia.org/wiki/Short_Message_Service

3.1.1 Tweet Structure

Structure
There are a lot of metedata in the tweets. In fact most of the data in a tweet object is metadata.

The data we acquire from Twitter is in the JSON data format. JSON or *JavaScript Object Notation* is an open standard format that uses human-readable text to transmit data objects consisting of attribute–value pairs[2].

For an example of the data structure of a tweet, see appendix C.

Content
A tweet is an astonishing compilation of data about who, where and when a tweet was posted.

The main content of a tweet is the message text itself. With the content we have fields for all the links, all the emoticons, and all hashtags that are present in a tweet.

Every tweet is posted by a user. All the data of a user is also present for each tweet. Here we have data on follower count, profile images, friend count, time zone, and many other profile related subjects.

For the sharing of a single tweet we have data fields such as 'favorite_count' and 'user_mentions'. We also have 'favorited' and 'retweet_count'.

```
u'in_reply_to_user_id': None,
u'retweet_count': 0,
u'id_str': u'390051769780142080',
u'favorited': False,
u'favorite_count': 0,
u'user_mentions': [
],
```

In addition we have the location of a given tweet. Where the tweet was posted, the name of the place, the coordinates of the tweet, the country, and the id of this place.

```
u'place': {
    u'full_name': u'Stavanger, Rogaland',
    u'url': u'https://api.twitter.com/1.1/geo/id/dee2255bd015b52c.json',
    u'country': u'Norway',
```

[2]Wikipedia on JSON: https://en.wikipedia.org/wiki/Json

```
    u'place_type': u'city',
    u'bounding_box': {
      u'type': u'Polygon',
      u'coordinates': [
        [
          [
            5.5655417,
            58.884420999999996
          ],
          [
            5.8687141,
            58.884420999999996
          ],
          [
            5.8687141,
            59.0608787
          ],
          [
            5.5655417,
            59.0608787
          ]
        ]
      ]
    },
    u'contained_within': [
    ],
    u'country_code': u'NO',
    u'attributes': {
    },
    u'id': u'dee2255bd015b52c',
    u'name': u'Stavanger'
  },
```


See all the different metadata types in appendix: C.

3.1.2 Twitter API

The Twitter API is a convenient way for lots of people to access data from twitter. Tweets, streams, timelines, profiles and more are available through the

API.

To provide easy access and conformity to industry standards, the API provides data in the JSON format.

While the API does not give access to 100% of the data from twitter, it gives a good representation of the tweets from the last 7 days.

Setup

To get access to the API there are a few requirements. You must have a twitter account, and the application to be used has to be registered with Twitter. Registering the application gives access to API keys. Then you have to used the keys to authenticate with Twitter before you access the API.

For a simple guide to this we have http://datascienceandprogramming.wordpress.com/2013/05/14/twitter-api/ as a good example.

The 4 authentication tokens you get from Twitter, app_key, app_secret, oauth_token, and oauth_token_secret, is used with the Twython library[3], as described below.

A simple example of how to use the API with Twython is shown below.

- Authenticate towards Twitter.
- Execute search query on Twitter.
- Print ID's of all retrieved tweets.

The code of the example is as follows:

```
1 twitter = Twython(APP_KEY, APP_SECRET, OAUTH_TOKEN, OAUTH_TOKEN_SECRET)
2 results = twitter.search(q='Search query', count='15')
3
4 for status in results['statuses']:
5     print status['id']
```

For more advanced use we have generators, and lots of parameters and API endpoints to use. Endpoints and search parameters will be described under section 3.1.2, *API endpoints options*. The Twyhon framework and its advanced usage can be explored more in the code and in the documentation of the framework[4].

[3] https://pypi.python.org/pypi/twython/

[4] https://twython.readthedocs.org/en/latest/usage/advanced_usage.html

Restrictions
The API has some access restrictions, or rather rate limitations. This is to be expected, as unlimited access would cripple the API.

Twitter limits request of a particular kind to 180 requests per 15 minutes. Which means that we can do 180 searches spread evenly over the time interval. Or we could do 180 requests as fast as possible and then wait. The rate limits can be explored thoroughly in the Twitter documentation[5]

As for the practical implications of the limitations, they are not a problem in our case. We get more then enough data. By using a generator and pour out tweets we get 1000 tweets in about 60 seconds. But then we have to wait 15 minutes. This is suboptimal. As we will crash the program at access denied from the API.

API endpoints
Twitter has made a lot of different endpoints available. The endpoints are divided into these categories: *Timelines*, *Tweets*, *Search*, *Streaming*, *Direct Messages*, *Friends & Followers*, *Users*, *Suggested Users*, *Favorites*, *Lists*, *Saved Searches*, *Places & Geo*, *Trends*, *Spam Reporting*, *OAuth*, and *Help*.

Of these categories we mainly use *OAuth*, *Help* and *Search*. Listing the endpoints used with parameters we get the list on page 20.

Searching
To perform a search we use the parameters described in the *API endpoints* table in section 3.1.2.

The query is a normal search string where Twitter will find tweets containing all words in the string, a boolean search.

```
1 query = "Finance Increase"
```

Further we can expand the search by using logic.

```
1 query = "Finance OR Investment AND Economy OR Growth"
```

The following query is the one used to create the original tweet sets. The original tweet sets are later used as the basis for the dictionaries. We get all tweets containing one of the words: *Finance*, *Investment*, *Economy*, and *Growth*.

```
1 query = "Finance OR Investment OR Economy OR Growth"
```

[5]Twitter: https://dev.twitter.com/docs/rate-limiting/1.1

Table 3.1: Used Twitter API endpoints

Category	Parameter	Description
search	-	Used for acquisition of tweets. A query can take the following parameters:
	q	A UTF-8, URL-encoded search query of 1,000 characters maximum, including operators. Queries may additionally be limited by complexity.
	count	The amount of tweets acquired in each request. Standard = 15, max = 100.
	geocode	Get tweets close to these coordinates.
	lang	The language we want tweets in. Very limited by the number of people that speaks that language.
	locale	Language specific. If the query is in Norwegian we get tweets in Norwegian.
	result_type	mixed, recent or popular. This is the general mix of tweets returned in a search. Recent is the newest tweets, while popular are tweets that are retweeted a lot and tweets from users with many followers.
	until	Gets tweets before the given time.
	since_id	We get tweets posted after the given time.
	max_id	We get tweets with ids lower then the one given.
	include_entities	This parameter has no practical application to us.
	callback	An optional place where Twitter can post tweets back to us.
authenticate	-	How we login and get access to the API.
	oauth_token	It is the authentication code or password to access the API.
rate_limit_status	-	The way we know if we are still inside the request limitations.
	resources	The elements that we want the status of. Currently that is search and help.

Adding other parameters such as count and language we can further improve our search. To execute such a search we get Python code like the following, where 'results' is a data structure containing tweets.

```
query = "Finance OR Investment OR Economy OR Growth"
results = twitter.search(q=query, count='15', language='no')
```

Mining optimization

The acquisition of tweets, the mining operation, was successful. We got lots of tweets. But we ran into a problem. Retweets. In some cases we got up to 90% retweets in a mining session. Many of the tweets being essentially the same one, only retweeted multiple times.

When we are using tweets to create dictionaries, duplicate data is a problem. Retweets are mostly duplicate data, so we removed them to increase the quality of the tweet set.

As nearly all retweets start with 'RT' we can easily sort them out. Then we get a query like this.

```
query = "Finance OR Investment OR Economy OR Growth AND -RT"
```

When using the twython framework and its cursor function we get a continuous stream of tweets. A problem with this is that twython's cursor basically executes multiple searches. Thus yielding the same tweets multiple times, unless you change the search. So we change the search to accommodate this and use the *max_id* parameter.

```
query = "Finance OR Investment OR Economy OR Growth"
results = twitter.cursor(
                twitter.search,
                q=query,
                count="100",
                lauage=language,
                max_id='the id of the last tweet')

for result in results:
    print result
```

API Caveats

The main caveats of the Twitter API are the request limitations and the limitations of the search engine.

As Twitter says: '*Please note that Twitter's search service and, by extension, the Search API is not meant to be an exhaustive source of Tweets. Not all Tweets will be indexed or made available via the search interface.*' and '*The limitations of the search engine of Twitter indexes only about a weeks worth of tweets.*'[6]

Although this is not a big problem, coding and data acquisition could have been simpler. The solution for the week limitation was to broaden the search to include more words. This resulted in a more varied dataset, and more tweets. We initially wanted to analyse tweets related to finance, so the diversity of the dataset could be a problem.

One caveat of the API is the rate limitations. This means we have to mine tweets over time. We should have set up a server and mined tweets with a cron[7] job every 15 minutes.

3.1.3 Tweet sets

We ended up with two datasets to be used. The Obama tweetset[8], which is a set of tweets containing around 1300 tweets about Obama and the election of 2008/2009. And a self compile dataset, referenced as the *Kiro* dataset, based on the words: *Finance*, *Investment*, *Economy*, and *Growth*.

In future work it would be natural to connect the search for tweets towards stocks in a better way. Another interesting thing would be to analyse how the Obama tweet set can be used to predict the stock market after the election. And see if we could extract trends or other useful finance related information.

Search words

The Kiro dataset is based on a four word query. The dataset represents a very limited part of Twitter. Neither does the search words represent a full range of finance words.

An improvement would use a wide variety of finance words to mine tweets. This would improve the relation to finance and the relevance of the dictionaries.

Structure

The self compiled datsets has one tweet per line. This is the JSON data object that is automatically imported into Python.

[6] Twitter: https://dev.twitter.com/docs/faq#8650

[7] Wikipedia on Cron: https://en.wikipedia.org/wiki/Cron

[8] Neal Caren of University of North Carolina. Tweet file: http://www.unc.edu/~ncaren/haphazard/obama_tweets.txt

The Obama tweet set only have the tweet text. We have one tweet per line. And no metadata present.

Caveats
Obama tweets are not ideal for sentiment analysis. It has to much political content to be easily comprehended. A political statement is not positive or negative. It is positive in the eyes on some and negative in the eyes of others. And there are people that think it is neutral because they do not care. Retweets are also present, so the actual data we get out of the tweet set is limited.

The Kiro dataset has a lot of retweets and neutral tweets in it. Therefore we have only used positive and negative tweets, ignoring all the neutral ones. This gives us more relevant data to work with and less noise. Although we should have used the neutral tweets to improve the dictionaries.

3.1.4 Trend Data

When mining larger sets of tweets, the rate limitations and week limitation is a challenge. The rate limitation defines how many queries we can do towards the API every 15 minutes. The week limitation is on the search index of the API, the search engine only indexes tweets a week back in time and not all tweets are indexed. The mining itself is quite easy. Just execute a search and store all the new unique tweets. The problem appear when we need huge amounts of tweets and a setup for continuous mining.

Each request to Twitter can return 100 tweets. This gives us a theoretical maximum of 1800 tweets per 15 min. Then the fact that we might already have the tweets we receive from Twitter, and that we might not get 100 tweets per query has an effect. In addition the Twitter API only indexes tweets a week back, forcing us to mine tweets over a longer period of time.

As we do not have a server to do the mining for us, we did it manually every now and then. Resulting in very few tweets per day ratio. An example of output from our mining script:

```
@chevron
Info -- Found 18 new tweets
Info -- Metadata file created, containing 1627 tweets
```

To get a broad search we have a list of search words. The words are mostly usernames, but also some hashtags and other words. A drawback with the search words are that they might not resemble the area of research, finance. A subset of the search words are:

@annaljunggren	@industrienergi	$sto
@ap_energi_miljo	@industrienergiu	stock
@asmundaukrust	@iraqoil_gas	@svheikki
@bp_america	@janezpotocnikeu	@tageerlend
@chedegaardeu	@knebben	@_tekna_
@chevron	@kobbaen	@tekniskmuseum
decrease	@linetrezz	@terjeaa
@dn_no	market	@theoilcouncil

The trend tweets are stored in files named with the search word. Each word having it's own file with tweets. Before using the trend tweets they are sorted by date. Example from the folder shows the structure of filenames. All the files with 'trend-' prefix contain tweets sorted by date.

economy.meta	@nikolaiastrup.meta	trend-Apr-20
@eiagov	@_nito_	trend-Apr-21
@eiagov.meta	@_nito_.meta	trend-Apr-22
@eirik_milde	@norskindustri	trend-Apr-23
@eirik_milde.meta	@norskindustri.meta	trend-Apr-24
@eirinolsen	@norskoljeoggass	trend-Apr-25
@eirinolsen.meta	@norskoljeoggass.meta	trend-Apr-26
@elonmusk	norway	trend-Apr-27
@elonmusk.meta	norway.meta	trend-Apr-28
@energyindepth	@oeddep	trend-Apr-29
@energyindepth.meta	@oeddep.meta	trend-Apr-30
@enr_gop	@ogjonline	trend-May-01
@enr_gop.meta	@ogjonline.meta	trend-May-02
@erlendjordal	@oilandgasiq	trend-May-03

All the used search words are stored in the file: '_search-terms'[9]. Most of the search words are based on an article[10] from 'Teknisk Ukeblad' where they list the Twitter handles that are most significant for the oil industry.

The main part of the Norwegian economy is based on oil and gas exports. This means that other parts of the Norwegian economy is dependent on the oil industry in many ways. As an example, we have all the service industry around oil and gas, such as food supply, air travel, financial services, etc. If the oil industry has a hard time, all the dependent companies will have a hard time to.

[9]Search word file: https://github.com/magnuskiro/master/blob/master/code/trend/_search-terms

[10]Teknisk Ukeblad: http://www.tu.no/petroleum/2014/04/05/dette-er-de-viktigste-twitrerne-for-oljebransjen

This leads us to conclude that OSEBX, the Oslo stock exchange, is largely dependant on the oil industry. This makes our choice of tweets reasonable.

3.1.5 Problems, Shortcomings, and Improvements

Retweets are a source of concern, they provide no new information. But they can provide information about impact and the importance of a given tweet. Retweets should be investigated more thoroughly in the future.

A shortcoming of the data mining is the search words. Are the words representative? Do we get good data or not? Are there other words that are better suited to get accurate results? A wide array of tests and analysis should be done to remove the problem.

3.2 Dictionaries

The dictionaries are lists of words used in the classification process. In each pair of dictionaries there are a list of positive words and a list of negative words.

The dictionaries are self compiled or downloaded. The downloaded dictionaries are very specific in their area. The LoughranMcDonald dictionaries only contain financial words, while the self compiled dictionaries contains words of all sorts.

The purpose of the dictionaries are to separate groups of words, and further to give a quantitative way to classify positive and negative tweets. We also want to look at the quality of the different dictionaries. And the method for dictionary compilation.

Monograms are consisting of one word. 'This', 'That' and 'Finance' are examples. Bigrams consist of two words. 'Apple cake' and 'Operatin system' are two examples. For trigrams we have three words to a word, 'big blue boots', 'fast lane car', and 'waterfall boat accident'.

We use the dictionaries to count mono-, bi-, and tri-grams. And we can say something about the quality of the dictionaries and the method for compilation.

The dictionaries are used in both classification methods, the simple word_count(4.2.1) classification and the more advanced classification using SVM(4.2.3) and Naive bayes(4.2.3) classifiers.

3.2.1 Downloaded Dictionaries

The downloaded dictionaries are dictionaries found on the Internet. They are compiled by others, and their quality can be questioned. The most significant feature of the downloaded dictionaries are that they contain words from a certain domain. The Obama dictionary contains words linked to politics, while the LoughranMcDonald dictionary only has words from the financial domain.

Obama
The Obama dictionary was created for the Obama tweet set[11] in relation to the us presidential election of 2008.

In the positive list[12] there are 2230 words. Whether or not the frequency of political words are higher than other types of words are uncertain. *Wisdom*, *truthful*, *profit*, and *intact* are words found in the positive list. Neither *intact* nor *wisdom* are words that can be described as positive or negative. This gives a clear indication that the dictionary can be improved.

There are 3905 negative words in the list of negative words[13]. The negative list include words such as *decrease*, *worried* and *tricky*. Duplicate words are also present, so some sort of improvement of this dictionary should be done.

Loughran & McDonald
Tim Loughran and Bill McDonald has a set of dictionaries available from the websites of University of Notre Dame[14].

These dictionaries have a lot of potential. We only use the positive and negative lists of words. The other lists could be used for a better measure of polarity or weighting of n-grams. The best use of this dictionary is for comparison of the different dictionaries.

3.2.2 Compiled Dictionaries

The compiled dictionaries are based on the two manually labeled tweet sets. The Kiro tweet set, and the Obama tweet set. We compiled three dictionaries

[11] Neal Caren of University of North Carolina. Tweet file: http://www.unc.edu/~ncaren/haphazard/obama_tweets.txt
[12] Neal Caren of University of North Carolina. Positive words: http://www.unc.edu/~ncaren/haphazard/positive.txt
[13] Neal Caren of University of North Carolina. Negative words: http://www.unc.edu/~ncaren/haphazard/negative.txt
[14] Bill McDonald, University of Notre Dame: http://www3.nd.edu/~mcdonald/Word_Lists.html

Table 3.2: LoughranMcDonald available dictionaries

Negative words	General list of negative words.
Positive words	General list of positive words.
Uncertainty words	Words like *may*, *maybe*, and *nearly*. Words that flag content to have no concrete sentiment.
Litigious words	Law related words, not much use for us.
Modal words strong	Strong descriptive words, such as *Always*, and *Strongly*
Modal words weak	Weak words on moods, such as *Somewhat*, and *Depends*.

from each dataset.

Details about the process of manually classifying tweets can be found in section 4.1.

Dictionary Compilation
The compilation of a dictionary is quite simple.

- 1: We import manually labeled tweets
- 2: We take all the positive tweets and extract words from them. This constitutes the positive words list.
- 3: We repeat step 2 with the negative tweets. This constitutes the negative words list.
- 4: We remove words that are present in both the positive and negative lists of words. The removal of duplicate words results in two dictionaries of unique words, either positive or negative.

Characteristics
For both datasets we compiled mono-, bi-, and trigrams. Giving us 6 sets of compiled dictionaries to test. All dictionaries has the drawback of personal bias. The personality of the person labeling the dataset also effects the dictionaries.

3.2.3 List of Dictionaries

Table 3.2.3, lists all the dictionaries used. For each entry in the table there is a dictionary of negative words and a dictionary of positive words.

Table 3.3: List of used Dictionaries

Name of dictionary	Description
Downloaded:	
Obama original	Monograms, in relation to the Obama tweet set.
LoughranMcDonald	Monograms, acquired from Bill McDonald's webpage.
Combined Obama original and LoughranMcDonald	Monogram. A combination of the previous two dictionaries.
Compiled:	
Kiro, Monogram	Compiled from the Kiro dataset. Containing monograms.
Kiro, Bigram	Containing words consisting of two separate words. (*an example*)
Kiro, Trigram	Containing words consisting of three separate trigrams. (*example of trigram*)
Obama, Monogram	Compile from the Obama tweet set, monograms.
Obama, Bigram	Containing words consisting of two separate words. (*another example*)
Obama, Trigram	Containing words consisting of three separate trigrams. (*also an example*)

3.3 Finance Data

The finance data consists of records from Oslo stock exchange, OSEBX, one record for each day. The values we use are: *quote_date*, *high*, *low*, and *close*.

Data acquisition
Obtaining the financial data is easy. Point and click to receive a csv file containing the data we need. Netfonds.no provides the content we need.

More specifically we get the data for the Oslo Stock exchange[15].

By creating a little script we get fresh data every time we run the script.

```
1 stock_exchange_history = urllib.urlopen(url_of_datafile).readlines()
2     for record in stock_exchange_history:
3         # do tomething with the data
4         print record
```

Structure
The data file contains data back to 1997. Data for one day on each line. Only the days the exchange have been open are present. Fields in the csv file are: *quote_date*,*paper*, *exch*, *open*, *high*, *low*, *close*, *volume*, and *value*.

[15]Netfonds data on OSEBX: http://www.netfonds.no/quotes/paperhistory.php?paper=OSEBX.OSE&csv_format=csv

Chapter 4

Sentiment Classification

Sentiment is described as "an attitude toward something; regard; opinion."[1]. The sentiment is the perceived positivity of the message that the user tries to communicate. Sentiment is in many cases a personal thing, and can change from person to person or from setting to setting. We think of the sentiment as a conveyed meaning of a message.

Some of the motivation for acquiring the sentiment of a tweet or a sentence, is that we can say something about a persons state of mind and from that predict behaviour. Further the aspect of sentiment in trading is interesting. Can sentiment analysis improve trading in any way? And how can the sentiment be used to predict trends.

In this thesis we have two main ways of classifying tweets. Word counting and training a classifier. Both methods require dictionaries of positive and negative words. Classifier use the dictionary to extract features from a tweet. And with the word counting we count the number of positive and negative words.

Sections in this chapter are as follows. Manual classifications, 4.1, where we see how people classify tweets. Classification with classifiers follows in section 4.2, containing: 'Word count classification', and classification with classifiers such as SVM and Naive Bayes. A comparison of the classifiers and associated results can be found in section 4.3, and a brief discussion come last, in section 4.4.

[1] Dictionary.com on Sentiment: http://dictionary.reference.com/browse/sentiment?s=t

4.1 Manual Classification

When labeling tweets manually there are a number of factors that affect the process. Among them are the quality of the tweet, state of mind of the classifier, language, and political affiliation.

The content describes the quality of the tweet in many ways. Does the tweet contain links or hashtags? Are users mentioned?

In many cases will a persons state of mind dictate the persons actions. This also has an effect on labeling tweets. A person with a positive state of mind can classify negative tweets as positive.

Note that all this happens in the brain during 3 to 60 seconds while reading the tweet text and is a very simple description of the process. Labeling a tweet follows this algorithm in many ways:

Step	Thought	Description
1	Have we seen this tweet before?	Skip it or use previous classification.
2	#Hashtags or links present?	Some hashtags are automatically positive or negative. Also remove noise such as users and links.
3	Sarcasm?	If sarcasm is present, put up a warning flag saying that it is the opposite of step 4.
4	Special words?	Find a word that triggers positive or negative impression.
5	Done	Label tweet as positive or negative.

Result files
The results file from the manual classification are comma separated variable files(csv) with three fields.

- Sentiment: Positive, neutral or negative. Represented by 1, 0 or -1.
- Tweet id, if it exists, else 'id'. It is a long number.
- The tweet text.

Obame tweet set
When labeling the Obame tweet set we found that there were many retweets in the dataset. Whether or not the data is representative for Twitter in general

is difficult to say. Tweets favoring Obame are positive for people who support Obame and negative for supporters of Romney. This makes the tweet set difficult to work with because tweets can be positive and negative at the same time, depending on who are reading them.

Kiro tweet set
The Kiro tweet set also has a lot of retweets, but they are not used later. Removing retweets increased the uniqueness of the data, as the same content do not show up multiple times. The search words the dataset is based on could be broader to increase the diversity of the tweets.

4.2 Classification

4.2.1 Word Count Classification

We describe the classification process and the different parts of it. The results and discussion section looks at the drawbacks and results of this method. The algorithm counts the positive and negative words. More positive words than negative means the tweet is positive and vice versa.

Polarity
The polarity of a given tweet is based on the difference in the amount of positive versus negative words.

Polarity is the difference between positive and negative words. The difference is expressed as the percentage of the total numbers of words.

We look at the difference between the negative and the positive word percentage. If the difference is positive we have a positive tweet, and if the difference is negative we have a negative tweet.

Drawbacks
There are some drawbacks to the word count classification, the dictionaries and the threshold. The threshold is described in section 4.2.2.

The threshold gives inaccuracy because tweets that have a polarity value that is the same as, or close to, the threshold might be classified wrong.

The quality of the dictionaries affects the word count classification. If the dictionaries had better quality we would get better results. An example of improvements would be to remove stop words before creating bi-, and monograms.

4.2.2 Threshold in Word Count Classification

The threshold is where tweets are either positive or negative. It is compared to the polarity value. For a tweet to be classified as positive the ratio of positive and negative words has to be above the threshold value.

The percentage of positive words minus the percentage of negative words gives the polarity value, or the positivity(how positive a tweet is) of a tweet. When actually deciding if a tweet is positive or negative we look at the polarity value. If the polarity value is greater than the threshold, the tweet is classified as positive.

The threshold is given to the classifier on initialization. We find which threshold is best in section 6.3, on page 51.

Examples of classification
Example tweets:

- t1 = “good that he was decreasing badly”
- t2 = “he was good for increase”
- t3 = “good or bad”

Classification of t1:

- positive words: ’good’
- pos = 1 / 6 = 0.16666
- negative words: ’decreasing’, ’badly’
- neg = 2 / 6 = 0.33333
- polarity = pos - neg = -0.1667
- threshold of 0 gives negative classification
- threshold of 0.1 gives negative classification
- threshold of -0.2 gives positive classification

Classification of t2:

- positive words: 'good', 'increase'
- pos = 2 / 5 (to av fem ord) = 0.4
- neg = 0 / 5 = 0
- negative words: none
- polarity = pos - neg = 0.4 - 0.0 = 0.4
- threshold = 0.4 – positive
- threshold = 0.5 – negative
- threshold = -0.1 – positive

Classification of t3:

- positive words: 'good'
- pos = 1 / 3 = 0.3333
- negative words: 'bad'
- neg = 1 / 3 = 0.3333
- polarity = pos - neg = 0
- threshold = 0 – positive
- threshold = 0.1 – negative
- threshold = -0.1 – positive

Drawbacks with the Threshold

When classifying with the word count classifier a problem occurs when we get an equal amount of positive and negative words. Then the polarity value becomes 0. This results in a situation where we have no indication of a tweet being positive or negative. This is an area where we can improve the classification.

The amount of tweets where the threshold and polarity are equal is shown in table 4.2.2 on page 36. The number of cases where we get an equal amount of positive and negative words are significantly reduced if the threshold is set to 0.1. We can also see indications that the diversity and quality of the dictionary plays a role in the classification correctness. The larger the dictionary the more likely it is that we do not get an equal amount of positive and negative words.

Table 4.1: Word Count results where Threshold value=0

id	Dictionary	Polarity=0	Tweets
	– Kiro compiled dataset –		
1	Monogram, Obame	234	997
2	Monogram LoughranMcDonald	543	997
3	Monogram, combined Obame and LoughranMcDonald	178	997
4	Kiro, Monogram, self compiled	53	997
5	Kiro, Bigram, self compiled	7	997
6	Kiro, Trigram, self compiled	28	997
7	Obame, Monogram, self compiled	14	997
8	Obame Bigram, self compiled	446	997
9	Obame Trigram, self compiled	931	997
	– Obame tweet set –		
10	Monogram, Obame	335	1365
11	Monogram LoughranMcDonald	854	1365
12	Monogram, combined Obame and LoughranMcDonald	345	1365
13	Kiro, Monogram, self compiled	233	1365
14	Kiro, Bigram, self compiled	462	1365
15	Kiro, Trigram, self compiled	1221	1365
16	Obame, Monogram, self compiled	37	1365
17	Obame Bigram, self compiled	52	1365
18	Obame Trigram, self compiled	92	1365

4.2.3 Using Classifiers

To test and compare classification methods we used Naive Bayes and Support Vector Machine classifiers.

Comparing the two classifiers we found that SVM is best. Although we see quite a span within the SVM results. Details of the results of the two classifiers are described in section 6.5.

The self compiled dictionaries based on the Kiro dataset was used to compare the two types of classifiers.

Each test with a classifier uses the Kiro monogram dictionary for feature extraction. Both datasets are used in testing. Feature extraction is the process where we find elements of a tweet that is relevant for the sentiment.

SVM

Support Vector Machine[2], is a classification algorithm that uses a supervised learning model. Supervised learning is used with the algorithm to recognize patterns and analyse data.

SVM builds a model, based on the training data, that split the data into two categories. New data will be assigned one of the two categories. This makes the classifiers a non-probabilistic binary linear classifier.

In more specific terms SVM creates multiple hyperplanes. The larger the distance from the hyperplane to the training data, the better the classification. The hyperplanes can be used for regression, classification or other tasks.

When using SVM there are different kernel modules that can be used to improve results. The kernel is often chosen based on knowledge about the dataset. In the nltk[3] library for Python there are a number of different kernels.

The kernels were tested to find out which kernel gave the best results. Section 6.4 describes the experiment and the results. As seen in the experiment the LinearSVC kernel performed best and was used in further experiments.

Naive Bayes Naive Bayes[4], is based on Bayes theorem. It is a simple probabilistic classifier. It uses strong(naive) independence assumptions with the theorem.

The classifier assumes that the features are independent, that they have no connection to each other, and that all features have equal importance. A conditional model is used with the probability model. Abstractly we get this:

[2]Wikipedia: https://en.wikipedia.org/wiki/Support_vector_machine
[3]Natural Language Toolkit http://www.nltk.org/
[4]Wikipedia: https://en.wikipedia.org/wiki/Naive_Bayes

$p(C|F_1, \ldots, F_n)$

In plain English we have the reformulated model that can handle larger datasets:
posterior $= \frac{\text{prior} \times \text{likelihood}}{\text{evidence}}$

Even more specifically the implementation that we use in 'nltk' are:

```
                      P(label) * P(f1|label) *...* P(fn|label)
P(label|features)=----------------------------------------
                      SUM[l]( P(l) * P(f1|l) *...* P(fn|l) )
```

Naive Bayes is efficient to train in a setting with supervised learning. It requires only a small training set. The classifier have worked well with complex real-world problems. Good results despite simplicity have been found theoretically sound. But Naive Bayes is still outperformed by algorithms such as 'random forests'.

4.3 Comparison

The Classifiers
We have looked at three methods of classifying sentiment, Naive Bayes classifier, SVM classifier, and the word count classifier. The classifiers have classified the two datasets with varying results.

All methods of classification uses the manually labeled tweet sets as the data source. We use the self compile dictionaries for feature extraction.

Experiments
In section 6.2, 6.3, 6.4, and 6.5 the methods described in this chapter are tested to validate the research questions, 1.3.

Results
The results of the classifiers are described in 7.3.4. There the good and bad parts of the classifiers are presented and discussed.

4.4 Comments

Biased Mind
Subjectivity is an issue. The datasets of manually labeled tweets are biased,

and based on a personal opinion and the state of mind in the moment of classification. Therefore we have to keep in mind that all the results are based on the assumption that everyone agrees on the manual labeling. This is of course a source of errors to be explored in future work, chapter 9.

We should explore the psychology of perception and classification. How do people perceive content differently? Is finance easier to label than politics?

Drawbacks

The quality of the dictionaries is a problem. Some of our observations give insight into the use of natural language in association with dictionaries. Further drawbacks are the threshold of the word count classification. This should be addressed in some smart way.

Code wise we should improve the testing of the classifiers, and remove the static use of dictionaries in the feature extraction.

Chapter 5

Trending

We look at trends on Twitter and in finance and try to see if there are any correlations between them. While defining the context of our research we made some assumptions.

The Oslo stock exchange, OSEBX, is largely an oil based exchange, and therefore we assume that oil related data will give a good indication of the behavior of OSEBX. Section 3.1.4 says something about why we choose tweets related to oil for the trend aggregation.

Further we wanted to look at trading that is not algorithm based. Algorithm trading is also known as speed trading. So we started looking at technical analysis. The technical analysis gives a base for decision support to the trader. The techniques of technical analysis gives us good indications about trends, but we want to take it further by involving sentiment analysis.

We start by describing and defining a trend in section 5.1. Followed by trends in finance, 5.2. Continuing with trends in relation to twitter, 5.3. And last we compare trends based on Twitter with finance trends, 5.4.

5.1 The trend is your friend

A trend is a series of changes that moves in the same direction over a period of time. It is often associated with fashion or stock trading. The definition of trend is "the general course or prevailing tendency"[1].

[1] Dictionary.com: http://dictionary.referencs.com/browse/trend

In finance the trend can be the deciding factor in buying or selling. Traders want to buy during a bullish trend, while the value is going up. And they want to sell before you get to the bearish trend, when the value goes down. Figure 5.1, on page 42, shows basic trends. When markets do not trend they move sideways in trading ranges[2].

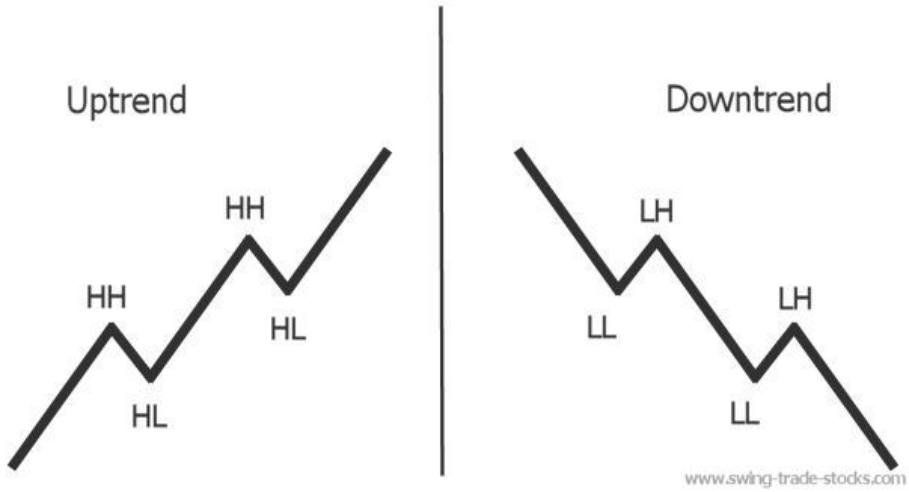

Figure 5.1: Basic Trend
Showing Higher Highs(HH), Lower Highs(LH), Higher Lows(HL), and Lower Lows(LL).

Moving average and *average directional index* are two techniques in technical analysis. Moving averages, MA, can help to confirm a move or determine a trend. Moving averages indicates average change over a period of time. The time frame is often adjusted to the time the trader likes to hold stocks before trading. Average directional index indicator, ADX, gives clear, easy-to-read picture of the market, shown in figure 5.2, on page 44. It shows if the trend strengthens or weakens. We focus on the two described techniques later in this chapter.

When using ADX or other trend indicators it is difficult to spot or predict the trend. When trading is based on a trend, the problem is to find the trend and

[2]Swing-trade-stocks.com: http://www.swing-trade-stocks.com/stock-trends.html

prolong it into the future. This is where we combine Twitter with traditional trading. If we could predict the trend based on public opinion mined from Twitter, we would have an advantage over the traditional trader.

Data is increasingly important for trend prediction. The better data we have, the more accurate the trend we predict can be. By focusing on sentiment we aim for the cooperation of information available on the Internet and the value of sentiment to predict trends. This will be more important in the future.

If the trend is your friend, you know how the market will move and make good decisions in accordance with it.

5.2 Trending in Finance

Moving average(MA), and average directional index(ADX) are the two trend indication techniques we test in this thesis.

Moving Average(MA)

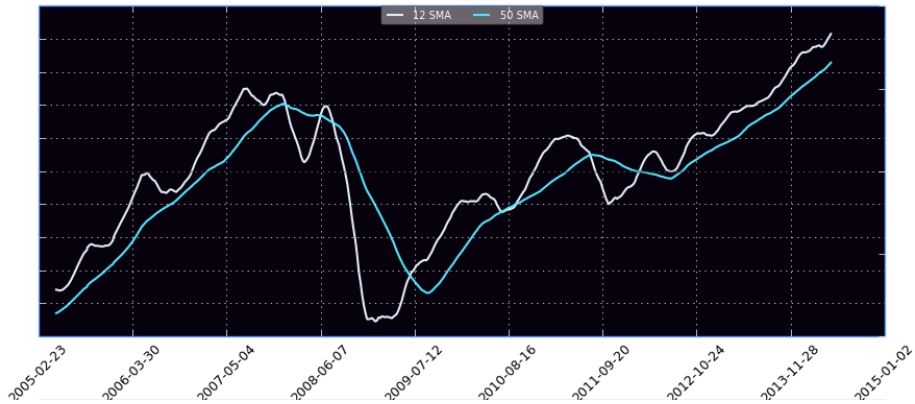

Figure 5.2: Moving Average example
The figure show the Moving Average for OSEBX from march 2005 until May 2014. The blue MA has a time frame of 50 weeks, while the white MA has a time frame of 12 weeks.

MA[3] is the unweighted mean of the previous n days. Given the closing price of the last n days as:

$p_M, p_{M-1}, \cdots, p_{M-(n-1)}$

We have that the simple moving average is:
$SMA = \frac{p_M + p_{M-1} + \cdots + p_{M-(n-1)}}{n}$

There are also other variants of the moving average. Such as cumulative MA, weighted MA and exponential MA. We use the simple version for MA trend.

Average Directional Index (ADX)

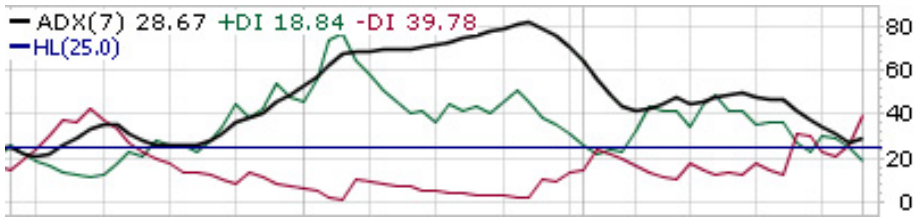

Figure 5.3: Average Directional Index example
An example of ADX. Green line is showing increase in price. Red line is decrease in price. The black line is the ADX.

In ADX, the further away from each other the red and green lines are, the stronger the trend is.

ADX[4] uses the positive directional indicator(+DI) and the negative directional indicator(-DI) in combination to determine a trend. The high, low and closing values are needed for the ADX to be calculated. The directional movement is calculated as follows:

```
UpMove = today's high - yesterday's high
DownMove = yesterday's low - today's low
if UpMove > DownMove and UpMove > 0, then +DM = UpMove, else +DM = 0
if DownMove > UpMove and DownMove > 0, then -DM = DownMove, else -DM = 0
```

[3]Wikipedia: https://en.wikipedia.org/wiki/Moving_average
[4]Wikipedia: https://en.wikipedia.org/wiki/Average_Directional_Index

When the directions are calculated we choose the time frame we want to investigate. And get -DI and +DI as follows:[5]

```
+DI = 100 times exponential moving average of +DM
      divided by average true range
-DI = 100 times exponential moving average of -DM
      divided by average true range
```

Where the time frame defines the number of periods used in the exponential moving average. And the average true is an exponential average of the true range. Resulting in the ADX:

```
ADX = 100 times the exponential moving average
      of the absolute value of (+DI - -DI)
      divided by (+DI + -DI)
```

Experiments
In the experiment, section 6.6 on page 52, we plot the graph for the finance trends. In the experiment we used data from OSEBX. The same time frame was used for stock data and tweet data, Apr 26 - May 26.

5.3 Twitter based Trends

We have two trends with twitter. First the trends Twitter themselves create based on words or hashtags that appear in many tweets. Second the trend we compile ourselves based on the trend data described in section 3.1.4, page 23.

Trends on Twitter
On Twitter.com there is a feature called 'Trends'. It shows the words that are used the most recently. A screen capture of it can be seen in figure 5.3, page 46.

The trends on Twitter are in many cases predictable and adds little value to trend compilation. The trend on Twitter is also specialised for each user, based on that users subscriptions and location. As an example at the Norwegian national day(May 17.) we would have trending words like *Norge*, *Norway*, and *#17Mai*. Today we already know that this will happen again next year.

[5] Wikipedia: https://en.wikipedia.org/wiki/Average_True_Range

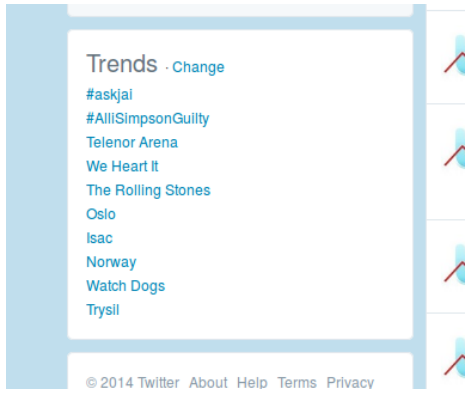

Figure 5.4: Trends on Twitter
Screen capture of trends from Twitter.

Twitter based sentiment trend
With the trend calculations based on sentiment we looked at methods used in finance and took inspiration from that. We ended up choosing the two previously mentioned methods MA and ADX as a base for our own work.

By using the two existing methods a base for our aggregated trend we gain a good indication of weather or not this is a viable way of creating trends with sentiment data.

The new element we use with the MA and ADX methods is the sentiment analysis. By using sentiment data we hope to create a sentiment trend. As far as we have research no one else has done this before.

As we do new and untested things we have no guarantee of success. But we have a good opportunity to test uses for sentiment analysis.

The trick to our method lies in the data manipulation. We manipulate the sentiment data we have to fit the existing format of stock data. Then we run the MA and ADX methods with the sentiment data to see if it works.

What we do, results in two methods of trend aggregation. Sentiment based MA and sentiment based ADX.

Using the trend data described in section 3.1.4 we construct the MA and ADX graphs based on sentiment. The data consists of tweets and is based on Twitter handles for users with big impact in the oil industry. There are two parts to the data. First tweets from Norwegian users, and secondly tweets from international users. To simplify the trend creation a bit we used English

tweets, ignoring tweets in Norwegian. After sorting the mined tweets by day, we classify all tweets for each day. The classification from each day is referred to as a trend day. The trend day contains the amount of positive tweets, the amount of negative tweets, and the total amount of tweets collected for that day.

```
trend_days = {
    'trend-Apr-30': {'neg': 112, 'pos': 131, 'tot': 243},
    'trend-May-19': {'neg': 523, 'pos': 1326, 'tot': 1849},
    'trend-May-18': {'neg': 59, 'pos': 110, 'tot': 169},
    'trend-May-15': {'neg': 1151, 'pos': 2255, 'tot': 3406},
}
```

Now we have some sentiment data to create a trend from. To simplify the trend aggregation and the graph plotting, we transform the data to fit into the format used for the finance data. The format of the data is the well known 'comma separated values', CSV.

```
Date,close,high,low,open,volume
20140423,559.41,562.45,557.74,561.13,0
20140424,562.89,565.65,559.84,559.41,0
20140425,566.24,566.24,562.41,562.89,0
20140428,566.82,567.14,564.55,566.24,0
```

To transform the data we do the following:

```
trend_days = get_tweet_trend_data()
keys = sorted(trend_days.keys())
date = ""
for i in range(1, len(keys)):
    if "Apr" in keys[i]:
        date = "201404" + str(keys[i].split('-')[2])
    elif "May" in keys[i]:
        date = "201405" + str(keys[i].split('-')[2])
    volume = trend_days[keys[i]]['tot'] * 1.0
    # (positive_tweets / total_amount_of_tweets)*scale
    high = (trend_days[keys[i]]['pos'] / volume) * 1000
    # (negative_teets / total_amount_of_tweets)*scale
    low = (trend_days[keys[i]]['neg'] / volume) * 1000
    openv = 0
    close = high - low
    print str(date) + "," + str(close) + "," + str(high) \
          + str(",") +str(low) + "," + str(volume / 10) + \
          "," + str(0)
```

We scale the values to plot a more representative graph. All the proportions are kept intact, so the MA and ADX are still valid. This method is tested in section 6.6 on page 52.

5.4 Comparing the Trends

Comparing the finance trend and the Twitter based trend is easy. We compare the two graphs, look for similarities, and draw conclusions. Drawing conclusions is the difficult part.

For plot examples see figure 6.6, page 54, and figure 6.6, page 55.

We look at the two plotted trends, the MA and the ADX. More specifically we compare the white MA of both graphs with each other, and the blue MA lines of each graph with each other. And look for similarities. With the ADX we will look for similar movements and turning points in the graphs. Also areas where the red and green lines stay apart.

Comparison should give some conclusions of the trustworthiness of the sentiment trend. For traders to trust in the sentiment analysis as a new way of predicting trends the sentiment would have to provide profit and scientifically proven results. At this stage there would be few who would rely on sentiment analysis as a trend indicator of the stock market. The trust lies in the technical analysis.

With the comparison we have the drawback of examining a stock exchange instead of a single stock. Comparing one stock at a time would give better indications of the accuracy of the sentiment trend. This should be explored further in the future.

5.5 Future work

For future improvements of the trend aggregation we should explore three aspects. One, the Norwegian market we started with. Two, the oil tweets. Three, find out whether or not oil tweets are a good indication of the OSEBX.

Additionally we should look into single stocks in the comparison of indicators. This way we can explore 'stock vs stock' Twitter trends, and find new information there.

Chapter 6

Experiments

This chapter aims to test methods described in earlier chapters. We describe setup and context for the experiments as well as what data was used.

The experiments described in this chapter are the following. Dictionary compilation, 6.1. The word count classification, bag of words method, 6.2. Threshold variation in section 6.3. Kernel choice for the SVM classifier is described in section 6.4. With the testing of classifiers coming next, 6.5. And the trend aggregation comes last in section 6.6.

6.1 Dictionary Compilation

Section 3.2 describes dictionaries and what they are used for. This section describes how the dictionaries are tested. The dictionaries are compiled from manually labeled tweets.

As a step in the sentiment classification dictionaries was created for the word count classification, 4.2.1. To find out which dictionaries performed well or not we tested the dictionaries with two data sets, the Kiro dataset and the Obama dataset.

This experiment enlightens part of research question 1, how we can find the sentiment of a tweet, section 1.3.1.

Dictionary quality
To test the dictionary quality the word count classification 6.2 does it for us.

All the dictionaries are tested with the two datasets and we can compare the results. How the dictionaries are compiled are described in 3.2.2 and A.4.

Removal of duplicate words
When creating the different dictionaries we remove duplicates from the positive and negative dictionary set. Words that are present in both the positive and negative dictionary is removed. By doing this we remove words that has no significance in the classification. But we also risk removing words with significance.

The code is described in appendix A.4 on page 81.

Results
Results from the testing of dictionaries are closely linked to the word count classification in 6.2 and described in table 7.3.1 on page 60.

6.2 Word Count Classification

Inspiration from section 2.2, with knowledge of tweets and dictionaries in section 3.1, builds this experiment. The background for the experiment and it's purpose is described in section 4.2.

This experiment used the technique described in section 4.2.1, the word count method of classifying a tweets sentiment.

This experiment tries to expand on the matter of how to determine the sentiment of a tweet, 1.3.1. And how Twitter can be used, indirectly, to aggregate a trend, 1.3.2. The method looks at the amount of positive and negative words and use that to decide the sentiment we have.

For all the dictionaries and datasets, the classification process. This gives the accuracy for all dictionaries for the two datasets. This in turn describes the efficiency of the word count classification.

The code for calculating the sentiment is as follows. It is also described in appendix A.5.1, on page 82.

```
1 positivity = positive_words_count / total_number_words
2 negativity = negative_words_count / total_number_words
3
4 polarity = positivity - negativity
5
6 if polarity > 0:
7     tweet is positive
```

```
8 else:
9     tweet is negative
```

The results of the classification can be found in section 7.3.1 on page 59.

6.3 Threshold Variation

In section 4.2.2 the description and purpose of the threshold variation is described. The threshold aims to find out how positive a tweet has to be before it is actually positive. Or rather how many more positive words than negative words do we need to call the tweet positive.

By varying the threshold we hoped to find an optimal point of which we could separate tweets based on polarity. From the threshold graphs in figure 7.3.2, page 64, we can see no clear distinction of one value being better than the other ones.

In essence what is done is that the word count classification is run multiple times where the threshold value is changed between -0.9 and 0.9. This gives a distribution of results which is plotted in graph 7.3.2 on page 63. The results will give an indication of what threshold is best for word counting.

The results of the threshold variation is described in section 7.3.2, on page 61. And in table 7.3.2 on page 62.

6.4 Choice of SVM Kernel

In extension of 4.2.3 all the SVM kernels were tested. While testing which kernel was best we used the Kiro dataset for training, and the Kiro monogram dictionary for feature extraction.

Using the self compile monogram dictionaries and all the different SVM kernels we get the results described in table 6.4, page 52.

6.5 Using Classifiers

For the classifiers the two datasets was tested with the two classifiers. This was done to find the best classifier and compare the results with the word count classification. By confirming that classifiers are better than word counting we can eliminate a factor of the classification uncertainties.

Table 6.1: SVM kernel test results table

Kernel	Failed	Correct	Accuracy
LinearSVC	7	990	0.9930
NuSVC	29	968	0.9709
NuSVR	422	575	0.5767
OneClassSVM	575	422	0.4233
SVC	422	575	0.5767
SVR	422	575	0.5767

Table 6.2: SVM classifier results table

Dataset	Type	Failed	Correct	Accuracy
Kiro	Monogram	7	990	0.9930
Kiro	Bigram	422	575	0.5767
Obama	Monogram	35	1330	0.9744
Obama	Bigram	507	858	0.6286

The description of the comparison on a conceptual level is described in section 4.3. Also of interest there is the element of the dictionaries in section 3.2. While we have taken inspiration and ideas for the experiment from section 2.2.

For the research questions to be enlightened further here is the trend aggregation in section 1.3.1 and the sentiment determination in section 1.3.2.

SVM
Results from testing SVM with different dictionaries are described in table 6.5 on page 52.

Naive Bayes
Results from testing Naive Bayes with different dictionaries are described in table 6.5 on page 53.

6.6 Trend Aggregation

Section 5.4 describes the conceptual parts of the trend comparison. This is the basis for the results of this experiment.

Table 6.3: Naive Bayes classifier results table

Dataset	Type	Failed	Correct	Accuracy
Kiro	Monogram	29	968	0.9709
Kiro	Bigram	29	968	0.9709
Obama	Monogram	59	1306	0.9568
Obama	Bigram	59	1306	0.9568

This experiment aims to test the relevance of positive and negative tweets in correlation with change in finance. There are two parts of this experiment. The finance part, and the twitter part. Can twitter be used to create a trend, 1.3.2. And how can we compare this to a finance trend, 1.3.3.

In the execution of the experiment the methods described in chapter 5 has been used. More specifically the trend aggregation under trends on twitter, 5.3. And the finance trend aggregation, 5.2.

The tweet data used is described in section 3.1.4, on page 23. And the finance data is described in section 3.3, page 29.

The results of the experiment is discussed in section 7.4.

Finance plot

The plotting of the graph is described in better detail in appendix A.6, page 83.

Plotting the finance trend we got figure 6.6, page 54.

Twitter plot

Code for plotting the trend graph is described in appendix A.6, page 83.

Plotting the moving average for the Twitter trend we get figure: 6.6, on page 55.

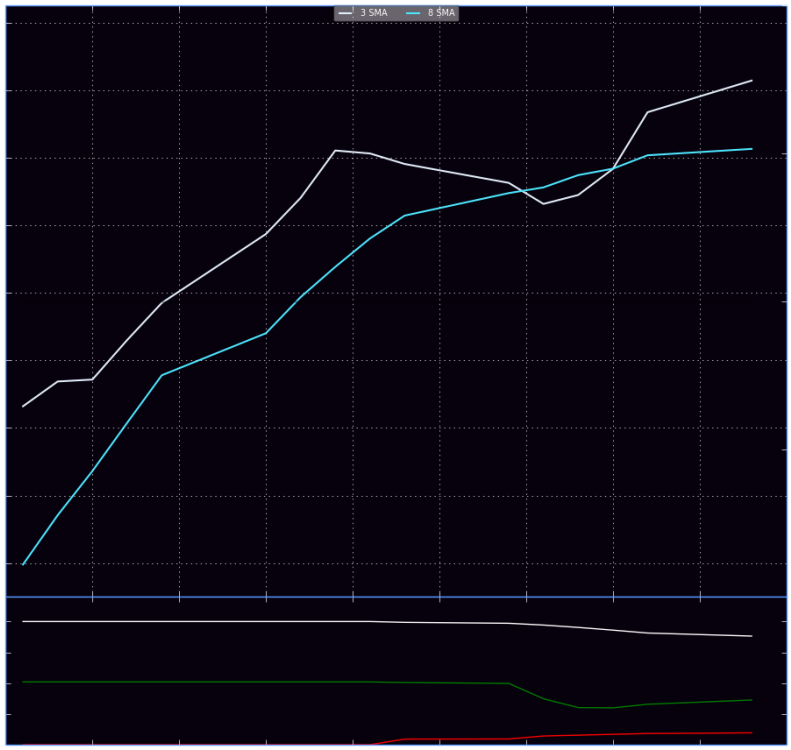

Figure 6.1: Finance trend plot
This is the plot for OSEBX. The plot show the moving average in the middle and the Average directional graph at the bottom. Time frame: Apr 26 - May 26.

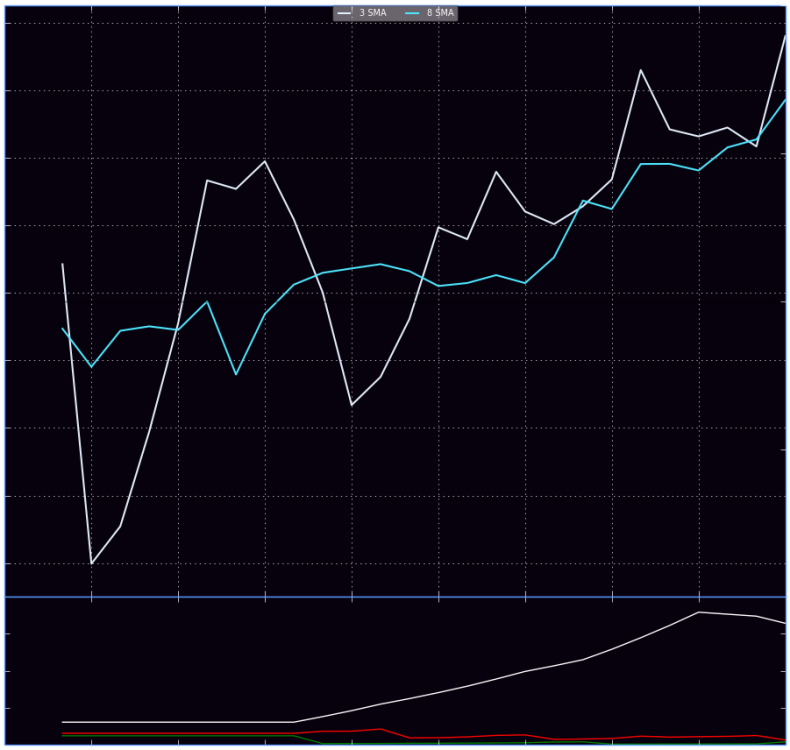

Figure 6.2: Tweet trend plot
This is the Twitter trend plot. The plot show the moving average in the middle and the Average directional graph at the bottom. Time frame: Apr 26 - May 26.

Chapter 7

Results and Discussion

The results of our research and the following discussion of them happen here. We touch the main parts of the thesis, described earlier. The data sources, section 7.1. Dictionaries, section 7.2. Classification of sentiment in section 7.3, and the trend in section 7.4. Last we have a general discussion about the results.

7.1 Data Source

Twitter as a data source proved to be OK. It was quite easy to obtain data. But increasingly difficult to get good amounts of data and unbiased data. The initial data have some restriction to search words. We should have mined a lot more data and manually classified more of it.

As far as the dictionaries go they work well. We have different results for different dictionaries and different datasets, but all in all the way to compile datasets work good. The dictionaries depend on the manually classified tweets, so a lot of the potential improvement lies there. Although we can improve the dictionaries by removing stop words and other words we know have no clear positive or negative sentiment.

In research question 1, 1.3.1, we asked what knowledge we can extract from tweets to find a sentiment. The dictionaries are a good way to compose sentiment from the different words in the manually labeled tweets.

Research question 3, 1.3.3, asks if Twitter has been useful in comparison to technical analysis. To some extent it has. We have mined and classified many tweets in the hope that the aggregated sentiment trend would show likeness to

plots of technical analysis. The data from Twitter has been used to acquire positive and negative tweets for the trend aggregation, and has also provided useful insight into the use of the Twitter API.

7.2 Dictionaries

As for the use of the dictionaries in association with the trend and research question 3, 1.3.3, we use the dictionaries for feature extraction in the classifiers. Besides the use in classifiers the dictionaries have no other practical uses in this thesis in relation to sentiment.

Error analysis
When looking at the duplicate words from the monogram dictionary based on the Kiro dataset we found some errors. As a selection of words found, we have *dangerous, bad, go, inc, let, up, or, need, good, if, no, are, and, of, on, the, is, as*. Here we can see that the words *good* and *bad* are represented. Which is not good. By removing the words from the dictionaries we have removed significant words in further classification, thus reducing correctness of the algorithm. This is one of the drawbacks of the monogram dictionaries.

When looking at the removed duplicate words for bigram and trigrams we found no indication of the same problem. As the uniqueness of bigrams and trigrams are a lot greater we end up with very few duplicates and only duplicates that has no significance to the over all classification. Although we might have other unknown problems.

Most stop words and other insignificant words are removed with the removal of duplicate words. The same thing cannot be said about the bigram and trigram dictionaries. There we have no stop words present in themselves, but they are frequently part of other words. For further improvements of classification with word counting and dictionary quality we should remove stop words, such as *as, is, on, off, and, or* etc, from the tweet/sentence before creating bi- and trigrams.

7.3 Classification

Manual labeling is a bit of a challenge to keep unbiased. The whole classification process is based on a persons opinion, which makes the results somewhat biased. To eliminate this error we should have multiple people do the manual classification and combine the results.

Word counting works to some extent. But presents varied results based on the dataset and the dictionary used. Again this comes down to the dataset and the dictionaries created. Also the way we separate tweets from each other with a threshold can be improved. Word counting works, but it is not very good.

Classifiers improve the classification quite a bit. With the SVM classifier we get around 98 percent correct classification on both data sets. Which is good.

The uncertainties lie in the biased, or potentially biased, data. We do not know if the data in itself is fully representable for the content on Twitter. Neither do we know if the manual classification is completely correct.

In total we gained some knowledge about dictionaries and their effect on classification. And we can confirm that classifiers work better than word counting.

We tested two ways of finding the sentiment of a tweet. The bag of words method described in section 4.2.1, and the trained classifier way with the SVM and Naive Bayes classifiers, 4.2.

In relation to research question 2, 1.3.2, the question about trend aggregation, we found that the sentiment classification was successful. The sentiment worked to some extent. We do not know how well, and we have to look into the part of credibility in future work.

7.3.1 Word Count Classification

The results from the word count classification is plotted in the tabled 7.3.1, page 60, and in the graph 7.3.1, page 61.

5, 6, 17, 18 are the dictionaries with best accuracy. Which is to be expected for the classification of the dataset the dictionary was created from.

But more interestingly, if we take the dictionaries created form one dataset, and look at the results of the classification on the other dataset. We compare 7, 8, 9 with each other and 13, 14, 15 with each other. Then we can see that the bigram and trigram dictionaries perform better than the monogram dictionaries.

Further we see indications that the quality of the dictionary plays an important role. The LoguhranMcDonald monogram dictionary performs quite good with both datasets. Even on par with the compiled dictionaries on opposite dataset. This indicates that a well crafted dictionary can perform as well as compiled dictionaries, and vice versa.

When comparing the results from one dataset to the other we can see indications of how the content of the dataset also plays a huge role in the results. Some tweets are more difficult to classify then others.

Table 7.1: Word Count Classification Results

id	Dictionary	Failed	Correct	Accuracy
	– Kiro compiled dataset –	a	b	b/(a+b)
1	Monogram, Obama	578	419	0.4203
2	Monogram LoughranMcDonald	491	506	0.5075
3	Monogram, combined Obama and LoughranMcDonald	416	581	0.5827
4	Kiro, Monogram, self compiled	115	882	0.8847
5	Kiro, Bigram, self compiled	17	980	0.9829
6	Kiro, Trigram, self compiled	18	979	0.9819
7	Obama, Monogram, self compiled	567	430	0.4313
8	Obama Bigram, self compiled	534	463	0.4644
9	Obama Trigram, self compiled	567	430	0.4313
	– Obama tweet set –	a	b	b/(a+b)
10	Monogram, Obama	855	510	0.3736
11	Monogram LoughranMcDonald	508	857	0.6278
12	Monogram, combined Obama and LoughranMcDonald	544	821	0.6015
13	Kiro, Monogram, self compiled	632	733	0.5370
14	Kiro, Bigram, self compiled	521	844	0.6183
15	Kiro, Trigram, self compiled	498	867	0.6352
16	Obama, Monogram, self compiled	493	872	0.6388
17	Obama Bigram, self compiled	37	1328	0.9729
18	Obama Trigram, self compiled	39	1326	0.9714

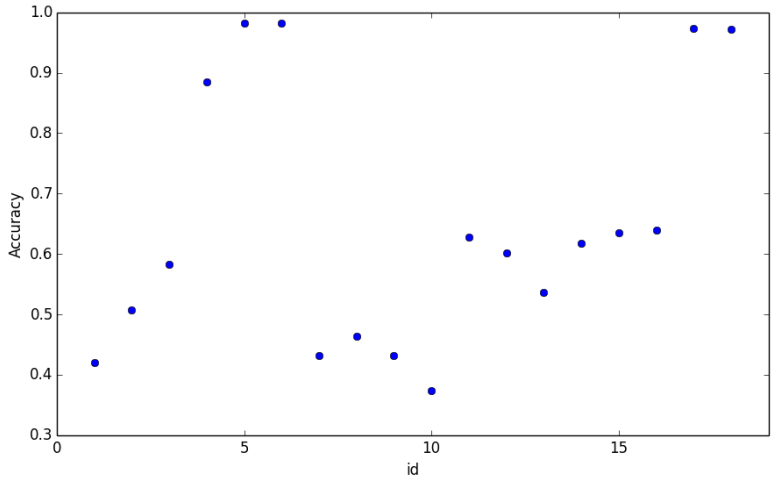

Figure 7.1: Dictionary Accuracy plot
The graph shows the plotted accuracies from the word count classification. The ids represents the ids from table 7.3.1, page 60

There are a lot of improvements that could be done. The word count classification is merely a convenient way to say something about the dictionaries used. We should expand our knowledge toward the quality of the dictionaries. And look for ways to eliminate words that are neither positive or negative.

7.3.2 Threshold Variation

Further we found the best average threshold value to be 0.1. From table 7.3.2, page 62, we have the threshold value, and the average classification accuracy among the 18 entries for each threshold value.

In figure 7.3.2, page 64, we list the results of the experimentation with the threshold. Table 7.3.2, page 62, lists the dictionaries and dataset used for which graphs in figure 7.3.2, page 64. 'Kiro dataset' and 'Obama dataset' columns tells which dataset that was classified in which graph.

Table 7.2: Average Threshold Accuracy

Threshold	Accuracy	Threshold	Accuracy
-	-	0.0	0.6479
-0.1	0.6316	0.1	0.6516
-0.2	0.6161	0.2	0.6511
-0.3	0.6059	0.3	0.6430
-0.4	0.5988	0.4	0.6305
-0.5	0.5888	0.5	0.6122
-0.6	0.5711	0.6	0.5934
-0.7	0.5423	0.7	0.5712
-0.8	0.5083	0.8	0.5457
-0.9	0.4881	0.9	0.5307

Table 7.3: Dictionary to Threshold graph plot values

Dictionary name and description	Kiro dataset	Obama dataset
Obama original, Monogram	1	10
LoughranMcDonald, Monogram	2	11
Combined Obama original and LoughranMcDonald, Monogram	3	12
Kiro, Monogram, self compiled	4	13
Obama, Monogram, self compiled	5	14
Kiro, Bigram, self compiled	6	15
Obama, Bigram, self compiled	7	16
Kiro, Trigram, self compiled	8	17
Obama, Trigram, self compiled	9	18

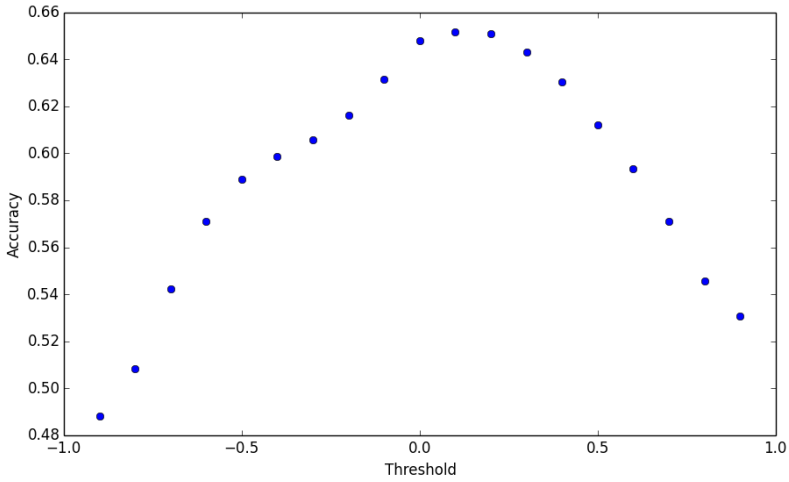

Figure 7.2: Average Threshold Accuracy
Graph plots of the 'Average threshold accuracy' table.

7.3.3 Using Classifiers

Not surprisingly the classifiers have better results than the bag of words method of classification. Further the SVM classifier with the LinearSVC kernel module performed best. Naive Bayes classifier was nearly as good. This answers part of research question 1, 1.3.1.

SVM
From results of the experiment, 6.5, we can draw some conclusions. The monogram dictionary performed better than the bigram dictionary. Probably due to the number of features we can extract from each tweet. Classifiers are more accurate than the word count classification method.

Naive Bayes
As we can see from the experiment, 6.5, the different dictionaries makes no difference for the Naive Bayes classifier. But if we compare the classifier with

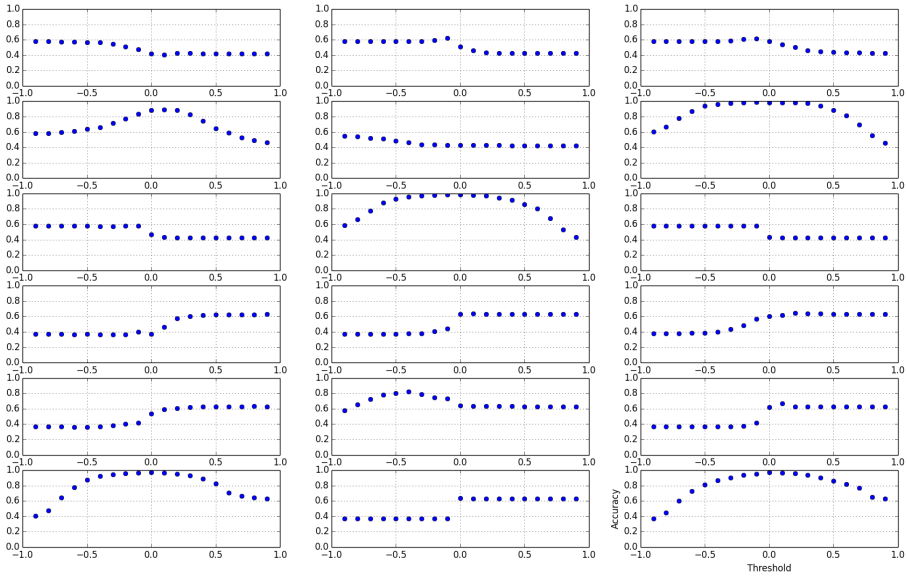

Figure 7.3: Threshold Variation Accuracy
The graphs plot the different variations of threshold. Counting is columns first; top left is 1, top mid is 7, top right is 13.

the results from the word count classification we can clearly see improvement. Naive Bayes did not perform as good as SVM.

7.3.4 Comparison

In table, 7.3.4, page 65, we highlight the results from the different classifications. We list the most noteworthy results from our experiments and compare them.

As we can see the classifiers give better results then the word count classification. And SVM is a bit better than Naive Bayes. Although the results from the word count classification indicates that the dictionaries play an important role in the results. We can also see that monograms are better for classifiers, while trigrams are better for word counting.

For the classifiers and the good results with the monogram dictionaries, we think that has to do with the number of features we get from a tweet. The more

Table 7.4: Comparison of Classifiers
Here we highlight the best results from classification tests.

Classifier	Dataset	Dictionary	Failed	Correct	Accuracy
Word Count	Kiro	Obama Bigram	534	463	0.4644
Word Count	Obama	LoughranMcDonald Monogram	508	857	0.6278
Word Count	Obama	Kiro, Trigram	498	867	0.6352
Naive Bayes	Kiro	Kiro Monogram	29	968	0.9709
SVM	Kiro	Kiro Monogram	7	990	0.9930

features we have the easier it is to classify and the more accurate the result.

7.4 The Trend

Comparing the two graph plots, figure 6.6, page 54, and figure 6.6, page 55, we can see minor similarities. All the moving averages have rightward growth. While for the tweet trend the 3 day interval preforms worse than the 8 day interval. There are no certain correlations between the two trends. And it is difficult to conclude on specifics.

Research questions(RQ), 2(1.3.2), and 3(1.3.3) are in some degree true. RQ2, we can aggregate a trend, the use for it, and the accuracy we know nothing of. RQ3 has the answer 'poor'. The sentiment trend compares poorly to the trend from the technical analysis.

7.5 Discussion

We have satisfying and unsatisfying results. The reasonable results should be expanded, retested and improved. Most notably we have clear indications that dictionaries, automatically compiled from a manually classified tweets set, can be used when classifying sentiment.

The accomplished research, based on the RQ, can be described as satisfactory. We set out to investigate how we can determine the sentiment of a tweet, 1.3.1, how Twitter can be used to aggregate a trend, 1.3.2, and how Twitter based trends compare the technical analysis in stock markets, 1.3.3.

RQ1 more specifically aims at the information of tweets, the knowledge we can extract from them, and classification of the sentiment of a tweet. In all respects we gained new knowledge. By creating dictionaries we extracted words from manually labeled tweets, giving us a way to classify new tweets. We ended up only using the tweet text for classification. Other metadata should have been explored. The best way of classifying tweets we found to be the SVM classifier.

RQ2 investigates the complicated part of aggregating a trend based on Twitter. This turned out to be difficult. Mainly by the lack of good ideas of how we could do it, but also the difficulty of transforming data to a usable format. In the end we transformed tweet data to the same format as stock data and used that for the trend aggregation.

RQ3 takes the trend a step further and compares technical analysis with the Twitter based sentiment trend. After plotting MA and ADX for both finance and Twitter data, we compared the graphs. The two plots can be seen in figure 6.6, page 54, and figure 6.6, page 54. In the figures we looked for similarities. For the MA we can argue that the blue lines, MA with 8 day intervals, are quite similar. We can make no definite conclusion that the Twitter trend is representative for the finance market. Technical analysis is still better, but sentiment analysis is a possibility. The ADX show no similarities between the Twitter sentiment trend and the finance trend.

Chapter 8

Conclusion

The Twitter API and mining of tweets started as a challenge. But after working out the quirks tweets was mined efficiently. The data worked, but the quality is questionable. The verdict of the data is that it provide results, but can be improved.

Tweets were classified manually and by classifier. Classifiers worked better than the bag of words method. The classifiers had high accuracy. Which is good. But also a cause for concern.

Aggregation of trends worked to some degree. The finance plotting work as expected, while the Twitter sentiment trend was difficult. The plotted trends had very few similarities and thus we conclude that technical analysis is still better than sentiment analysis for predicting the stock market.

The questions we set out to answer, and the achievements of those, can be summarised as follows. We can determine the sentiment of a tweet by counting words, but trained classifiers work better. Trends can be aggregated with data from Twitter, but the trend created so far is not very useful. Last we can safely say that technical analysis of stock markets work better than the current sentiment analysis.

Chapter 9

Future Work

Through the time spend writing this thesis a lot of ideas and edging areas of research have been touched, albeit many of them briefly and only with little interest. This chapter look into all the things not quite touching the core of this thesis.

9.1 Twitter

Twitter should be used for further mining, specific areas of interest, and specific users and hashtags should be investigated. The finance section should also be an area of focus. The importance of retweets should also be investigated. How does retweets propagate a tweet through social media, and what impact does it have?

Twitter API
We should look at the different endpoints of the API and see how we can use them smartly [1]. There are definitely unused options that can improve the sentiment analysis, or data mining aspects discussed in this thesis.

Tweet sets
There should have been a lot more data and it should have been more diverse. To achieve this, the natural path would be to define a set of finance words that

[1] Twitter: https://dev.twitter.com/docs/api/1.1

represent the finance area of Twitter. Then we should use those words to mine Twitter for finance relevant tweets. The tweets would be used to create better dictionaries and provide better sentiment data in trend aggregation.

Special Twitter content
Twitter has it's own convention of symbols that is used. Symbols like #hashtags, @usernames and stock markers, $STO. What is the contribution of those? Do these symbols add special meaning to a tweet? Questions like these, and other questions about use and frequency should be answered. We think that this can contribute to the sentiment classification and trend aggregation.

Language analysis
Some form of language analysis should be done with tweets to see if the language used on Twitter can be described in a good systematic way. This would find specific words, sentences, punctuation and other syntactic sugar. An example of this is 'bc'. It is an abbreviation that, in it's context, means 'because'. The existing information from SMS linguistics[2] should be tested on Twitter to see it's relevance.

9.2 Dictionaries

For the dictionaries and the compilation of them we should look further into the quality of them. Improvements with stop words and further elimination of neutral words is also an option. This could be some kind of qualitative analysis of words and their sentiment. Combinations of mono-, bi-, and trigrams should also be tested. Do we get better results if we extract features that are mono-, bi-, and trigram?

9.3 Sentiment

The future of sentiment classification lies with the classifiers. But the feature extraction can be improved. This is a natural place to start further research into sentiment analysis. A further test of training data, and test sets should be done to validate the accuracy of the classifiers.

[2]Wikipedia: https://en.wikipedia.org/wiki/Texting_Slang

We should cross classify and test the classification with the SVM classifier, and do quality assurance on the work with it. Other classifiers, and other data should be explored to confirm the findings we have so far.

Is neutral negative or positive?
If a tweet is neutral, where the amount of positive words and negative words are the same, how do we then classify the tweet? The edge case where we, with the threshold, found that many tweets were classified wrongly. How can this problem be solved?

Word weighting
How can we use weighting of words in sentiment classification? Are there a way where we can assign a value to some specific words to improve the bag of words classification? Is 'good' as positive as 'excellent'? Or 'bad' as negative as 'evil'?

How people express sentiment
We should explore the psychology of perception and classification. How do people perceive content differently? Is finance easier to label than politics? A bit of a psychological twist is to look into the natural language of people, and find out how we express a sentiment. What trigger an event and what part of the event is significant to the sentiment, and can knowledge about how people express sentiment improve upon the classification of sentiment?

Use of sentiment analysis on news
As a test we can change the dataset, and focus on news. Particularly finance related news, and focus on the title and ingress of the news. In many cases the sentiment, and all the necessary information is presented in title and ingress. The title and ingress can in many ways be compared to tweets, short messages containing a sentiment.

9.4 Trend

Credibility
The credibility of a tweeter might have an impact on the sentiment. We do not know. Reputation and social reach are two factors that cannot be ignored when classifying tweets. Do these factors have an effect on the trend at all? These factors should be considered when aggregating a trend.

Trend Data
The data we have acquired should be further analysed. Is the data good? Is the data representative for twitter? Can we use the data in other ways? Is the quality of the data any good? The analysis of the trend data is important for further improvements for the trend aggregation. We have over 44 thousand tweets waiting to be analysed.

Trend Aggregation
We have to look for other methods of aggregating a trend. This is to improve the Twitter sentiment trend we have created, but also to validate the accuracy and correctness of the trend we have created. Other parts of the data might be used, and should be explored.

Bibliography

Luciano Barbosa and Junlan Feng. Robust sentiment detection on twitter from biased and noisy data. 2010. Coling 2010: Poster Volume, pages 36–44, Beijing, August 2010.

Lee Becker, George Erhart, David Skiba, and Valentine Matula. Avaya: Sentiment analysis on twitter with self-training and polarity lexicon expansion. 2013. Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Seventh International Workshop on Semantic Evaluation (SemEval 2013), pages 333–340, Atlanta, Georgia, June 14-15, 2013. c 2013 Association for Computational Linguistics.

Johan Bollen, Huina Mao, and Xiaojun Zeng. Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8, 2011. ISSN 1877-7503. doi: http://dx.doi.org/10.1016/j.jocs.2010.12.007. URL http://www.sciencedirect.com/science/article/pii/S187775031100007X.

Gregory W. Brown and Michael T. Cliff. Investor sentiment and the near-term stock market. *Journal of Empirical Finance*, 11(1):1 – 27, 2004. ISSN 0927-5398. doi: http://dx.doi.org/10.1016/j.jempfin.2002.12.001. URL http://www.sciencedirect.com/science/article/pii/S0927539803000422.

B. Connor, R. Balasubramanyan, B. Routledge, and N. Smith. From tweets to polls: linking text sentiment to public opinion time series. 2010. URL http://www.cs.cmu.edu/~nasmith/papers/oconnor+balasubramanyan+routledge+smith.icwsm10.pdf.

Nicholas A. Diakopoulos and David A. Shamma. Characterizing debate performance via aggregated twitter sentiment. 2010. URL http://dmrussell.net/CHI2010/docs/p1195.pdf.

John A. Doukas, Constantinos Antoniou, and Avanidhar Subrahmanyam. Sentiment and momentum. January 10, 2010. Updated May 20, 2011. Available at SSRN: http://ssrn.com/abstract=1479197 or http://dx.doi.org/10.2139/ssrn.1479197.

A. Java, X. Song, T. Finin, and B. Tseng. Why we twitter: Understanding microblogging usage and communities. 2007. 9th WebKDD and 1st SNA-KDD workshop on web mining and social network analysis, 2007.

Annika Jubbega. Twitter as driver of stock price. Master's thesis, BI Norwegian School of Management, 2011.

Yung-Ming Li and Tsung-Ying Li. Deriving market intelligence from microblogs. *Decision Support Systems*, 55(1):206 – 217, 2013. ISSN 0167-9236. doi: http://dx.doi.org/10.1016/j.dss.2013.01.023. URL http://www.sciencedirect.com/science/article/pii/S0167923613000511.

Michael Speriosu, , Nikita Sudan, Sid Upadhyay, and Jason Baldridge. Twitter polarity classification with label propagation over lexical links and the follower graph. 2011. Proceedings of EMNLP 2011, Conference on Empirical Methods in Natural Language Processing, pages 53–63, Edinburgh, Scotland, UK, July 27–31, 2011. c 2011 Association for Computational Linguistics.

Alexandra Stevenson. The social media stock pickers, Oct 23 2012. URL http://search.proquest.com/docview/1114502067?accountid=12870. Copyright - Copyright Financial Times Ltd. 2012. All rights reserved.; Last updated - 2012-10-23.

Xue Zhang, Hauke Fuehres, and Peter A. Gloor. Predicting stock market indicators through twitter "i hope it is not as bad as i fear". *Procedia - Social and Behavioral Sciences*, 26(0):55 – 62, 2011. ISSN 1877-0428. doi: http://dx.doi.org/10.1016/j.sbspro.2011.10.562. URL http://www.sciencedirect.com/science/article/pii/S1877042811023895. ¡ce:title¿The 2nd Collaborative Innovation Networks Conference - COINs2010¡/ce:title¿.

Appendix A

The Code

The code is complete in the sense of doing what it should. Or making an attempt to. The results of the coding are up for discussion.

For functionality we have code for mining tweets, compiling dictionaries, classifying tweets, and aggregating a trend based on data from twitter.

For this chapter we start of with the structure of the code, A.1. Then touch the technology and libraries, A.2, used up to this point. Before we look at the data mining, A.3, dictionary compilation, A.4, classification, A.5, and trend aggregation, A.6. Last we describe the comparison, A.7, of the trends and ending with the issues, A.8, of the code.

A.1 Structure

There are four main folders. One for classification, one for mining tweets, one for the dictionaries and on for the files associated with trend.

Code for classification is split into four files. One file for utilities, helper functions that does not touch logic. And three for classification: manual classification, word counting classification, and classification with classifiers. List_threshold_results plots the results from the threshold variation in graphs.

The dictionary code is split onto two files, helpers and utility in one, and logic and execution in the other.

Trend code is in the trend folder. There we have mining_utils, which is a replica of the same file in the Twitter folder. This file only helps with the acquisition of tweets. The trend_mining file executes the mining operations, and

acquires tweets from Twitter based on a set of search words. The trend_tweet_-sorting file sorts tweets from all the raw search word data files into files based on date. So all tweets from the same date ends in the same file. We also have the trend_compilation file, which contains code for compiling and plotting trend data and financial data.

Last we have the 'twitter' directory which has code for extracting tweets from twitter. This code is used for creating the datasets that the dictionaries are built from. We have the mining_utils which is helper code for connection and writing files. While the mining_operations file does the logic of managing the acquired tweets.

The important files are listed under. '- something' are folders, while the others are actual Python files. The indentation shows which files are in which folders.

```
- code
    - classification
        classification_utils.py
        list_threshold_results.py
        manual_classification.py
        svm_bayes_classification.py
        word_count_classification.py
    - dictionaries
        dictionaries.py
        dictionary_utils.py
    - trend
        mining_utils.py
        trend_classification_utils.py
        trend_compilation.py
        trend_mining.py
        trend_tweet_sorting.py
        ma_adx_plotter.py
    - twitter
        mining_operations.py
        mining_utils.py
    graph_plots.py
```

A.2 Technology and Libraries

The used Python libraries are: ConfigParser, ast, codecs, io, os, twython, time, matplotlib, urllib, re, nltk, sklearn. Most of them are standard, while some are not. Some are self explanatory.

The libraries that are not shipped with standard Python are described in the following paragraphs.

Twython Library for connection and integration with the Twitter API. The source and documentation can be found on github: https://github.com/ryanmcgrath/twython

nltk The natural language tool kit (nltk) is a library that provides functionality for working with human language. It has functionality such as classifiers and tokenization tools. See more at http://www.nltk.org/.

sklearn Scikit-learn provides functionality for learning algorithms, machine learning and classification. We use this library to provide the kernels for classification with classifiers. http://scikit-learn.org/

matplotlib Matplotlib is a library for graph plotting. And that is what we use it for. http://matplotlib.org/.

A.3 Data Retrieval

As we have two data sources we split it into finance data and data from twitter.

A.3.1 Twitter

The mining operation can be seen as three steps. This happens in *mining_operations.py* and *mining_utils.py*.

Step 1
Authenticating and connecting to twitter.

```
1 conf = open('auth.cfg').read()
2 config = ConfigParser.RawConfigParser(allow_no_value=True)
3 config.readfp(io.BytesIO(conf))
```

```
4
5  # getting data from conf object.
6  APP_KEY = config.get('twtrauth', 'app_key')
7  APP_SECRET = config.get('twtrauth', 'app_secret')
8  OAUTH_TOKEN = config.get('twtrauth', 'oauth_token')
9  OAUTH_TOKEN_SECRET = config.get('twtrauth',
10                                 'oauth_token_secret')
11
12 # creating authentication object for twython twitter.
13 twitter = Twython(APP_KEY,
14                   APP_SECRET,
15                   OAUTH_TOKEN,
16                   OAUTH_TOKEN_SECRET)
```

Step 2
Query execution.

```
1 results = twitter.cursor(twitter.search,
2                          q=query,
3                          count="100",
4                          lauage=language)
```

Where we have the query, and language as input variables to the *cursor_extraction* function.

Step 3
Data extraction and storage.

```
1  # opens new file with today's date and time now as filename
2  filename = destination_folder + "/dataset-" + \
3      strftime("%d-%b-%Y_%H:%M:%S")
4  # opens the file for appending.
5  data_set = open(filename, 'a')
6
7  for result in results:
8      # skip previously acquired tweets
9      if str(result['id']) in str(previous_tweets_list):
10         continue
11     else:
12         previous_tweets_list.append(result['id'])
13
14     # store tweet to file for later use.
```

```
15      data_set.write(str(result) + "\n")
```

We open the storage file for writing and gives it a name based on the time of creation. Then we traverse all results and store tweets we have not seen before.

A.3.2 Finance

We get finance data from http://www.netfonds.no/. Specifically we get the data about Oslo stock exchange[1] We also sort out the data we do not want by breaking at a certain date.

We do this by using the urllib library in Python:

```
# get data file from internet. (csv)
stock_exchange_history = urllib.urlopen(url).readlines()
records = []
for record in stock_exchange_history:
    # earlier records are not interesting.
    if "2014" in record:
        records.append(record.strip().lower().split(","))
    if "20140414" in record:
        break
return records
```

A.4 Dictionary compilation

For the dictionary compilation we use the manually classified datasets to create mono-, bi-, and trigram dictionaries.

Starting off the process we run compilation for all datasets:

```
data_files = [
    # [name/description, name of file containing tweets]
    ["kiro", "tweets_classified_manually"],
    ["obama", "obama_tweets_classified_manually"]
]

for item in data_files:
    # get labeled tweets
    # tweets[0] are the positive ones,
    # tweets[1] are the negative ones.
```

[1] OSEBX data: http://www.netfonds.no/quotes/paperhistory.php?paper=OSEBX.OSE&csv_format=csv

```
    tweets = get_tweets_from_manually_labeled_tweets(
             classification_base + item[1])

    compile_monogram_dictionaries(tweets, item[0])
    compile_bigram_dictionaries(tweets, item[0])
    compile_trigram_dictionaries(tweets, item[0])
```

Mono-, bi-, and trigrams creation parts are quite similar:

```
for text in tweets[0]:
    positive_dict.extend(text.split(" "))
# negative
for text in tweets[1]:
    negative_dict.extend(text.split(" "))

filename += "-monogram"
save_dictionaries(positive_dict, negative_dict, filename)
```

Where the 'positive_dict =' would be the only part changing.
Bigram:

```
bigrams_list = get_bigrams_from_text(text)
positive_dict = positive_dict + bigrams_list
```

Trigram:

```
trigrams_list = get_trigrams_from_text(text)
positive_dict = positive_dict + trigrams_list
```

We get the bigrams and trigrams by:

```
# if we have no words to work with, return empty list.
text = clean_text(text)
if not text:
    return []

bigrams_list = []
# for all bigram tuples
for tup in bigrams(text.split(' ')):
    # find the tuple texts
    # (u'text1', u'text2')
    m_obj = re.search(r'\(u\'(.+)\', u\'(.+)\'\)',
                      str(tup).decode())
    if m_obj:
        # combine tuple parts to bigram
        word = m_obj.group(1) + " " + m_obj.group(2)
```

```
16          # if bigram don't exist
17          if word not in bigrams_list:
18              # add it to list.
19              bigrams_list.append(word)
20  return bigrams_list
```

Where we essentially generate sets of words in tuples and then extract the combined word by string conversion and regex.

Remove duplicate words

Then we conveniently remove duplicates between the positive and negative dictionary before we write the dictionaries to file.

```
1   primary_dictionary = get_lines_from_file(
2       primary_dictionary_name)
3   secondary_dictionary = get_lines_from_file(
4       secondary_dictionary_name)
5
6   duplicate_words = []
7   # for all words in the primary dictionary.
8   for pw in primary_dictionary:
9       #print w
10      # for all words in the secondary dictionary.
11      for sw in secondary_dictionary:
12          # if primary word equals secondary word
13          if pw == sw:
14              # remove word from both dictionaries.
15              primary_dictionary.remove(pw)
16              secondary_dictionary.remove(sw)
17              duplicate_words.append(pw)
18              # as we don't have duplicates within a list,
19              # we skip to next pw
20              break
21
22  # rewrite the updated lists to file
23  write_array_entries_to_file(primary_dictionary,
24                              primary_dictionary_name)
25  write_array_entries_to_file(secondary_dictionary,
26                              secondary_dictionary_name)
27  write_array_entries_to_file(duplicate_words,
28                              "duplicate-words")
```

A.5 Sentiment Classification

For both classifiers we have utility code for loading data and dictionaries, but the important parts are the classification. Which is shown here.

A.5.1 Word Count

For each tweet we do the same thing. Count words and find the difference between positive and negative words. Here is the code for classifying a tweet.

```
# sanitize text.
tweet = sanitize_tweet(tweet)

# get word count for tweet
word_count = len(tweet.split(' ')) * 1.0

# get word count of pos and neg words.
pos = get_word_count(positive_dict, tweet) / word_count
neg = get_word_count(negative_dict, tweet) / word_count

# storing the polarity value
polarity.append(pos - neg)

if polarity[-1] == threshold:
    edge_cases += 1

# adding sentiment value (True/False),
# for the last classified tweet.
if polarity[-1] > threshold:
    # positive tweet
    results.append(True)
else:
    # negative tweet
    results.append(False)
```

A.5.2 Using Classifiers

The execution of the classification can be described in the following steps.

Classifier initialization
First we have to initialize the classifier, then train it. We send the class of

the kernel as an input argument, such that *classifier_class* is 'SklearnClassifier(LinearSVC())'.

```
# get the training set.
training_set = nltk.classify.apply_features(
    extract_features_from_text,
    tweets)

# create the classifier.
classifier = classifier_class.train(training_set)
```

Where the extract_features_from_text function is important. Here the 'dictionary' variable is the combined list of the kiro-monogram-negative.txt and kiro-monogram-positive.txt dictionaries.

```
dictionary = get_list_of_possible_words_in_tweets()
document_words = set(text)
features = {}
# more precision to iterate the word of a tweet then the
# whole dictionary.
# extracts all words that are both in the dictionary and
# in the tweet
for word in document_words:
    features['contains(%s)' % word] = (word in dictionary)
return features
```

Classification
The code for classifying a tweet is quite simple.

```
# runs classify() on the given classifier class in the
# nltk library.
results.append(classifier.classify(
    extract_features_from_text(tweet)))
```

A.6 Trend aggregation

The trend aggregation is a split process. Where the mining of tweets, sorting, classification, and trend compilation happens in separate steps. Most important is the trend compilation. The sorting process only reads tweets and stores them the file that corresponds with the posting date of the tweet. Mining is done on the same principles as described earlier, only with other search words.

A.6.1 Compilation

First we get positive and negative tweets from the dataset. And store that in a dict, with the filename, or day, as key.

```
pos, neg = 0, 0

# get tweets for this day.
trend_day_tweets = [ast.literal_eval(tweet) for tweet in
    codecs.open(trend_base + trend_day_filename,
                'r',
                "utf-8").readlines()]

# sentiment for each tweet.
sentiment = []

# for tweets in this day
for tweet in trend_day_tweets:
    # get words of two or more characters.
    tweet_text = [e.lower() for e in sanitize_tweet(
          tweet['text']).split() if len(e) >= 2]

    # get sentiment
    sentiment.append(classifier.classify(
        extract_features_from_text(tweet_text)))

    # if the last tweet was positive.
    if sentiment[-1]:
        pos += 1
    else:
        neg += 1

return [pos, neg, pos + neg]
```

Then we transform the tweet data to fit the structure of finance data.

```
def transform_tweet_data():
    #Date,close,high,low,open,volume
    #'trend-Apr-29': {'neg': 21, 'pos': 101, 'tot': 122},
    #print "data,h-l,pos,neg,open,tot"
    trend_days = get_tweet_trend_data()
    keys = sorted(trend_days.keys())
    date = ""
    for i in range(1, len(keys)):
```

```
9           if "Apr" in keys[i]:
10              date = "201404" + str(keys[i].split('-')[2])
11          elif "May" in keys[i]:
12              date = "201405" + str(keys[i].split('-')[2])
13          volume = trend_days[keys[i]]['tot'] * 1.0
14          # (positive_tweets / total_amount_of_tweets)*scale
15          high = (trend_days[keys[i]]['pos'] / volume) * 1000
16          # (negative_teets / total_amount_of_tweets)*scale
17          low = (trend_days[keys[i]]['neg'] / volume) * 1000
18          openv = 0
19          close = high - low
20          print str(date) + "," + str(close) + "," + str(high) \
21                + str(",") + str(low) + "," + str(volume / 10) \
22                + "," + str(0)
```

A.7 Comparison

For the comparison of trend graphs we have acquired some code. What we need is the MA and the ADX.

The MA is calculate by:

```
1 def movingaverage(values, window):
2     weigths = np.repeat(1.0, window) / window
3     smas = np.convolve(values, weigths, 'valid')
4     return smas  # as a numpy array
```

And for the ADX we have this:

```
1  def ADX():
2      window = 14  # modified by kiro
3      PositiveDI, NegativeDI = calcDIs(window)  # modified by kiro
4
5      xxx = 0
6      DXs = []
7
8      while xxx < len(date[1:]):
9          DX = 100 * ( (abs(PositiveDI[xxx] - NegativeDI[xxx])
10                       / (PositiveDI[xxx] + NegativeDI[xxx])))
11
12         DXs.append(DX)
13         xxx += 1
14
```

```
15      #print len(DXs)
16      ADX = ExpMovingAverage(DXs, window)  # modified by kiro
```

The hundred and some lines used to plot the fancy graphs can be seen in the code file: https://github.com/magnuskiro/master/blob/master/code/trend/ma_adx_plotter.py.

A.8 Issues

The quality of the code in general is quite good. But whether or not the logic works out all the time is uncertain. Some of the functions and methods can also be improved.

Operating System Impediments
There might be some OS related specifics that we do not know about. Linux of the Debian family has been used. So we expect that you know what you are doing when testing the code on Windows or Mac.

If at all a problem, it will be with pre installed packages that will not show up in the requirements or library parts discussed earlier. Thats because we do not know which packages was preinstalled.

Also some Linux tools has been used in scripts. Those will not work on Windows, but might on Mac.

Appendix B

Web resources

List of potential places to get further information.

http://hum.csse.unimelb.edu.au/emnlp2013/papers.html

http://neuro.imm.dtu.dk/wiki/Twitter_sentiment_analysis

http://provalisresearch.com/products/content-analysis-software/wordstat-dictionary/sentiment-dictionaries/

http://www3.nd.edu/~mcdonald/Word_Lists.html

http://hopey.netfonds.no/paperhistory.php?paper=STL.OSE&csv_format=txt

http://sentdex.com/

Appendix C

Tweet Data Structure


```
{
  u'contributors': None,
  u'truncated': False,
  u'text': u'WO2013149663A1 Estimating Anisotropic Resistivity Of A
Geological Subsurface $STO #G01V #G01V11 http://t.co/yyPFEJSdIj',
  u'in_reply_to_status_id': None,
  u'id': 390051769780142080,
  u'favorite_count': 0,
  u'source': u'<a href="http://w.pat.tc" rel="nofollow">TwittlyDumb</a>',
  u'retweeted': False,
  u'coordinates': {
    u'type': u'Point',
    u'coordinates': [
      5.7326363,
      58.9645836
    ]
  },
  u'entities': {
    u'symbols': [
      {
        u'indices': [
          77,
          81
        ],
```

```
        u'text': u'STO'
      }
    ],
    u'user_mentions': [

    ],
    u'hashtags': [
      {
        u'indices': [
          82,
          87
        ],
        u'text': u'G01V'
      },
      {
        u'indices': [
          88,
          95
        ],
        u'text': u'G01V11'
      }
    ],
    u'urls': [
      {
        u'url': u'http://t.co/yyPFEJSdIj',
        u'indices': [
          96,
          118
        ],
        u'expanded_url': u'http://w.pat.tc/WO2013149663A1',
        u'display_url': u'w.pat.tc/WO2013149663A1'
      }
    ]
  },
  u'in_reply_to_screen_name': None,
  u'in_reply_to_user_id': None,
  u'retweet_count': 0,
  u'id_str': u'390051769780142080',
  u'favorited': False,
```

```
  u'user': {
    u'follow_request_sent': False,
    u'profile_use_background_image': True,
    u'default_profile_image': False,
    u'id': 163877216,
    u'verified': False,
    u'profile_text_color': u'333333',
    u'profile_image_url_https': u'https://si0.twimg.com/profile_images
/2309783804/355j4shhjrh4rqb5vsys_normal.jpeg',
    u'profile_sidebar_fill_color': u'DDEEF6',
    u'entities': {
      u'url': {
        u'urls': [
          {
            u'url': u'http://t.co/apqPEHN3aC',
            u'indices': [
              0,
              22
            ],
            u'expanded_url': u'http://w.pat.tc',
            u'display_url': u'w.pat.tc'
          }
        ]
      },
      u'description': {
        u'urls': [

        ]
      }
    },
    u'followers_count': 299,
    u'profile_sidebar_border_color': u'C0DEED',
    u'id_str': u'163877216',
    u'profile_background_color': u'C0DEED',
    u'listed_count': 8,
    u'profile_background_image_url_https': u'https://abs.twimg.com/
images/themes/theme1/bg.png',
    u'utc_offset': 32400,
    u'statuses_count': 247688,
```

```
    u'description': u'New patent information from WIPO.
IPC-based hashtags for realtime subject searching.',
    u'friends_count': 203,
    u'location': u'Tsukuba, Japan',
    u'profile_link_color': u'0084B4',
    u'profile_image_url': u'http://a0.twimg.com/profile_images/
2309783804/355j4shhjrh4rqb5vsys_normal.jpeg',
    u'following': False,
    u'geo_enabled': True,
    u'profile_banner_url': u'https://pbs.twimg.com/profile_banners/
163877216/1359154591',
    u'profile_background_image_url': u'http://abs.twimg.com/images/
themes/theme1/bg.png',
    u'screen_name': u'w_pat_tc',
    u'lang': u'en',
    u'profile_background_tile': False,
    u'favourites_count': 10,
    u'name': u'World Patents Mapped',
    u'notifications': False,
    u'url': u'http://t.co/apqPEHN3aC',
    u'created_at': u'Wed Jul 07 14:08:23 +0000 2010',
    u'contributors_enabled': False,
    u'time_zone': u'Tokyo',
    u'protected': False,
    u'default_profile': True,
    u'is_translator': False
  },
  u'geo': {
    u'type': u'Point',
    u'coordinates': [
      58.9645836,
      5.7326363
    ]
  },
  u'in_reply_to_user_id_str': None,
  u'possibly_sensitive': False,
  u'lang': u'en',
  u'created_at': u'Tue Oct 15 09:49:23 +0000 2013',
  u'in_reply_to_status_id_str': None,
```

```
u’place’: {
  u’full_name’: u’Stavanger, Rogaland’,
  u’url’: u’https://api.twitter.com/1.1/geo/id/dee2255bd015b52c.json’,
  u’country’: u’Norway’,
  u’place_type’: u’city’,
  u’bounding_box’: {
    u’type’: u’Polygon’,
    u’coordinates’: [
      [
        [
          5.5655417,
          58.884420999999996
        ],
        [
          5.8687141,
          58.884420999999996
        ],
        [
          5.8687141,
          59.0608787
        ],
        [
          5.5655417,
          59.0608787
        ]
      ]
    ]
  },
  u’contained_within’: [

  ],
  u’country_code’: u’NO’,
  u’attributes’: {

  },
  u’id’: u’dee2255bd015b52c’,
  u’name’: u’Stavanger’
},
u’metadata’: {
```

```
    u'iso_language_code': u'en',
    u'result_type': u'recent'
  }
}
```