# Sentiment Analysis

## Förmågan att göra opinionsundersökningar i realtid med hjälp av Twitter

Inledning 1

Syfte 1

Avgränsning 1

Sentiment Analysis 2

Definition av Sentiment Analysis 2

Historia 2

Användningsområden 3

Twitter – vad är det? 3

Följa användare 4

Innehåll 4

Format 4

Natural Language Processing(NLP) 5

Textklassifikation 5

Språkmodellen 7

Maskininlärning 7

Bag of Words i Python 8

Informationsinsamling 8

Informationsutvinning 11

Data-insamling 12

Naive Bayes nätverk 13

# Inledning

Detta är ett arbete inriktar sig på Sentiment Analysis och hur stora communitys som Facebook och Twitter använder denna teknik för att se i realtid vad dess användare skriver om. Med hjälp av SA kan man alltså i realtid se hur hela världens twitterflöde är genom att använda sig utav olika nyckelord. Det har visat sig att om man analyserar tweets som innehåller nyckelord om t ex en fotbollsmatch, säg Manchester Utd – Arsenal och söker twitterflöden innehållande de nyckelorden och sedan analyserar ifall det är positiva/negativa tweets. Skulle det vara mer positiva tweets om Manchester Utd än det laget dem möter vid en given match har man sett att Man Utd oftast vinner. Detta för att den totala datan som analyseras i realtid är så enormt stor och fansen och experterna som twittrar positivt/negativt om matchen eller ett lag har sällan fel.

Ett av problemen med SA när man granskar twitterflöden är sarkasm, det är väldigt svårt för en dator att förstå om ett inlägg på Twitter är sarkastiskt.

Sentiment Analysis på Twitter och andra sociala verktyg kan ge dig, till skillnad från förut, en fokus grupp på över en miljon människor vilket man såklart förstår är något helt otroligt(<http://www.youtube.com/watch?v=YGlhCCwYVh4>).

## Syfte

Syftet med detta projektarbetet är att försöka ge en grundläggande insikt om vad Sentiment Analysis är, hur det fungerar, vad det finns för verktyg tillgängliga och hur man i Python kan bygga sitt egna Sentiment Analysis verktyg.

## Avgränsning

Trots att Sentiment Analysis(SA) är relativt nytt finns det otroligt mycket material om det. Inom SA används även en rad olika tekniker. För att göra detta arbete hanterbart inom tidsramen och utrymmet för detta projektarbete har jag valt att inrikta mig på dessa tekniker som behövs för att kunna utföra SA: Data Mining, Natural Language Processing och Machine Learning. Jag kommer att förklara hur de är uppbyggda och ge ett exempel på ett eget SA-verktyg jag skapat i Python. För att samla in data att analysera har jag valt att använda Twitters API som gör det möjligt att streama tweets i realtid. Jag har då valt att ha ett avsnitt i arbetet som förklarar hur Twitter fungerar och vad det är för något.

# Sentiment Analysis

## Definition av Sentiment Analysis

Sentiment Analysis(SA) kan beskrivas som ett sätt att ta reda på människors åsikter om alltifrån politik och ekonomi till filmer eller andra produkter som personer köper (Feldman, 2013). Sentiment Analysis(SA) är alltså en ihop slagning av flera olika element inom Artificiell Intelligens. Översätter man Sentiment till Svenska betyder det ungefär *känsla*, alltså känslan i språket eller känslan som uppstår när man läser en viss text. Det går i grunden ut på att man använder *Natural Language Processing*(NLP), *Information retrieval*(som är en del av NLP) och *Computational Linguistics* för att bestämma polariteten(positivt, neutralt eller negativt) hos ett specifikt objekt. Det går alltså ut på att avgöra attityden i ett textstycke och på så sätt se trender från allmänheten i till exempel det sociala nätverket Twitter.

## Historia

Sentiment Analysis(SA) är en relativt ny teknik och den första riktiga forskningen kring Sentiment Analysis gjordes av Minqing Hu och Bing Liu år 2004. De försöker med hjälp av Opinion mining, som betyder att man samlar in data från exempelvis 1000-tals olika recensioner. Allt eftersom e-handel ökar och hemsidor som amazon.com, cdon.com och e-bay.com låter användare själva recensera olika produkter så samlade (Hu & Liu, 2004) in en massa olika recensioner. Dessa e-handelssidor låter även användare betygsätta recensioner så de kunde enkelt säga om en recension var positiv eller negativ. De använde sig utav NLP(förklaras mer i detalj nedan) för gå igenom alla recensioner där NLP identifierar ordklasserna efter hur de fungerar i satserna (Hu & Liu, 2004). Detta var startskotten för SA och idag finns det flera stora företag som använder sig utav SA för att i realtid kunna få feedback på till exempel en ny produkt eller reklamkampanj.

## Användningsområden

Dagens samhälle kretsar oerhört mycket kring olika typer av social media som Twitter, Facebook, bloggar och andra forum där folk uttrycker åsikter och synpunkter. Detta är en guldgruva för företag eller vissa individer som snabbt vill ha reda på hur folket tycker om till exempel en produkt, en reklamkampanj eller valkampanjer. SA erbjuder dessa organisationer att få otroligt snabb feedback från en otroligt stor mängd data, det är som att ha en fokusgrupp på flera miljoner människor i realtid (Feldman, 2013).

Man brukar klassificera meningar till två principiella klasser med hänsyn till subjektivitet: objektiva meningar som innehåller faktiskt information och subjektiva meningar som innehåller explicita åsikter, synpunkter och tro om specifika enheter(saker) (Feldman, 2013).

# Twitter – vad är det?

Twitter är en online tjänst som låter människor interagera med varandra genom att skicka så kallade ”tweets” som är textmeddelanden som är begränsade till 140 tecken. Man måste vara en registrerad användare för att kunna läsa och posta tweets medan icke registrerade användare endast kan läsa dem. Man kan komma åt Twitter via deras hemsida twitter.com, SMS eller genom deras mobil-app (Twitter Inc., 2014).

År 2012 var Twitter populärt över hela världen och hade då 500 miljoner registrerade användare som tillsammans postade 340 miljoner tweets per dag. Twitter hade då även hand om 1.6 miljarder sökningar per dag. Det nuvarande rekordet för antal tweets sattes den 1 januari 2013 i den Japanska tidzonen då antalet tweets var 33 388 tweets per sekund. Det senast uppmätta datan visar att 200 miljoner användare skickade över 400 miljoner tweets dagligen där nästan 60% av tweetsen var skickade från mobila enheter (Lunden, 2012).

## Följa användare

När man väl är registrerad på Twitter kan man följa andra användare som använder Twitter, detta kallas för att man *följer* någon. Kontra att en användare kan ha *följare.*

## Innehåll

När en användare på Twitter postar en Tweet är den automatiskt inställd på att visas för alla, dock kan användare själva välja att vissa Tweets endast ska synas för användare som följer det Twitter-kontot. Under en två-veckors period i augusti 2009 utförde Pear Analytics en undersökning där man försökte ta reda på vad användare skriver på Twitter. Man delade in de olika typerna av Tweets i 6 olika kategorier och resultatet blev följande:

* Meningslöst babbel – 40%
* Konversationer – 38%
* Retweets – 9%
* Själv upplyftande tweets – 6%
* Spam – 4%
* Nyheter – 4%  
  (Kelly, 2009)

## Format

Användare kan gruppera poster(Tweets) beroende på vad Tweeten handlar om genom att använda sig utav hashtags(#), till exempel om något händer på LiU kan man då skriva #LiU i sin Tweet och andra användare kan då söka efter hashtaggen #LiU och få fram alla tweets som handlar om LiU(innehåller #LiU). På samma sätt kan man använda ”@” följt av ett användarnamn för att nämna eller svara på en annan användares tweet. Man kan ”omposta” ett meddelade från en annan användare och då dela en annans tweet på sin egen sida, denna funktionen kallas för *retweet* och symboliseras genom att det står ”RT” i tweeten.

Eftersom att Twitter har satt en maxgräns på 140 tecken per tweet används mycket förkortningar och slang i tweets. Man har även börjat använda förkortningar för URL-adresser som t ex bit.ly, goo.gl och twitters egna t.co. Detta gör att man kan posta URL-adresser i sina Tweets utan att de tar upp för mycket plats (http://en.wikipedia.org/wiki/Twitter).

# Natural Language Processing(NLP)

Som nämnt ovan skickas det över 400 miljoner tweets på Twitter varje dag. Detta blir otroligt mycket data och nästan all av den datan är skrivit i naturligt språk, med undantag för en del tweets som innehåller mycket förkortningar och bild-länkar, men ändå otroligt mycket vanligt innehåll med. En agent som vill införskaffa sig kunskap måste förstå(i alla fall delvis) det tvetydliga och oklara språk som människor faktiskt använder sig utav. För att lösa detta problem från perspektivet att införskaffa sig data om vad allmänheten har för åsikter om en viss person, företag, stad, politik eller en helt vanlig digitalkamera(eller någon annan produkt) måste man använda sig utav följande: text klassifikation, informationsinsamling(information retrieval) och informationsutvinning (Russel & Norvig, 2010, ss. 860-865).

## Textklassifikation

När man utför textklassifiering , även kallat kategorisering behöver man någon typ av text för att sedan bestämma till vilken av de fördefinierade klasserna den texten går under. Språkidentifiering och genre klassificering är två exempel på textklassifikation. Även SA som detta projekt handlar om är en typ av textklassifiering men då handlar det mer om att avgöra om en text är positiv eller negativ. Till skillnad från (Russel & Norvig, 2010) som tar upp exempel på hur man kan skilja spam-mail mot vanlig mail beskrivs här istället hur man kan skilja spam-tweets(meningslöst babbel och spam) mot värdefulla tweets(ham-tweets) som har någon form av innebörd. I detta projekt är värdefulla tweets sådana tweets som innehåller objektiv information och till viss del subjektiv information (Feldman, 2013).

**Ham-tweets:** Former NFL player found dead in Florida hotel room after complaining of stomach illness: http:\/\/t.co\/Q0KooFkjmT - (@ABC)

**Ham-tweets:** Florida State quarterback Jameis Winston has won the Manning Award, given to the nation's top quarterback.

**Ham-tweets:** Florida State redshirt freshman Jameis Winston wins Manning Award, honoring nation's top QB (ESPN)

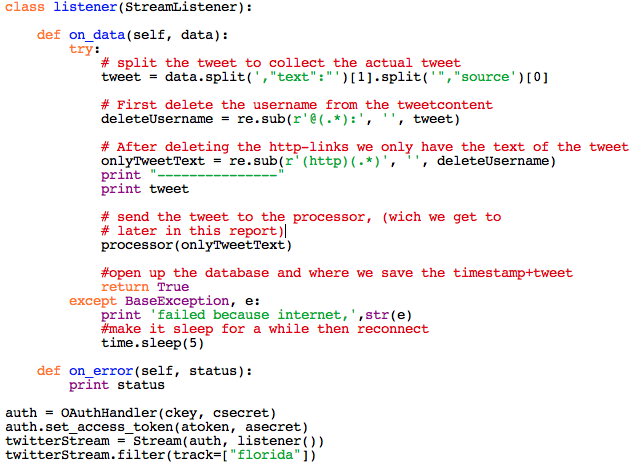
**Ham-tweets:** I'm not even a Florida State fan, but the fact is that team is going to be better next year.

**Spam-tweets:** Can't wait to go florida!!!!!!!

**Spam-tweets:** @TescoBabyClub: Follow us and RT if you would like to #win a trip to @Disneyland Florida for you and your family

**Spam-tweets:** @ginnyyiam: Can it just be February 14th please #florida\u201d I WANNA GO HURRY UP CALENDAR HURRY PLZ PLZ I WANNA GET MY TAN ON

*Dessa tweets är hämtade 2014-01-08 23:00 genom att köra ett Python-script som jag skrivit. Scriptet ser ut på följande sätt:*



*Script 1.1 – Ett script som ”lyssnar” på alla tweets som skickas och tar ett nyckelord för att scanna av tweets och sparar de tweets som innehåller nyckelordet.*

Från dessa tweets kan man börja få en idé om vad som just nu pågår i Florida. Dessa tweets är hämtade i realtid och när detta skrivs(2014-01-08 23:00) verkar många twittra om en quarterback som vunnit ett pris och även många som twittrar om att en före detta NFL spelare hittats död. Medan en del andra spam-tweets som inte har något riktigt innehåll kan filtreras bort. För att göra detta behöver man en samling av vad som kan definieras som värdefulla tweets(ham-tweets) och spam-tweets för att kunna analysera tweetsen i en *Supervised learning model*. Som man kan se verkar spam-tweets innehålla mycket versaler och utropstecken. Man skulle kunna skapa en tecken *n*-gram modell över spam-tweets och ham-tweets eller så kan man för hand skapa olika egenskaper som till exempel ”antalet utropstecken som avslutar en mening” (Russel & Norvig, 2010).

### Språkmodellen

Det finns två vägar att gå när man pratar om klassifikation, de är båda komplement till varandra. Med språk-modellens synsätt så definierar man en språkmodell med ett *n*-gram för **P**(*Tweet*|*spam-tweet*) som tränar på spam-tweets, och en modell för **P**(*Tweet*|*ham-tweet*) som tränar på ”bra” tweets. När det är gjort kan man klassificera en ny tweet med hjälp av Baye’s regel:

där *P*(*c*) är uppskattat endast genom att man adderar antalet spam-tweets och ham-tweets. Denna teknik fungerar bra för att upptäcka spam (Russel & Norvig, 2010).

### Maskininlärning

Det andra sättet är genom maskininlärning där man representerar meddelandet(tweeten) i en grupp utav kännetecken och/eller värd-par för att sedan använda en klassifikationsalgoritm *h* till känntecksvektorn **X**. Man kan göra språkmodellen och maskininlärningssättet kompatibla genom att tänka att *n*-grammen är kännetecken. De olika kännetecknena är ord i vokabuläret: ”a”, ”aardvark”, …, och värdet är antalet gånger som ett specifikt ord förefaller i meddelandet/tweeten. Om det finns 100 000 ord i språkmodellen så har känneteckensvektorn en längd på 100 000 men eftersom tweets ofta är korta och ett ord kanske bara finns med en gång har de känneteckensvärde på 0. Denna unigram representationen har kommit att kallas för **bag of words**-modellen. Man kan tänka på modellen som att man lägger en massa ord från träningskorpusen i en påse för att sedan välja ett ord i taget. I vilken ordning orden ligger har därmed ingen betydelse. En unigram-modell ger alltså samma sannolikhet för en text oavsett vilken ordning orden ligger i. *N*-gram modeller som är mer sofistikerade kan dock ge olika sannolikheter beroende på vilken ordning orden ligger i (Russel & Norvig, 2010, s. 866). Med de mer sofistikerade *N*-gram modellerna som bigram och trigram så är kännetecknena i kvadrat eller att man tar kubikroten ur kännetecknena och på så sätt kan man lägga in andra icke-*n*-grams kännetecken som till exempel tiden då tweeten är skickad, om det är en URL eller bild i tweeten, ID nummer(användarnamn) för den som skicka tweeten eller om den användaren tidigare har skickat spam-tweets eller ham-tweets och mycket annat. När man har valt ut ett antal kännetecken så kan man applicera dessa på en rad olika ”supervised learning” tekniker, de populäraste för textkategorisering är *k*-nearest neighbors, support vector machines, decision trees, naive Bayes och logistic regression. Alla dessa har tidigare applicerats på spam-igenkänning och har oftast en säkerhet på 98%-99% och har man en riktigt bra design över kännetecken kan man uppnå 99.9% säkerhet (Russel & Norvig, 2010).

#### Bag of Words i Python

Från att ha samlat in en otrolig mängd med Tweets går det att göra en simplare SA som ger en överblick det valda nyckelordet som i detta fall kommer att vara ”Obama”. För att samla ihop en ”Bag of Words” används två stycken textfiler med positiva och negativa engelska ord där det är totalt 2006 positiva ord och 4783 negativa ord (Hu & Liu, 2004). Efter att ha lagt in dessa ord i en SQL-databas med värdena 1 för positiva och -1 för negativa kördes dessa ord mot <http://thesaurus.com>, som är en hemsida där man kan se synonymer till ord, för att bygga på databasen med fler positiva/negativa ord. Scriptet samlade alltså in synonymer för alla de positiva och negativa orden, lade in dem i databasen och satte respektive värde på ordet. Detta gav en databas med totalt 9857 ord med positiva/negativa ord, övriga ord som inte är med klassas då per automatik som neutrala med värde 0.

### Informationsinsamling

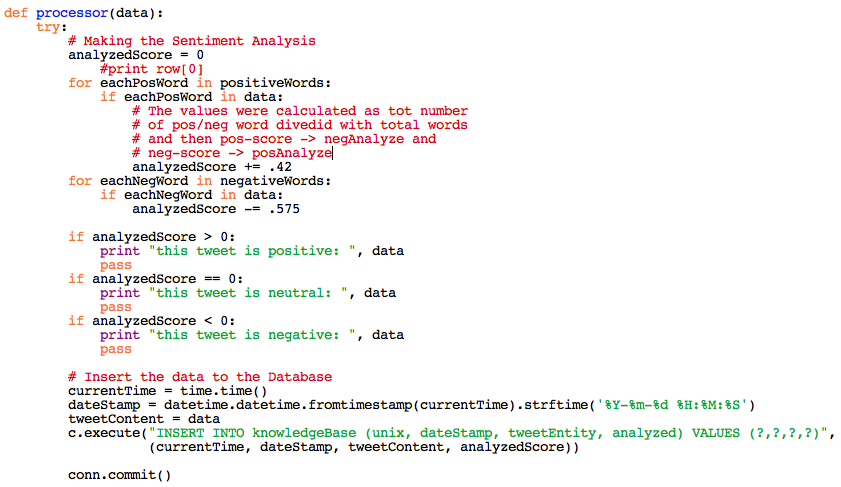
Med informationsinsamling(IR) menas att man letar efter dokument eller i detta fall tweets som är relevanta för det nyckelordet vi söker efter. De mest välkända exempel på detta är sökmotorer på internet som Google. Där kan en användare skriva in ett sökord som [AI book] i sökmotorn och få fram en lista med relevanta sidor. Här förklaras hur en sådan sökmotor fungerar och hur den istället har använts för att söka upp relevanta tweets. Man kan karakterisera en IR-modell på följande sätt:

1. **En korpus av dokument.** Varje system måste bestämma vad det är man ska behandla som ett dokument: en paragraf, en sida, brödtext eller i detta fall en tweet.
2. **Frågor(queries) i det format eller språk som krävs.** En fråga eller nyckelord kan vara en lista med olika ord eller så kan det vara en specifik fras. Det kan innehålla Boolean operatorer som [Obama AND Economy] eller icke-Boolean operatorer som [Obama NEAR Economy].
3. **En samling med resultat.** Detta är en samling av dokument som IR-systemet anser vara relevant till frågan/nyckelordet. I detta SA-projekt består det av en SQL-databas full med insamlade tweets.
4. **En presentation av resultaten.** Detta kan vara så enkelt som en lista med dokument eller databas med alla tweets.

De tidigare IR-systemen använde sig utav en **Boolean keyword model.** Varje ord i ett dokument/tweet ses som ett Booleansk kännetecken som är sant om tweeten innehåller det ordet och falsk om det inte gör det. Så det kännetecknande ordet ”Florida” blir då sant för de ovanstående ham och spam tweetsen men falsk för många andra tweets som inte innehåller ordet ”Florida” (Russel & Norvig, 2010).

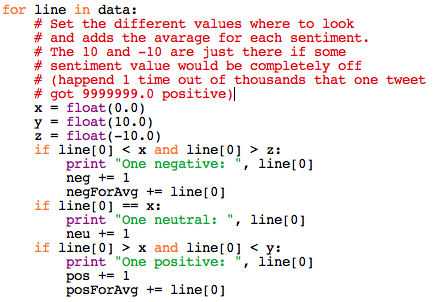
För att samla in information om ett valt nyckelord körs *Script 1.1*.Scriptet är ett python-script som använder sig utav modulen **Tweepy** (Roesslein, 2009) och **SQLITE3** för att söka efter tweets i realtid och sedan spara ner dem i en SQL-databas. Totalt antal tweets som samlades in uppgick till 2430 st under loppet av 15 min(mellan 2014-01-08 02:30-02:45) med ”obama” som nyckelord.

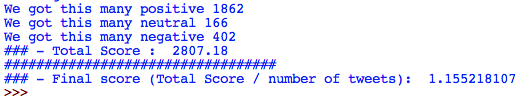
Vidare kördes nedanstående script för att räkna ut det sentimenta värdet för varje enskild tweet som innehöll ordet ”obama”.



*Script 1.2. Detta script analyserar varje enskild tweet mot bag-of-words, finner den ett positivt ord i en tweet adderas .42 till analyzedScore och subtraherar .575 för negativa ord i tweeten. Därefter lagras de i en databas.*

Efter att ha samlat in alla tweets och analyserat dem följt av att lagra dem i en SQL-databas körs ytterligare ett *Script 1.3*.





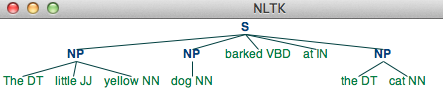
*Script 1.3. räknar ut vad det sentimentala värdet i genomsnitt blev för alla tweets och under scriptet syns resultatet från körningen.*

Man kan då säga att det sentimentala värdet i tweets innehållande ”obama” under den givna tiden var i genomsnitt positivt med ett värde på där 0 är neutralt.

## Informationsutvinning

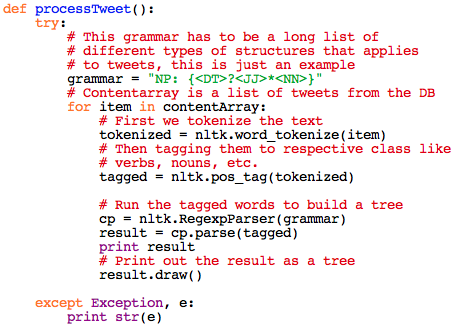
Det finns en del olika metoder man kan använda sig av när man ska utvinna information från en text. Det enklaste sättet kallas för ”**attribute-base extractio**” som antar att hela texten refererar till ett enda objekt och att uppgiften är att extrahera attribut utifrån det objektet. För att göra detta så måste man definiera en mall för varje attribut man vill extrahera. Mallen är definierade med hjälp utav reguljära uttryck (Russel & Norvig, 2010). Man kan skriva reguljära uttryck direkt in i ett python-script. I *Script 1.1* kan man se några reguljära uttryck som byter ut(tar bort) användarnamn och länkar i en tweet för att få en renare och enklare text.

Ett lite mer avancerat system är ”**relation extraction**” systemet som kan hantera flera objekt och relationen mellan dem. Ett typiskt relationsbaserat extraktionssystem är FASTUS som är specialiserat på nyhetstexter som handlar om företagssammanslagningar och uppköp (Russel & Norvig, 2010). Det verktyg som använts i detta projekt heter *Natural Language Toolkit*(NLTK).Ett exempel taget från (Bird, Klein, & Loper, Natural Language Processing with Python, 2009) använder meningen ”the little yellow dog barked at the cat” och genom att använda NLTK modulen i python och *Script 1.4* får man ut följande resultat:



Denna mening skickade jag via en tweet som enligt vissa nyckelord sparades i databasen och analyserades genom att tweeten först delades upp i ”Tokens”. Det andra steget taggar orden till respektive ordklass med NLTK’s inbyggda pos\_tagsom (Bird, Klein, & Loper, Natural Language Processing with Python, 2009) syns i *Script 1.4*. Det tredje steget har hand om ”**basic groups**” vilket betyder nominalfraser, verbfraser osv. vilket resulterade i trädet ovan (Russel & Norvig, 2010). Målet med det tredje steget är alltså att man ska kunna använda en formell grammatik för att beskriva strukturer i en oändlig massa olika meningar eller tweets och hur man ska representera dessa med hjälp av syntax träd som exemplet ovan (Bird, Klein, & Loper, Natural Language Processing with Python, 2009). Detta görs även det i *Script 1.4* på dessa rader:  
cp = nltk.RegexpParser(grammar)

result = cp.parse(tagged)



*Script 1.4. Detta script tokeniserar en text, taggar orden till respektive ordklass för att sedan jämföras med ”grammar-listan”. Den grammar som visas i scriptet måste såklart byggas ut för att kunna hantera flera olika typer av meningsuppbyggnader.*

# Diskussion