LUNDS UNIVERSITET
Ekonomihögskolan

Institutionen för informatik

Negativa attityder och misslyckade Open Source-projekt

En attitydanalys av commit-meddelanden i digitala förvaringsplatser som prediktor för misslyckade Open Source-projekt

Kandidatuppsats 15 hp, kurs SYSK16 i Informatik

Författare: Erik Berggren
Axel Ungewitter

Handledare: Björn Svensson

Rättande lärare: Christina Keller
Benjamin Weaver

Negativa attityder och misslyckade Open Source-projekt: En attitydanalys av commit-meddelanden i digitala förvaringsplatser som prediktor för misslyckade Open Source-projekt

ENGELSK TITEL: Negative attitudes and Open Source-project failure: A sentiment analysis of commit messages in software repositories as a predictor for Open Source-project failure

FÖRFATTARE: Erik Berggren, Axel Ungewitter

UTGIVARE: Institutionen för informatik, Ekonomihögskolan, Lunds Universitet

EXAMINATOR: Christina Keller, Professor

FRAMLAGD: maj, 2020

DOKUMENTTYP: Kandidatuppsats

ANTAL SIDOR: 58

NYCKELORD: projektmisslyckande, Open Source-projekt, attityder, digitala förvaringsplatser, repository, Git

SAMMANFATTNING (MAX. 200 ORD):
I föreliggande studie utformades en modell utifrån hypotesen att negativa attityder i commit-meddelanden predicerar misslyckande i Open Source-projekt. Denna modell testades kvantitativt med ett bekvämlighetsurval av digitalt förvarade projekt eller så kallad repositories (n = 990) på GitHub. Ett egenutvecklat verktyg i form av en webbapplikation användes till att utvinna samtliga data och ett instrument för attitydanalys, VADER, användes för värdering av text. En enkel logistisk regressionsanalys visade att modellen hade dålig passform vilket indikerar att den inte kan användas till att predicera projektmisslyckande i Open Source. Således förkastades hypotesen och bakomliggande faktorer till resultatet diskuteras. I vidare forskning rekommenderas att modellen omarbetas, förslagsvis genom att utgå från annan textdata eller lägga till andra projektegenskaper såsom antal *issues*.

Innehåll

Figurer

Tabeller

1 Introduktion

1.1 Inledning

Projektmisslyckande är ingen ovanlig företeelse, inte minst inom mjukvaruutveckling (Linberg, 1999). I de årliga rapporterna från *The Standish Group* estimeras det att runt 30 % av alla mjukvaruprojekt avbryts i förtid, ca 50 % får dubbelt så hög kostnad och enbart en bråkdel av de projekt som lyckas sker inom utsatt tidsram och budget (The Standish Group International, 1994). I liknande studier har statistiken sedan mitten av 90-talet varit ungefär densamma - andelen framgångsrika projekt har ökat med liten marginal (El Emam & Koru, 2008; Rubinstein, 2007; The Standish Group International, 2015). Således är antalet misslyckade projekt många medan lyckade projekt är få. Mängder av forskning tar upp möjliga orsaker till varför mjukvaruprojekt på ett eller annat sätt misslyckas (Linberg, 1999; Pinto & Mantel, 1990; Pressman, 1998).

Återkommande orsaker till projektmisslyckande är att det bl.a. saknas adekvat projektledning, definieras orealistiska och otydliga krav sett till tillgängliga resurser, förekommer bristfällig kommunikation mellan intressenter och aktörer, samt uppstår missnöje hos utvecklare m.m. (Verner, Sampson & Cerpa, 2008). Närmare bestämt är *missnöje bland utvecklare* en vanlig konsekvens när det uppstår konflikter och negativitet i sociala sammanhang (Liu, Chen, Chen & Sheu, 2011). Missnöje går att upptäcka förhållandevis enkelt i organisationer, eftersom det sker ett utbyte av information i form av formella och informella möten mellan projektledare och utvecklare (Alvesson, 2014). Därmed finns även nödvändiga åtgärder för att motverka missnöje och andra negativa attityder, som till exempel tekniska och byråkratiska kontrollverktyg (Alvesson, 2014).

I dagens digitala samtid kan mjukvaruprojekt existera under andra förutsättningar, t.ex. vara helt internetbaserade och tillåta i princip vem som helst att bidra utan någon form av kompensation (Hertel, Niedner & Herrmann, 2003; Kogut & Metiu, 2001). En allt vanligare projektform är *Open Source* (von Hippel, 2001). Projekt som bedrivs genom Open Source blir ledda, utvecklas och används av “online communities”, och till skillnad från traditionella projekt, som måste uppfylla affärsmål från intressenter och kunder, kan dessa existera endast för att sprida kunskap och innovation samt dela med sig av ny mjukvara (Hertel, Niedner & Herrmann, 2003; von Hippel, 2001).

Hur kommer det sig då att vissa Open Source-projekt, som bedrivs av en online community och inte använder affärsmål som ska leda till kommersiell eller organisatorisk framgång, uppfattas vara misslyckanden?

Många misslyckanden i organisationer beror på brister i samarbete (Tarricone & Luca, 2002). Dessa brister är olika motivationsnivåer, avsaknad av öppen kommunikation och undermåliga mellanmänskliga färdigheter, såsom dålig hänsyn till andras känslor och negativa *attityder* (Tarricone & Luca, 2002). Dessa brister i samarbete och negativa attityder mellan gruppmedlemmar synliggörs i verbala samtal eller genom icke-verbala tecken såsom ansiktsuttryck (Hancock, Landrigan & Silver, 2007). Däremot blir det svårare att identifiera

känslolägen i textbaserad kommunikation, t.ex. chattmeddelanden (Hancock, Landrigan & Silver, 2007), eftersom det just saknas icke-verbala tecken. Detta är särskilt relevant för Open Source då mycket av kommunikationen mellan användare, utvecklare eller andra bidragare sker i textform via email, på forum eller andra plattformar (Guzzi, Bacchelli, Lanza, Pinzeger & Van Deursen, 2013).

Det finns därför ett behov av att utveckla mätverktyg för att bättre förstå positiva och negativa attityder i textmeddelanden, särskilt om attityder kan vara tecken till misslyckade samarbeten som sker digitalt, t.ex. i Open Source-projekt.

1.2 Teori

Denna studie är icke-experimentell med en kvantitativ ansats. Syftet med studien är att utforma en modell med hjälp av de hypoteser som kommer presenteras senare i uppsatsen. Studien kommer se ut på följande vis:

Efter föregående inledning av problemområde kommer först en (1.2) teoridel. Denna teoridel fungerar som en litteraturgenomgång och presenterar tidigare forskning kring de områden som är relevanta för studiens syfte. Därefter formuleras de (1.3) hypoteser som blir grunden för utformningen av en modell som testas i studien. Slutligen kommer (2) metod, (3) resultat, (4) diskussion och (5) slutsats i sedvanlig ordning.

1.2.1 Open Source

Open Source *(öppen källkod)* är mjukvara som huvudsakligen utvecklas, distribueras och blir underhållen av privatpersoner (till skillnad från leverantörer eller företag) med avsikt att dela med sig av källkod till allmänheten samt bidra till innovation (Lakhani & Von Hippel, 2004). Open Source-utvecklare skapar och delar källkod med varandra utan kompensation, men motiveras exempelvis av att arbetet är givande och roligt samt att de får äran att bidra med högkvalitativa insatser till ett Open Source-projekt (Lakhani & Von Hippel, 2004; Lerner & Tirole, 2002). Från ett ekonomiskt perspektiv så kan negativa påföljder uppstå för utvecklare som verkar inom Open Source, som t.ex. en avsaknad av inkomst och en bristande motivation att utföra sina primära arbetsuppgifter (Lerner & Tirole, 2002). Utvecklingen av Open Source mjukvara ger däremot utvecklare fördelar i form av ökade programmeringskunskaper och möjligheter till framtida jobberbjudanden (Lerner & Tirole, 2002).

Även företag och organisationer kan fritt använda Open Source-mjukvara i sina verksamheter (Hauge, Ayala & Conradi, 2010) och i en del fall är det till och med företag som ansvarar för eller står bakom mjukvaran (Stam, 2009), till exempel Facebooks React (Staff, 2016). Vissa organisationer anpassar sina affärsmetoder efter Open Source och väljer att upprätta ett samarbete med fristående Open Source-communities (Stol & Fitzgerald, 2014; Wesselius, 2008). Om företag dock väljer att utnyttja fri källkod för kommersiella syften kan det resultera i missnöje hos Open Source-communities (Dahlander & Magnusson, 2005). Ett exempel på en sådan situation var när *Apple* först lanserade sitt Open Source-operativsystem *Darwin* år 2000 (West, 2003). Vissa delar av operativsystemet använde proprietär programvara (*stängd källkod)* som krävde hårdvara från *Apple* för att fungera, samt använde en Open Source-licens (*BSD-licens*) som tillät kommersiella företag att använda och sälja den fria källkoden som stängd

källkod (West, 2003). Dessa anledningar resulterade i en kontrovers bland de Open Source-communities som utvecklade operativsystemet (West, 2003).

En annan kontrovers som väckte oro bland Open Source-communities var när Open Source-databashanteraren *MySQL* blev förvärvad av företaget *Sun*, som i sin tur blev förvärvad av *Oracle* (Nyman & Lindman, 2013). En följd av förvärven var att *MySQL* fick många proprietära licenser och funktioner, vilket ledde till att de ursprungliga grundarna av *MySQL* utvecklade en ny version (*MariaDB*) som var fullständigt Open Source (Nyman & Lindman, 2013).

Dahlander och Magnusson (2005) förklarar att missnöjen kan uppstå från krockar mellan Open Source-organisationers och communities motivationer. Organisationer som jobbar med Open Source drivs mer av en ekonomisk och teknologisk motivation, snarare än social motivation - vilket är mer populärt bland självständiga utvecklare (Bonaccorsi & Rossi, 2006). Därför måste organisationer göra sitt bästa för att förstå kulturer och värderingar som existerar i Open Source-communities.

Utvecklingen av Open Source-mjukvara bygger på att det finns ett intresse för samarbete, att hjälpa, bidra med och dela med sig av ny kunskap till andra privatpersoner online (Dabbish, Stuart, Tsay & Herbsleb, 2012; Dahlander & Magnusson, 2005). Det behövs därför möjligheter för privatpersoner att få tillgång till och bevara digitala objekt, vilket kan göras med hjälp av digitala förvaringsplatser för mjukvara, eller s.k *software repositories* (Jantz & Giarlo, 2005).

1.2.2 Digitala förvaringsplatser (repositories) & facilteringstjänster

Digitala förvaringsplatser för mjukvara (software repositories) har lett till nya möjligheter när det gäller utveckling och datalagring i Open Source-miljöer (D'Ambros, Gall, Lanza & Pinzger, 2008). Dessa förvaringsplatser faciliteras som tjänster av företag, däribland återfinns exempelvis SourceForge, Bitbucket, Gitlab och GitHub. Tjänsterna skiljer sig en aning åt i sina tillvägagångssätt, men har i grund och botten samma huvudsakliga syfte - vilket är att versionshantera kod, möjliggöra kollaborativa insatser och på ett eller annat sätt förvara privata och offentliga mjukvaruprojekt (D'Ambros et al. 2008). Digital preservation av mjukvara har öppnat upp för en helt ny värld av samarbete mellan utvecklare och bidrar till en utveckling av nya arbetsmetoder, visualiseringstekniker och analyser (D'Ambros et al. 2008).

Med hjälp av dessa faciliteringstjänster som erbjuds av t.ex. GitHub kan utvecklare över hela världen ta del av källkod, manipulera den och föreslå ändringar som efter kontroller (så kallad *code reviews* eller kodgenomgångar) kan inkorporeras i nästa version av mjukvaran (Kalliamvakou, Gousios, Blincoe, Singer, German & Damian, 2014; Tsay, Dabbish, & Herbsleb, 2014). På sätt och vis tillåter plattformen en form av demokratiskt samarbete där kodkvalité sätts i fokus (Tsay, Dabbish & Herbsleb, 2014). Utvecklingsformen är speciellt fördelaktig i Open Source-projekt eftersom den huvudsakliga utvecklingen inte alltid görs av mjukvarans officiella ägare, underhållare eller egna utvecklare (Kalliamvakou et al. 2014). Istället sker utvecklingen i samband med att användare och externa utvecklare uppmärksammar och rapporterar problem genom att skapa så kallad *issues*, och kommer med praktiska förslag på nya funktioner eller lösningar på gamla problem i form av *merge/pull requests* i mjukvarans digitala förvaringsplats (Kalliamvakou et al. 2014).

Eftersom facilteringstjänster lagrar historiken av alla *commits* (bidrag, ändringar eller tillägg i källkoden) som gjorts i repositories blir det enkelt att följa utvecklingen av ett Open Source-projekt (Alali, Kagdi & Maletic, 2014). När det görs en commit måste det följa med ett kortfattat

commit-meddelande som beskriver och dokumenterar de ändringar som gjorts (Alali, Kagdi & Maletic, 2014; Santos & Hindle, 2016). Dessa meddelanden varierar i längd och innehåll, och enligt en undersökning som gjordes av Alali, Kagdi och Maletic (2014) innehöll de vanligaste commit-meddelanden i 9 olika repositories ord såsom "fix", "add", "test", "file" och "new", och vanligtvis kombinerades orden med varandra. Oftast indikerar inte commit-meddelandets innehåll ifall kodens uppbyggnad är bra eller inte (Santos & Hindle, 2016) så därför har meddelandet inte någon större påverkan när en person vill upptäcka en talangfull utvecklare. Däremot så har de sociala aspekterna som existerar vid samarbete över Open Source-plattformarna fortfarande en stor betydelse när ägare väljer bidragsgivare till ett Open Source-projekt (Tsay, Dabbish & Herbsleb, 2014).

1.2.3 Känslor i textbaserad kommunikation

För att kommunicera online och använda plattformar som förvarar Open Source-projekt krävs det på ett eller annat sätt att *textbaserad kommunikation* används (Guzzi et al. 2013). Enligt Hancock, Landrigan och Silver (2007) har individer utvecklat och anpassat sina uttrycksförmågor i textbaserade meddelanden för att lättare förmedla känslor och bedöma andras tonlägen. Hancock, Landrigan och Silver (2007) undersökte ifall personer kunde visa positivitet och negativitet i textbaserade interaktioner utan att göra tydliga referenser till sitt känsloläge (som t.ex. "*Jag är glad*", som förmedlar tydligt att en person är *glad*). Hancocks, Landrigan och Silver (2007) resultat visade att de språkliga uttryck som bäst visade positivitet var när flera ord och utropstecken användes, medan de uttryck som visade negativitet var ett frekvent användande av negationer och ord som var negativt känsloladdade.

Idag är människor så vana vid textbaserad kommunikation att det är förhållandevis enkelt att förstå sociala koder och texter på mikrobloggar såsom Facebook samt Twitter, och det blir mer vanligt att personliga uppgifter samt arbetsmaterial görs tillgängliga för allmänheten (Dabbish et al. 2012). Även GitHub, som främst erbjuder tjänster såsom digitala förvaringsplatser, har integrerat sociala funktioner som tillåter användare att samarbeta utan organisatoriskt sammanhang eller tillhörighet (Dabbish et al. 2012). Dabbish et al. (2012) fann att det som kännetecknar GitHubs kommunikation är främst feedback från utvecklare, och att deras användare är drivna av engagemang för innovation, gemenskap och till och med egen uppmärksamhet.

1.2.4 Attityder

För att bättre förstå känslor i textmeddelanden och framförallt *attityder*, har nya analysverktyg utvecklats. En *attityd* kan förklaras som - enligt Charles Darwin (2009) - *uttryck* för känslor som går att *iaktta*. Darwin (2009) menade också att personers attityder kan variera i styrka, vilket betyder att vissa attityder kan vara mer eller mindre positiva och negativa. På ett liknande sätt definierar Tesser och Shaffer (1990) en attityd som antingen en positiv eller negativ reaktion mot en stimulus, som t.ex. en händelse, person eller ett koncept. På ett liknande sätt beskriver Gilovich, Keltner, Chen och Nisbett (2016) en attityd vara bestående av en affektkomponent, dvs. känsla, som kan variera i negativitet och positivitet.

Attitydanalys, ett instrument avsett för mätning av *attityder* (även mer känt som *sentiment analysis*), är en automatiserad metod som kan användas för att identifiera styrkan av positiva samt negativa känslolägen i korta texter, såsom paragrafer från recensioner (Pang & Lee, 2008). Attitydanalysen blev runt 2001 ett populärt verktyg för att undersöka forskningsfrågor om bl.a.

åsikter och beteenden i textform, och uppkom i samband med den ökande användningen av maskininlärning och dataset som blev tillgängliga online (Pang & Lee, 2008).

Attitydanalyser har till och med använts för att upptäcka attityder och känslolägen som framkommer i texter inom mjukvaruutveckling (Guzman, Azócar, & Li, 2014; Jurado & Rodriguez, 2015; Sinha, Lazar & Sharif, 2016). I en studie av Sinha, Lazar och Sharif (2016) genomfördes en attitydanalys för att ta reda på ifall utvecklare tenderar att skriva känsloladdade ord i commit-meddelanden. Författarna extraherade data från 28,466 repositories och gjorde en attitydanalys på totalt 2,130,474 commit-meddelanden (Sinha, Lazar & Sharif, 2016).

Enligt Sinha, Lazar & Sharif (2016) hade 74.7 % av commit-meddelandena ett neutralt känsloläge, som oftast var fyllda med tekniska termer som endast beskrev de ändringar i mjukvaran som gjorts. Resterande commit-meddelanden visade att 18.1 % var negativa, och 7.2 % var positiva, vilket sammantaget innebar mer än dubbelt så många negativa meddelanden jämfört med positiva (Sinha, Lazar & Sharif, 2016). Enligt Dabbish, Stuart, Tsay och Herbsleb (2012) kan utvecklare förstå vad andra försöker åstadkomma genom att hänvisa till tidigare problem i källkoden, vilket kan förklara varför det inte behövs mer utvecklade svar än tekniska termer.

Det finns dock en uppfattning att känslor har stor betydelse i mjukvaruutveckling och bör övervakas för att bättre upptäcka och förstå problem i projekt (Jurado & Rodriguez, 2015). I en studie av Wrobel (2013) undersöktes känslors påverkan på produktivitet bland mjukvaruutvecklare, och det visades att negativa attityder försämrade utvecklares produktivitet. Frustration var den attityd som var mest riskabel, medan ilska var den enda negativa attityd som bidrog till en aning förhöjd produktivitet (Wrobel, 2013). Attityder hos utvecklare har därmed visats påverka mjukvaruprojekt på olika sätt (Jurado & Rodriguez, 2015).

1.2.5 Projektmisslyckanden inom mjukvaruutveckling

Ett *projektmisslyckande* kan definieras som ett projekt vars budget, resultat och affärsmål inte levde upp till förväntningarna, men också som ett projekt som blivit nedlagt (Linberg, 1999). Om ett projekt ska uppfattas som ett misslyckande behöver endast en av dessa två definitioner uppfyllas, och eftersom mjukvaruprojekt sällan lever upp till förväntningar på grund av höga krav från beställare, händer det att mjukvaruprojekt tolkas av projektledare och utvecklare som misslyckanden (Linberg, 1999). Nedlagda projekt leder vanligtvis till förödande ekonomiska konsekvenser för organisationer och företag (Ahonen & Savolainen, 2010). Därmed blir det essentiellt för projektledare att ha klara och välgrundade visioner om sina affärsmål samt att de kan förmedla dessa visioner till sina medarbetare. Om dessa krav uppfylls tenderar det att bli bättre arbetsmoral och mer positiva attityder hos medarbetarna i en organisation (Aga, Noorderhaven & Vallejo, 2016).

Enligt Linberg (1999) bör en projektledare ha en klar bild av vad som uppfattas vara ett misslyckande respektive en succé inom ett projekt, så att realistiska krav kan specificeras och att projektets medlemmar blir ense om vad som kan tolkas vara framgångsrikt. Linberg (1999) undersökte med hjälp av intervjuer hur utvecklare definierar och ser på framgångsrika samt misslyckade projekt, och hur deras perspektiv skiljer sig från de krav som sätts av mjukvaruindustrin. Utvecklare anser att ett lyckat mjukvaruprojekt både ska uppfylla krav från industrin (dvs budgeten, tidsschemat och arbetsprestationer), men framförallt att produkten erhåller en hög kvalité som uppskattas av kunder (Linberg, 1999).

För att bättre förstå vad som orsakar problem i ett projekt så har tecken till misslyckande undersökts (Kappelman, McKeeman & Zhang, 2006). I en studie av Kappelman, McKeeman och Zhang (2006) blev 138 erfarna projektledare tillfrågade om att rangordna de vanligaste fallgroparna för ett IT projekt, och de fann att vanliga tecken var brist på engagemang och kommunikation från ledningen samt odokumenterade kravspecifikationer, milstolpar, förfallodatum och riskhantering.

1.2.6 Framgång och misslyckande i Open Source-projekt

De processer som finns i Open Source-projekt skiljer sig däremot från vanliga mjukvaruprojekt på många vis, och borde därför inte ha samma tecken på projektmisslyckande. Open Source-projekt använder i de flesta fall inte traditionella förberedande modeller för mjukvaruutveckling och behöver inte uppfylla krav från administrativa avdelningar (Llanos & Castillo, 2012). Istället förbereds projektets krav genom konversationer mellan utvecklare via mejl, chattfunktioner eller internetforum, och anpassas efter diskussioner (Llanos & Castillo, 2012; Guzzi et al. 2013). Därför går inte vanliga tecken till misslyckade IT-projekt, såsom kommunikation från ledning och förfallodatum, att applicera på Open Source lika väl.

Vad som definierar ett framgångsrikt respektive misslyckat Open Source-projekt är debatterbart eftersom digitala utvecklingsmiljöer, affärsmodeller och dess trender är i ständig förändring (West & Gallagher, 2006), men enligt Subramaniam, Sen och Nelson (2009) är ett Open Source-projekt framgångsrikt om det befinner sig i ett sent stadium och har fått ett starkt intresse bland andra utvecklare och användare som aktivt bidrar till projektet. Om ett Open Source-projekt har kommit långt i utvecklingsprocessen och fått många aktiva användare så är sannolikheten större för utvecklarna att få feedback på sina bidrag, vilket kan leda till att mjukvaran förbättras ytterligare i form av buggfixar och ny funktionalitet (Subramaniam, Sen & Nelson, 2009).

Det kan då argumenteras för att ett misslyckat Open Source-projekt är motsatsen till ett framgångsrikt projekt - när samarbete mellan utvecklare och användare försvunnit, underhåll och utveckling avstannat, eller att projektet övergivits och blivit lagt på is, vilket till exempel sker när ett Open Source-projekt *arkiveras*. På ett liknande sätt menar Ehls (2017) att ett misslyckat Open Source-projekt är ett projekt vars antal aktiva utvecklare har minskat. Tecken till varför utvecklare väljer att lämna ett Open Source-projekt är bl.a långvariga brister i produktens funktioner och utvecklarnas samarbetsförmågor (Ehls, 2017). Även plötsliga händelser, som t.ex. förändringbeslut kring strategier och slutmål som går emot utvecklares tidigare visioner, får aktiva medlemmar att lämna ett Open Source-projekt (Ehls, 2017).

För att förstå varför Open Source-projekt misslyckas och *överges* intervjuade Coelho och Valente (2017) utvecklare som just arbetat med nedlagda Open Source-projekt. Författarna kategoriserade intervjusvaren och tog fram en lista med övergripande anledningar till varför Open Source Projekt misslyckas, vilket var följande:

(1) att de blev utkonkurrerade av andra projekt vars syfte var densamma,

(2) att det som utvecklades blev utdaterat,

(3) brist på tid och (4) intresse,

(5) utdaterad programmeringsteknik,

(6) lågt underhåll av projektet,

(7) konflikter mellan utvecklare. (Coelho & Valente, 2017)

Den sistnämnda anledningen i listan, dvs. konflikter mellan utvecklare, är även en vanlig fallgrop i vanliga mjukvaruutvecklingsprojekt. Konflikter mellan utvecklare eller medarbetare leder ofta till försämrad kommunikation och lagbeslut, ökad frustration samt lägre arbetsmoral (Acuña, Gómez & Juristo, 2009; Liu et al. 2011).

1.2.7 Konfliktrisker i Open Source-projekt

Online-samarbeten tillåter att nya verk och projekt kan skapas utan att hindras av intressenter, men det händer lätt att konflikter uppstår inom virtuella communities, eftersom sättet att samarbeta samt kommunicera blir helt och hållet datormedierat och textbaserat (Elliott & Scacchi, 2003; Kollock & Smith, 1996). Kommunikation som sker online för med sig problem såsom irrelevanta kommentarer, urartade online-bråk mellan utvecklare, och kränkande språk (Kollock & Smith, 1996) som typiskt inte hade uppstått på ett kontor. De största anledningarna till att konflikter uppstår online är bl.a. de stora kulturskillnaderna, olika intresseområden och ambitionsnivåer, samt de otydliga och oftast anonyma identiteterna som personer använder online (Kollock & Smith, 1996).

Hur dessa konflikter ska hanteras i Open Source rent praktiskt råder det en allmän kunskapsbrist om, men Elliott & Scacchi (2003) konstaterar att den tillgängliga historiken av textmeddelanden och diskussioner online kan hjälpa medlemmar i ett Open Source-projekt att bättre förstå problemen som finns. Information kring projektets normer och kultur kan användas till att analysera situationer och komma med lösningsförslag (Elliot & Scacchi, 2003). Med hjälp av den enorma mängd data som genereras borde information kunna utvinnas och användas till att upptäcka och förhindra problem som riskerar leda till projektmisslyckande.

1.2.8 Sammanfattning

Utifrån tidigare presenterad litteratur framgår det att en av anledningarna till misslyckade mjukvaruprojekt kan härledas till konflikter, vilket grundas i utvecklares samarbetsförmågor och attityder gentemot varandra och projektet som helhet. Specifikt har misslyckande inom Open Source-projekt även definierats som projekt som inte längre underhålls, tappat sin användarbas, avbrutits i förtid eller som på andra sätt tvingats lägga ned på grund av bl.a. föråldring, bristande intresse och interna konflikter. I Open Source-sammanhang är dessa projekt, i fruset tillstånd, tillgängliga genom sina digitala förvaringsplatser och dess utvecklingsdata är redo att utvinnas för djupgående analyser. Detta öppnar upp för nya sätt att undersöka projektstatus, mäta dess hälsa, osv, vilket i sin tur kan användas till att styra projektet i rätt riktning och på så vis undvika misslyckande.

1.3 Syfte och hypotes

Syftet med den här studien är att testa en modell som ämnar predicera projektmisslyckande utifrån data som hämtas från Open Source-projekts digitala förvaringsplatser. I föreliggande studie hypotetiseras att ett samband existerar mellan utvecklares attityder i commit-meddelanden i Open Source-projekt och dess status.

Begreppet *attityd* förklaras som ett *iakttagbart uttryck av känslor i olika styrkor* (Darwin, 2009; Gilovich et al. 2016), och i modellen kommer attityder på liknande vis motsvara de iakttagbara attityder och styrkor som återfinns i *textbaserad kommunikation*.

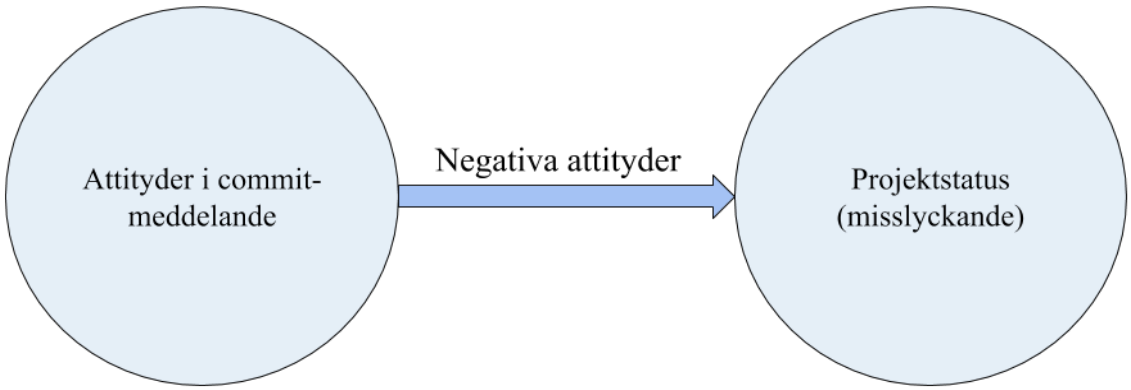
Open Source-projektets *status* är i modellen polariserad som antingen *framgångsrik* eller *misslyckad*. Ett misslyckat Open Source-projekt kan argumenteras för att vara ett projekt som blivit lagt på is på grund av minskat intresse från utvecklare samt användare, vilket på ett eller annat sätt har lett till att det *övergivits*. Övergivna projekt skulle kunna likställas med *arkivering av ett repository*, som enligt GitHub Help (2020) beskrivs som att projektets ägare signalerar för intressenter att projektet inte längre aktivt underhålls och inte borde användas. Därmed kan det sägas att ett *misslyckat Open Source-projekt* kan likställas med att ett repository blivit *arkiverat*.

Som ett resultat av ovanstående argumentation formas följande hypoteser:

H_0 (Nollhypotes): Det finns inget samband mellan attityder (oberoende variabel) och projektstatus (beroende variabel).

H_A (Alternativ hypotes): Det finns ett samband mellan attityder (oberoende variabel) och projektstatus (beroende variabel) som säger att negativa attityder i commit-meddelanden predicerar projektmisslyckande i Open Source.

I linje med den alternativa hypotesen ovan utformas följande modell:



Figur 1: Negativa attityder i commit-meddelanden predicerar projektmisslyckande

2 Metod

2.1 Design

Studien hade en icke-experimentell kvantitativ ansats och tillämpade en *tvärsnittsdesign*. I en tvärsnittsstudie inhämtas data vid endast ett tillfälle för att upptäcka mönster och förekomster av det som undersöks (Levin, 2006). Oftast samlas det in data med hjälp av intervjuer och enkäter (Levin, 2006), men i den här studien samlades det in data från software repositories som redan fanns tillgängliga online och denna data blev föremål för statistisk dataanalys.

2.2 Urval

Urvalet av repositories gjordes med hänsyn till eventuella restriktioner på möjliga plattformar (t.ex. GitHub, GitLab, BitBucket och Azure DevOps). Den plattform som valdes i det här fallet var GitHub, som tillåter mest utvinning av data och som dessutom är den största med över 44 miljoner repositories och 40 miljoner användare (GitHub, 2019; GitHub API v3, 2020). GitHub begränsar antalet förfrågningar i timmen till 30 (för att motarbeta överbelastning och missbruk av Githubs servrar) utan någon form av autentisering (GitHub API v3, 2020), vilket för den här studiens ändamål var alldeles för lite då utvinningen av data hade tagit för lång tid. Däremot, med hjälp av autentisering som exempelvis OAuth, en form av säker inloggning som tillåter tredjepartsapplikationer att få begränsad åtkomst till användarens resurser (Hardt, 2012), tillåts istället 5000 vanliga förfrågningar i timmen och 30 förfrågningar i minuten som nyttjar GitHubs sök-API (GitHub API v3, 2020).

För utvinning av data med hjälp av GitHubs API används sökparametrar. De sökparametrar som användes i denna studie var följande:

- Över 250 stjärnmarkeringar.
- Publik status.
- Arkiveringsstatus.

Stjärnmarkeringar (en indikator för popularitet) fungerar som favoritmarkeringar för GitHub-användare, och användes för att filtrera bort mindre och oetablerade repositories med få uppdateringar och utvecklare.

Synligheten på dessa repositories behövde vara publika så att allmänheten kan ta del av koden, vilket ökar sannolikheten för att det just handlar om Open Source. Motsatsen, dvs. privata repositories, går inte att utvinna data från.

Dessutom valdes repositories utifrån arkiveringsstatus för att finna Open Source-projekt som var aktiva respektive arkiverade. När ett repository arkiveras går det inte längre att lägga till ändringar i källkoden eller bidra till dess community med nya issues, merge och pull requests, commits eller kommentarer (GitHub Help, 2020).

För att jämföra arkiverade och icke-arkiverade repositories gjordes två stickprov. Det ena stickprovet gjordes för datan från arkiverade projekt medan det andra stickprovet gjordes för data från aktiva projekt. Antalet repositories begränsades till högsta möjliga, dvs. 100 per förfrågan, vilket återigen inte var ett aktivt val utan snarare en restriktion i GitHubs API.

För varje repository hämtades de senaste 100 commits (från dagens datum och bakåt i tiden). Utöver detta gjordes ingen vidare filtrering eller urval.

Med tanke på att det användes särskilda sökparametrar och det inte gick att kontrollera för ordningen som data utvanns, gick det inte heller att garantera ett slumpmässigt urval. Därför borde urvalet för repositories betraktas som ett bekvämlighetsurval och inget annat.

2.3 Instrument

I föreliggande studie användes ett egenutvecklat verktyg för att hämta och sammanställa data (https://gitextraction.app), samt ett instrument (VADER, https://github.com/cjhutto/vaderSentiment) för attitydanalys av ord och meningar. Det sistnämnda är sedan tidigare förankrat i forskning (Hutto & Gilbert, 2014; Islam & Zibran, 2018; Jongeling, Sarkar, Datta & Serebrenik, 2017).

Med hjälp av VADER går det att utvinna känslotillstånd ur korta texter från mikrobloggar (Hutto & Gilbert, 2014). VADER analyserar varje ord i en textrad och finner dess valens (dvs *grad av känsloläge* eller *styrka av attityd*) genom att mäta ordets polaritet och intensitet utifrån ett lexikon anpassad efter attitydanalyser (Hutto & Gilbert, 2014). Varje ord får en valens som sträcker sig från -4 (väldigt negativ känsla) till +4 (väldigt positiv känsla). Till exempel har ordet “good” en valens som är +1.9, medan ordet “horrible” har en valens som är -2.5 (Hutto & Gilbert, 2014).

Därefter får en mening ett övergripande värde (*compound score)* genom att ordens valens summeras och sen normaliseras resultatet så att den passar i en skala från -1 till +1 (Hutto & Gilbert, 2014). Om det övergripande värdet är större än +0.05 bedöms meningen ha ett positivt känsloläge, och om värdet på motsvarande sätt är mindre än -0.05 har meningen ett negativt känsloläge (Hutto & Gilbert, 2014). Ett värde som är mellan trösklarna anses ha ett neutralt känsloläge. Men att sammanställa ett övergripande värde är inte obligatoriskt vid användning av VADER, eftersom personer kan visa tvådelade känslor som pågår samtidigt. I många fall kan personers känslolägen och reaktioner från stimuli alternera mellan positivt och negativt så hastigt att olika känslor uppfattas pågå simultant (Schimmack & Colcombe, 2007), vilket kan göra att de känslor som går att utvinna från textrader blir tvetydiga. Därför kan VADER även presentera en menings positiva, negativa och neutrala värden parallellt med varandra, så att det blir lättare att utföra multidimensionella beräkningar av meningens “totala” attityd (Hutto & Gilbert, 2014).

VADER valdes eftersom det är mer anpassat efter textbaserad kommunikation på sociala medier (Hutto & Gilbert, 2014). Jämfört med andra mer etablerade och populära attitydanalysprogram såsom LIWC, som till exempel har använts för att skilja mellan lyckliga och olyckliga romantiska par genom att analysera chattmeddelanden (Hancock, Landrigan & Silver, 2007), så valdes VADER eftersom den även finner sentiment i bl.a. internetslang, uttryckssymboler och akronymer (Hutto & Gilbert, 2014). VADER visade goda resultat i en teststudie eftersom den kunde analysera text lika väl som ett urval testpersoner, och i en kontext där sociala medier användes visade den i flera fall bättre resultat än många andra attitydanalysprogram, bl.a LIWC

och ANEW (Hutto & Gilbert, 2014). Men eftersom utvecklingsmiljöer (såsom Open Source) använder många tekniska termer, så finns det däremot en risk att VADER inte kan analysera ord och meningar relaterade till mjukvaruutveckling med lika hög noggrannhet som för text från sociala medier (Islam & Zibran, 2018).

2.4 Procedur

2.4.1 Pilotstudien

Det första som gjordes var att genomföra en småskalig pilotstudie för att testa om det var möjligt att utvinna data från GitHub. Pilotstudiens datautvinningen gjordes med hjälp av Postman (https://www.postman.com/). Postman är ett verktyg för hantering och testning av API:er (applikationsprogrammeringsgränssnitt). API:er gör det lättare för olika programvaror att kommunicera och bli sammankopplade med varandra på ett strukturerat sätt (Biehl, 2016).

Samtliga sökningar på GitHub gjordes genom att använda parametrarna som redogjordes för i urvalsavsnittet. En sökning måste alltid innehålla en söksträng (query) med minst en parameter (GitHub API v3, 2020). Nedan följer ett exempel på en fullständig sökning och vad beståndsdelarna innebär (för en mer djupgående genomgång och förklaring rekommenderas GitHubs dokumentation):

https://api.github.com/search/repositories?per_page=100&q=archived:true+is:public+stars:>=250

Markerar början på anropet.
Antal repositories som går att se åt gången.
Är skiljetecken för separation av sökparametrar.
Markerar början på själva söksträngen.
Är sökparametrar.
Är definitionstecken.
Är parametervärden.

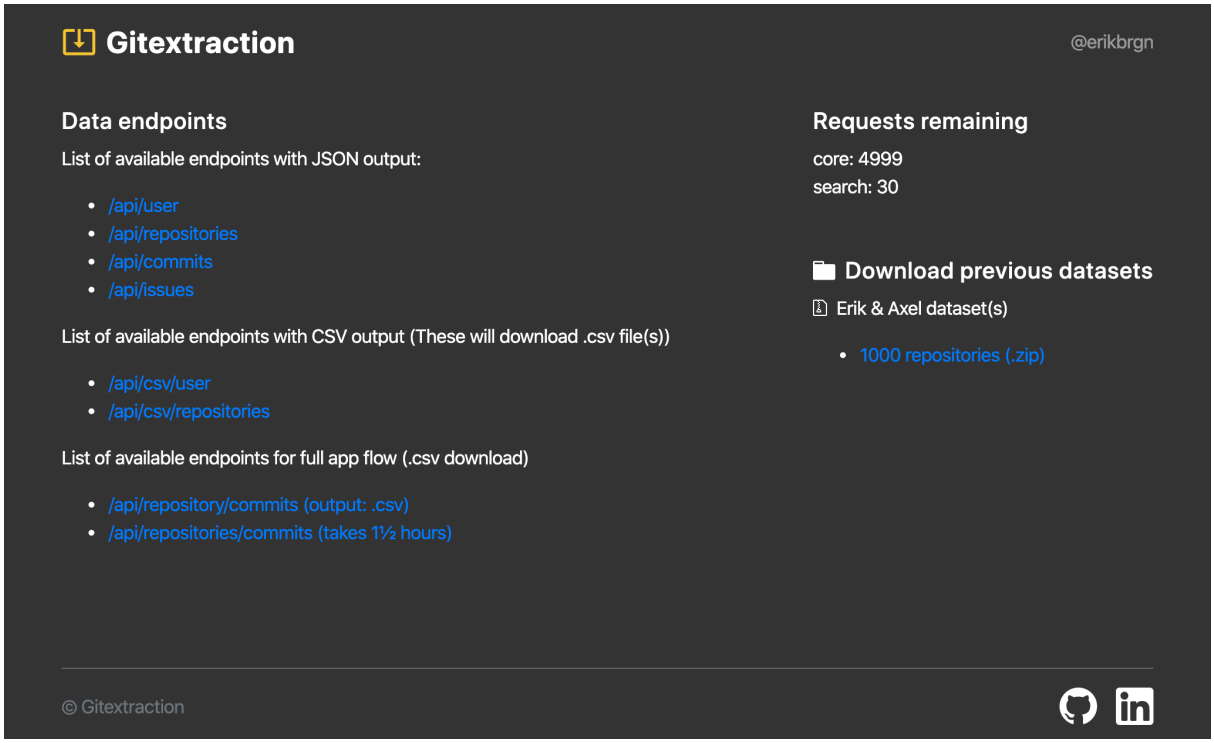
Olika sökparametrar testades för att stifta bekantskap med GitHubs API. Det hämtades bl.a. 100 repositories och senaste 100 commits för ett slumpmässigt valt repository.

Samtidigt, separat från pilotstudiens datautvinning, testades även VADER, instrumentet för attitydanalys, på ett fåtal commit-meddelanden (se tabell 1).

Tabell 1: Exempel på commit-meddelanden och hur de poängsätts av VADER. Skalan för negativitet, neutralitet och positivitet är från 0 till 1. Total attityd är från -1 till +1.

Commit-meddelande	Negativitet	Neutralitet	Positivitet	Total attityd (*compound*)
Fixed crappy implementation of templates.	.47	.53	-	-.56
Reverted temporary fix #32341.	-	1.00	-	0
Updated formatting to better reflect common practices.	-	.67	.33	.44
Readme should be good now.	-	.58	.42	.44
Ffs... Please don't use unnecessary spacing.	.38	.40	.23	-.36

2.4.2 Utvecklingen av Gitextraction



Figur 2: Användargränssnittet för Gitextraction

Efter pilotstudiens genomförande påbörjades utvecklingen av en webbapplikation som kunde användas till att utvinna större mängder data från GitHub. För denna utvinning behövde applikationen uppfylla tre funktionella kriterier:

1. Applikationen behövde kunna autentisera sig genom GitHub, t.ex. med hjälp av OAuth.
2. Applikationen behövde kunna kommunicera med GitHubs API:er.
3. Applikationen behövde integrera instrumentet VADER för att kunna göra en attitydanalys.

Det införskaffades en domän (https://gitextraction.app) som applikationen kopplades till för att främja vetenskaplig transparens, så att vem som helst ska kunna utvinna data på samma sätt som i den här studien (se avsnittet om reliabilitet). Själva applikationen utvecklades med hjälp av "Flask" (https://flask.palletsprojects.com/), ett webbramverk skrivet i programmeringsspråket Python. Sett till tids- och resursbegränsningar underlättades utvecklingen av att använda ett ramverk såsom Flask.

I utvecklingen av webbapplikationen var det första steget att implementera OAuth. Ett krav för att OAuth ska fungera är att webbapplikationen använder en säker och krypterad anslutning, vilket tillhandahölls genom att införskaffa och använda ett SSL-certifikat (en legitimationskontroll för webbsidor samt servrars säkerhet). Dessa certifikat genererades med hjälp av certbot (https://certbot.eff.org/). När certifikaten var på plats registrerades applikationen i GitHub som konsument till deras OAuth. Anledningen till varför det fanns ett krav på inloggning till GitHub var för att maximera antalet förfrågningar. Utan inloggning tilläts endast 30 förfrågningar *i timmen* och med inloggning var det istället 30 förfrågningar *i minuten*.

Det andra steget i applikationens utveckling bestod av att testa GitHubs API:er. Tre olika testanrop gjordes för att hämta data om den inloggade användaren, 100 repositories och de 100 senaste commit-uppdateringarna som fanns i det första hämtade repository:t. Samtlig data presenterades i JSON-format, eller "JavaScript Object Notation", direkt i webbläsaren.


```
{"incomplete_results":false,"items":[{"archive_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/{archive_format}
{/ref}","archived":true,"assignees_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/assignees{/user}","blobs_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/git/blobs{/sha}","branches_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/branches{/branch}","clone_url":
v0.x-archive.git","collaborators_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/collaborators{/collaborator}","comments_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/comments{/number}","commits_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/commits{/sha}","compare_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/compare/{base}...
{head}","contents_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/contents/{+path}","contributors_url":"https://ap
archive/contributors","created_at":"2009-05-27T16:29:46Z","default_branch":"moved","deployments_url":"https://api.github.com/r
archive/deployments","description":"Moved to https://github.com/nodejs/node","disabled":false,"downloads_url":"https://api.git
archive/downloads","events_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/events","fork":false,"forks":7678,"forks_count":7678,"forks_url":"https://api.github.com/repos/nodejs/node-v0.x-archiv
v0.x-archive","git_commits_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/git/commits{/sha}","git_refs_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/git/refs{/sha}","git_tags_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/git/tags{/sha}","git_url":"git:/
archive.git","has_downloads":false,"has_issues":true,"has_pages":false,"has_projects":true,"has_wiki":true,"homepage":"","hook
/nodejs/node-v0.x-archive/hooks","html_url":"https://github.com/nodejs/node-v0.x-
archive","id":211666,"issue_comment_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/issues/comments{/number}","issue_events_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/issues/events{/number}","issues_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/issues{/number}","keys_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/keys{/key_id}","labels_url":"https:
v0.x-archive/labels{/name}","language":null,"languages_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/languages","license":null,"merges_url":"https://api.github.com/repos/nodejs/node-v0.x-
archive/merges","milestones_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/milestones{/number}","mirror_url":null
archive","node_id":"MDEwOlJlcG9zaXRvcnkyMTE2NjY=","notifications_url":"https://api.github.com/repos/nodejs/node-v0.x-archive/r
```


Figur 3: Exempel på data i JSON-format

För att lättare presentera data i tabellformat gjordes samma testanrop som tidigare, men denna gång formaterades datan om till tabeller i filformatet .csv (Comma Separated Values). Detta gjordes för att möjliggöra ett bearbetnings- och importerbart filformat. Slutligen gjordes ett anrop för att hämta 100 repositories för att sedan utvinna 100 commits för varje repository. Detta visade sig dock vara problematiskt eftersom GitHubs API:er endast tillåter 30 förfrågningar i minuten. För att komma runt problemet lät vi applikationen hämta samtliga commits för 30 repositories, pausa tills antalet förfrågningar var återställda och sedan successivt fortsätta med inhämtningen.

Det tredje steget bestod av att integrera VADER i applikationen. Utgångspunkten var att instrumentet skulle göra analysen av meningarna allteftersom de inhämtades från GitHub. Eftersom attitydanalyser är anpassade för att analysera en mening åt gången behövde commit-meddelanden bli uppdelade i separata meningar. Varje mening analyserades med instrumentet, och därefter summerades resultatet för varje mening och delades med det antal meningar som fanns i varje commit. På så vis beräknades commit-meddelandets genomsnittliga attityd.

Efter att ha fått ett genomsnitt per commit behövde vi få reda på genomsnittet för ett helt repository, vilket gjordes på ett liknande sätt som tidigare. Vi adderade commit-genomsnitten och delade summan med antalet analyserade commits. Slutligen landade detta i ett genomsnitt för ett helt repository.

2.4.3 Datautvinning

I den slutliga utvinningen av data kombinerades samtliga steg till ett enhetligt flöde. Först hämtades data från 500 arkiverade repositories och därefter hämtades data från 500 icke-arkiverade repositories. För varje repository hämtades även de senaste 100 commitsen. För varje commit delades tillhörande commit-meddelande upp i meningar som var och en analyserades med hjälp av VADER. Poängen för varje mening blev summerat och delad med antalet analyserade meningar. Denna beräkning gjordes för varje commit. Poängen för varje commit lades därefter också ihop och delades med antalet hämtade commits.

Poängen för varje repository lades till i den ursprungliga listan med data från arkiverade och icke-arkiverade repositories. Proceduren avslutades med att listan konverterades till en datatabell som sedan exporterades i form av en .csv-fil. Därefter påbörjades den statistiska analysen.

2.5 Statistisk analys

2.5.1 Databearbetning (Data screening)

All data bearbetades och analyserades med hjälp av IBM SPSS Statistics Software version 26.

Det första som gjordes var att finna och filtrera bort repositories som inte var inom vår räckvidd av urval för att undvika outliers (dvs avvikande värden som skiljer sig från andra värden). Efter att ha sorterat efter de lägsta värdena i "antalet commits" så upptäcktes flera repositories som endast hade 1-10 commit-meddelanden registrerade. För att undvika extrema minimi- och maximivärden så togs dessa dataresultat bort. Därefter fick vi en absolut frekvens som var 990 repositories.

Eftersom värdena från attitydanalysen var i decimalform och uppfattades som otydliga transformerades det övergripande värdet (*compound score*) till en ny variabel där alla värden multiplicerades med 100. Istället för att använda VADERs skala, som sträcker sig från -1 till +1, så blev istället skalan -100 till +100. Denna omvandling gjordes för att underlätta läsförståelsen. För att lättare skilja på den beroende variabeln transformerades även arkivering från *false* och *true* till respektive 0 (ej arkiverad) och 1 (arkiverad) i enlighet med de rekommendationer som återfinns i SPSS Survival Manual (Pallant, 2013).

2.5.2 Dataanalys

Dataanalysen började med att finna beskrivningar av variablernas värden för att ta reda på information såsom medelvärden, standardavvikelser, snedfördelning och toppighet. De variabler som undersöktes var det transformerade övergripande värdet (*compound score* eller som sedan kallas för *attitydskalan*) samt separata medelvärden för en repositories positiva, negativa och neutrala värde. Denna analys utfördes med hjälp av ett oberoende t-test där samtliga variabler grupperades utifrån arkiveringsstatus. Därefter undersöktes ifall alla variabler var normalfördelade, och fann att det övergripande värdet visade tydligast vilket känsloläge som ett repository hade eftersom alla tre känslolägen (positivitet, negativitet och neutralitet) finns med i skalan. Därför valdes det att endast analysera vidare det övergripande värdet, dvs. skalan för attityd eller *compound score*.

Det övergripande värdet för attityden i ett projekt sattes som prediktorvariabel i modellen. Sedan utfördes en enkel logistisk regressionsanalys för att testa huruvida prediktorvariabeln kunde användas till att predicera den beroende variabeln, dvs. arkivering, eller inte.

2.6 Validitet

Validitet är ett centralt begrepp i forskningssammanhang och berör huvudsakligen hur slutsatser i en studie kan vara felaktiga på så sätt att studiens fynd skiljer sig från verkligheten (Boudreau, Gefen & Straub, 2001; Feldt & Magazinius, 2010). Validitet är inget som kan garanteras eller bevisas utan bör betraktas som ett mål som arbetas mot, t.ex. genom att tidigt upptäcka och eliminera validitetshot (Feldt & Magazinius, 2010).

Det finns olika former av validitet som är särskilt relevanta för den här studien. Det första är konstruktvaliditet, vilket handlar om hur väl det som mäts stämmer överens med det som tros mätas samt att inget annat mäts (Boudreau, Gefen & Straub, 2001; Straub, 1989). I den här studien rörde det sig om attityder hos utvecklare och projektmisslyckande i Open Source. Studien har hög konstruktvaliditet om mätinstrumentet mäter det avsedda konstruktet och inget annat. Ett sätt att öka konstruktvaliditeten i en studie är att använda ett liknande instrument för samma ändamål och jämföra resultaten, vilket kallas för konvergent validitet (Carlson & Herdman, 2012). Är resultaten snarlika finns det stöd för att instrumenten mäter det de är gjorda för.

Det andra begreppet kallas extern validitet och handlar om huruvida studiens resultat är representativt för populationen eller inte (Feldt & Magazinius, 2010; Ferguson, 2004). I det här fallet skulle detta innebära hur väl fynden representerar de projekt som inte ingick i studien. Det är en fråga om de slutsatser som görs kan generaliseras, dvs. att de gäller i sammanhang utanför studien. Ett sätt att höja generaliserbarheten är genom att göra ett slumpmässigt urval (Ferguson, 2004). I den här studien kunde detta inte säkerställas eftersom GitHubs API svarar på förfrågningar med något de kallar för “best match” (GitHub API v3, 2020). Sannolikt innebär det att urvalet formas utifrån de parametrar som skickas med i förfrågan men eftersom vi inte kunde vara helt säkra på detta gick det inte att garantera ett slumpmässigt urval vilket följaktligen påverkade studiens externa validitet.

2.7 Reliabilitet

Precis som validitet är viktigt inom forskning är begreppet reliabilitet likaså. Reliabilitet kan definieras som ett uttryck för noggrannhet eller exakthet hos ett mätinstrument (Boudreau, Gefen & Straub, 2001). Ett reliabelt instrument är konsekvent i sina mätningar. Till exempel skulle en våg som visade samma vikt för samma objekt vid upprepade tillfällen klassificerats som reliabel eller pålitlig, men om vågen istället hade visat olika vikter varje gång utan att objektets egenskaper på något sätt förändrats hade vågen bedömts som opålitlig.

I den här studien var det viktigt att VADER (instrumentet för attitydanalys) var pålitligt i värderingen av meningar. Instrumentet kan bedömas ha hög reliabilitet om det är konsekvent i sin värdering eller poängsättning av samma ord eller hela meningar. Till exempel ordet “good”, som tidigare nämnts poängsätts +1.9, måste konsekvent värderas lika om instrumentet ska kunna betraktas som reliabelt.

När det handlar om en studies reliabilitet i sin helhet är begreppen reproducerbarhet och replikerbarhet relevanta (Goodman, Fanelli & Ioannidis, 2016). Det förstnämnda, reproducerbarhet, handlar om ifall samma resultat kan erhållas vid användning av metod och data från originalstudien (Goodman, Fanelli & Ioannidis, 2016). Det andra, replikerbarhet, handlar i stort sett om samma sak förutom att det istället samlas in ny data som blir föremål för analys (Goodman, Fanelli & Ioannidis, 2016). För att stärka den här studiens reproducerbarhet finns studiens dataset tillgängligt via webbapplikationen. I enlighet med att öka replikerbarheten finns där även möjlighet att utvinna ny data på samma liknande vis som denna studie. Alternativt kan kodexempel (se appendix 1) användas till att utforma en ny applikation för liknande ändamål.

2.8 Etik

I vetenskaplig forskning är det viktigt att följa de etiska rekommendationer och även regler som satts upp. Utöver regler mot plagiat och fabricering av data, undersökningsprocessen och dataanalyser, förekommer även följande rekommendationer (Recker, 2013):

- Använd inte publicerad data utan tillkännagivande eller också opublicerad data utan godkännande eller tillkännagivande.
- Tacka samtliga inblandade, vare sig det är deltagare, kollegor eller studenter utifrån deras bidrag till studien.
- Använd inte andra forskares opublicerade artiklar, information, idéer, koncept eller data utan godkännande från ägare.
- Använd arkiverad data enbart i enlighet med källans regler och riktlinjer.

Samtliga rekommendationer är särskilt relevanta och har noga beaktats inför varje steg i den här studien. Sett till användningen av ett sentiment analysis-instrument var det viktigt att instrumentet i fråga var tillgängligt och fick användas i forskningssammanhang. Sett till utvunnen data var det inget som på något sätt bröt mot regler om anonymitet eller samtycke, eftersom en del av urvalet bestod i att hämta allmän data. Dessutom fanns det inget i den hämtade datan som kunde identifiera eller exponera enstaka individer i analysen av attityder.

3 Resultat

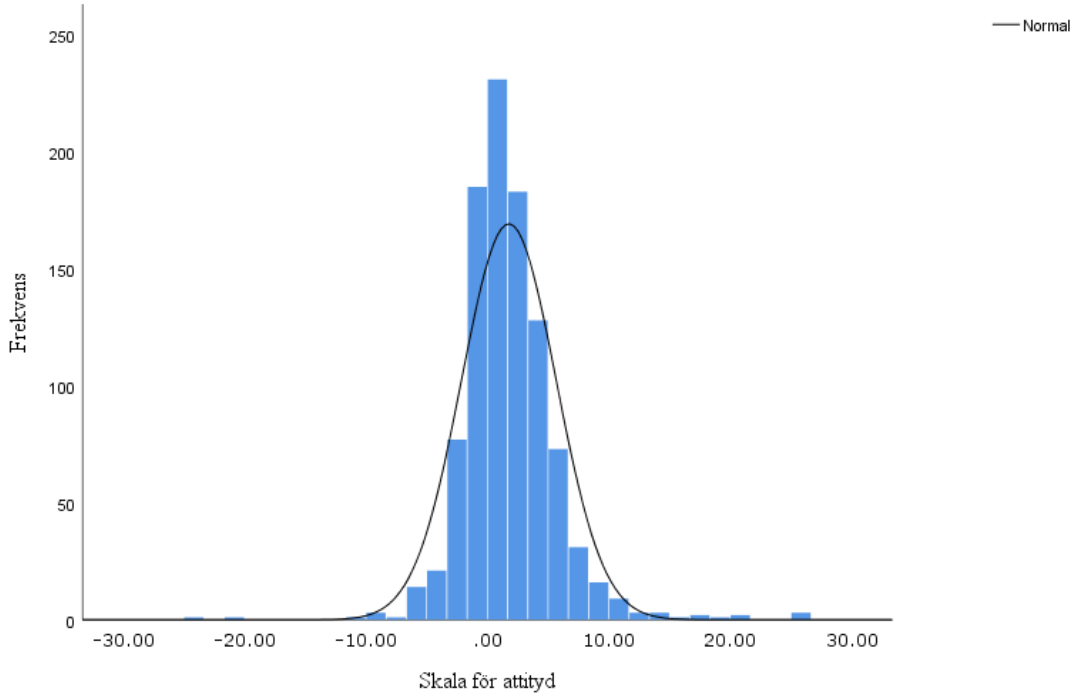
3.1 Föranalys

Antalet analyserade repositories var 990, varav 492 arkiverade och 498 icke-arkiverade repositories (se Appendix 2 för samtliga repositories). Totalt fanns det 45 programmeringsspråk med i datan. De största språken var JavaScript (28.1 %, *n* = 278), Java (9 %, *n* = 89) och Python (8.4 %, *n* = 83). I skalan för attityd (*avg. compound)* erhölls ett medelvärde på 1.76 (*n* = 990) och standardavvikelse på 3.9 (Se tabell 2 för skillnader i medelvärden mellan arkiverade och icke-arkiverade repositories). Medelvärdet visade att de flesta commit-meddelanden hade en neutral attityd.

Tabell 2: Medelvärde (M), standardavvikelse (SD) för icke-arkiverade (n = 498) och arkiverade (n = 490) repositories. Negativitet, positivitet och neutralitet är på en skala 0 till 100.

Mätskala	Ej arkiverad		Arkiverad	
	M	SD	M	SD
1. Attityd (*avg. compound*)	1.74	4.13	1.79	3.64
2. Negativitet	2.94	1.79	3.03	2.96
3. Positivitet	4.46	3.74	4.48	2.54
4. Neutralitet	92.51	4.59	92.48	3.88

Skalan för attityd bedömdes vara normalfördelad efter inspektion av dess histogram (se figur 4). Normalfördelningen bekräftades delvis av resultatet för skalans snedfördelning (skewness) som



Figur 4: Histogram över fördelningen av värden på attitydskalan. Kurvan illustrerar en typisk normalfördelning.

låg på .86, medan värdet för toppighet (kurtosis) låg på 8.7. Sett till toppighet var värdet en bra bit över det rekommenderade värdet på 3.

En femprocentig borttagning av repositories med högsta och lägsta värden visade sig inte påverka medelvärdet avsevärt (5 % trimmed mean = 1.62).

3.2 Huvudanalys

En logistisk regressionsanalys utfördes för att testa ifall uppvisad attityd genom text kan användas för att förutsäga projektmisslyckande. Modellen innehöll en oberoende variabel, eller här en så kallad prediktorvariabel (attityd), och den beroende variabeln (arkivering). Modellen visades inte vara signifikant ($\chi 2$ (1, n = 990) = .05, p = .82), vilket indikerar att modellen inte kunde användas till att skilja arkiverade från icke-arkiverade repositories. Samtidigt visade ett Hosmer & Lemeshow test att modellen passade någorlunda väl, $\chi 2$ (8, n = 990) = 8.57, p > .38, där ett signifikansvärde över .05 är önskvärt.

Eftersom modellen endast bestod av en prediktorvariabel, skalan för attityd, var variabelns "individuella" sannolikhet för predicering av projektmisslyckande samma som för hela modellen. Variabelns B-värde var .004, p = .82, dvs. värdet som här kan användas till att beräkna sannolikheten för att ett repository är arkiverat eller inte.

Inga outliers observerades i den logistiska regressionsanalysen.

4 Diskussion

Hypotesen i föreliggande studie baserades på att det i tidigare forskning fanns belägg för att undersöka förhållandet mellan utvecklares attityder och projektstatus. Exempelvis visade viss forskning på att attityder kan mätas genom att analysera attityder i de textmeddelanden som utvecklare skriver när de gör bidrag, eller så kallad commits, till Open Source-projekt (Guzman, Azócar, & Li, 2014; Sinha, Lazar & Sharif, 2016). På motsvarande sätt argumenterades det för att arkivering av projekt kunde användas till att beskriva projektstatus, eller mer specifikt projektmisslyckande, då en konsekvens av misslyckande i Open Source är bl.a. att projektet läggs på is på obestämd tid och en brist på aktivt samarbete (Ehls, 2017).

Närmare bestämt hypotetiserades det att mätningen av attityder i text, specifikt i commit-meddelanden, kunde användas till att predicera projektmisslyckande. I enlighet med hypotesen utformades en modell som blev föremål för testning.

4.1 Diskussion av resultat

Resultatet visar på att det inte finns ett samband mellan attityder och projektstatus. Med andra ord kan den utformade modellen ej användas till att förutsäga huruvida ett projekt kommer misslyckas eller ej. Därmed bör den alternativa hypotesen, H_A, förkastas och nollhypotesen, H_0, ej förkastas.

Varför den alternativa hypotesen förkastas kan ligga i de många outforskade kulturerna och samarbeten som uppstår i en virtuell community (Kollock & Smith, 1996). Något som utmärker människor är deras förmåga att uttrycka känslor (Darwin, 2009) och hantera konflikter, men däremot anpassas och förändras språkbruket vid datormedierad kommunikation (Elliot & Scacchi, 2003). Sättet som människor kommunicerar med hjälp av text och visar attityder online förändras beroende på tidpunkt, kultur och kontext (Kollock & Smith, 1996), och om kommunikation i t.ex. Open Source-utveckling jämförs med mikrobloggar samt online-spel, så har dessa communities helt unika sätt att samarbeta och kommunicera virtuellt.

Misslyckande i Open Source-projekt var svårt att definiera, eftersom vad som uppfattas vara ett misslyckande kan skilja sig från ett projekt till ett annat. De problem och samarbetssvårigheter som går att finna i vanliga misslyckanden inom mjukvaruprojekt, såsom brist på ledarskap, specifikationskrav och tid (Kappelman, McKeeman & Zhang, 2006) behöver inte heller nödvändigtvis stämma överens med Open Source-projekt - just eftersom Open Source-projekt inte har samma krav eller förutsättningar som traditionella projekt (Llanos & Castillo, 2012). De problem som ledde till att Open Source-projekt övergavs i Coelho och Valentes (2017) studie gav ett givande perspektiv till varför Open Source-projekt kan misslyckas, men området hade behövt nyanseras ytterligare för att tydliggöra eventuella riskfaktorer för misslyckande i Open Source.

Konflikter och attityder brukar dock vara en generell riskfaktor som präglar flera typer av mjukvaruprojekt och samarbeten (Jurado & Rodriguez, 2015; Liu et al. 2011; Tarricone & Luca, 2002). Därmed kan sambandet mellan konflikter och Open Source beaktas och undersökas

närmare i framtida studier för att klargöra dess egenskaper i Open Source-sammanhang. Till exempel kan det vara så att mätning och analys av attityder i textform inte är en särskilt bra indikator för utvecklares faktiska attityder, känslotillstånd eller inställning till ett projekt eller att de helt enkelt inte uttrycks i commit-meddelanden på det sätt som förväntas. Alternativt skulle det kunna vara så att själva textunderlaget helt enkelt är fel för den här typen av analys. Detta diskuteras mer djupgående i metoddiskussionen.

Attitydanalyser har tidigare visat goda resultat för att undersöka och mäta attityder som finns online (Hancock, Landrigan & Silver, 2007), men i vissa situationer föredrar människor att delge information på ett mer neutralt och tekniskt vis (Dabbish et al. 2012). Likt resultaten från Sinha, Lazar & Sharif (2016) studie visar det sig att de flesta commit-meddelandena är *neutrala*. Trots att commit-meddelanden är en form av textbaserad kommunikation används de främst för att beskriva ändringar som skett i källkoden (såsom "new" och "error") (Alali et al. 2014), vilket kan förklara den höga frekvensen av neutrala meningar.

Poängen med commit-meddelanden är att de objektivt och neutralt ska beskriva ändringar som gjorts i källkoden (Dabbish et al. 2012), men detta bör inte hindra utvecklare från att skriva vad de känner för. Eftersom Open Source medför en typ av demokratisk utvecklingsform där kodkvalité är i fokus (Tsay, Dabbish, & Herbsleb, 2014) hade det inte varit orimligt om commits med tydlig negativitet från utvecklare aldrig integreras i projektet. Med andra ord kan Open Source-projektledare efter diskussion med andra utvecklare välja bort commits som anses vara negativt laddade. Istället kan det vara av intresse att analysera text som inte kan exkluderas på ett sådant sätt, t.ex. inlägg på diskussionsforum.

Detta är dock enbart spekulationer, men modellen hade kunnat anpassats till att istället analysera ifall frekventa t.ex. merge och pull requests samt issues kan vara tecken till misslyckade projekt. I samband med denna reflektion gjordes ett försök att testa tillägget av antal issues för ett projekt i modellen. Jämfört med tidigare modell verkade den nya vara bättre på att förutsäga projektmisslyckande. Detta skulle kunna fungera som utgångspunkt för framtida forskning i kombination med attitydanalys av issues, merge och pull requests.

4.2 Metoddiskussion

Som tidigare nämnts är tecken till misslyckade Open Source-projekt inaktiva communities för utvecklare (Ehls, 2017), men att operationalisera projektstatus visade sig vara en utmaning för den här studien. Att likställa arkiverade repositories med övergivna och misslyckade projekt var ett antagande som ansågs vara rimligt utifrån de egenskaper som kännetecknar Open Source-projekt (Dahlander & Magnusson, 2005; Dabbish et al. 2012; GitHub Help, 2020). Däremot saknades tidigare forskning som explicit gjorde en sådan jämförelse eller härledning. Det finns därför en risk att metoden inte mätte projektstatus på rätt vis och att modellen således har låg konstruktvaliditet. Anledningen till att ett repository arkiveras kan istället bero på att utvecklingen har blivit pausad eller färdigställd. Poängen är att ett arkiverat repository inte nödvändigtvis betyder att det har misslyckats.

Framtida studier borde utforska alternativa sätt att operationalisera projektstatus för att se ifall det blir en skillnad i resultat. Ett förslag är att undersöka repositories som avsevärt har förlorat aktiva utvecklare och som saknar frekventa statusuppdateringar från dess community. Detta kan t.ex. göras med en modifierad variant av den webbapplikation som utformades för den här studiens ändamål. Med utvinning av data från Open Source-plattformar som GitHub finns det

möjlighet att analysera andra förutsättningar för att bättre förstå samarbete, attityder och tecken till misslyckande i Open Source.

4.2.1 Instrument

Valet av instrument, dvs VADER, kan ha påverkat resultatet på grund av domänspecifika skäl. Eftersom VADER är utvecklat för att undersöka korta textmeddelanden på sociala medier (Hutto & Gilbert, 2014) så är verktyget egentligen inte anpassat efter text som förekommer inom utvecklingsmiljöer. Det finns en risk att instrumentet inte poängsätter eller värderar tekniska termer och kontexter för utveckling med lika hög noggrannhet (Islam & Zibran, 2018). Om en liknande studie skulle genomföras i framtiden rekommenderas därför användningen av ett instrument som är mer anpassat efter en domän för mjukvaruutveckling - ifall ett sådant instrument finns tillgängligt och är tillämpbart.

Instrumentets validitet är också viktigt att diskutera eftersom det påverkar studiens resultat. Tidigare nämndes det att det finns en risk att textunderlaget är fel för den här typen av analys. Om inte instrumentet passar texten, eller tvärtom, riskerar värderingen av ord och meningar bli missvisande. Förekomsten av vissa ord eller meningar kanske poängsätts fel i sammanhanget, vilket skulle betyda att resultatet tillskrivs fel attityd. Det skulle kunna finnas begrepp som instrumentet värderar som neutrala men som utvecklare snarare skulle säga är negativa. Återigen blir detta en fråga om domänspecificitet, dvs. om instrumentet är anpassat för området som analyseras.

4.2.2 Urvalet

För att kunna göra en mätning som inkluderar relevanta repositories hämtades data med hjälp av sökparametrar. Detta gjordes för att undvika repositories som saknar en community och som inte kan klassificeras som Open Source-projekt.

I studien gjordes dessutom ett bekvämlighetsurval. Ett bekvämlighetsurval innebär, som tidigare nämnts, att urvalet inte bör ses som representativt för hela populationen, dvs. repositories som stämmer in på de sökparametrar som användes. Resultatet är inte heller generaliserbart för alla repositories utanför parametrarnas omfång eller för repositories på andra plattformar eftersom det kan finnas skillnader i deras förutsättningar. Det skulle kunna vara så att repositories på en annan plattform, exempelvis GitLab, har fler eller färre utvecklare med negativa attityder. Alternativt kan GitLab ha funktioner som inte GitHub har, där dessa funktioner rentav minskar risken för att konflikter ska uppstå mellan utvecklare. Sammantaget går det därför inte att säga att resultatet är representativt för alla repositories, vilket betyder att studiens externa validitet påverkas.

5 Slutsats

Syftet i föreliggande studie var att testa en modell som utgick från följande hypotes: Negativa attityder i commit-meddelanden predicerar misslyckande i Open Source-projekt. Modellen togs fram och testades med hjälp av data från GitHub, och trots att studiens alternativa hypotes kunde förkastas, vilket innebär att modellen inte kunde användas till att predicera projektmisslyckande, rekommenderas vidare forskning för att se över och eventuellt omarbeta modellen. Att modellen inte kunde användas, så som det hypotetiserades, beror sannolikt på att typen av text, dvs. commit-meddelanden, var huvudsakligen neutral eller att det instrument som användes för attitydanalys inte är anpassat för det här området.

Ett förslag till vidare forskning skulle vara att dels använda andra instrument för attitydanalys och dels göra analysen på annan text som genereras i samband med Open Source-projekt och digitala förvaringsplatser, t.ex. merge och pull requests samt issues. Dessutom hade en annan typ av data, så som antal issues eller status på issues, kunnat integreras i modellen för att göra den mer träffsäker.

Sammantaget hade modellen praktiskt kunnat användas för att få bättre insyn eller förståelse för negativa attityder och samarbetsproblem som kan uppstå och vara svåra att upptäcka i Open Source-projekt.

Appendix 1

Kodexempel från Gitextraction. Exemplet gäller extraheringen av projekt och attitydanalys av commit-meddelanden. I koden är importeringen av externa bibliotek exkluderad.

```
@app.route("/api/repositories/commits")
    def repositoriesCommits():
        if not github.authorized:
            return redirect(url_for("github.login"))
        def getRepositories(page="1", perPage="100", query=None):
            while True:
                searchRemaining = getRateLimit()
                if searchRemaining == 30:
                    break
                else:
                    sleep(30)
            resp =
github.get("/search/repositories?page={page}&per_page={perPage}&q={query}".format(page=pa
ge, perPage=perPage, query=query))
            assert resp.ok
            repositories = resp.json()["items"]
            totalCount = 0
            analyzer = SentimentIntensityAnalyzer()
            # start on 1 because intitial fetching of repos is considered as one search
request
            nr = 1
            for repository in repositories:
                query = "/search/commits?per_page=100&q=repo:{repoName}+author-
date:<2020-04-25".format(repoName=repository["full_name"])
                response = github.get(query, headers={"Accept" :
"application/vnd.github.cloak-preview"})
                assert response.ok
                commits = response.json()["items"]
                avgNegRepo = 0
                avgPosRepo = 0
                avgNeuRepo = 0
                avgCompoundRepo = 0
                nrOfSentences = 0
                for commit in commits:
                    # Get the message
                    message = commit["commit"]["message"]
                    # replace newline with punctuation
                    message = message.replace("\n", " ").replace("\r", "")
                    # Split the message into sentences
                    sentences = message.split(". ")
                    # Initiate avg vars
                    avgNeg = 0
                    avgPos = 0
                    avgNeu = 0
                    avgCompound = 0
                    for sentence in sentences:
                        # Analyse each sentence
                        vs = analyzer.polarity_scores(sentence)
                        # Sum of all the sentence scores in the message
```

```
                        avgCompound += vs['compound']
                        avgNeg += vs['neg']
                        avgPos += vs['pos']
                        avgNeu += vs['neu']
                        nrOfSentences += 1
                        # print("{:-<65} {}".format(sentence, str(vs)))
                    # Divide by number of sentences in message to get average score
                    avgCompound = avgCompound / len(sentences)
                    avgNeg = avgNeg / len(sentences)
                    avgPos = avgPos / len(sentences)
                    avgNeu = avgNeu / len(sentences)

                    # Sum the average of messages
                    avgCompoundRepo += avgCompound
                    avgNegRepo += avgNeg
                    avgPosRepo += avgPos
                    avgNeuRepo += avgNeu

                # Divide by number of commits to get average for whole repo
                avgCompoundRepo = avgCompoundRepo / len(commits)
                avgNegRepo = avgNegRepo / len(commits)
                avgPosRepo = avgPosRepo / len(commits)
                avgNeuRepo = avgNeuRepo / len(commits)
                print("Average compound: {compound} \nAverage negative:
{negative}\nAverage positive: {positive}\nAverage neutral:
{neutral}".format(compound=avgCompoundRepo, negative=avgNegRepo, positive=avgPosRepo,
neutral=avgNeuRepo))

                # Add values back to json
                repository['average_compound'] = avgCompoundRepo
                repository['average_negative'] = avgNegRepo
                repository['average_positive'] = avgPosRepo
                repository['average_neutral'] = avgNeuRepo
                repository['nr_of_commits'] = len(commits)
                repository['nr_of_sentences'] = nrOfSentences

                # repeat logic
                nr += 1
                totalCount += 1
                if nr == 30:
                    while True:
                        searchRemaining = getRateLimit()
                        if searchRemaining == 30:
                            nr = 0
                            break
                        else:
                            sleep(30)
                print("{index}\t{repoName}".format(index=str(totalCount),
repoName=repository["full_name"]))
            return repositories

        # Archived repositories
        #
        # >= 250 stars
        # page 1
        repositories = getRepositories(query="archived:true+is:public+stars:>=250")
```

```
        # page 2
        repositories += getRepositories(page="2",
query="archived:true+is:public+stars:>=250")
        # page 3
        repositories += getRepositories(page="3",
query="archived:true+is:public+stars:>=250")
        # page 4
        repositories += getRepositories(page="4",
query="archived:true+is:public+stars:>=250")
        # page 5
        repositories += getRepositories(page="5",
query="archived:true+is:public+stars:>=250")

        # Not archived
        #
        # >= 250 stars
        # page 1
        repositories += getRepositories(query="archived:false+is:public+stars:>=250")
        # page 2
        repositories += getRepositories(page="2",
query="archived:false+is:public+stars:>=250")
        # page 3
        repositories += getRepositories(page="3",
query="archived:false+is:public+stars:>=250")
        # page 4
        repositories += getRepositories(page="4",
query="archived:false+is:public+stars:>=250")
        # page 5
        repositories += getRepositories(page="5",
query="archived:false+is:public+stars:>=250")

        # Normalize results, i.e flattening of json
        df = json_normalize(repositories)

        # Convert to csv format
        resp = make_response(df.to_csv())
        resp.headers["Content-Disposition"] = "attachment;
filename=repositories_final.csv"
        resp.headers["Content-Type"] = "text/csv"
        resp.headers["Content-Encoding"] = "utf-8"

        return resp
```

Appendix 2

Tabell över utvunna repositories.

index	id	Namn på repository
0	211666	node-v0.x-archive
1	206084	legacy-homebrew
2	18275356	pop
3	3122202	RxJS
4	32665718	upterm
5	3755875	skrollr
6	53658802	weex
7	321278	npm
8	21481439	hosts
9	26554	reddit
10	6094683	bootstrap
11	138634298	MS-DOS
12	88876034	WeChatPlugin-MacOS
13	138806129	download
14	51994692	electronic-wechat
15	93076012	chromeless
16	32866454	nuklear
17	43998576	git-recipes
18	30003816	powerlevel9k
19	1136075	iscroll
20	198770415	github-do-not-ban-us
21	73929422	oni
22	8257106	JSQMessagesViewController
23	4702560	PHPExcel
24	5101823	dashing
25	11133743	Effeckt.css
26	10536180	AngularJS-Learning
27	12601374	react-devtools
28	15345331	bolt
29	38003903	react-redux-starter-kit
30	2155793	ViewPagerIndicator

31	105229748	awesome-algorithm
32	90786246	TypeScript-React-Starter
33	9603714	Alcatraz
34	10020773	gekko
35	26509369	rkt
36	2874001	Android-PullToRefresh
37	5297437	pdf2htmlEX
38	268027	jquery-cookie
39	113402414	xray
40	99127696	tfjs-core
41	23082332	SlackTextViewController
42	38066334	caffe2
43	11446311	VVDocumenter-Xcode
44	53991171	BottomBar
45	46891997	browser-laptop
46	728994	fine-uploader
47	59939691	marp
48	48301282	apex
49	9467421	vis
50	31524768	nuclide
51	45632603	react-router-redux
52	37798806	aria2gui
53	156648725	eeeeeeeeeeeeeeeeeeeeeeeeeee eeeeeeeeeeeeeeeeeeeeeeeeeee eeeeeeeeeeeeeeeeeeeeeeeeeee eeeeeeeeeeeeeeeeeee
54	18427187	shadowsocks-qt5
55	112380358	android-ktx
56	1451060	ActionBarSherlock
57	75566993	universe
58	17149519	KVOController
59	49760504	TrumpScript
60	43443499	WeChatLuckyMoney
61	78849439	Awesome_APIs
62	13426154	odometer
63	1762028	BlocksKit

64	42332863	filebrowser
65	6486840	shadowsocks-go
66	1922703	wysihtml5
67	50301368	maybe
68	53877810	fbctf
69	261335	progit
70	6804324	generator-angular
71	15126429	Mvc
72	702550	zendframework
73	8877512	ListViewAnimations
74	9787757	godep
75	179786310	cssfx
76	51420370	react-router-tutorial
77	5417266	ShowcaseView
78	51740069	fuck-login
79	22498444	postcss-cssnext
80	40706	blueprint-css
81	63731800	Keyframes
82	34408310	code-prettify
83	40262072	Empire
84	7694208	twoway-view
85	599070	SAMKeychain
86	65851975	open_nsfw
87	4812336	otto
88	99805855	deep-learning-coursera
89	19777812	EhViewer
90	58605580	why-did-you-update
91	10697582	node-jscs
92	44739044	go-git
93	33822149	govendor
94	637935	phpunit-mock-objects
95	7209439	oauth2_proxy
96	33456123	PermissionScope
97	43789644	vue-strap
98	488514	bundler
99	6566487	PhantomCSS

100	31001422	sqlbrite
101	21782593	swift-style-guide
102	500370	html5demos
103	13779494	android-proguard-snippets
104	5363235	component
105	64762103	in-view
106	5275559	neat
107	4553246	grace
108	2942615	NineOldAndroids
109	36187165	vibrant.js
110	30484966	rtv
111	23393132	emojione
112	3269452	ECSlidingViewController
113	1445827	libextobjc
114	10188673	android-volley
115	14816993	vim.js
116	125564566	wxappUnpacker
117	3287301	llvm
118	10091574	hopscotch
119	2103283	FormatterKit
120	17996971	augmented-traffic-control
121	16297741	extract-text-webpack-plugin
122	44498352	AndroidWiFiADB
123	79033973	chunkwm
124	97779740	why-did-you-update
125	90786593	TypeScript-Vue-Starter
126	10420996	WP-API
127	47903333	store
128	9477941	finalterm
129	5199579	vim
130	1171324	mobile-boilerplate
131	82605180	panelkit
132	25003905	webcomponentsjs
133	20849184	TLYShyNavBar
134	76130347	create-react-app-typescript
135	186760137	electron-ssr-backup

136	39473685	go-torch
137	87123146	glamorous
138	914985	Silex
139	42825774	wooyun_public
140	3006455	shumway
141	41855272	WebDriverAgent
142	7293210	jquery.payment
143	75118280	Store
144	5969840	turbolinks-classic
145	22629932	gometalinter
146	65648767	babel-preset-env
147	19346768	ace
148	2493754	phantomjs-node
149	2346623	hubot-scripts
150	32153390	gofpdf
151	18609451	duo
152	28836678	Gadgetbridge
153	7296156	bootstrap-datetimepicker
154	78880128	instabot
155	76672223	luna
156	1459821	pow
157	2581357	tess-two
158	48850847	adapta-gtk-theme
159	45750356	mention-bot
160	15397085	android-butterknife-zelezny
161	16907502	color
162	47807407	react-web
163	44620036	typings
164	5589723	pthreads
165	849558	sstoolkit
166	20452776	ExSwift
167	42753362	VisualStudioUninstaller
168	5542032	viz.js
169	9023386	PhysicsJS
170	1543431	libuv
171	12962729	ui-select

172	128623312	hands-on-ml-zh
173	21339768	react-tutorial
174	61623700	SSM
175	66208923	ramme
176	10480910	ActionBar-PullToRefresh
177	15467734	zone.js
178	21212616	gvr-android-sdk
179	45365298	JavaScriptServices
180	2848123	MKNetworkKit
181	2160835	swig
182	5289665	launchrocket
183	15324383	bootstrap-treeview
184	34597195	friends
185	96975695	EPPlus
186	2125788	osx-gcc-installer
187	65428555	SpechtLite
188	53731544	hain
189	61702909	deepmask
190	8222042	mosca
191	5188794	cheddar-ios
192	1380117	lantern
193	32432671	network-connection-class
194	10466461	brick
195	8523763	MCSwipeTableViewCell
196	5872657	robospice
197	16226063	conceal
198	48759915	frontend-boilerplate
199	18727860	DraggablePanel
200	13887383	webscalesql-5.6
201	2031640	sensu
202	22284914	structs
203	21567832	MITMf
204	28523724	react-chartjs
205	2982576	Bootstrap-Image-Gallery
206	1282929	processing-js
207	38141050	WiFi-Pumpkin-deprecated

208	4368712	jersey
209	10992987	YouTubeCenter
210	9873882	CoVim
211	45007173	anthelion
212	2803190	vim-powerline
213	38300942	tellform
214	5499296	algorithms
215	13175334	mdwiki
216	17208050	Telegram
217	9461481	docker-registry
218	30160120	hubpress.io
219	5345965	infinity
220	2536618	thinkphp
221	4459589	syte
222	38110583	android-percent-support-lib-sample
223	66245850	mini.css
224	14964923	FoundationPress
225	78433294	oklog
226	1734612	chocolatey
227	31057802	jsblocks
228	65319101	pyflame
229	110989998	propel
230	89502824	create-next-app
231	9475692	fartscroll.js
232	3287353	clang
233	26663083	charted
234	74927215	MSEC
235	58266220	NEKit
236	25370072	Sia
237	861781	arachni
238	3483944	homebrew-php
239	92652320	npx
240	6049573	our-boxen
241	15242752	ngReact
242	4571340	elasticsearch-rtf

243	27791908	gvr-unity-sdk
244	28800891	browserhtml
245	20344662	KestrelHttpServer
246	23806326	djinni
247	24257743	kali-nethunter
248	18203752	ConfuserEx
249	21246279	android-Camera2Basic
250	5438867	angular-google-maps
251	83614695	android-ConstraintLayoutExamples
252	23272510	heapster
253	36069295	agar.io-clone
254	56941978	brew
255	29429361	bettercap
256	4154667	massive-js
257	16630339	vue-touch
258	23063531	weixin-java-tools
259	58052865	react-native-fetch-blob
260	14029328	videojs-contrib-hls
261	481811	murder
262	11535888	CMS
263	123383102	canner
264	56771677	Free-Security-eBooks
265	3027949	appframework
266	4864799	batarang
267	10081171	gopm
268	23401311	awesome-angularjs
269	31786762	device-year-class
270	65587921	NMessenger
271	4640263	Nestable
272	13512328	fleet
273	35370819	PagingMenuController
274	1840417	luna
275	13421516	FreeFlow
276	18806010	pourover
277	8769058	goread

278	11558916	disunity
279	50328557	AppleDNS
280	71159589	SignalR
281	61919271	react-monocle
282	932626	pfff
283	40686014	contrib
284	14457098	bootstrap-validator
285	248959	jquery-tokeninput
286	80753020	react-trend
287	1955800	paperwork
288	3313788	sublimetext-markdown-preview
289	102909215	pwa-starter-kit
290	11271755	cocoapods-xcode-plugin
291	870225	kod
292	31394531	react-isomorphic-starterkit
293	7491531	html-inspector
294	23831045	deliver
295	22774923	migrate
296	31077672	awesome-typescript-loader
297	82193141	luminoth
298	9214768	legacy-linuxbrew
299	103934044	rasa_core
300	21445180	hamburger-button
301	38793780	telepot
302	39787202	ScpToolkit
303	1606665	jQuery-Tags-Input
304	9736937	laravel-breadcrumbs
305	86204319	hacker-news-pwas
306	58152695	ansible-container
307	50056934	distributedlog
308	16396294	angular-formly
309	25452970	kotterknife
310	22472365	react-starter
311	10626680	card.io-iOS-SDK
312	2280086	envoy

313	40522844	etherwallet
314	3790489	angular-ui-OLDREPO
315	38568813	hack.chat
316	15117001	vue-validator
317	4057468	recess
318	6801434	ObjectiveSugar
319	7268697	drywall
320	9207280	crosswalk
321	55811475	deploykit
322	71946001	stickybits
323	44044446	it-ebooks-archive
324	3626870	HoloEverywhere
325	31404980	gb
326	38478437	tsf
327	853845	HockeyKit
328	28521302	regexper-static
329	63352907	the-road-to-learn-react
330	50817818	ChatKit-OC
331	597840	gizzard
332	43410867	ReactJS-AdminLTE
333	19968235	Sketch-Toolbox
334	33685883	Messenger-for-Desktop
335	79487641	goldfish
336	26302821	android-ripple-background
337	82003080	wireguard-monolithic-historical
338	679377	popcorn-js
339	12229237	dind
340	93097839	guide
341	122112163	nebulet
342	9372336	raft
343	1262196	MKStoreKit
344	1279067	add-to-homescreen
345	3044325	dot-vimrc
346	2242185	event-stream
347	49312071	way-cooler

348	4696667	Bootstrap-Form-Builder
349	20636942	react-autocomplete
350	114287404	ci
351	10603681	skyline
352	71402554	shave
353	49533308	wmail
354	2624543	android-ocr
355	70867540	grid
356	90922472	aesthetic
357	2161666	GrowingTextView
358	44591594	universal-starter
359	130092	snipmate.vim
360	65798229	easy-log-handler
361	57380736	zmirror
362	15714062	IdentityServer3
363	9360778	vincent
364	6376011	backgrid
365	451490	jquery.maskedinput
366	4334944	Sami
367	91639446	tensorflow-build-archived
368	12655652	json11
369	26403221	snapshot
370	36896680	Unbox
371	1409544	capybara-webkit
372	98561896	MineCraft-One-Week-Challenge
373	21517184	Simple-Web-Server
374	77749667	ex-baiduyunpan
375	17444456	bcloud
376	37272972	react-native-dribbble-app
377	63073314	phphub5
378	79538924	dubbo-spring-boot-starter
379	912604	ua-parser
380	16480576	laravel-translatable
381	10712489	ReactiveViewModel
382	10298623	XPrivacy

383	110947712	stacks-cli
384	1760455	320andup
385	28524244	TrueCraft
386	138501426	rogue.js
387	104287441	now-examples
388	12601393	react-art
389	54614463	zazu
390	56186554	Moon
391	40907450	gl-react-native-v2
392	49527035	fb.resnet.torch
393	59494640	commit
394	1840036	skynet-archive
395	84739133	Marathon
396	53587175	cooking
397	667163	github-services
398	82990603	sites-using-cloudflare
399	61632427	vim-go-tutorial
400	74011714	tui-go
401	35588993	DKChainableAnimationKit
402	1424005	commons
403	111558	gh-unit
404	91576794	whats-new-in-swift-4
405	1337406	retire
406	34876145	node-convergence-archive
407	16491880	Identity
408	43278409	LayaAir_Discard
409	2125665	deployinator
410	14202056	hunter
411	291056	teambox
412	46881155	node-chakracore
413	7530454	octokit.objc
414	7400568	regexper
415	325827	msysgit
416	46555171	Rosie
417	69279243	HubFramework
418	27856783	tinderbox

419	10626798	card.io-Android-SDK
420	66378700	react-faq
421	149249432	go-under-the-hood
422	1578548	persona
423	12801352	TextJustify-Android
424	3878981	d3-plugins
425	2871177	requestbin
426	59078087	firebase-jobdispatcher-android
427	40664963	laravel-graphql
428	7070571	emojify.js
429	39902722	minigrid
430	80666384	Dotzu
431	7757171	QBImagePicker
432	1394574	Custom-Metaboxes-and-Fields-for-WordPress
433	25601444	LollipopShowcase
434	115532492	retrofit2-kotlin-coroutines-adapter
435	42280005	TinyDancer
436	2764438	NSDate-TimeAgo
437	1610564	fabric8
438	122942337	btt-touchbar-presets
439	4128408	Durandal
440	25973357	react-d3
441	37307032	music
442	49966855	html
443	46213845	hexo-theme-apollo
444	76630616	Potatso
445	50753708	vue-sui-demo
446	11827104	vim-mode
447	922017	eco
448	58910553	react-starter-kit
449	28202609	peloton
450	35617117	zend-code
451	80234065	composer

452	26255685	wechat-api
453	91250793	dingdang-robot
454	22615096	osxcollector
455	2442469	JSONStream
456	44090231	vue-antd
457	61155730	react-universally
458	100997690	ayo
459	21876647	http2
460	107142574	kotlin-guides
461	24718628	goddd
462	1777643	BareMetal-OS-legacy
463	98404512	permissions4m
464	5968664	objective-c-style-guide
465	24654062	CNPPopupController
466	21592194	PunchClock
467	52304529	cpustat
468	161859	ioctocat
469	20775197	noc-book
470	7864067	groundwork
471	35280797	zend-eventmanager
472	65903301	tiger
473	63655543	baffle
474	4798	DotNetOpenAuth
475	25601850	talon-twitter-holo
476	71740230	chain
477	38273206	nsp
478	2072864	Golden-Grid-System
479	595090	zeromq.node
480	44144887	vrview
481	3181606	exhibitor
482	23505879	nd4j
483	29552587	SwiftMoment
484	253952	QuincyKit
485	7083261	libphenom
486	8941531	chromeview
487	11463670	lean

488	8921879	htmlbars
489	14991049	Scout2
490	67552785	reframe.js
491	20778549	foundation-apps
492	1134253	Heimdall
493	61771898	nnvm
494	23534015	xenko
495	29161493	LolliPin
496	53558755	babylon
497	68908781	wechat-app-demo
498	6015874	boxen
499	9866547	fries
500	28457823	freeCodeCamp
501	177736533	996.ICU
502	11730342	vue
503	10270250	react
504	13491895	free-programming-books
505	45717250	tensorflow
506	2126244	bootstrap
507	21737465	awesome
508	14440270	You-Dont-Know-JS
509	60493101	coding-interview-university
510	291137	ohmyzsh
511	85077558	developer-roadmap
512	1062897	gitignore
513	121395510	CS-Notes
514	41881900	vscode
515	6498492	javascript
516	83222441	system-design-primer
517	31792824	flutter
518	943149	d3
519	2325298	linux
520	29028775	react-native
521	123458551	Python-100-Days
522	9384267	electron
523	21289110	awesome-python

524	54346799	public-apis
525	63537249	create-react-app
526	132464395	JavaGuide
527	35955666	the-art-of-command-line
528	23088740	axios
529	23096959	go
530	63476337	Python
531	27193779	node
532	132750724	build-your-own-x
533	126577260	javascript-algorithms
534	2561582	animate.css
535	14098069	free-programming-books-zh_CN
536	20580498	kubernetes
537	1039520	youtube-dl
538	51117837	models
539	3470471	Font-Awesome
540	90796663	puppeteer
541	100060912	terminal
542	24195339	angular
543	576201	three.js
544	20929025	TypeScript
545	460078	angular.js
546	34526884	ant-design
547	19415064	computer-science
548	1863329	laravel
549	22790488	java-design-patterns
550	7691631	moby
551	23083156	material-ui
552	112507086	30-seconds-of-code
553	3678731	webpack
554	21540759	awesome-go
555	44571718	awesome-vue
556	167174	jquery
557	88464704	vue-element-admin
558	33614304	thefuck

559	36535156	redux
560	160640094	LeetCodeAnimation
561	3228505	atom
562	44838949	swift
563	1861458	reveal.js
564	596892	flask
565	557980	socket.io
566	7600409	shadowsocks-windows
567	4164482	django
568	507775	elasticsearch
569	8843683	Chart.js
570	237159	express
571	33015583	keras
572	9309093	Semantic-UI
573	133442384	deno
574	54173593	storybook
575	70107786	next.js
576	6296790	spring-boot
577	18408635	Apollo-11
578	14747598	json-server
579	3544424	httpie
580	26500787	FiraCode
581	10744183	netdata
582	4311796	markdown-here
583	8514	rails
584	128398636	architect-awesome
585	67274736	element
586	486550	html5-boilerplate
587	3955647	lodash
588	21872392	awesome-machine-learning
589	724712	rust
590	3402537	Front-end-Developer-Interview-Questions
591	1334369	resume.github.com
592	5108051	opencv
593	36633370	awesome-selfhosted

594	1424470	moment
595	36040894	gatsby
596	81975372	interviews
597	11180687	hugo
598	103633984	nodebestpractices
599	1181927	bitcoin
600	3638964	ansible
601	151834062	advanced-java
602	7508411	RxJava
603	156018	redis
604	1362490	requests
605	62607227	tech-interview-handbook
606	3214406	meteor
607	101296881	every-programmer-should-know
608	612230	nvm
609	23357588	protobuf
610	120538304	nocode
611	12256376	ionic
612	9185792	incubator-echarts
613	843222	scikit-learn
614	65252	jekyll
615	19872456	react-router
616	24953448	material-design-icons
617	15204860	papers-we-love
618	52631841	realworld
619	50264296	bulma
620	107111421	Front-End-Checklist
621	26898879	awesome-public-datasets
622	1700621	normalize.css
623	49970642	yarn
624	65600975	pytorch
625	46629305	hacker-scripts
626	28428729	awesome-android-ui
627	138839979	DeepLearning-500-questions
628	20904437	gin

629	23974149	materialize
630	63539055	awesome-mac
631	45986162	TensorFlow-Examples
632	20300177	guava
633	529502	scrapy
634	69629434	freecodecamp.cn
635	5152285	okhttp
636	3100121	nw.js
637	55076063	Awesome-Hacking
638	22670857	awesome-react
639	1148753	spring-framework
640	24560307	babel
641	16408992	neovim
642	75104123	prettier
643	51148780	architecture-samples
644	40416236	big-list-of-naughty-strings
645	3065454	impress.js
646	34302698	serverless
647	21737266	awesome-nodejs
648	99598595	parcel
649	892275	retrofit
650	74791366	clean-code-javascript
651	48378947	frp
652	241576270	fucking-algorithm
653	53809090	anime
654	33791743	x64dbg
655	32484381	free-for-dev
656	21700699	awesome-ios
657	98029592	learn-regex
658	15111821	grafana
659	5271882	build-web-application-with-golang
660	15428480	AdminLTE
661	16752620	gogs
662	22887094	tesseract
663	47018239	awesome-interview-questions

664	83844720	face_recognition
665	127988011	mall
666	22119721	git-flight-rules
667	9852918	Ghost
668	22458259	Alamofire
669	26689598	awesome-courses
670	62367558	hyper
671	74293321	svelte
672	77189043	vue2-elm
673	12888993	core
674	1828795	AFNetworking
675	4086616	shadowsocks
676	5483330	you-get
677	36502	git
678	10187082	pm2
679	138393139	the-book-of-secret-knowledge
680	4710920	dubbo
681	88011908	project-based-learning
682	11167738	gulp
683	147350463	33-js-concepts
684	29268051	material-design-lite
685	3432266	kotlin
686	83999700	Best-websites-a-programmer-should-visit
687	2935735	brackets
688	68957920	awesome-wechat-weapp
689	14712850	syncthing
690	11225014	etcd
691	41912791	v2ray-core
692	127407957	leetcode
693	15062869	jest
694	81598961	cpython
695	1129010	jQuery-File-Upload
696	11177928	Projects
697	7569578	discourse

698	19148949	MPAndroidChart
699	6838921	prometheus
700	1663468	pdf.js
701	5923215	hexo
702	12791642	caffe
703	114747226	faceswap
704	172953845	code-server
705	18708860	github-cheat-sheet
706	29290473	XX-Net
707	14370955	hackathon-starter
708	12972263	fullPage.js
709	15653276	android-open-project
710	15634981	godot
711	12820537	slate
712	26066727	mermaid
713	21413198	immutable-js
714	7190986	shadowsocks-android
715	58028038	HelloGitHub
716	7741856	pixi.js
717	58836534	algorithm-visualizer
718	65794292	styled-components
719	11551538	koa
720	111583593	scrcpy
721	11267509	glide
722	13807606	fzf
723	70198875	lottie-android
724	2573058	foundation-sites
725	67186968	chinese-poetry
726	27442967	fastlane
727	39464018	incubator-superset
728	42408804	traefik
729	75830968	deeplearningbook-chinese
730	42751014	clipboard.js
731	139824423	100-Days-Of-ML-Code
732	91573538	12306
733	8681349	huginn

734	667006	video.js
735	82227585	design-patterns-for-humans
736	931135	Leaflet
737	6093316	DefinitelyTyped
738	952189	backbone
739	60844036	ShadowsocksX-NG
740	1200050	phantomjs
741	17375436	es6features
742	29207621	caddy
743	90528830	awesome-flutter
744	9393759	phaser
745	1903522	php-src
746	10865436	frontend-dev-bookmarks
747	32948863	awesome-react-native
748	56717493	HEAD
749	41058054	CppCoreGuidelines
750	128907699	dayjs
751	64558143	AndroidUtilCode
752	15019962	tldr
753	71995937	nuxt.js
754	26516210	certbot
755	10788737	android_guides
756	1844251	todomvc
757	698041	async
758	35866694	Rocket.Chat
759	1644196	julia
760	80945428	nest
761	10446890	ijkplayer
762	5239185	quill
763	191113739	interview_internal_reference
764	42283287	preact
765	70905478	Deep-Learning-Papers-Reading-Roadmap
766	45512989	gold-miner
767	63266213	og-aws
768	21600440	awesome-awesomeness

769	14194174	what-happens-when
770	17165658	spark
771	27729880	grpc
772	15452919	go-ethereum
773	115478820	awesome-scalability
774	155220641	transformers
775	108252892	proxyee-down
776	26783295	kong
777	18049133	slick
778	349241	underscore
779	8575137	butterknife
780	2562751	zxing
781	21935031	awesome-cpp
782	30969188	react-boilerplate
783	48626335	vue-cli
784	61929586	programmer-job-blacklist
785	128624453	taro
786	67962648	vuetify
787	14098121	500lines
788	101394335	ant-design-pro
789	873328	sentry
790	34824499	leakcanary
791	858127	pandas
792	25315643	nylas-mail
793	1283503	request
794	4578002	python-patterns
795	83119431	AiLearning
796	21648001	awesome-java
797	114523184	JCSprout
798	28167802	weui
799	3620194	select2
800	137147386	pure-bash-bible
801	317757	Modernizr
802	3774328	wrk
803	3482588	SecLists
804	22362099	angular-styleguide

805	71948498	localstack
806	35969061	styleguide
807	156157055	lessons
808	117372806	front-end-interview-handbook
809	122692377	hangzhou_house_knowledge
810	481366	gitflow
811	138547797	mkcert
812	43441403	strapi
813	2579314	Faker
814	3577919	beego
815	70431106	machine-learning-for-software-engineers
816	29891188	standard
817	94498635	carbon
818	76838017	iina
819	39176269	vuex
820	6452529	rethinkdb
821	25136308	fetch
822	14705691	awesome-php
823	3721224	swiper
824	56969116	trackerslist
825	54544023	new-pac
826	41986369	tidb
827	667561	faker.js
828	42956467	awesome-macos-command-line
829	1064563	netty
830	36891867	angular-cli
831	64355429	iview
832	2329766	dotfiles
833	105919803	Detectron
834	112567264	BaiduPCS-Go
835	458058	symfony
836	29887499	open-source-ios-apps
837	65388917	PythonDataScienceHandbook

838	38558578	nativefier
839	11620669	CodeHub
840	63477660	Java
841	1864363	composer
842	13840241	devdocs
843	50447720	swift-algorithm-club
844	6007295	jieba
845	154747577	bert
846	15585444	Hover
847	32578467	Charts
848	313419	SDWebImage
849	74175805	istio
850	13078968	postcss
851	46273445	uppy
852	5070389	EventBus
853	47394776	lerna
854	61893399	awesome-react-components
855	109343098	open-source-mac-os-apps
856	18280236	gitbook
857	158703981	libpku
858	2500088	gitlabhq
859	1631024	chosen
860	2096579	marked
861	24841635	date-fns
862	106017343	tailwindcss
863	51980455	alacritty
864	20619036	pi-hole
865	8859474	jadx
866	72907253	spring-boot-examples
867	2541284	cheerio
868	94367677	formik
869	13584262	webtorrent
870	32215970	mobx
871	32247847	rxjs
872	17728164	terraform
873	790359	sequelize

874	23783375	ResumeSample
875	23736449	particles.js
876	42920477	layui
877	29261473	minio
878	1801829	ember.js
879	5532320	polymer
880	1028340	ace
881	37958358	tesseract.js
882	238316428	COVID-19
883	53238813	brew
884	4484451	vimrc
885	3757512	sails
886	95189138	p3c
887	3602123	hammer.js
888	2700474	fastjson
889	995750	Vundle.vim
890	8989842	the-way-to-go_ZH_CN
891	95876775	project-guidelines
892	17803236	rclone
893	28751632	linux-insides
894	119811010	tabler
895	32689863	manim
896	4037197	YouCompleteMe
897	2700159	druid
898	7607075	Probabilistic-Programming-and-Bayesian-Methods-for-Hackers
899	18044526	streisand
900	63484632	fastText
901	60325062	awesome-deep-learning-papers
902	27574418	nerd-fonts
903	16607898	drone
904	62812261	kubernetes-the-hard-way
905	7548986	framework
906	12244426	nprogress

907	52281283	mastodon
908	9603240	pure
909	308770	devise
910	6988020	sheetjs
911	1481305	python-guide
912	164608222	flutter-go
913	11276147	cmder
914	146633589	arthas
915	24635156	sweetalert
916	597879	mongoose
917	19141383	blog
918	140687430	uni-app
919	9754983	tutorials
920	25880891	flow
921	31085130	lottie-web
922	93152223	SmartRefreshLayout
923	162279822	Motrix
924	173228436	ghidra
925	23405758	leveldb
926	53127403	apollo
927	958314	nodemon
928	30203935	metabase
929	57222302	gym
930	101684374	wtfpython
931	59652928	ngx-admin
932	2056312	html2canvas
933	2293158	metasploit-framework
934	47071941	redux-saga
935	49935814	ItChat
936	22067521	imgui
937	33263118	uBlock
938	23069399	bootstrap-material-design
939	73681508	wepy
940	21393871	awesome-javascript
941	79510167	pipenv
942	1254497	CodeMirror

943	18840003	react-starter-kit
944	2810455	the_silver_searcher
945	15308499	Sortable
946	40997482	vim
947	1580851	PhotoSwipe
948	7212645	spacemacs
949	23338716	react-select
950	103953059	modern-js-cheatsheet
951	19620844	octotree
952	55886798	BaseRecyclerViewAdapterHelper
953	60630844	monaco-editor
954	3606624	ReactiveCocoa
955	481872	vagrant
956	133251103	dive
957	33895378	dragula
958	18347476	iosched
959	1116542	aria2
960	31629751	cypress
961	168008797	calculator
962	16677706	awesome-sysadmin
963	49016322	components
964	124171501	mpvue
965	34757182	interactive-coding-challenges
966	145553672	funNLP
967	53370988	lighthouse
968	170326929	Awesome-Design-Tools
969	6731432	shellcheck
970	51863547	handson-ml
971	20965586	SwiftyJSON
972	34453060	awesome-electron
973	401025	hub
974	6766558	Hystrix
975	72495579	gitea
976	70198664	lottie-ios
977	15045751	compose

978	53631945	ripgrep
979	38934449	react-redux
980	20429943	the-swift-programming-language-in-chinese
981	92807616	English-level-up-tips-for-Chinese
982	184456251	PowerToys
983	130464961	bat
984	1614410	FFmpeg
985	1451352	mocha
986	3431083	GPUImage
987	21552971	awesome-shell
988	8686855	intro.js
989	1420053	guzzle
990	734934	pug
991	50151075	react-redux-links
992	24976755	HanLP
993	12670444	bluebird
994	113752225	profile-summary-for-github
995	7240954	fonts
996	5625464	pyenv
997	21696302	awesome-swift
998	301742	tornado
999	15183485	The-Art-Of-Programming-By-July

Referenser

Acuña, S. T., Gómez, M., & Juristo, N. (2009). How do personality, team processes and task characteristics relate to job satisfaction and software quality?, *Information and Software Technology,* vol. 51, no. 3, pp. 627-639

Ahonen, J. J., & Savolainen, P. (2010). Software engineering projects may fail before they are started: Post-mortem analysis of five cancelled projects, *Journal of Systems and Software,* vol. 83, no. 11, pp. 2175-2187

Alvesson, M. (2014). Kommunikation, makt och organisation: närläsning och multipla tolkningar, Studentlitteratur, pp. 79-102

Boudreau, M. C., Gefen, D., & Straub, D. W. (2001). Validation in information systems research: A state-of-the-art assessment, *MIS quarterly,* pp. 1-16

Bonaccorsi, A., & Rossi, C. (2006). Comparing motivations of individual programmers and firms to take part in the open source movement: From community to business, *Knowledge, Technology & Policy,* vol. 18, no. 4, pp. 40-64

Biehl, M. (2016). RESTful API Design (Vol. 3), *API-University Press,* pp. 19

Carlson, K. D., & Herdman, A. O. (2012). Understanding the impact of convergent validity on research results, *Organizational Research Methods,* vol. 15, no. 1, pp. 17-32

Coelho, J., & Valente, M. T. (2017). Why modern open source projects fail, *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pp. 186-196

Dabbish, L., Stuart, C., Tsay, J., & Herbsleb, J. (2012). Social coding in GitHub: transparency and collaboration in an open software repository, *Proceedings of the ACM 2012 conference on computer supported cooperative work*, pp. 1277-1286

Dahlander, L., & Magnusson, M. G. (2005). Relationships between open source software companies and communities: Observations from Nordic firms, *Research policy*, vol. 34, no. 4, pp. 481-493

Darwin, Charles. (2009). The expression of the emotions in man and animals – Introduction, afterword and commentaries by Paul Ekman, Anniversary Edition, London: HarperCollinsPublishers

D'Ambros, M., Gall, H., Lanza, M., & Pinzger, M. (2008). Analysing software repositories to understand software evolution, *Software evolution*, Springer, Berlin, Heidelberg, pp. 37-67

Ehls, D. (2017). Open source project collapse–sources and patterns of failure

El Emam, K., & Koru, A. G. (2008). A replicated survey of IT software project failures, *IEEE software*, vol. 25, no. 5, pp. 84-90

Elliott, M. S., & Scacchi, W. (2003). Free software developers as an occupational community: resolving conflicts and fostering collaboration, *Proceedings of the 2003 international ACM SIGGROUP conference on Supporting group work*, pp. 21-30

Feldt, R., & Magazinius, A. (2010). Validity threats in empirical software engineering research-an initial survey, *Seke,* pp. 374-379

Ferguson, L. (2004). External validity, generalizability, and knowledge utilization, *Journal of Nursing Scholarship,* vol. 36, no. 1, pp. 16-22.

Gilovich, T., Keltner, D., Chen, S., & Nisbett, R. E. (2016). Social Psychology (Vol. 4).

GitHub. (2019). The state of Octoverse, Available online: https://octoverse.GitHub.com/ [Accessed 14 April 2020]

GitHub API v3. (2020). REST API v3, Available online: https://developer.github.com/v3/ [Accessed 11 May 2020]
GitHub Help. (2020). Archiving a GitHub repository, Available online: https://help.github.com/en/github/creating-cloning-and-archiving-repositories/archiving-a-github-repository [Accessed 4 May 2020]
Goodman, S. N., Fanelli, D., & Ioannidis, J. P. (2016). What does research reproducibility mean?, *Science translational medicine*, vol. 8, no. 341, pp. 341
Guzman, E., Azócar, D., & Li, Y. (2014). Sentiment analysis of commit comments in GitHub: an empirical study. *Proceedings of the 11th Working Conference on Mining Software Repositories*, pp. 352-355
Guzzi, A., Bacchelli, A., Lanza, M., Pinzger, M., & Van Deursen, A. (2013). Communication in open source software development mailing lists. *2013 10th Working Conference on Mining Software Repositories (MSR),* IEEE, pp. 277-286
Hancock, J. T., Landrigan, C., & Silver, C. (2007). Expressing emotion in text-based communication. *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 929-932
Hardt, D. (2012). The OAuth 2.0 authorization framework, *RFC 6749*
Hauge, Ø., Ayala, C., & Conradi, R. (2010). Adoption of open source software in software-intensive organizations–A systematic literature review, *Information and Software Technology*, vol. 52, no. 11, pp. 1133-1154
Hertel, G., Niedner, S., & Herrmann, S. (2003). Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel, *Research policy*, vol. 32, no. 7, pp. 1159-1177
Hutto, C. J., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth international AAAI conference on weblogs and social media*
Islam, M. R., & Zibran, M. F. (2018). SentiStrength-SE: Exploiting domain specificity for improved sentiment analysis in software engineering text. *Journal of Systems and Software,* vol. 145*,* pp. 125-146
Jantz, R., & Giarlo, M. J. (2005). Digital preservation: Architecture and technology for trusted digital repositories, *Microform & imaging review*, vol. 34, no. 3, pp. 135-147
Jongeling, R., Sarkar, P., Datta, S., & Serebrenik, A. (2017). On negative results when using sentiment analysis tools for software engineering research, *Empirical Software Engineering*, vol. 22, no. 5, pp. 2543-2584
Jurado, F., & Rodriguez, P. (2015). Sentiment Analysis in monitoring software development processes: An exploratory case study on GitHub's project issues, *Journal of Systems and Software*, vol. 104, pp. 82-89
Kappelman, L. A., McKeeman, R., & Zhang, L. (2006). Early warning signs of IT project failure: The dominant dozen, *Information systems management*, vol. 23, no. 4, pp. 31-36
Kalliamvakou, E., Gousios, G., Blincoe, K., Singer, L., German, D. M., & Damian, D. (2014). The promises and perils of mining GitHub, *Proceedings of the 11th working conference on mining software repositories*, pp. 92-101
Kollock, P., & Smith, M. (1996). Managing the virtual commons, *Computer-mediated communication: Linguistic, social, and cross-cultural perspectives*, pp. 109-128
Kogut, B., & Metiu, A. (2001). Open-source software development and distributed innovation, *Oxford review of economic policy*, vol. 17, no. 2, pp. 248-264
Lakhani, K. R., & Von Hippel, E. (2004). How open source software works:“free” user-to-user assistance, *Produktentwicklung mit virtuellen Communities*, Gabler Verlag pp. 303-339
Lerner, J., & Tirole, J. (2002). Some simple economics of open source, *The journal of industrial economics*, vol. 50, no. 2, pp. 197-234

Levin, K. A. (2006). Study design III: Cross-sectional studies, *Evidence-based dentistry*, vol. 7, no. 1, pp. 24-25

Linberg, K. R. (1999). Software developer perceptions about software project failure: a case study, *Journal of Systems and Software*, vol. 49, no. 2-3, pp. 177-192

Liu, J. Y. C., Chen, H. G., Chen, C. C., & Sheu, T. S. (2011). Relationships among interpersonal conflict, requirements uncertainty, and software project performance, *International Journal of Project Management*, vol. 29, no. 5, pp. 547-556

Llanos, J. W. C., & Castillo, S. T. A. (2012). Differences between traditional and open source development activities, International Conference on Product Focused Software Process Improvement, Springer, Berlin, Heidelberg, pp. 131-144

Nyman, L., & Lindman, J. (2013). Code forking, governance, and sustainability in open source software, *Technology Innovation Management Review*, vol. 3 no. 1

Recker, J. (2013). Scientific research in information systems: a beginner's guide. Springer Science & Business Media

Rubinstein, D. (2007). Standish group report: There's less development chaos today, *Software Development Times*, vol. 1

Santos, E. A., & Hindle, A. (2016). Judging a commit by its cover, *Proceedings of the 13th International Workshop on Mining Software Repositories-MSR*, vol. 16, pp. 504-507

Schimmack, U., & Colcombe, S. (2007). Eliciting mixed feelings with the paired-picture paradigm: A tribute to Kellogg (1915), *Cognition and Emotion*, vol. 21, no. 7, pp. 1546-1553

Sinha, V., Lazar, A., & Sharif, B. (2016). Analyzing developer sentiment in commit logs. *In Proceedings of the 13th International Conference on Mining Software Repositories*, pp. 520-523

Staff, C. A. C. M. (2016). React: Facebook's functional turn on writing Javascript, *Communications of the ACM*, vol. 59, no. 12, 56-62

Stam, W. (2009). When does community participation enhance the performance of open source software companies?, *Research Policy*, vol. 38, no. 8, pp. 1288-1299

Stol, K. J., & Fitzgerald, B. (2014). Inner source--adopting open source development practices in organizations: a tutorial, *IEEE Software*, vol. 32, no. 4, pp. 60-67

Straub, D. W. (1989). Validating instruments in MIS research, *MIS quarterly*, pp. 147-169

Subramaniam, C., Sen, R., & Nelson, M. L. (2009). Determinants of open source software project success: A longitudinal study, *Decision Support Systems*, vol. 46, no. 2, pp. 576-585.

Tarricone, P., & Luca, J. (2002). Successful teamwork: A case study

Tesser, A., & Shaffer, D. (1990) Attitudes and attitude change, *Annual Review of Psychology*, vol. 41, pp. 479-523

The Standish Group International (1994). CHAOS: Project Failure and Success Research Report, *Standish Group International, Inc*

The Standish Group International (2015). CHAOS Report 2015, *Standish Group International, Inc*

Tsay, J., Dabbish, L., & Herbsleb, J. (2014). Influence of social and technical factors for evaluating contribution in GitHub, *Proceedings of the 36th international conference on Software engineering,* pp. 356-366

Pallant, J. (2013). SPSS survival manual. McGraw-Hill Education (UK)

Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis, *Foundations and Trends® in Information Retrieval*, vol. 2, no. 1–2, pp. 1-135

Pinto, J. K., & Mantel, S. J. (1990). The causes of project failure, *IEEE transactions on engineering management*, vol. 37, no. 4, pp. 269-276

Pressman, R. (1998). Fear of trying: the plight of rookie project managers, *IEEE software*, vol. 15, no. 1, pp. 50-51

Verner, J., Sampson, J., & Cerpa, N. (2008). What factors lead to software project failure?, *2008 Second International Conference on Research Challenges in Information Science,* IEEE, pp. 71-80

Von Hippel, E. (2001). Learning from open-source software, *MIT Sloan management review*, vol. 42, no. 4, pp. 82-86

Wesselius, J. (2008). The bazaar inside the cathedral: Business models for internal markets, *IEEE software*, vol. 25, no. 3, pp. 60-66

West, J. (2003). How open is open enough?: Melding proprietary and open source platform strategies. *Research policy*, vol. 32, no. 7, pp. 1259-1285

West, J., & Gallagher, S. (2006). Patterns of open innovation in open source software, *Open Innovation: researching a new paradigm*, vol. 235, no. 11

Wrobel, M. R. (2013). Emotions in the software development process. *2013 6th International Conference on Human System Interactions (HSI),* IEEE, pp. 518-523