Employee Sentiment for Factor Investing in the Corporate Bond Market

Daniel Stancl

Academic Supervisor:	Prof. Philip Treleaven
Industry Supervisors:	Daniel Beresford
	Lucette Yvernault

A dissertation submitted in partial fulfillment of the requirements for the MSc degree in Computational Statistics and Machine Learning at University College London

Department of Computer Science
University College London

September 2020

Declaration

This report is submitted as a part requirement for the degree MSc Computational Statistics and Machine Learning at UCL.

I, Daniel Stancl, hereby proclaim that I wrote this work on my own under the leadership of my supervisors and that the references include all resources and literature I have used.

I grant a permission to reproduce and to distribute copies of this report document and all corresponding source codes in whole or in part.

London, August 24, 2020

Daniel Stancl

Daniel Stancl

Abstract

This thesis investigates the relationship between employee sentiment, proxied by Glassdoor reviews and ratings, and excessive returns on corresponding bonds. While sentiment analysis is well studied for probing into how companies are perceived by investors or the general public, it is a novel idea to exploit sentiment of employees, which enables us to capture very important information for assessing companies' governance. Although a few studies scrutinising the relation employee sentiment and future stock returns have already appeared, this is the first attempt, to the best of my knowledge, to place this analysis to the universe of corporate bonds.

This research was conducted in collaboration with Fidelity International and entails a series of experiments that examine how employees' feelings might be utilised as an indicator of returns on corporate bonds. This project was prepared as a separate piece of work, in cooperation with an ESG-oriented research group under the supervision of Prof. Philip Treleaven.

The thesis consists of the following four components - a data retrieval pipeline, two experiments and an exploratory analysis:

1. Data Retrieval and Database Pipeline. This work engineers a pipeline for scraping employee reviews and related information from Glassdoor using a Python's Selenium-based crawler and subsequently storing them in the database built using Django making the data easily available for possible future endeavours of other students and researchers.

2. Comparison of Different Sentiment Scoring Methods in Generating Alpha. This experiment investigates the utilisation of NLP sentiment analysis of employee reviews into a multi-factor credit scoring model, as a part of ESG investing framework, and their power in generating alpha by own constructed factor long-short portfolio.

3. Comparison of the Best Sentiment Scoring Method against the Usage of Reviews in Generating Alpha. This study further examines whether NLP sentiment analysis provides any additional piece of information compared with a simpler proxy for exmployee sentiment, star ratings, for generating excessive returns of factor portfolios.

4. Exploratory Analysis of Ratings and Reviews. Since analyses of reviews and ratings

from Glassdoor and similar platforms are quite scarce, mainly because of unavailability of any public API, a thorough exploratory analysis of the scraped data is conducted as a part of this thesis and is presented at Appendix.

The following sentiment scoring methods are considered:

1. Score pros and cons separately.

2. Score reviews consisting of concatenated pros and cons.

3. Combine both approaches stated above with weighting them according to the text length.

The key findings of this thesis are:

1. A multi-factor credit scoring model utilising sentiment scoring method can generate higher alpha compared with a baseline without a sentiment factor by XY %.

2. An approach considering NLP employee sentiment is able to provide us with better performance than the one with a simplified estimation of sentiment proxied by reviews. The difference is XY %.

Daniel Stancl, Employee Sentiment Analysis for ESG Investing
Supervisors: Prof. Philip Treleaven and Daniel Beresford

Acknowledgements

Firstly, I would like to express the sincere gratitude to my academic supervisor, Prof. Philip Treleaven, for his support throughout writing of this final project and an invaluable amounts of his advice. At the same time, I would like to thank to Jan Pratnicki, a PhD student at UCL, for the time he dedicated to me.....

Contents

List of Figures **viii**

List of Tables **ix**

List of Code Snippets **x**

List of Acronyms **xi**

1 Introduction **1**

1.1 Motivation 1
1.2 Research Objectives 2
1.3 Research experiments 2
1.4 Scientific Contributions 3
1.5 Thesis Structure 3

2 Background and Literature Review **5**

2.1 Factor Investing in the Corporate Bond Market 5
2.1.1 Definition of Selected Factors 6
2.1.2 Multi-Factor Models 8
2.2 ESG Investing and Its Adoption 9
2.3 Sentiment Analysis and Machine Learning Advances 10
2.3.1 Sentiment Analysis and Optimisation Basics 10
2.3.2 Advances in NLP and Sentiment Analysis- Neural Networks 13
2.4 BERT for Sentiment Analysis 16
2.4.1 Architecture 16
2.4.2 Pre-Training and Fine-Tuning 18
2.5 Sentiment Analysis in ESG Investing 19

3 Data - Selection, Sources and Exploration **22**

3.1 Company Selection Criteria 22
3.2 Employee Sentiment Data 24
3.2.1 Data for fine-tuning 29

3.3 Market Data . . . 29
3.4 Data Pre-Processing . . . 31
3.4.1 Employee Sentiment Data . . . 31
3.4.2 Market data . . . 31

4 Methodology **33**
4.1 Tools Developed for Data Retrieval . . . 33
4.1.1 Glassdoor Scraper . . . 33
4.1.2 Yahoo Scraper . . . 38
4.2 Database and Pipeline . . . 40
4.3 Sentiment Analysis . . . 41
4.3.1 BERT Implementation . . . 43
4.3.2 Approaches to Employee Sentiment Scoring . . . 44
4.4 Proposed Factor Model . . . 45
4.5 Test Methodology . . . 46

5 Employee Sentiment in Generating Return **48**
5.1 Ratings . . . 48
5.2 Comparison of Different NLP Sentiment Scoring Methods . . . 51

6 Discussion **52**
6.1 Interpretation of Results . . . 52
6.2 Limitations . . . 52
6.3 Application . . . 52
6.4 Future Work . . . 52

7 Conclusion **53**

Bibliography **54**

Appendix **60**

List of Figures

2.1 Meaning of ESG 9
2.2 GloVe: Nuances between similar words 13
2.3 2D Convolutional layer Source: (Goodfellow et al., 2016) 15
2.4 RNNs have internal state 15
2.5 Multi-head attention 17

3.1 An example of employee review posted on Glassdoor 24
3.2 Distribution of review 26
3.3 Correlation of monthly changes in sentiment across stock markets 29
3.4 Correlation of monthly changes in sentiment across market sectors 30

4.1 Glassdoor - Login portal 35
4.2 Glassdoor - Company search 36
4.3 Glassdoor - 'Continue reading' button 37
4.4 Yahoo Finance 39
4.5 Django - Database structure 42

5.1 Correlation of returns - Long-only portfolios 50

List of Tables

3.1 Distribution of companies across stock market sectors 23
3.2 Distribution of companies issuing bonds across stock market sectors 23
3.3 Number of reviews per company across stock markets 26
3.4 Number of reviews per company across market sectors 26
3.5 Descriptive statistics - Ratings across stock markets 27
3.6 Descriptive statistics - Ratings across market sectors 27
3.7 Summary statistics of employee ratings 27
3.8 Descriptive statistics - Ratings across stock markets 28
3.9 Descriptive statistics - Ratings across market sectors 28
3.10 Summary statistics of employee ratings 28
3.11 Caption 30

5.1 Returns of the long-short portfolios (ratings); T=21 (number of monthly returns) 49
5.2 Returns of the long portfolios (ratings); T=21 (number of monthly returns) . . 49
5.3 Return of the long multi-factor portfolios (ratings); T=21 51

List of Code Snippets

4.1 Import web driver module 34
4.2 Glassdoor - Login 35
4.3 Glassdoor - Continue reading 37
4.4 Glassdoor - Counting reviews on a single page 37
4.5 Glassdoor scraper - application 38
4.6 Generic form of Yahoo Finance URL 38
4.7 Yahoo and Wikipedia scraper - application 39
4.8 Django Model - Class instantiation 41
4.9 Django - Writing a record if not exist 42
4.10 BertTokeniser - Review transformation 43

List of Acronyms

ANN	Artificial neural network
BERT	Bi-diractional Encoder Representations from Transformers
BoW	Bag of words
CNN	Convolutional neural network
ESG	Environmental, social and corporate governance
ML	Machine Learning
NLP	Natural Language Processing
RNN	Recurrent neural network
SGD	Stochastic gradient descent

1. Introduction

The chapter introduces problems discussed in this thesis, outlines the motivation for this work, its research objectives, and empirical methods used to achieve them, concluding with an overview of the structure of the dissertation.

1.1 Motivation

Corporate bonds represent a convenient way of obtaining capital for stable companies all around the world. The US corporate bond market capitalisation is about 40 % of the global market, the largest of any country (Sifma, 2020). After the borrowing pile-up in 2019, the outstanding debt of U.S. non-financial companies rose to the unprecedented $13.5 trillion at the end of that year (Çelik et al., 2020). Although the EU and UK markets are considerably smaller, they still constitute an interesting investment opportunity. The research purely devoted to the drivers of corporate bond returns started appearing decades after the presence of equity literature. Still there persist disputes over definitions of factors used for portfolio construction.

Factor investing relies on the idea of the possibility to generate portfolio returns beyond the market premium by harvesting excessive risk premium, which is associated with selected characteristics. Furthermore, a side effect of this investment approach is an enhancement of portfolio diversification as these drivers do not have to be necessarily highly correlated with macroeconomic variables, thus exhibit low beta with a market.

This thesis extends the recent research of factor models for corporate bond portfolios, for example, by Houweling and Van Zundert (2017). That and other studies successfully adopted a range of well-documented factors in equity literature, but definitions of factors vary across individual papers. These factors, initially explored for stock returns are low risk, value, size, momentum and carry. These characteristics are, therefore, covered in this work, though they are not examined due to the limited availability of corporate bond data.

In recent years, the soar of responsibility for sustainability issues among investors has been observed. This resulted in integrating alternative non-financial data into their investment decision process. These unconventional risk assessment criteria are environmental, social and governance (ESG) factors. Even though the first studies emerged already in the 1970s, adoption of these gauges has been sluggish, mainly because of concerns over their potentially negative impact on portfolio returns.

However, the ubiquity of alternative data, including textual or satellite and many other, and the possibility of investigating them gradually accelerated consideration of such information during portfolio constructions. This was enabled by a rise in computational resources together with invention of new powerful machine learning (ML) algorithms. Among others, sentiment analysis, a subfield of natural language processing (NLP), whose aim is to analyse people's sentiment, emotions or opinions expressed in the text, has attracted considerable attention of academic and commercial researchers (Liu, 2012).

Glassdoor, one of the world's largest job sites, where employees share information about their current and former employers, contains amounts of data potentially exploitable as one of the predictors of performance of companies' securities. The relationship between employee satisfaction and future stock returns has already been investigated and started with a study of Green et al. (2019) used reviews from Glassdoor as well. This master's dissertation aims to place a similar technique into the corporate bond market.

1.2 Research Objectives

The main objective of this thesis is to investigate whether employees sentiment about companies has any statistically significant predictive power for the returns on corporate bonds. To achieve this, a series of experiments, described in Section 1.3 has been conducted to pick the bonds for the long-short portfolios.

Subsequently in the simplified manner, the portfolio returns are calculated and investigated whether the employee sentiment has a predictive to estimate future bond prices trends. The results are also compared with momentum-based portfolio returns.

1.3 Research experiments

The following research experiments are conducted in this thesis:

1. Investigation of Employee Sentiment based on Ratings in Generating Return. This experiment investigates the utilisation of sentiment analysis of employee ratings for factor credit scoring model and its power in generating return using own constructed factor long-short and long-only portfolios. On this occasion, the company's sentiment score is derived as arithmetic mean of its ratings.

2. Comparison of Different Sentiment Scoring Methods in Generating Return. This study further examines whether utilisation of NLP sentiment analysis of employee reviews

provides any additional piece of information compared with a simpler proxy, star ratings, for generating returns of factor portfolios. In order to calculate sentiment for each company, the following approaches are considered:

- Score reviews consisting of concatenated pros and cons and then compute arithmetic mean.
- Follow the approach stated above and replace the averaging method with weighted arithmetic mean based on the length of the reviews.

In all cases, pre-trained language model - Bidirectional Encoder Representations from Transformers (BERT) - is used for estimating sentiment score.

1.4 Scientific Contributions

This thesis contributes to the existing literature in the following ways:

1. Factor model Incorporating Employee Sentiment for the Corporate Bond Market. This thesis builds the model fed by monthly bond data and company's sentiment calculated from Glassdoor reviews and ratings. This model derives the factor that is automatically used for portfolio construction. Subsequently, the framework calculates estimated returns.
2. Data Retrieval and Database Pipeline. This work engineers a pipeline for scraping employee reviews and related information from Glassdoor using a Python's Selenium-based crawler and subsequently storing them in the database built using Django making the data easily available for possible future endeavours of other students and researchers.
3. Exploratory Analysis of Ratings and Reviews. Since analyses of reviews and ratings from Glassdoor and similar platforms are quite scarce, mainly because of unavailability of any public API, a thorough exploratory analysis of the scraped data is conducted as a part of this thesis and is presented at Appendix.

1.5 Thesis Structure

The rest of the thesis is organised as follows. Chapter 2 provides an overview of related research fields, starting with factor investing in the corporate bond market, followed by an ESG approach to the risk assessment and sentiment analysis and its utilisation in ESG investing. Chapter 3 outlines company selection criteria together with sources, characteristics and pre-processing of the data used for this study. Chapter 4 then introduces tools developed for data retrieval and storage. It continues with approaches used for employee sentiment scoring and corresponding proposed modifications to the factor model. Chapter 5 evaluates the ability of employees sentiment in generating alpha, which is subsequently discussed, followed by a future research proposal in Chapter 6. Finally, Chapter 7 reviews key findings and scientific contributions of this work and concludes this report by presenting the most promising direction for research.

2. Background and Literature Review

This chapter provides background information on key research concepts this thesis focuses on such as factor investing in the corporate bond market and an ESG approach to risk management. This is followed by a survey on sentiment analysis and the used language model - BERT. The final section reviews existing literature on sentiment analysis in ESG investing.

2.1 Factor Investing in the Corporate Bond Market

A formal notion about factor investing was initially made in Ross (1976), where the author followed on his previous co-authored study Ross et al. (1973). There, Ross expressed the future returns are correlated with various financial factors, i.e. there exists a linear relationship between them. In those times, this investment framework was proposed as an alternative to two paramount theoretical approaches - mean variance model (MPT) (Markowitz, 1952) and state preference theory (SPT) (Hirshleifer, 1966). Importantly, the relaxation of various rigid assumptions, including normal distribution or quadratic preference for MPT and SPT respectively, was suggested as appealing characteristics of factor investing (Ross, 1976).

Nowadays, corporate bond literature on factor investing predominantly springs from the equity's counterparts. Even some contemporary studies attempted to directly apply characteristics explaining stock excessive returns into the universe of companies' debt investing (Chordia et al., 2017), however, this attitude is rather doubted by the more recent research like Israel et al. (2017), where authors pointed out that pertinent risks across credit and equity markets differ. Israel et al. (2017) further warned that, even though corporate bond and equity prices are not independent, they do not react equally to asset value changes. Moreover, there are distinct factors for both equities and bonds, which affect their prices, and also corporate bond and equity markets are not identical. It is, therefore, appropriate credit investing to be considered from its own perspective.

Endeavours have been thus dedicated to finding variables having alpha specifically in the corporate bond market. These characteristics, although inspired by the world of equities, now have their own definitions for corporate bonds, but some disputes around have been lingering. While papers devoted to multi-factor credit portfolios are still scarce, there is, nowadays, evidence for **low risk** (also called as **quality**), **value**, **size**, **momentum** and **carry** (the latter one is not investigated in this thesis).

These characteristics are usually used to construct long-only or long-short portfolio in a way that a certain percentage of best performing bonds, according a given factor, are bought, and the worst doing are sold. Subsequently, alpha of the portfolio can be estimated using CAPM as

$$R_t = \alpha + \beta_t \cdot \text{Premium} + \varepsilon_t, \tag{2.1}$$

where R_t is a return on the portfolio, and Premium is designed to reflect the corporate bond market premium. According to Chen et al. (2014b), it is appropriate to distinguish between investment-grade and high-yield bonds, as these represent two distinct spheres of credit universe. For this reason, depending on the bond class, investment-grade and high-yield index, respectively, are used as Premium in model (2.1) to evaluate excess returns.

There are also more advanced regression models to describe portfolio returns, which achieve higher R-squared, namely:

- Fama-French three-factor model (Fama and French, 1993),
- Fama-French-Carhart four-factor model (Carhart, 1997),

nevertheless, for the sake of the scope of this thesis, these frameworks are not tested here.

2.1.1 Definition of Selected Factors

Low risk

Long-position investing in low-risk assets has been documented since the work of Haugen and Heins (1972), where the hypothesis that higher risk generated excessive returns was rejected. There, researchers' experiments, moreover, suggested that low-risk portfolio provided higher average reward in the long run. Many years later, Frazzini and Pedersen (2014) supported the proposition that investing in low-beta assets yields higher alpha, and provided evidence on a broad range of financial instruments including equity, futures, government and corporate bonds. In this study, ratings and maturity were used as a measure of risk for corporate and Treasury bonds respectively.

This thesis utilises a definition of Ilmanen (2011), and then also followed by Houweling and Van Zundert (2017), suggesting to long buy bonds with higher ratings and shorter maturity and short sell worse rated, long-dated ones.

Value

Value investing in equity markets has been also well documented for decades with the first occurrence already in the 1970s (Basu, 1977). In this experiment, Basu proposed cheap stocks tended to outperform their more expensive counterparts in terms of risk-adjusted return. In order to determine how much expensive a given share is, prices of a security are commonly compared with a company's fundamental value (e.g. earnings).

Correia et al. (2012) were among the first researchers who investigated the applicability of the concept of value in the corporate bond market. They tested different accounting-based and market-based (both equity and credit) information for explaining ex-post credit returns and found strong evidence that the difference in implied and actual credit spreads had a robust predictive power in explaining future bond returns. Houweling and Van Zundert (2017) further restricted themselves only on credit-related risk measures and derived "true" credit spread from a cross-sectional regression as

$$S_i = \alpha + \sum_{r=1}^{R} \beta_r I_{ir} + \gamma M_i + \delta \Delta S_i + \varepsilon_i, \tag{2.2}$$

where S_i denotes a credit spread of bond i, I_{ir} stands for a dummy variable denoting a rating class bond i belongs to, M_i is a maturity and ΔS_i captures the three-month change in the credit spread.

Accordingly, bonds exhibiting a higher difference between actual and implied credit spreads should be longed, while the ones with low disparity are intended to be shorted.

Size

Also, the relationship between the total market value of stocks and future returns has been examined for decades beginning with a seminal work of Banz (1981), where Banz on the sample of NYSE common provided evidence that smaller companies generated a higher risk-adjusted return in average.

For a long time, there was no study, which would show how alpha in the corporate bond market is associated with a size factor. The first attempted was made by Houweling and Van Zundert (2017), where the authors demonstrated there was a statistically significant positive relation between corporate bond excessive returns and total company's public debt. Importantly, the results hold both for multi-factor long-only and single factor long-short portfolios. Similarly, the hypothesis incorporating a size factor was experimentally proven by Henke et al. (2020), for single- and multi-factor long strategies.

Momentum

Jegadeesh and Titman (1993) documented that long buying stocks, which had exhibited well in the past, and short selling the ones with the poor performance was a strategy leading to excessive returns. This was best shown for a 6-month holding period with the same time frame of hindsight. However, the researchers also found a large portion of winners' and losers' momentum had been likely to vanish for long-term portfolios within two years since the formation.

Jostova et al. (2013) showed, on the large sample of U.S. corporate bonds between the years 1991 and 2011, strong momentum profitability in this market for the 3-, 6-, 9- and 12-month formation and holding periods. Moreover, their study proposed momentum was predominantly driven by high-yield bonds, which explained the shortage of momentum before 1991. The presence of momentum in the European corporate bond markets was analysed and confirmed by Barth et al. (2017), and the authors as well claimed it mainly arose in the non-investment bond universe. Validity of momentum credit portfolios was further endorsed by multi-factor corporate bond literature [(Houweling and Van Zundert, 2017), (Israel et al., 2017)]. The latter studies relied on six months of trailing credit returns, Israel et al. (2017) beside observed a half-year equity momentum.

2.1.2 Multi-Factor Models

Portfolio construction using a multi-factor model is pretty similar to the version of a single-factor model. Only instead of choosing securities according to one characteristic, different portfolios of an equal size are built, which then are combined to one asset pool (Houweling and Van Zundert, 2017). This approach should benefit from a low correlation between individual cross-sectional drivers and thus provides us with higher returns.

In the last 5 years, more attention has been paid to the investigation of the validity of multi-factor models in the corporate bond markets. Israel et al. (2017) demonstrated the ability of a combination of four characteristics - carry, defensive, value and momentum - to generate positive economically significant risk-adjusted profit. Furthermore, their inference showed to be valid both for long-and-short and long-only portfolio strategies.

Houweling and Van Zundert (2017) conducted separate comprehensive analyses for investment-grade and high-yield corporate bonds. They gave evidence that a combination of four factors - low risk, value, size ad momentum - were able to provide a reasonable balance between more prudent and risky characteristics, measured by Sharpe ratio, for both credit classes. Moreover, their multi-factor model always generated higher alpha, in CAPM and Fama-French-Carhart model frameworks, than low-risk and momentum portfolios. This excessive return was, furthermore, statistically significant at 1% level in all cases, while value and size portfolios, which performed higher alpha, reached this degree of significance only in 1 out of 2 cases for high-yield bonds and not single once for investment-grade class.

Five characteristics - carry, quality, value, size and momentum - were used and combined by

Henke et al. (2020). Their analysis, which was again conducted on U.S. investment-grade and high-yield bonds, manifested that the multi-factor model outperformed single-factor ones in terms of risk-adjusted returns, and the authors of this study these results attributed exactly to a low correlation between individual factors.

2.2 ESG Investing and Its Adoption

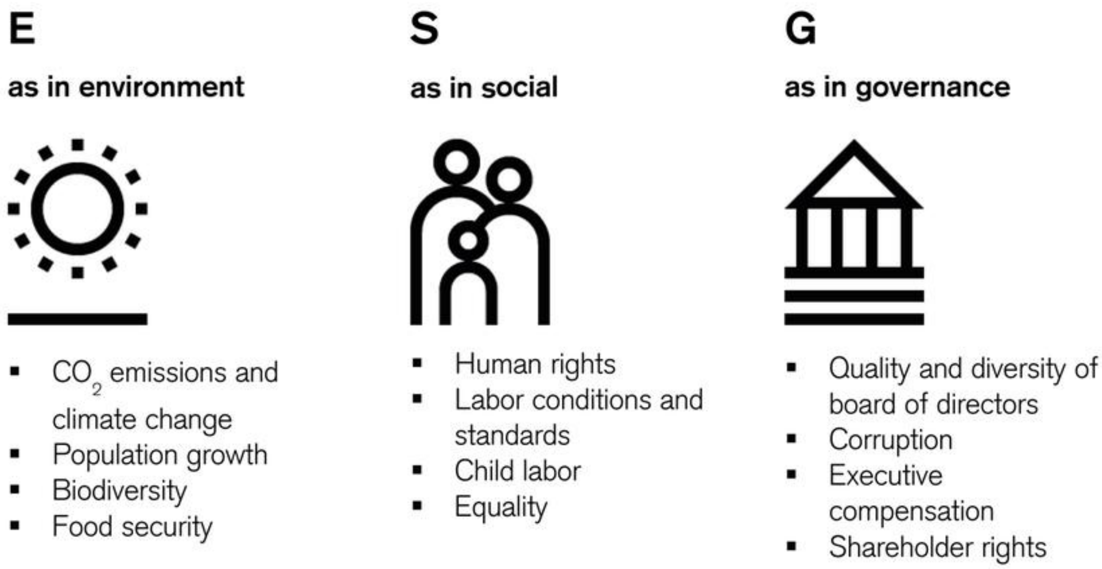


Figure 2.1: Meaning of ESG
Source: (ESG, 2018)

Throughout the time, two interwoven terms - ESG investing and socially responsible investing (SRI) - were referred by academics, nonetheless, an agreement on their definitions and appropriate names was missing (Eccles and Viviers, 2011). On of the first mention considering these two concepts fitting together was made by Sparkes (2001), where the author concerned with ethical and responsible investment. He postulated that SRI is an investment approach taking ESG and related criteria into account with an aim to maximise risk-adjusted returns. The phrases ESG and related criteria are used intentionally, as the researchers in the more recent literature were inclined to the stance that terms ESG and alternative data being interchangeable (In et al., 2019). The authors of the latter study referred to sources of such data to be, among others, satellites or smartphones, and reflected the broad width of ESG data.

However, concerns over the potential negative impact of socially responsible investment process on the funds' returns, altogether with low expressed ethical responsibility in the past century may have dragged an adoption of these non-financial assessment instruments. Notwithstanding early evidence non-financial characteristics did not necessarily sacrifice portfolio returns [(Mallin et al., 1995) for the U.K., (Guerard, 1997) for the U.S.], this scepticism persisted (Sparkes, 2001), therefore, a lot of empirical research has been devoted to this issue since then.

Friede et al. (2015) conducted an extensive analysis of more than 2000 empirical studies published since the 1970s, where the authors investigated the relation between ESG criteria and the financial performance of companies. Their work provided evidence for a positive correlation between these two measures with findings there were many opportunities on multiple equity and non-equity markets for ESG investment. However, they also pointed out the perception of investors to this asset management approach were still pessimistic with a notion of negative effects of accounting for non-financial criteria on portfolio returns. The reluctance of investors and fund managers to rely on alternative data was ascribed to the lack of understanding of incorporation this information into the investment decision process.

Adoption of ESG factors by asset managers in the U.S. and Europe was probed by Van Duuren et al. (2016), where the authors focused on individual ESG dimensions and evaluated to which a degree this information was utilised by investors. They identified a perception of an integrating of alternative data into portfolio construction differs between the U.S. and Europe, with European managers being more favourable to this approach. This dissimilarity was attributed to the perspective on SRI across continents as European peers deemed this to be more close to fundamental investing (FI). They further found predominantly governance dimension to be the centre of interest, which was deemed to be closely related to long-run corporate financial performance and thus this companies' assessment aligns with the concept of FI. The researchers, moreover, concluded the conscientious investors preferred company-level analyses to the industry-level ones, which was interestingly similar to the asset managers' attitude in the 1980s as earlier explored by Chugh and Meador (1984).

2.3 Sentiment Analysis and Machine Learning Advances

2.3.1 Sentiment Analysis and Optimisation Basics

Sentiment analysis is a sub-field of statistical NLP whose roots stretched already to 1950. For example, Harris (1951) tried to find automatic methods for analysing of various language structures. However, this empiricist approach was subsequently prevailed by rationalists for a couple of decades (Manning and Schutze, 1999). The resurgence of the statistical-based branch has been then observed since 2000 with a growth of industry applications, publicly available textual data sets and a rise of computational resources (Liu, 2012).

A current definition of this practice can be that: "sentiment analysis, also called opinion mining, is the field of study that analyzes people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes." Liu (2012)

Liu (2012), furthermore, distinguished analysis of sentiment on three different levels:

- **Sentence** - The aim is to decide between positive, negative and neutral subjectivity of a single sentence.
- **Document** - This task broadens the scope of the previous one and focuses on determining of sentiment of whole documented, which was considered by Pang et al. (2002) among the first.
- **Entity and Aspect** - This approach tries to reveal not only the sentiment of a sentence or document, but also find out what explicitly people like and do not like expressed in a given text (Hu and Liu, 2004).

There are two main approaches to determine the sentiment of text contents - rule-based, and automatic or machine-learning-based - or also a combined one. The first method uses dictionaries of positively- and negatively-valenced words or phrases, and the score for a given sentence or document is simply derived by counting positive and negative occurrences. This approach can be further extended by assigning a score, on a continuous scale, to single elements, that expresses the valence of word elements, and the sentiment of the text is then based on the sum of these quantities.

These techniques are, however, far from being powerful. Therefore, they were substituted by machine-learning-based approaches. These are usually based on word corpora of highly pre-processed textual data (the text transformation, feature engineering and their role in sentiment analysis are well described by Haddi et al. (2013)). The corpus and training data are then used to create a document-term matrix, which carries term occurrences in individual sentences or documents, and a conventional machine learning (ML) algorithms such as logistic regression, random forest, support vector machine etc., are utilised to learn polarity of words via model weights. These models can be called bag-of-words (BoW) models.

Mathematically, the task described in the paragraph above thus can be expressed as a simple multi-class classification problem

$$
\begin{aligned}
\hat{\mathbf{y}} &= \operatorname{softmax}\left(f_A(\mathbf{x})\right), \ \mathbf{x} \in \{0,1\}^n \\
s.t. \ \min \ & \mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}; \mathcal{D}),
\end{aligned}
\tag{2.3}
$$

where $\hat{\mathbf{y}}$ represents a vector of probabilities given algorithm assign to individual output classes and $\mathbf{y}$ is a one-hot vector of depicting an underlying truth. $\mathcal{D}$ is here used to emphasise the loss function is minimised over the whole data sample, in practice usually over a validation set. $\mathbf{x}$ stands for a vector of ones and zeros capturing word or n-gram occurrence in an analysed text. $f_A(\cdot)$ denotes any predictive algorithm, softmax is a function mapping output values of

$f_A(\cdot)$ into the probability space

$$\text{softmax}(x_j) = \frac{e^{x_j}}{\sum_{i=1}^{K} e^{x_i}} \tag{2.4}$$

and $\mathcal{L}$ represents a convex loss function, for example commonly used cross-entropy, which is formulated as

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}; \mathcal{D}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{C} y_k^{(i)} \log \hat{y}_k^{(i)} \tag{2.5}$$

is commonly used. In equation (2.5), index i refers to observation i out of N samples, and k indexes output classes.

Since there is no analytical solution for the problem like (2.3), numerical iterative methods are used. The most simple technique is a gradient descent, first introduced long ago by Cauchy (1847). This algorithm minimises a differentiable function L until convergence as

$$\mathbf{w}_{n+1} := \mathbf{w}_n - \alpha \nabla L(\mathbf{w}_n),\ n = 0, .., N \tag{2.6}$$

where $\mathbf{w}_n$ represents model's weight, e ∇L denotes a gradient of the given function L, and α is a learning rate or also called step-size. In full gradient descent, the whole training set is used to evaluate $\nabla L(\mathbf{w}_n)$. More recently, this approach has been supplanted by stochastic gradient descent (SGD) (Robbins and Monro, 1951). SGD uses only random samples training examples to estimate gradient and thus tends to be much faster than full gradient descent. Furthermore, the noise introduced by random sampling helps the algorithm to avoid stopping in a local, but non-global, minima (Bottou, 1991).

This thesis relies on a modern adaptive optimisation algorithm introduced in Kingma and Ba (2014) - Adam. Adam was proposed as a speed and memory efficient combining strengths of two other methods.

- AdaGrad (Duchi et al., 2011) - A varaiation of SGD with per-parameter adaptive learning rate substantially improving the robustness of an optimisation process.
- RMSProp (Tieleman and Hinton, 2012) - Another adaptive algorithm utilising moving averages of past gradients to evaluate a current step.

The Adam's single update step is more complex than gradient descent's one, however, it can be still summarised in a few formulas. Let m_0 (first moment estimate), v_0 (second moment estimate) and t (time step) be 0. Furthermore, set α=0.001 (learning rate), $\beta_1 = 0.9$ and $\beta_2 = 0.999$ (decay rates for moment estimates) and $\varepsilon = 10^{-8}$ (correction term), which are default parameters' values according to Kingma and Ba (2014). Suppose $\mathcal{L}$ be our loss function, and $\mathbf{w}$ denotes model's parameters. Moreover, g_t denotes a gradient of loss function $\mathcal{L}$ with respect to model's weights $\mathbf{w}_t$, and g_t^2 represents an element-wise square, i.e. $g_t^2 = g_t \odot g_t$.

The, the update step is illustrated in Algorithm 1. Further details behind the choice of parameters can be found in the source paper (Kingma and Ba, 2014).

Algorithm 1: *Adam*: Update step
Source: Kingma and Ba (2014)

while $\mathbf{w}_t$ *not converged* **do**
  $t \leftarrow t + 1$
  $g_t \leftarrow \nabla\mathcal{L}(\mathbf{w}_{t-1})$
  $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) g_t$ (update biased first moment estimate)
  $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) g_t^2$ (update biased second moment estimate)
  $\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}$ (correct for bias)
  $\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^t}$ (correct for bias)
  $\mathbf{w}_t \leftarrow \mathbf{w}_{t-1} - \alpha \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}}$
end
return $\mathbf{w}_t$

2.3.2 Advances in NLP and Sentiment Analysis- Neural Networks

A considerable disadvantage of BoW is its disregard of semantic similarity between words and phrases. As research on artificial neural networks (ANNs) ramp up, the simple, but a bit toothless BoW was replaced by a continuous high-dimensional representation of words, so called a word embedding (usually vectors of a dimension between 50 and 300 are used), which was initially introduced in Skip-Gram model (Mikolov et al., 2013), and soon after in Glove (Pennington et al., 2014a).

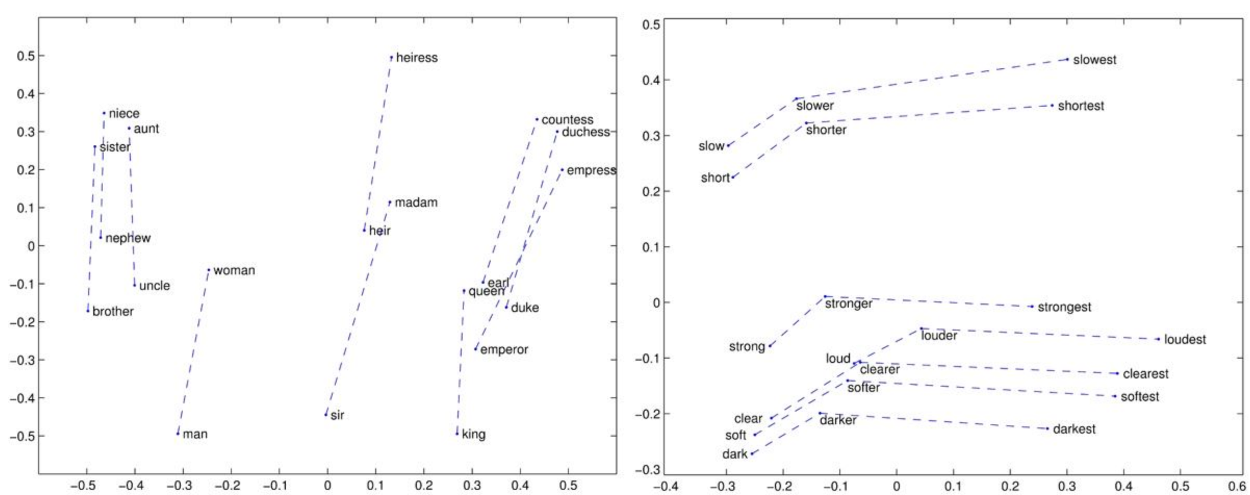


Figure 2.2: GloVe: Nuances between (i) man-woman, and (ii) adjective-comparative-superlativ
Source: (Pennington et al., 2014b) and (Pennington et al., 2014a)

These word embeddings can be used in ML algorithms either as a layer with pre-trained weights, which can be fixed throughout the training of the model or updated via back-propagation in a common way, or one can replace this by a single feed-forward fully-connected layer.

For the models utilising embeddings instead of the BoW, input representation given by equation (2.3) is no more suitable, and the problem can be re-defined as

$$\begin{aligned} \hat{\mathbf{y}} &= \text{softmax}\left(f_A(\mathbf{X})\right), \; \mathbf{X} \in \{0,1\}^{n \times m} \\ s.t. \; \min \; &\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}; \mathcal{D}), \end{aligned} \tag{2.7}$$

where $\mathbf{X}$ represents concatenated one-hot vectors of size n to the matrix of size $n \times m$ and m denotes a sequence length. Since the length of sequences differs, they are padded so that they can be processed by ML models.

Moreover, in order to achieve nowadays state-of-the-art results, 'classical' ML algorithms were replaced by neural-network-based counterparts with significantly more trainable parameters. The following variants (or possibly their combinations) belong to the most popular choices:

- **Convolutional neural networks (CNNs)** - This type of network utilises convolutional layers, which use kernel filters with shared weights across the whole input object that are applied to neighbourhood features (LeCun et al., 1998). Although they were initially designed for image classification, Kim (2014) amongst others successfully tailored this technique to various sentence classification problems, including sentiment analysis, and got state-of-the-art performance on multiple tasks.

  An undoubted advantage of CNNs for text classification in comparison with RNNs or general language models such as BERT is fewer trainable parameters, which results in less cumbersome and significantly shorter training. They, moreover, usually require only little hyperparameter tuning (Kim, 2014).

- **Recurrent neural networks (RNNs)** - This class of artifical neural networks (ANNs), which was initially derived from feed-forward ANNs, uses an updated hidden (internal) state, which enables a model to store past information, and thus are perfectly fit for dealing with time-series or sequence data.

  RNNs were firstly introduced in the seminal paper of Rumelhart et al. (1986), however, their architecture was strongly modified because vanilla RNNs suffer from vanishing or exploding gradient which completely cramps their fitting process. This stems from training these networks via backpropagation through time, which is a consequence of recursively updated hidden state, i.e.

$$\mathbf{h}^{(t+1)} = \mathbf{W}^\top \mathbf{h}^{(t)}, \tag{2.8}$$

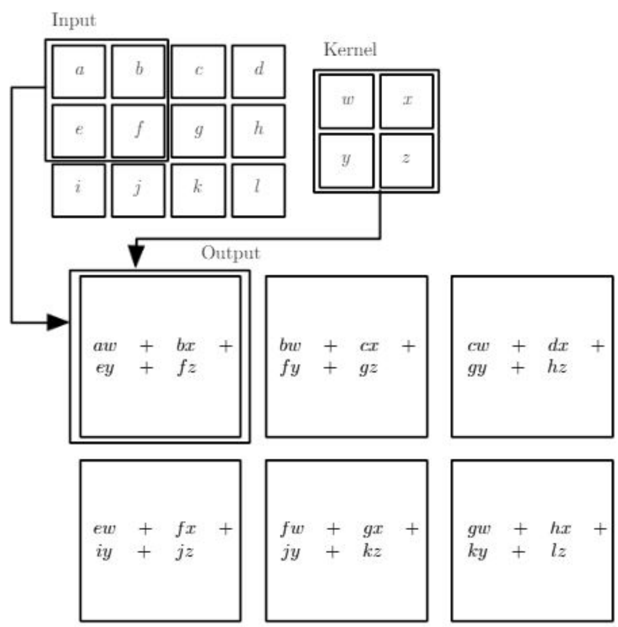


Figure 2.3: 2D Convolutional layer
Source: (Goodfellow et al., 2016)

which can halt training far from the optimum or can cause a model's weights will never converge. (Goodfellow et al., 2016).

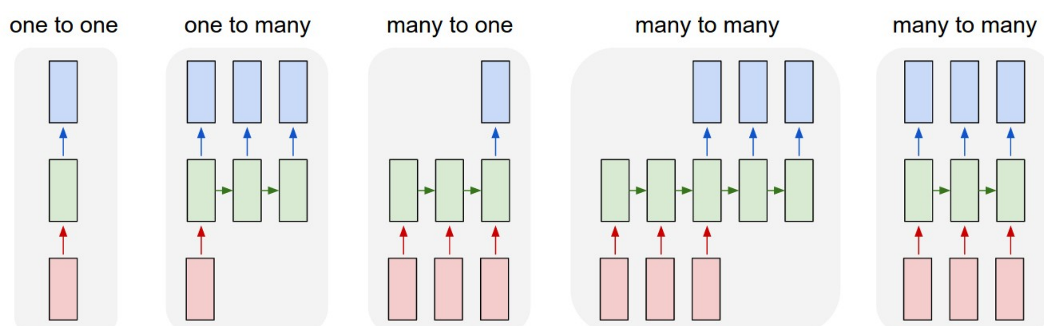


Figure 2.4: RNNs have internal state
Source: Karpathy (2015)

Motivated by these problems, Hochreiter and Schmidhuber (1997) and Cho et al. (2014) presented two mechanisms, long short-term memory and gated recurrent unit respectively, employing a forget gate which controls an extent of past information to be kept. These helped to tackle the vanishing gradient but did not solve the exploding gradient, which was meanwhile simply resolved using clipping (Pascanu et al., 2013).

- **BERT**

2.4 BERT for Sentiment Analysis

This thesis uses the language model firstly presented by Devlin et al. (2018). BERT has been a new state-of-the-art language model that "is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers" (Devlin et al., 2018). Consequently, the pre-trained model can be fine-tuned by adding a fully-connected layer on various tasks while achieving state-of-the-art results.

2.4.1 Architecture

BERT is based on the multi-layer bi-directional Transformer introduced in Vaswani et al. (2017). The key component of this model is the Transformer block consisting of two sub-layers, a multi-head self-attention, that is composed of a few parallel attention layers, and a position-wise fully-connected ANN. These two sub-layers are accompanied with residual connections (He et al., 2016) and a normalisation layer (Ba et al., 2016) to ease an optimisation process. Since a very good description of the transformer is provided in Vaswani et al. (2017), only the very crucial components are elaborated in this section.

Attention

The attention is the cornerstone of BERT. The attention mechanism was firstly used by Bahdanau et al. (2014) to enhance the performance of an encoder-decoder RNN model in neural machine translation tasks. The authors modelled the conditional probability of a next token, y_i, in the output sequence as

$$p(y_i|y_1, ...y_{i-1}, \text{input seq.}) = \text{Encoder-Decoder}(y_{i-1}, s_i, c_i), \tag{2.9}$$

where s_i is the model's hidden state use for emitting the output token and c_i denotes a context vector, which contains information about the sequence that is concentrated around the position i.

In order to compute c_i, they made use of an additive attention defined as a weighted sum:

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j, \tag{2.10}$$

where α_{ij} denotes a probability, an output of attention mechanism, and h_j represents annotation output by a decoder of the model. Attention, or also an *alignment model* according to Bahdanau et al. (2014) is then defined as

$$\alpha_{ij} = \text{softmax}\left(a(s_{i-1}, h_j)\right), \tag{2.11}$$

where a is a feed-forward network which is trained altogether with the whole model.

The alignment model can be understood as a network which tries to estimate the importance of h_j, considered with the previous hidden state s_{i-1}, for the next output token y_i. In other words, an attention mechanism decides which part of the input sequence the encoder-decoder should focus on.

For BERT, the additive attention were replaced by scaled dot-product attention introduced by (Vaswani et al., 2017), who defined it as

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^{\top}}{\sqrt{d_k}}\right)\mathbf{V}, \tag{2.12}$$

where $\mathbf{Q}, \mathbf{K}, \mathbf{V}$ represent queries, keys and values packed in matrices which determines a task attention functions are asked for (Vaswani et al., 2017), and a scaling factor $\frac{1}{\sqrt{d_k}}$ is used to avoid vanishing gradient for too large output values. While both types of attention mechanism are conceptually similar, the latter one is much more efficient in terms of computation and memory.

Where an attention mechanism is quite powerful itself, it was shown the performance can be further boosted by the utilization of multi-head attention. This is attained by running several attentions in parallel, whose output values are concatenated and then again put through the function once again yielding the ultimate output of the attention mechanism. This technique enables the model to jointly inference information from different positions in the input sequence (Vaswani et al., 2017).

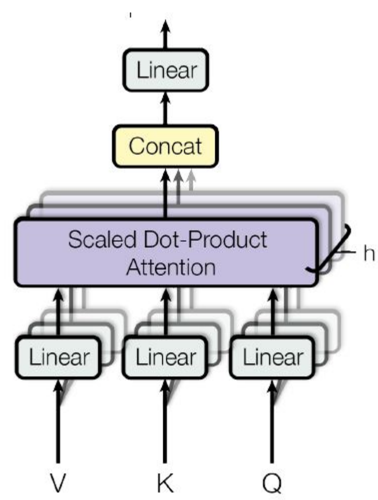


Figure 2.5: Multi-head attention
Source: Devlin et al. (2018)

,

2.4.2 Pre-Training and Fine-Tuning

The power of BERT lies in the model is pre-trained in an unsupervised way, and then can be simply fine-tuned on various down-stream tasks regardless of it is a single-sequence problem like sentiment analysis, or a pair-sequence exercise such as question answering or machine translation. This versatility is possible by a tailored input/output representation described in Devlin et al. (2018). This is based on special tokens and input and position embeddings helping the model to differentiate between single- and pair-sequence tasks, and between sequences A and B for the latter problem.

BERT is then pre-trained on large corpora like English Wikipedia, which contains a couple of thousands million words, with 2 tasks Devlin et al. (2018):

1. **Masked LM** - For the sake of simplicity, this is done via masking a certain percentage of tokens from input word embeddings, and these masked elements are then predicted by the model. The model is optimised with a minimisation of cross-entropy loss.

2. **Next sequence prediction** - This tasks helps significantly improves the performance of BERT on a two-sequence problems by understanding the relationship between a couple of sequences. In this case, training data are represented by pairs of sequences and the model should predict whether the second sequence follows the first one, or it is just a random one. The objective is to minimise binary cross-entropy loss.

After this stage, fine-tuning is on a turn. Fine-tuning of language models has been inspired by an accomplishment of a transfer learning in computer vision (CV), where researchers were attempting to deal with over-fitting of large-scale networks when trained on insufficient large data sets. For example, Donahue et al. (2014) achieved state-of-the-art results on multiple CV by transferring a models on a different-purpose task by fine-tuning the last layer. Long et al. (2015) adapted various the then superb pre-trained classification convolutional neural networks on semantic segmentation and again got the very best results with a solid margin. In the latter case the authors left unfrozen a few last layers during the training. Both approaches correspond with a suggestion not to transferred all model's layers at once as it may lead to forgetting already well learnt weights (Yosinski et al., 2014)

Later in NLP, fine-tuning had proved to be proficient between related tasks - sentiment analysis (Severyn and Moschitti, 2015), machine learning translation (Sennrich et al., 2015) or question answering tasks (Min et al., 2017). However, attempts to transfer models between dissimilar problems had been failing (Mou et al., 2016) until the endeavours of Howard and Ruder (2018) or BERT (Devlin et al., 2018)

In case of BERT, fine-tuning is pretty straightforward. Due to attention mechanisms and the specific model architecture, BERT is capable to handle various down-stream tasks by adding:

- a task-specific output layer that is appended at the top of the pre-trained model,
- special tokens (described in (Devlin et al., 2018)) helping BERT to understand which problem should be learnt on.

Importantly, fine-tuning is significantly less expensive than pre-training itself, and therefore, in a few hours this model can be adapted to different NLP tasks on a single GPU Devlin et al. (2018).

2.5 Sentiment Analysis in ESG Investing

Research on employee sentiment (regardless of we are talking about sentiment analysis in realm of NLP or not) and its relevance to the performance of securities followed on the findings on investors' and managements' sentiment and corresponding stock returns. For example, (Chen et al., 2014a) found evidence between the stock returns and investors' opinion expressed via social media. In this study, the authors used articles and commentaries from Seeking Alpha. This and similar studies, however, cannot be assigned to the pure ESG literature.

(Huang, 2018) analysed more than 14.5 million product reviews posted by customers on Amazon.com. They constructed portfolio each month, longing on stocks of companies with abnormal high customer ratings and shorting for the ones with poor ratings within the past months. An abnormal rating was defined as a difference between past-month mean rating and the average over the last year. Through the years from 2004 to 2015, the authors of this study were able to achieve monthly alpha between 0.51 % and 0.60 %, which was highly statistically significant, on equal-weighted portfolio measured by Fama-French and Fama-French-Carhart models. The excessive returns with review-weighted portfolios even rose to the range of 0.77 % and 0.79 %. They, furthermore, showed that abnormal ratings are positively correlated with future companies' returns and earnings surprises, suggesting firms' fundamental information and related share prices are carried by customer opinions.

More recently, studies utilising crowd-sourced data sets of employee feelings have emerged. Green et al. (2019) used a data set of 1 million Glassdoor reviews for more than 1200 unique firms.The researchers found a positive correlation between stock returns and improvement in the employer rating. At the end of each quarter in the period spanning over almost 9 years, they were constructing three different portfolios according to rating change in the last quarter. These then consisted of companies from

- the top 20 %,
- the middle 60 %, and
- the bottom 20 %

the ranking of change in ratings. The equal-weighted portfolio of the top 20 % companies yielded higher average monthly returns and alpha (according to the Fama-French-Carhart model) than the bottom counterparts by 0.84 % and 0.88 % respectively, and these differences were claimed to be statistically significant at the 1% level. For value-weighted portfolio the differences was lower roughly by 0.1 % but with the same level of statistical significance.

Green et al. (2019), furthermore, noticed the effect was more detectable among current employees, novice ones and those ones writing longer reviews or working in a country of the headquarter. In other analyses, the authors found that ratings and reviews were related to the companies' cash flow and other financial fundamental information, which resulted in the predictability of stock performance.

The similar findings that an impact of employee sentiment and stock returns are more prevalent among the certain groups of workers were made by Chen et al. (2020). They used even larger data set containing more than 1.7 million reviews of current employees for more than 3,600 firms. In this study, however, the researchers employed a different methodology, and used employee sentiment index (ESI),

$$ESI = \frac{\#\text{positive reviews} - \#\text{negative reviews}}{\#\text{reviews}}, \tag{2.13}$$

instead of a change in the mood of employees. They concluded stocks associated with low ESI tended to outperform the lower-rated counterparts. Nonetheless, the authors labelled all reviews with the rating of 4 or higher as positive, and negative otherwise, which might be a bit inconsistent with Green et al. (2019), who had found a mean and median rating was 3.2 and 3.0 respectively, therefore the threshold of 4.0 may have lead to the biased data.

At the same time, quarterly (or possibly monthly) difference in ratings seem to be a more valuable indicator than ratings themselves because these are relatively well explained by various determinants, including company's size, return on assets, past stock returns and others, controlled for time and firms (Green et al., 2019). On the sample of roughly 16,000 quarter-firm-level observations for 1,200+ companies, the authors reported R-squared of 0.649 and 0.074 for rating and change in rating respectively regressed on the aforementioned variables.

The propitious results that employee expectations are associated with future stock returns were also confirmed by Sheng (2019), who further supported employees' feelings expressed online have amassed a huge amount of information, that can be harvested for the incorporation into the investment decision process, despite the fact employees have no superior information about future returns of their firms [(Benartzi, 2001), (Cohen, 2009)].

Sheng (2019) on the sample of approximately 1 million Glassdoor reviews for 1,422 companies showed a positive correlation between abnormal positive outlook and future excessive returns. They, therefore, looked at the outlook, which is optional information filled in with reviews, and defined positive outlook as a fraction of reviews with positive outlook. Abnormal outlook was then calculated as a different between positive outlook in the past month and the mean over three prior months. The companies were then sorted according to that factor and tercile portfolios were created. It was found out that monthly excess returns of the high portfolios were higher by 0.86 % in average measured in the Fama-French-Carhart framework. Moreover this difference was statistically significant at the 1% level. The authors then construct a long-short portfolio buying the top and selling the bottom tercile companies, which results in the ca 0.7% monthly excess return.

Even though there are multiple promising findings on the relevance of employee sentiment and returns in the stock market, to the best of my knowledge, there is no study investigating the relationship between this sentiment and corporate bond returns. This aligns with the research focused on factor investing in the corporate bond market which is much less developed compared with an equity-oriented literature.

On the other hand, more recently, ESG-oriented analyses of sentiment in a combination with big data and ML appeared in other spheres. Serafeim (2020) used two data sets - ESG Performance scored by MSCI and more than 250,000 ESG-related articles for roughly 8,000 companies gathered by TruValue Labs used for the calculation of public ESG sentiment momentum - to show that long-short portfolios built upon the superior/inferior ESG performance and negative/positive sentiment momentum yielded significant alpha over the period of ten years.

3. Data - Selection, Sources and Exploration

This chapter outlines the selection criteria determining which companies are included in the study. Sources, characteristics together with some exploratory analysis, and pre-processing of market and employee sentiment data are also presented.

3.1 Company Selection Criteria

This thesis is based on information from three different sources:

- **Bloomberg** - market data on corporate bonds elaborated in Section 3.3,
- **Glassdoor** - employee sentiment data described in Section 3.2,
- **Yahoo Finance** - basic information about companies, including the sector and industry a given firm operates in, the country of headquarter, total revenue and a number of employees.

The data set used for this thesis is based on the sample of companies listed on three stock indices - S&P 500, FTSE 100, EURO STOXX 50 - and thus representing three different markets in the U.S., the U.K. and Europe.[1] This first choice criterion was determined by:

- companies included stock market indices represent an easily available source of firms that can be scraped for this study.
- the very limited amount of employee reviews and ratings for Asian markets. This is likely to be caused by a tendency of people to use different online platforms in these markets, such as it is the case with other online social media (Choi, 2020).

Due to the limited time assigned to the final project, this thesis works with the employee data in the relatively short period from July 1, 2018, to June 30, 2020, because the lengthy scraping, which is justified in Section 4.1.1. I, furthermore, narrowed the data set only on the

[1] U.K and Europe companies are underrepresented compared with U.s. ones, however, it is not problematic as the firms are mainly studied within a single market.

companies with at least 10 Glassdoor reviews over the monitored time frame of two years. This results in a sample of 605 companies spanning across 11 market sectors (Lake, 2020), and their distribution in individual markets is displayed in Table 3.2.

	S&P 500	**FTSE 100**	**EURO 50**
Basic materials	18	9	4
Communication services	20	10	3
Consumer cyclical	63	15	6
Consumer defensive	36	11	5
Energy	23	2	2
Financial services	70	17	9
Healthcare	61	4	4
Industrials	69	10	7
Real estate	23	2	1
Technology	64	2	3
Utilities	25	4	3
Total	472	86	47

Table 3.1: Distribution of companies across stock market sectors

The sample of 605 firms is used for the exploratory analysis of ratings and reviews presented in Appendix. However, since not all the companies issue bonds, I further retain only the companies using public debt. Eventually, I end up with the data set consisting of xyz business entities.

	S&P 500	**FTSE 100**	**EURO 50**
Basic materials	14	9	4
Communication services	15	10	3
Consumer cyclical	43	15	6
Consumer defensive	25	11	5
Energy	20	2	2
Financial services	58	17	9
Healthcare	45	4	4
Industrials	46	10	7
Real estate	15	2	1
Technology	46	2	3
Utilities	23	4	3
Total	350	86	47

Table 3.2: Distribution of companies issuing bonds across stock market sectors

Last, besides the selection criteria outlined above, I also couple information on market data for Alphabet Inc. with the employee reviews for Google since there are significantly more Glassdoor records for the latter company, which is the major subsidiary of the aforementioned holding conglomerate.

3.2 Employee Sentiment Data

Glassdoor is one of the largest job sites in the world offering millions of job listings and represents a still expanding database of company reviews written by current or former employees. There are also interview reviews and salary reports. As of July 2020, there are posted about 60 million reviews, salaries and insights on roughly 1 million employers. The websites are visited by approximately 50 million unique visitors each month (Glassdoor, 2020). I was given written express permission from a representative of Glassdoor to use all publicly available data via an e-mail.

All employee reviews are anonymous and consist of numerous elements describing the relationship between an employee and a company and his or her opinion on the firm.

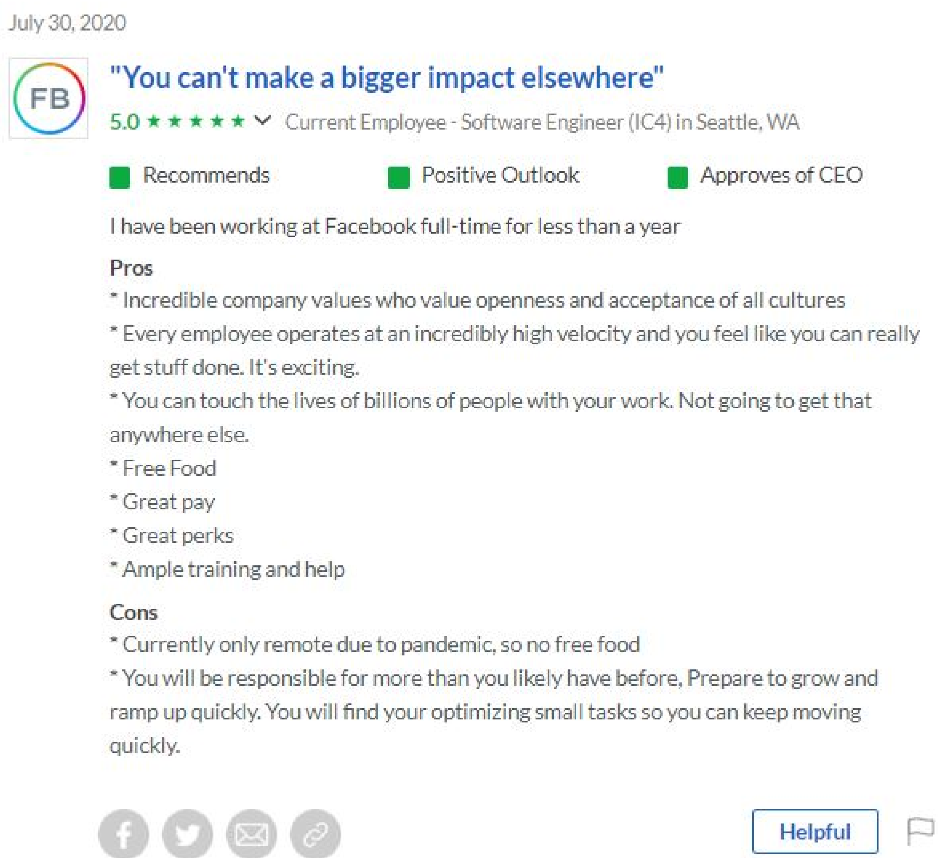


Figure 3.1: An example of employee review posted on Glassdoor
Source: Glassdoor

Individual components of the reviews are:

- Date - Date a review was posted.
- Main title - This is also used as a review link, and there is no specification on the content of this field.

- Star rating - The rating on scale 1 to 5 which consists of five sub-components: *Work/life balance, Culture & Values, Carrer Opportunities, Compensation and Benefits* and *Senior Management*. For this thesis, only the aggregated score is used, however, the proposed database intended for potential future use should contain also these sub-elements.
- Job title - An item specifying job title altogether with a information on whether the review author e is current or former employee.
- Job location - Location of a unit an employee has been or working at.
- Recommendation - Two pre-defined options: *Recommends, Does not recommend*
- Outlook - Three pre-defined choices: *Positive, Neutral, Negative*
- CEO - Three pre-defined values: *Approves of CEO, No opinion of CEO, Disapproves of CEO*
- Contract - The line containing the type of contract: *full-time, part-time, intern*. It also provides information about the length of the contact.
- Pros - Aspects the author likes. Minimum of 5 words
- Cons - Aspects the author likes. Minimum of 5 words
- Advice to management

For the analysis in this thesis, only reviews written in English were scraped. Moreover, I confine myself only to the reviews added only by full-time or part-time employees since they make up for the vast majority of all records, and intern reviews may not be entirely comparable or equivalent in weight and detail to reviews of employees with much longer work experience, due to the different nature of an intern's position.

With respect to all the criteria from sections 3.1 and 3.2, I gathered the data set of 392,408 reviews for 605 companies in the period of 2 years. For the firms issuing bonds, there are 283,002 records for 353 firms. All the data were scraped with a tool built for this thesis. This web crawler together with its application is thoroughly described in Section 4.1.1.

One valuable data insight is that a distribution of a number of reviews strongly varies on the levels across companies, stock markets and market sectors as well. There are 648.60 reviews per company in average with a standard deviation of 1681.00. The selected quantiles are provided below and accompanied by a corresponding histogram depicted in Figure 3.2.

<u>Quantlies:</u>

- 10^{th} - 39.40
- 25^{th} - 83.00
- 50^{th} - 214.00

- 75^{th} - 559.00
- 90^{th} - 1361.20
- 100^{th} - 27455.00

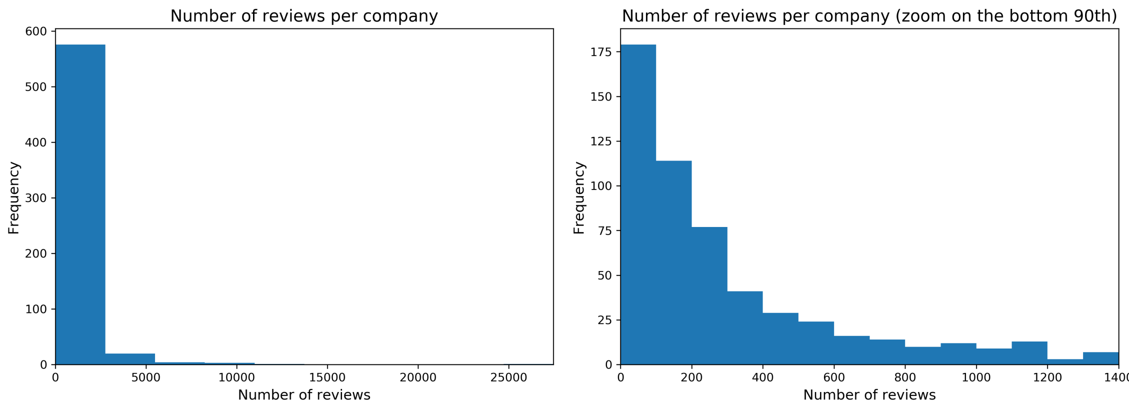


Figure 3.2: Distribution of review

Tables 3.3 and 3.4 then display summary statistics according to the stock market indices and sectors respectively.

Market index	Total	Mean	Std	10^{th}	Q1	Med.	Q3	90^{th}
S&P 500	336,930	713.83	1859.45	44.1	93.0	222.0	588.0	1443.9
FTSE 100	33,497	389.50	698.33	27.0	62.3	134.5	322.3	968.5
EURO STOXX 50	21,973	467.51	772.73	42.2	69.5	190.0	534.5	1043.8

Table 3.3: Number of reviews per company across stock markets

Market sector	Total	Mean	Std	10^{th}	Q1	Med.	Q3	90^{th}
Basic Materials	3,798	122.52	173.51	29.0	47.5	62.0	123.5	237.0
Communications	22,444	680.12	936.59	41.80	73.0	205.0	827.0	2146.8
Consumer Cyclical	92,959	1106.65	3325.23	63.0	111.5	226.5	692.8	1983.9
Consumer Defensive	44,658	858.81	1849.34	49.1	92.3	303.0	630.0	1590.1
Energy	6,742	249.70	338.39	32.0	52.0	105.0	272.5	707.4
Financial Services	59,490	619.69	915.15	57.5	113.5	265.0	609.5	1549.0
Healthcare	31,891	462.19	588.24	72.0	127.0	256.0	587.0	910.0
Industrials	40,102	466.30	627.61	44.0	97.3	259.5	569.8	1032.5
Real Estate	3,552	136.62	227.54	12.0	18.0	72.5	145.0	233.0
Technology	83,958	1216.78	2324.17	62.2	127.0	341.0	1041.0	3641.4
Utilities	2,806	87.69	61.32	30.1	47.0	73.0	104.0	186.8

Table 3.4: Number of reviews per company across market sectors

Furthermore, there are differences the ratings distribution over individual stock market indices and also market sectors. These results are provided in Tables 3.5 and 3.6. Other tables describing the variation in a higher level of granularity can be found in Appendix.

Last but not least, it is desirable to point out there are nuances between ratings of full-time versus part-time and current versus former employees. This information can be read from Table 3.7, where are, among others, a category of 'Not specified' employees, which may be caused by various factors.

- the piece of information regarding the contract and employee relationship is not explicitly displayed alongside a review,
- errors during parsing textual data from Glassdoor, which can stem from a non-standard formatting for given reviews.

Market index	Mean	Std	Q1	Median	Q3
S&P 500	3.59	1.23	3.0	4.0	5.0
FTSE 100	3.61	1.20	3.0	4.0	5.0
EURO STOXX 50	3.91	1.15	3.0	4.0	5.0

Table 3.5: Descriptive statistics - Ratings across stock markets

Market sector	Mean	Std	Q1	Median	Q3
Basic Materials	3.56	1.23	3.0	4.0	5.0
Communication Services	3.61	1.27	3.0	4.0	5.0
Consumer Cyclical	3.62	1.22	3.0	4.0	5.0
Consumer Defensive	3.4	1.25	3.0	4.0	4.0
Energy	3.66	1.18	3.0	4.0	5.0
Financial Services	3.59	1.2	3.0	4.0	5.0
Healthcare	3.48	1.29	3.0	4.0	5.0
Industrials	3.59	1.27	3.0	4.0	5.0
Real Estate	3.67	1.36	3.0	4.0	5.0
Technology	3.78	1.15	3.0	4.0	5.0
Utilities	3.54	1.31	3.0	4.0	5.0

Table 3.6: Descriptive statistics - Ratings across market sectors

Employee	Total ratings	Mean rating	Std	Q1	Med.	Q2
Full-time	316,117	3.63	1.23	3.0	4.0	5.0
Part-time	58,240	3.56	1.17	3.0	4.0	5.0
Not specified	18,043	3.50	1.29	3.0	4.0	5.0
Current	220,135	3.80	1.16	3.0	4.0	5.0
Former	154,224	3.35	1.27	3.0	4.0	5.0
Not specified	18,041	3.50	1.29	3.0	4.0	5.0
Total	392,408	3.61	1.23	3.0	4.0	5.0

Table 3.7: Summary statistics of employee ratings

In Tables 3.8, 3.9, 3.10 shows the equivalent statistics to Tables 3.5, 3.6, 3.7 (with some additional pieces of information), just confined on the companies issuing bonds within the monitored period. Unfortunately, it is obvious there is no chance to conduct any statistically

Market index	Total companies	Total ratings	Mean	Std	Q1	Median	Q3
S&P 500	350	274,426	3.60	1.24	3.0	4.0	5.0
FTSE 100	2	5,341	3.74	1.14	3.0	4.0	5.0
EURO STOXX 50	1	3,235	4.13	1.02	4.0	4.0	5.0

Table 3.8: Descriptive statistics - Ratings across stock markets

significant analysis for companies listed on FTSE 100 and EURO STOXX 50, therefore, the analysis of employee sentiment is confined to bonds of companies listed on S&P 500.

Other useful statistics and data insights concerning the Glassdoor data used for this thesis can be found in Appendix.

Market sector	Total ratings	Mean	Std	Q1	Median	Q3
Basic Materials	2,498	3.60	1.20	3.0	4.0	5.0
Communications	18,781	3.57	1.28	3.0	4.0	5.0
Consumer Cyclical	71,644	3.66	1.21	3.0	4.0	5.0
Consumer Defensive	29,423	3.35	1.26	3.0	4.0	4.0
Energy	3,085	3.61	1.20	3.0	4.0	5.0
Financial Services	43,584	3.59	1.22	3.0	4.0	5.0
Healthcare	25,564	3.44	1.30	3.0	4.0	5.0
Industrials	23,060	3.55	1.28	3.0	4.0	5.0
Real Estate	1,779	3.66	1.42	3.0	4.0	5.0
Technology	61,954	3.76	1.17	3.0	4.0	5.0
Utilities	1,630	3.64	1.31	3.0	4.0	5.0

Table 3.9: Descriptive statistics - Ratings across market sectors

Employee	Total ratings	Mean rating	Std	Q1	Med.	Q2
Full-time	226,117	3.62	1.24	3.0	4.0	5.0
Part-time	43,539	3.57	1.18	3.0	4.0	5.0
Not specified	13,346	3.50	1.29	3.0	4.0	5.0
Current	157,649	3.79	1.16	3.0	4.0	5.0
Former	112,008	3.35	1.27	3.0	4.0	5.0
Not specified	13,345	3.50	1.29	3.0	4.0	5.0
Total	283,002	3.60	1.23	3.0	4.0	5.0

Table 3.10: Summary statistics of employee ratings

Apparently, there are some differences in a distribution of ratings across stock market sectors. Even more interestingly, the analysis of monthly changes in employees sentiment suggests the sentiment develops non-identically across stock market sectors and stock market indices. This is illustrated by correlation matrices depicted by Figures 3.3 and 3.4.

Specifically, there are several sectors negatively correlated on FTSE 100 and EURO STOXX 50. It is noticeable, a few sectors exhibiting a negative relation on these two markets, on the other hand, are strongly correlated on S&P 500. These detailed correlation matrices can be found in Appendix.

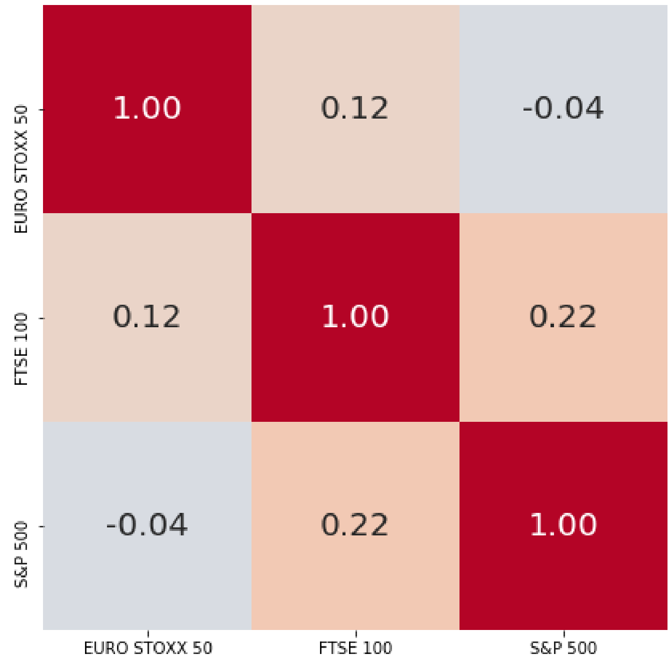


Figure 3.3: Correlation of monthly changes in sentiment across stock markets

3.2.1 Data for fine-tuning

I also had to gather a data set of Glassdoor reviews used for fine-tuning of BERT. For this purpose, I merged data from two sources:

- a data set with an anonymised companies downloaded from Kaggle[2],
- residual reviews, that were scraped during obtaining employee data for this dissertation and were out of the two-year period.

This endowed me with a set of XYZ examples, which were split in a training and validation set of sizes XY and YZ, respectively.

3.3 Market Data

This thesis uses corporate bond ask and bid prices and interest rate. The source of all these data is Bloomberg and they will be retrieved with Bloomberg Terminal software in a combination with *Rblapi*[3], which is the API package for R software. The license necessary for using Bloomberg service was provided by UCL.

[2] Data source: www.kaggle.com/fireball684/hackerearthericsson

[3] https://cran.r-project.org/web/packages/Rblpapi/Rblpapi.pdf

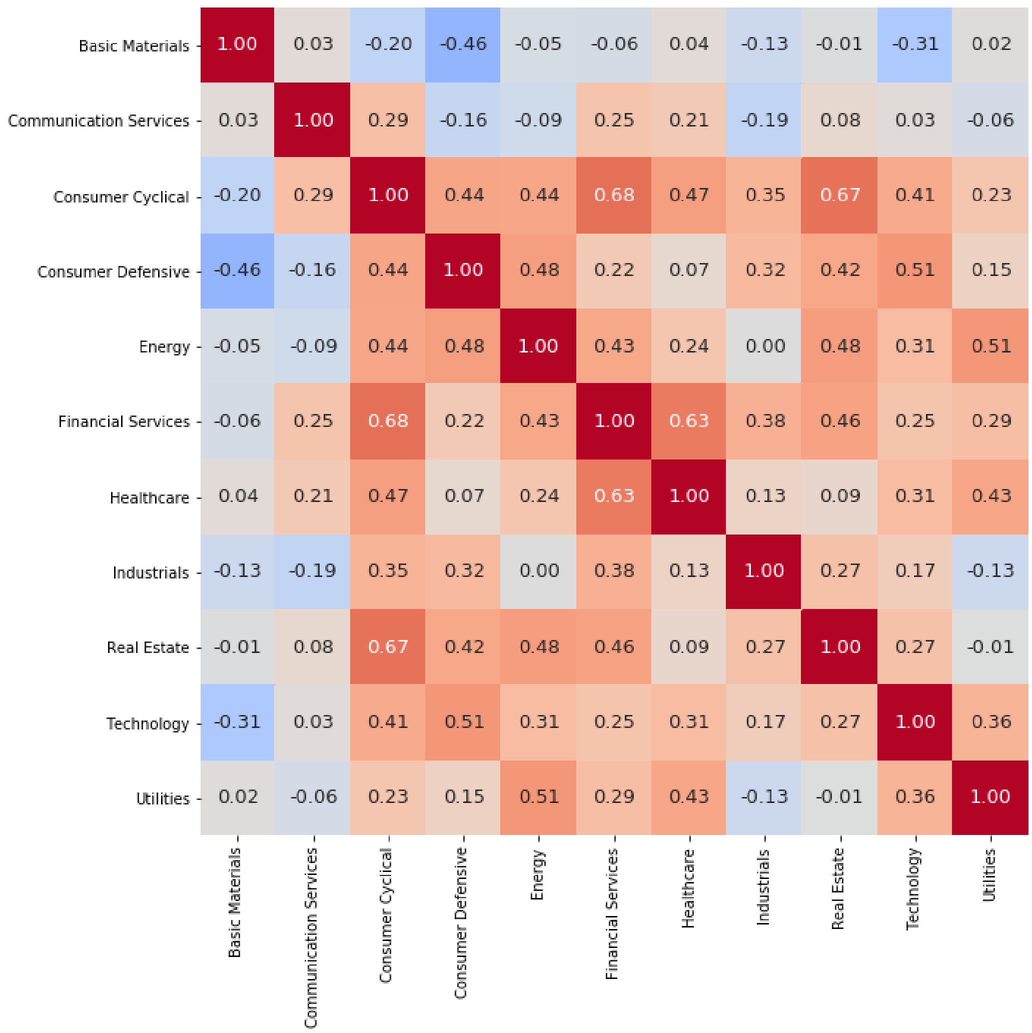


Figure 3.4: Correlation of monthly changes in sentiment across market sectors

In summary, I collected XYZ monthly returns for S&P 500 firms in total. These data were retrieved using a "lookup" function integrated in Bloomberg Terminal, which were not able reach already maturated bonds. Some bonds may have also been omitted due to a nature of the limit imposed on search results. For further studies, it would thus be more beneficial to use some licensed data to cover the larger scope of corporate bonds.

Considering the methodology of return calculation covered in the next section, Table 3.11 displays descriptive statistics on th

Table 3.11: Caption

3.4 Data Pre-Processing

3.4.1 Employee Sentiment Data

In this thesis, no pre-processing of the employee ratings data was not conducted as they were not fed into any algorithm required standardised data. Therefore, it was unblemished to leave ratings ranges from 1 to 5. On the other hand, there was a necessity to transform reviews data so that they can be read by BERT. For this purpose, first the reviews had to be tokenised. In this project, the pre-trained '*bert-base-cased*' tokeniser delivered by Hugging Face was used.

As some reviews were too long, it was beneficial to set a maximum length and cut those exceeding this limit. This step is appropriate because shortening a small fraction of sequences resulted in losing only a tiny amount of information while training and evaluation time is considerably reduced. The maximum length of 300 was used in this study.

After this procedure, a token <CLS> had to be added before each review. This is done in order a language model will be able to distinguish which down-stream task should be done. Furthermore, each sequence was appended with a token <SEP> that determines the end of that sequence.

As all input sequences must be of the same length, all of them were padded with respect to the longest sequence (or the length limit). This was achieved by adding <PAD> tokens in a combination with generation attention mask. This mask is represented by a vector of ones and zeros, which is an indication for BERT which input tokens should be considered for loss calculation and subsequent optimisation.

Finally, both arrays, the tokenised input and attention mask, were transformed to the PyTorch tensor, which is a standardised input type format used for deep learning models built with *PyTorch*[4].

3.4.2 Market data

For the purpose of simplicity, the monthly return on the corporate bond was calculated as

$$R_t = \frac{Ask\_price_t - Ask\_price_{t-1} + Interest}{Ask\_price_{t-1}},\ t = 2, ..., T, \tag{3.1}$$

when bonds are longed, and

$$R_t = \frac{Bid\_price_{t-1} - Bid\_price_t - Interest}{Bid\_price_{t-1}},\ t = 2, ..., T, \tag{3.2}$$

[4]Deep learning library; www.pytorch.org/

if they are shorted. Furthermore, when the bonds performance was trailed over past 3 months for determining the momentum, the return was determined as

$$R_t = \frac{Price_{t-3} - Price_t}{Price_{t-3}},\ t = 4, ..., T, \tag{3.3}$$

where $Price_t$ was computed as a mean value of bid and ask prices.

While this approach does not perfectly reflects the methodology of how corporate bond returns should be calculated, it should provide us with sufficient evidence whether the proposed trading strategy represents any meaningful signal for portfolio construction. On the other hand, this can lead to overstating portfolios returns as transaction costs are neglected, which is discussed in the discussion Chapter 6.

4. Methodology

This chapter consists of five parts. First, it presents scraping tools devised for gathering data from Glassdoor and Yahoo Finance. Subsequently, it provides an outline of the retrieval pipeline and database. A description of different approaches to scoring employee sentiment and incorporation of these analysis results into the multi-factor model follow. The closing section covers test methodology.

4.1 Tools Developed for Data Retrieval

Two different scraping tools were developed for this thesis. First, the *selenium*-based[1] web crawler for automated browsing through Glassdoor and parsing employee reviews. Second, a relatively simple program based on standard Python HTTP and parsing libraries, *requests*[2] and *Beautiful Soup*[3], that is intended for getting information from Yahoo Finance. The latter one is also capable of downloading auxiliary data from Wikipedia.

Both scrapers are designed to store the acquired data set either as a CSV/Excel file, or in a database created with SQL statements generated with another Python package, *Django*[4].

4.1.1 Glassdoor Scraper

Due to a design of the Glassdoor web pages, the scraper comprises multiple components that are briefly described in the **Components** subsection below. Everything is accompanied by supportive justifications, illustrations and code snippets. All the codes presented in this section are parts of my work. The high-level application of this tool is then presented in the **Application** subsection.

Since browsing the Glassdoor website requires user interaction, it is necessary to use a browser automation library such as *Selenium*. This enables us to build a robot that goes through web pages and interacts with them like a human does, but in an almost completely automatic way.

[1] pypi.org/project/selenium
[2] pypi.org/project/requests
[3] pypi.org/project/beautifulsoup4
[4] pypi.org/project/Django and docs.Djangoproject.com/en/3.1

It is important to mention that libraries such as *selenium* usually require a web driver for their full functionality. This thesis relies on *ChromeDriver*[5].

Before diving into the individual components of scraper, I would like to introduce the whole scraping process briefly. The bot needs to handle following tasks:

1. signing in to a Glassdoor account,
2. finding the first review page of a company,
3. unrolling all reviews exceeding a certain length on the page,
4. scraping all reviews on the page,
5. moving on to the next page for the company,
6. repeating steps 3 to 5 until the last page containing reviews is reached. Then, terminate scraping and start with another company.

During the process outlined above, the program is required to locate various elements in the web page HTML. These functions used for this task are, therefore, introduced at this place so that they can be easily referenced later in the text. Let import and initialised methods and classes. The class "webdriver.Chrome(**args)", where **args represents argument options,

```
from selenium import webdriver
driver = webdriver.Chrome(**args)
```

Code Snippet 4.1: Import web driver module

then have the following functions available for detecting various content.[6]

- "find_element_by_class_name"
- "find_element_by_id"
- "find_element_by_name"
- "find_element_by_xpath"

The choice of a strategy using for locating elements depends on the convenience of a given solution. In other words, it depends on a style how to code is written. In some cases, it is possible to locate elements with more than one choice, sometimes a coder is enforced to stick to an only available option.

[5] chromedriver.chromium.org

[6] The list below contains only the methods used in this project. There are other functions, which can be found at www.selenium-python.readthedocs.io/locating-elements.html. The attributes follow an HTML convention.

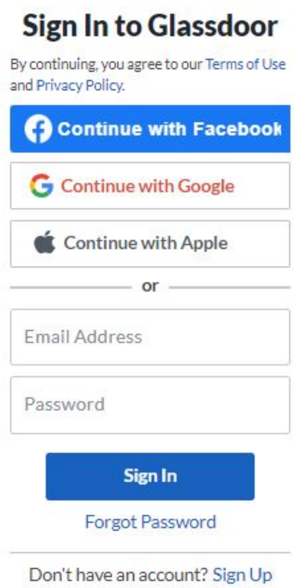


Figure 4.1: Glassdoor - Login portal
Source: Glassdoor

Components

Surfing the Glassdoor websites needs a user or a robot to be signed in so that reviews in the full scope will be available. The first crucial component of the scraper is then responsible for logging in with user credentials. The login portal is portrayed in Figure 4.1, and a corresponding high-level source code handling all necessary actions is provided in Code Snippet 4.2

```python
from selenium import webdriver
driver = webdriver.Chrome(**args)
## code continues... ##

driver.find_element_by_name('username').send_keys(<email>)
driver.find_element_by_name('password').send_keys(<password>)
driver.find_element_by_xpath('//button[@type="submit"]').click()
```

Code Snippet 4.2: Glassdoor - Login

As personal data, including an e-mail and a password, are not appropriate to be stored in the source code, the better practice is to create a separate JSON[7] file. This file, which contains such sensitive information, is then opened and parsed directly within the main file. This procedure is also followed in this work.

Another step in getting employee reviews is to search for a company we are interested in. In

[7] json.org

general, there are two main ways to do this.

1. A user-friendly approach is to find a firm by inserting its name into the company search on the web pages (see Figure 4.2). However, this method was not used because it proved to be quite unstable, as the bot was often re-routed to URLs that cannot be loaded. Moreover, more than one company can be found with the same name, therefore, reviews for an incorrect firm can be unintentionally scraped.

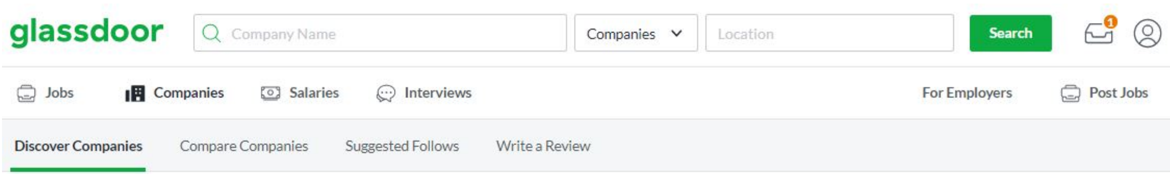


Figure 4.2: Glassdoor - Company search
Source: Glassdoor

2. The alternative is to manually find a URL address to the first page of reviews for the required company. While this method is less comfortable since it requires the user to find these URL links manually, it is much faster and reliable in general than the first approach.

Once the page with reviews for a given company is obtained. By default they are sorted by the popularity of individual records. This is not the most desirable option when one wants to scrape records over a certain time period without handling a number of reviews out of that time range. It is thus more efficient to order the reviews by date. This can be easily achieved by appending to the URL address the following statement "sort.sortType=RDsort.ascending=false".

The cornerstone of crawling through the Glassdoor web pages lies in unrolling reviews by clicking on the "Continue reading" button (see Figure 4.3), which is located at the bottom of most reviews exceeding a certain length. Without clicking on "Continue reading" (when present), it is impossible to retrieve the full text of the review from the HTML code. After scraping a review, it takes about 1.5-2.5 seconds for the website to refresh and for another clicking element (for the next review) to become available. This idle period may sometimes even prolong, which is why an error handling mechanism is desired, which is implemented as shown in (pseudo) Code Snippet 4.3. For this error handling, it is more efficient to loop over shorter breaks rather than a single longer one because the second option might impede the speed of scraping.

As soon as the reviews are unrolled, particular elements, described in Section 3.2, from individual posts can be parsed. This is achieved with *re*, a standard python module. Since this procedure consists of multiple functions built using a few simpler functions, covering all the details is beyond the scope of this report. The source code can be found in the cited GitHub repository.

```
import time
success = 0

while success == 0:
    try:
        driver.find_elements_by_xpath(<`Continue reading` path>)[0].click()
        success += 1
    except:
        time.sleep(0.5)
```

Code Snippet 4.3: Glassdoor - Continue reading

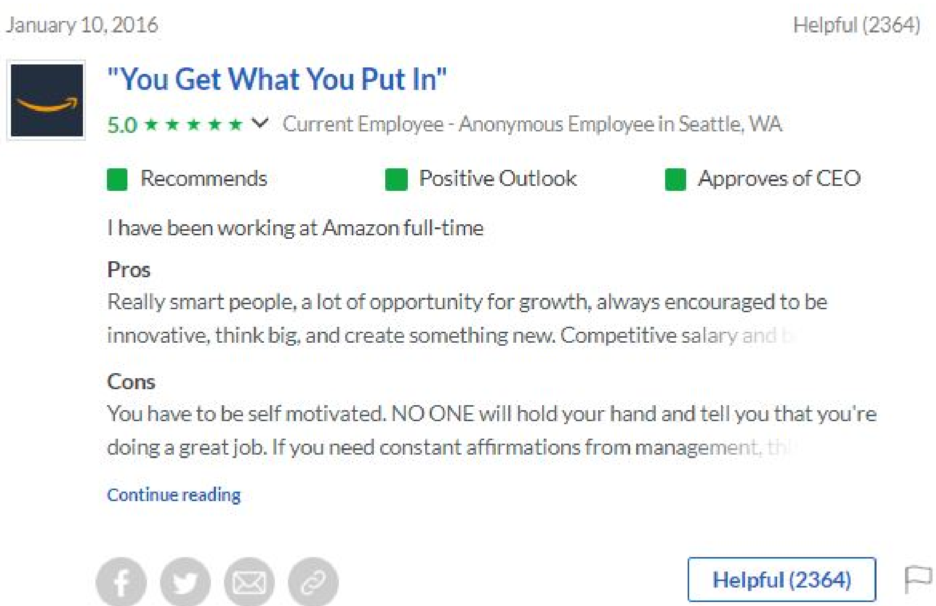


Figure 4.3: Glassdoor - 'Continue reading' button
Source: Glassdoor

The last step ensuring smooth operation of the tool is to stop scraping when a page with no review is reached, which is possible on Glassdoor. To avoid browsing empty pages indefinitely, we can simply count reviews on a page and terminate if desired (see Code Snippet 4.4).

```
reviews = driver.find_elements_by_xpath('//li[@class="empReview cf"]')
reviews.extend(driver.find_elements_by_xpath(
    '//li[@class="noBorder empReview cf"]'))

if len(reviews) == 0:
    `TERMINATE SCRAPING`
```

Code Snippet 4.4: Glassdoor - Counting reviews on a single page

Application

Notwithstanding the complexity of the scraper itself, its use is pretty simple, and everything can be executed from a terminal. Furthermore, the scraper allows to directly specify various optional parameters. The simplest application of my tool can be illustrated with the Code Snippet 4.5.

This command runs scraping from the main file. First, "chrome_drive_path" is necessary to specify as a web driver constitutes an essential component of scraper. Credentials are represented by a JSON file containing a user e-mail and password. Company names along with corresponding URL links are fed through two text files. These can be generated from an Excel/CSV sheet with another python application in the repository. The last two arguments, "min_date" and "mysite_path", specify the latest date of review, we desire to obtain, and a path to a Django application defining our database respectively.

```
python main.py  --chrome_driver_path '/mnt/c/data/chromedriver/'
                --credentials '/mnt/c/data/credentials.json'
                --companies '/mnt/c/data/companies.txt'
                --url '/mnt/c/data/urls.txt'
                --min_date '2018-06-30'
                --mysite_path '/mnt/c/mysite'
```

Code Snippet 4.5: Glassdoor scraper - application

Other optional with corresponding documentation can be found in the README.md file stored in the GitHub repository of this thesis.

4.1.2 Yahoo Scraper

Yahoo scraper tailored for this work is conceptually significantly lighter compared with the Glassdoor tool described above. This is owed to the availability of all necessary information immediately after loading of a company's web page. Thus it is not necessary to build a bot interacting with a website. This program is, therefore, built with well-known python modules, *requests* and *Beautiful Soup*. The former packages is used for obtaining the HTML source code and the latter one for parsing text.

All the pieces of information, that are used in this work, are retrievable from two sections - Profile and Financials (see Figure 4.4). These sections are easily accessible for all companies listed on any stock exchange because of a generic form of the URL address as shown in Code Snippet 4.6.

```
url = f'https://finance.yahoo.com/quote/{<stock>}/{<section>}?p={<stock>}'
```

Code Snippet 4.6: Generic form of Yahoo Finance URL

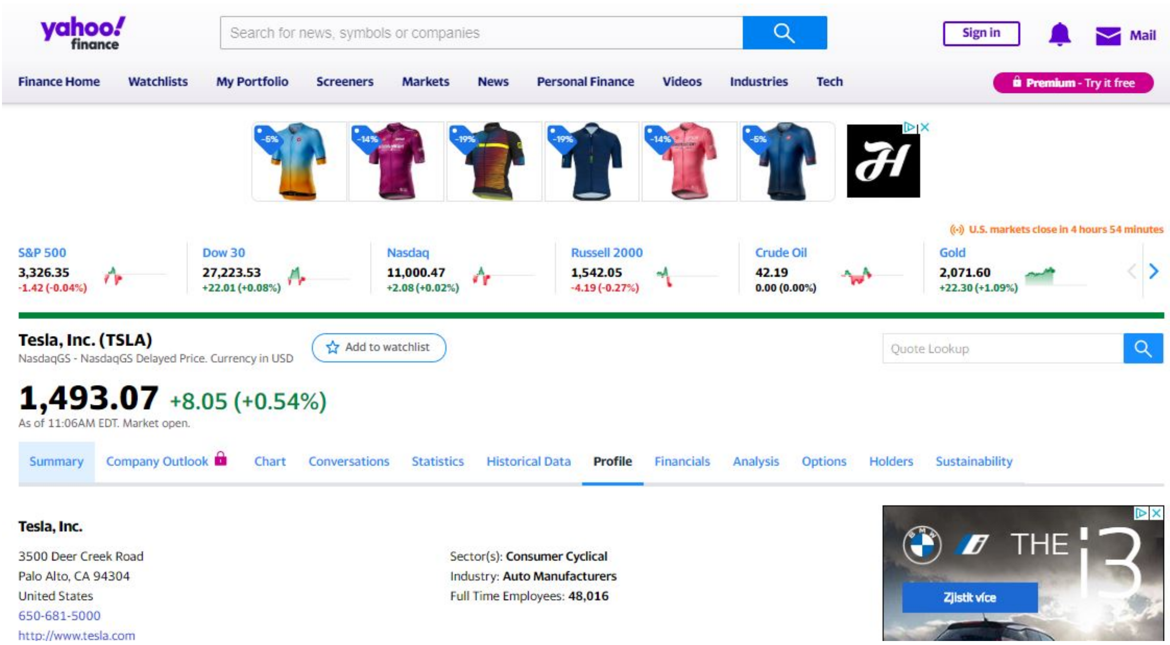


Figure 4.4: Yahoo Finance
Source: Yahoo Finance

The stock indices can be either gathered manually, or Yahoo Scraper has a built-in feature, which can download indices for stocks listed on S&P 500, FTSE 100 or EURO STOXX 50 indices from Wikipedia. This Wikipedia crawler works similarly as the Yahoo one because all the data are obtainable directly from the first loading of the page. This program is, therefore, built using a combination of *requests*, *Beautiful Soup* and *re* modules. The last one helps with the text parsing. Getting data from Wikipedia is the least difficult out of all three cases described, as all the data are well formatted in tables.

Application

Similarly to the application of Glassdoor crawler, this program can be again directly executed from a terminal. A possible application is in Code Snippet 4.7.

```
python main.py --stock_indices 'S&P 500 | FTSE 100 | EURO STOXX 50'
               --mysite_path '/mnt/c/mysite'
```

Code Snippet 4.7: Yahoo and Wikipedia scraper - application

This command first runs the Wikipedia scraper and gets all companies (with their stock symbols) currently listed on S&P 500, FTSE 100 and EURO STOXX 50. These data are temporarily stored in the scraper *object*. Subsequently, information about firms are gathered from Yahoo Finance and the individual records are immediately pushed to the *Django* database.

4.2 Database and Pipeline

The backbone of the database consists of two interconnected tables, which are administrated by *Django* and proved to be working with *SQLite*[8] or *MySQL*[9] databases at the back-end. For the projects of lower scale, *SQLite* is completely sufficient. Furthermore, there is no need for running the database on a server when *SQLite* is used. Therefore, this project is based on *SQLite*.

The only limitation of *SQLite* is an impossibility of concurrent writing into the database. Importantly, attempting to simultaneously write into the database is not prevented by *SQLite*. It is, therefore, completely on the users of the software to make sure there are no events of concurrency. If this paradigm is not obeyed, *SQLite* will not be able to handle this order, which will result in the corruption of the whole database. While there are techniques for repairing such crippled files, there is still a non-negligible danger of losing the whole work.

The two tables that constitute the database here are intended for storing companies' information (*Company* model) and employee reviews (*Review* model) respectively. These tables are created with the *Django* class *models.Model* which is responsible for generating database queries. This object consists of one or more fields that represent column names in the output table. There are multiple standard field formats supported by *Django*[10], and all of them are rooted at *django.db.model*. This project works with the following ones.

- CharField
- IntegerField
- FloatField
- TimeField
- ForeignKey - The field representing a many-to-one relation, i.e. various inputs in one table are strictly linked to the one record in the other one.

The items above usually require to set a few parameters. Individual fields have been, therefore, initialised as shown in Code Snippets 4.8. The concrete *models.Model* schemes implemented in this project can be found in Appendix.

First, the parameter *max_length* determines the maximum length of character input. Another one, *blank*, allows to insert an empty string, which is an alternative for the *null* value in case of integers and floats. The argument *unique* ensures each company occurs exactly once in the *Company* table. Insisting on unique records is necessary since the connection with the other table through *models.ForeignKey* would not be working otherwise. The last yet covered parameter is *on_delete* that specifies an action to take with records if their superordinate is

[8] www.sqlite.org/
[9] www.mysql.com/
[10] All of them can be found at docs.djangoproject.com/en/3.1/topics/db/models/

```python
from django.db import models
max_length=<max_length>

class Company(models.Model):
    company = models.CharField(max_length=max_length, blank=True, unique=True)

    def __str__(self):
        return self.company

class Review(models.Model):
    entityField = models.ForeignKey(to=Company, on_delete=models.CASCADE)
    characterField = models.CharField(max_length=max_length, blank=True)
    integerField = models.IntegerField(null=True)
    floatField = models.FloatField(null=True)
    timeField = models.TimeField(null=True)

    def __str__(self):
        return self.characterField
```

Code Snippet 4.8: Django Model - Class instantiation

deleted. An option *models.CASCADE* enforces all subordinates entries to be removed with their superordinates. Finally, an inner function *__str__(self)* defines the value to be printed in the database overview.

Since the *Review* model depends on the *Company* one, users have to build the latter table first. The ultimate structure is resembled with Figure 4.5. My system, moreover, ensures the uniqueness of *Review* and *Company* records. This is attained by scanning a database for an input to save, and the new record is pushed in if and only if the same record does not exist, as shown in Code Snippet 4.9, where *args contains values for the model's fields.

4.3 Sentiment Analysis

In this thesis, sentiment analysis is used as a tool for analysing employees' opinions about their company. I, therefore, calculate sentiment proxied by either ratings or reviews retrieved from Glassdoor for a company. The sentiment scores are calculated in rolling time windows, which are covered in Subsection 4.3.2.

Generally, the sentiment score for company i at some time period j is given as

$$\text{Sentiment}_{ij} = \frac{\sum_{k=1}^{N_{ij}} w_k^{(ij)} \cdot r_k^{(ij)}}{\sum_{k=1}^{N_{ij}} w_k^{(ij)}}, \; k = 1, ..., N_{ij}, \tag{4.1}$$

where N_{ij} is the total number of reviews for company i at time period j. $r_k^{(ij)}$ denotes either

```
from MyDjangoModels import Review
def checkIfRecordExists(record):
    try:
        Review.objects.get(*args)
        return True
    except Company.DoesNotExist:
        return False

def writeRecord(record):
    if not checkIfRecordExists(record):
        reviewRecord = Review(*args)
        reviewRecord.save()
    else:
        pass
```

Code Snippet 4.9: Django - Writing a record if it does not already exist

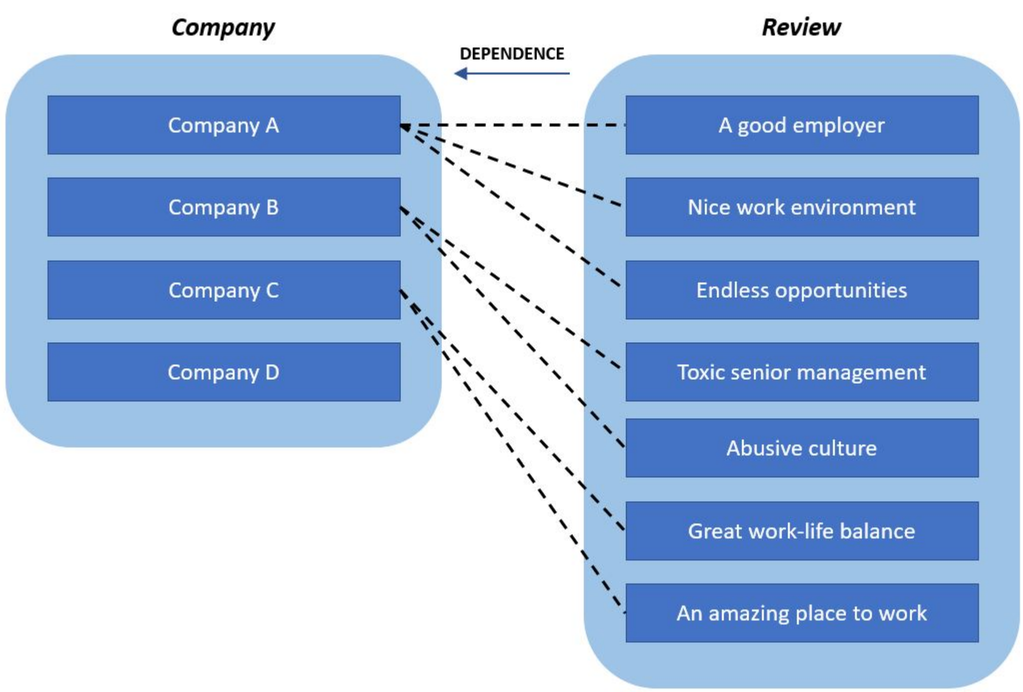


Figure 4.5: Django - Database structure

a star rating, ranged from 1 to 5, or probability of a review being positive, when reviews are used. The probability that the given review being positive is returned by the language model BERT. $w_k^{(ij)}$ represents a weight of a given rating or a review, which is again further described in Subsection 4.3.2.

4.3.1 BERT Implementation

In this thesis, BERT is used as a language model to estimate the probability of a review being positive. Each review consists of two concatenated parts, pros and cons. To make reviews of a suitable format for the model, they must be first padded to the same length. Moreover, special tokens must be attached and attention mask needs to be generated as described in Section 3.4. All these steps can be easily done with a pre-trained tokeniser delivered by Hugging Face. The transformed input can be, therefore, directly obtained with a few lines of codes.

```
from transformers import BertTokenizer
tokenizer = BERT.from_pretrained('bert—based—cased')
transformed_review = tokenizer.encode_plus(
    review,
    max_length=max_length,
    truncation=True,
    add_special_tokens=True
    pad_to_max_length=True,
    return_attention_mask=True
    return_tensors='pt'
)
```

Code Snippet 4.10: BertTokeniser - Review transformation

Once the input is transformed, it can be directly passed to the BERT model tailored for sentiment analysis. The architecture of the model used in this master's project can be summarised as follows:

$$\text{Model} = \{\text{BERT} \rightarrow \text{DROPOUT} \rightarrow \text{FULLY-CONNECTED}\}.$$

Here, the base version of BERT is used (the model with 12 hidden layers with an embedding size of 768 units and the 12-head attention). After this block, a dropout layer, with a dropout probability of 0.3, is implemented to regularise our model. Finally, a fully-connected layer, followed by a softmax function, is placed. The output of this whole model is then represented with a vector of size 3.

$$\text{Model(review)} = \begin{bmatrix} \text{Prob. of being positive} \\ \text{Prob. of being neutral} \\ \text{Prob. of being negative} \end{bmatrix} = \begin{bmatrix} p^{+} \\ p^{0} \\ p^{-} \end{bmatrix} \tag{4.2}$$

Since the ultimate, fully-connected layer needs to be trained, the model is fine-tuned with all weights being frozen except for the ones in the very last layer. The fine-tuning is conducted with the following parameters, which follow the recommendations of Devlin et al. (2018).

- Objective: Minimisation of cross-entropy loss

- Number of epochs: 4
- Batch size: 32
- Optimiser: AdamW
- Learning rate: $4 \cdot 10^-5$

The whole model is written in *PyTorch*[11]. Besides, Hugging Face's implementations of the pre-trained BERT and AdamW are used. Moreover, the model and the corresponding definitions of a training and validation step and epoch are wrapped with *PyTorch Lightning*[12]. Encapsulation with this high-level library brings a couple of advantages. First, it allows a programmer to avoid a lot of boilerplate code and thus makes the script more readable for other developers. Second, the code for the model and its optimisation is completely identical regardless of device (i.e. CPU, GPU, TPU) used for a computation. These two points combined together further implies explicitly allocation of *PyTorch* models and tensors to a device within a script is no more necessary as *PyTorch Lighting* handles this job on itself.

4.3.2 Approaches to Employee Sentiment Scoring

The sentiment score is evaluated either based on employees' ratings or reviews. There are following three approaches tested in this thesis.

1. **Ratings** - The simplest method for determining sentiment score for a given company i in period j is to calculate this metric directly on the star ratings. In this case, each record has the same weight. Equation 4.1 can be, therefore, simplified as

$$\text{Sentiment}_{ij} = \frac{\sum_{k=1}^{N_{ij}} r_k^{(ij)}}{N_{ij}}. \tag{4.3}$$

2. **Unweighted reviews** - Another approach uses reviews as a proxy to calculate sentiment score. On this occasion, each review is scored with BERT. The sentiment of an individual review is defined as

$$\text{sentiment}(\text{Model}(\text{review}_k^{(ij)})) = p_k^+ - p_k^-,\ k = 1, ..., N_{ij}, \tag{4.4}$$

where p_+ and p_- are output probabilities of the language model. In this study, the additive method is preferred to the ratio one, i.e.

$$\text{sentiment}(\text{Model}(\text{review}_k^{(ij)})) = \frac{p_k^+}{p_k^-},\ k = 1, ..., N_{ij}, \tag{4.5}$$

because the latter one can generate undesirable outliers in the data if the review is as-

[11] www.pytorch.org
[12] www.pytorch-lightning.readthedocs.io

signed with a very high probability, close to 1, to be positive, and a very low probability, close to 0, to be negative (or also vice versa). On the other hand, formula from equation 4.4 ensures

$$\text{sentiment}(\text{Model}(\text{review}_k^{(ij)})) \in [-1, 1]. \tag{4.6}$$

I also include both positive and negative probabilities, p_k^+, p_k^-, because disregarding, for example, p_k^- from equation 4.4 leads to the losing of information about whether the model returns a higher probability the review being neutral or negative. It would have resulted in an ambiguity like

$$r_1^{ij} = \begin{bmatrix} 0.3 \\ 0.65 \\ 0.05 \end{bmatrix} = \begin{bmatrix} 0.3 \\ 0.05 \\ 0.65 \end{bmatrix} = r_2^{ij}, \tag{4.7}$$

which is, of course, an undesired information loss.

Finally, the sentiment for the whole company in an observed time frame is then again calculated as in (4.3) with $r_k^{(ij)}$ being given as

$$r_k^{(ij)} = \text{sentiment}(\text{Model}(\text{review}_k^{(ij)})) = p_k^+ - p_k^-,\ k = 1, ..., N. \tag{4.8}$$

3. **Weighted reviews** - The last investigated method further takes the length of reviews into consideration. The review length is measured by its total number of characters, and represents the weight of a review in (4.1). An aggregated sentiment score can be thus expressed with the following formula

$$\text{Sentiment}_{ij} = \frac{\sum_{k=1}^{N}(\text{review\_length}_k^{(ij)}) \cdot r_k^{(ij)}}{\sum_{k=1}^{N} \text{review\_length}_k^{(ij)}}. \tag{4.9}$$

The sentiment scores are calculated in one-month and three-month periods, starting with the beginning of a calendar month. Furthermore, time series of month-to-month changes in sentiment score are also computed, and used as a bond performance indicator.

4.4 Proposed Factor Model

The proposed model is conceptually similar to those presented in Section 2.1 (Factor Investing in the Corporate Bond Market). In order to construct the full long-short portfolio, we build long and short decile portfolio with respect to the companies' sentiment factor, which is calculated according to the methodology in Section 4.3. In this case, a definition of a decile portfolio slightly differs from the one normally used when top and bottom 10 per cent of bonds are picked to be long-bought and short-sold, respectively. In my case, top and bottom companies instead of bonds are selected with respect to the sentiment factor. This sorting is also done as the first step in the bond selection process.

Since there is usually more than one type of a bond per company, as the firm may issue debt securities with different maturities etc., another ordering rule is desirable to introduce in order to avoid buying or selling all types of bonds issued by picked companies. In this thesis, the low-risk factor is arbitrary chosen as an auxiliary characteristic for sorting the bond universe. Recalling a definition of the low-risk factor, the bond with the shortest maturity is picked to buy and the longest-maturity debt security is to be sold for top and bottom companies, respectively

I introduced that additional measure for the bond selection in order to cap investor's exposure to the individual firms. I also assume it would be unreasonable to buy and short all types of bonds issued by the selected top and bottom companies without further consideration of their performance.

4.5 Test Methodology

For the testing of the efficacy of the employee sentiment factors, mean monthly returns on the portfolios are calculated. The monthly return on a bond in the portfolio follows the data pre-processing from Section 3.4, i.e.

$$R_{l,t} = \frac{Bid\_price_{l,t} - Bid\_price_{l,t-1} + Interest_l}{Bid\_price_{l,t-1}},\ l = 1, ..., |L_t|,\ t = 1, ..., T \tag{4.10}$$

if the bond is longed, and

$$R_{l,t} = \frac{Ask\_price_{l,t-1} - Ask\_price_{l,t}}{Ask\_price_{l,t-1}}, \tag{4.11}$$

if it is shorted. In equations 4.10 and 4.11, l denotes bond l from the pool of all bonds of size L_t available to trade at time t. Again, this definition of bond returns is rather simplified as it neglects elements such as compound interest or transaction costs (the latter one might be cumbersome to estimate). Furthermore, monthly interest payments are assumed to be held for the purpose of simplicity.

The return on the whole portfolio is then calculated as an arithmetic mean, i.e.

$$R_t = \frac{\sum_{l \in \mathcal{B}_t} R_{l,t}}{|\mathcal{B}_t|},\ t = 1, ..., T \tag{4.12}$$

where $\mathcal{B}_t$ denotes indices of bonds chosen for the portfolio at time t. The arithmetic mean is used because the unweighted bond portfolios are considered. This means each bought or sold bond is assumed to have the same weight in the portfolio.

Even though the definition of returns are simplified, and thus does not have to perfectly match the reality, I believe it can still provide us with a valuable insight whether the utilisation of employee sentiment is proficient in picking good and poor performing bonds respectively. The holding period for all bonds is one calendar month. The formation period is determined according to the length of time frame used for the sentiment score calculation highlighted in Section 4.3, i.e.

- one month,
- three months.

The results of this analysis are further compared with the returns of the momentum-based portfolio, which is used as a baseline. In this case, the holding period is again one month and the formation period lasts three months.

Besides mean returns, standard deviation is also computed so that we can test the returns for statistical significance. For this purpose, I calculate t-value

$$|t\text{-value}| = \left| \frac{\bar{R}}{\sqrt{T \cdot \frac{\sum_{t=1}^{T}(R_t - \bar{R})^2}{T-1}}} \right|, \tag{4.13}$$

where $\bar{R}$ denotes the mean monthly return. Finally, t-value together with a degree of freedom, which equals T[13], is used for obtaining p-value.

Moreover, correlation among sentiment-based and momentum-based portfolios are investigated. In this case, I use the Pearson's correlation coefficient

$$\rho_{ij} = \frac{\sum_{t=1}^{T}(R_t^S - \bar{R}^S)(R_t^M - \bar{R}^M)}{\sqrt{\sum_{t=1}^{T}(R_t^S - \bar{R}^S)^2 \cdot \sum_{t=1}^{T}(R_t^M - \bar{R}^M)^2}}. \tag{4.14}$$

In the equation above, R^S and R^M corresponds to the sentiment-based and momentum-based portfolio returns, respectively.

[13] T is the total number of portfolio month returns, thus it gives us the number of observations. Importantly, I assume monthly returns to be independent according to the efficient-market hypothesis (Fama, 1970).

5. Employee Sentiment in Generating Return

This chapter first evaluates an incorporation of a factor, derived from numerical ratings as a proxy for sentiment, into the factor model and its ability to generate return. Subsequently, this technique is compared with the model utilising factors that are derived from different NLP sentiment scoring methods. The results are studied against the returns momentum- and low risk-based portfolios. This study suggests that taking employee sentiment into account can generate neco neco.

The experiments according to the methodology described in Sections 4.4 and 4.5 were run and the results are now summarised in Section 5.1 and 5.2. In all experiments below, the following notation is used.

- **1M** - Employee sentiment calculated as a month average;
- **∆1M** - Month-to-month change in **1M** employee sentiment;
- **3M** - Employee sentiment calculated as mean over three calendar months;
- **∆3M** - Monthly change in **3M** employee sentiment;
- **Low risk** - Bonds are chosen using a simplified low-risk strategy, i.e. bonds with the shortest maturity are long-bought and one with the longest are short-sold;
- **Momentum** - Bonds are chosen using a momentum-based strategy.

Furthermore, as suggested in Section 3.2 the results are presented only for S&P 500. This is because an analysis for companies listed on other two indices, FTSE 100 and EURO STOXX 50, is inadequate due to a severe lack of firms I was enabled to collect bond data for.

5.1 Ratings

The returns on portfolios based on ratings and momentum are summarised in tables below. The holding period of 1 month was used for in all cases.

	1M	Δ1M	3M	Δ3M	Low risk	Momentum
Mean return	-0.37 %	-0.33 %	-0.33 %	-0.24 %	-0.35 %	0.08 %
St. dev.	1.19%	1.14 %	1.35 %	1.62 %	2.19 %	2.74 %
|t|	1.43	1.34	1.11	0.68	0.72	0.13
p-value	0.17	0.20	0.28	0.50	0.48	0.90
Annualised return	-4.36 %	-3.93 %	-3.85 %	**-2.87 %**	**-4.07 %**	0.92 %

Table 5.1: Returns of the long-short portfolios (ratings); T=21 (number of monthly returns)

Table 5.1 depicts the results of the analysis conducted. In this case, all the returns are negative, which might be disappointing. However, if one focuses only on the long-bought bonds, the returns of the long portfolios, a significant improvement can be observed (see Table 5.2). It, therefore, seems there is a flaw in a short strategy.

Ignoring the absolute term, we can still compare performance of individual strategies relatively. Even though all sentiment-based portfolios generate worse returns than the momentum-based one, they exhibit considerably lower variation. It can be also recognised the strategies based on month-to-month changes, instead of absolute sentiment score, tend to perform better. Furthermore, a longer formation period looks to be beneficial as well. The mean monthly return of the **Δ3M** strategy is higher by 0.13 percentage point in comparison with the **1M** strategy. However, the difference is not statistically significant as $t = 0.28$ for two-sample t-test.

	1M	Δ1M	3M	Δ3M	Low risk	Momentum
Mean return	0.51 %	0.53 %	0.50 %	0.72 %	0.30 %	1.13 %
St. dev.	1.45%	1.52 %	1.35 %	1.20 %	0.44 %	4.09 %
|t|	1.62	1.59	1.69	2.75	3.17	1.27
p-value	0.12	0.13	0.11	0.01	0.00	0.22
Annualised return	6.31 %	6.50 %	6.15 %	**9.00 %**	3.70 %	14.48 %

Table 5.2: Returns of the long portfolios (ratings); T=21 (number of monthly returns)

Now, consider the return of long-only portfolios. These returns are overstated due to neglecting all transaction costs, although it makes a good sense to compare individual performance relatively. Again the sentiment-based strategies tend to perform worse than the momentum-based one. Nevertheless, there is even a larger difference between the variance of sentiment- and momentum-based portfolios. Low-risk portfolios yield the lowest average return, however, standard deviation is unequivocally smallest.

The **Δ3M** is again a strategy with the highest yield out of all sentiment-based approaches. More importantly, this portfolio has a considerable low value of standard deviation of its returns. Together, mean return is statistically significant at the level of 0.05.

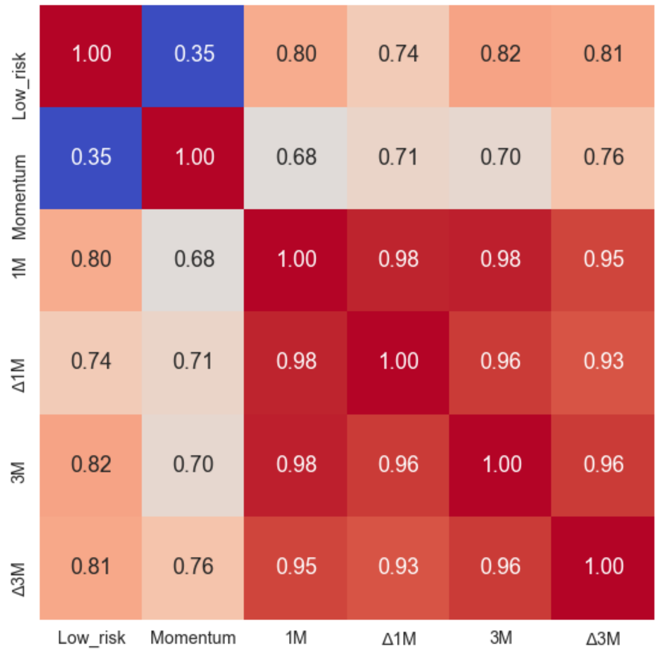


Figure 5.1: Correlation of returns - Long-only portfolios

Furthermore, correlation between the portfolios' returns can be also investigated. This metric for long-only strategies is depicted by 5.1. Clearly, all the sentiment-base portfolios exhibit high pair-wise correlation (all correlation coefficients exceed 0.93). Furthermore, sentiment-base strategies seems to demonstrate higher correlation for the portfolios with the same formation period.

The pair-wise correlation between the returns of the low-risk-based portfolio and the sentiment-based ones is still pretty strong. In this case, the value ranges from 0.74 to 0.82 for **Δ1M** and **3M**, respectively. The relationship between sentiment-based portfolios and the moment-based one is further a bit weaker with a correlation coefficient dropping between 0.68 and 0.76 for **1M** and **Δ3M**. Moreover, $\rho(\text{Low risk, Momentum}) = 0.35$, which proves these two strategies are suitable to combine to increase the level of diversification.

I, therefore, also constructed four multi-factor long-only portfolios to investigate the effect of combination of two or three factors on the mean return and variance. For this purpose, the top $|\mathcal{B}_t|/2$ or $|\mathcal{B}_t|/3$ from single portfolios are picked for the multi-factor one in case of two or three factors respectively. For a reminder, $|\mathcal{B}|$ denotes the total number of bonds chosen for each of single-factor portfolios. In this case, $|\mathcal{B}|$ is 30. The results of this analysis are displayed in Table 5.3. In that table, **LR** and **M** are used as an abbreviations for **Low risk** and **Momentum**, respectively. **Full** represents the portfolio constructed with **LR+M+Δ3M** strategy and with and consisting of full-sized single portfolios.

Interestingly, even the most diversified **Full** portfolio, containing 90 bonds, performs worse

	LR+M	LR+Δ3M	M+Δ3M	LR+M+Δ3M	Full
Mean return	0.58 %	0.45 %	0.83 %	0.57 %	0.72 %
St. dev.	1.93 %	0.79 %	2.40 %	2.04 %	1.76 %
|t|	1.39	2.60	1.58	1.29	1.87
p-value	0.18	0.02	0.13	0.21	0.08
Annualised return	7.24 %	5.50 %	**10.41 %**	7.09 %	8.98 %

Table 5.3: Return of the long multi-factor portfolios (ratings); T=21

than the single **Δ3M** one, consisting of 30 bonds, considering comparable results, but the higher variance of the former one. Unfortunately, adding a **Δ3M** criterion to **LR+M** and keeping the same size of the whole portfolio leads to slightly poorer results both in terms of average monthly returns and their corresponding standard deviation. Also, although **LR+Δ3M** tends to generate higher results than the single **Δ3M** portfolio, this increment is rather overweight by a significant rise in the returns' variance.

5.2 Comparison of Different NLP Sentiment Scoring Methods

6. Discussion

In the beginning, this chapter discusses the significance and interpretation of results derived in the previous chapter. Then, the limitations of my approach altogether with potential applications are considered. Eventually, the chapter proposes recommendations for future work conducted by other researchers.

6.1 Interpretation of Results

6.2 Limitations

6.3 Application

6.4 Future Work

7. Conclusion

This chapter highlights key findings and scientific contributions of this study, which among others include the developed scraping and storing tools (now freely available to other researchers). The dissertation is concluded by setting a direction for future research.

Bibliography

What is ESG? https://www.credit-suisse.com/be/en/asset-management/esg-investing.html, 2018. Accessed: July 2020.

J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

R. W. Banz. The relationship between return and market value of common stocks. *Journal of financial economics*, 9(1):3–18, 1981.

F. Barth, H. Scholz, and M. Stegmeier. Momentum in the European corporate bond market: The role of bond-specific returns. *The Journal of Fixed Income*, 27(3):54–70, 2017.

S. Basu. Investment performance of common stocks in relation to their price-earnings ratios: A test of the efficient market hypothesis. *The journal of Finance*, 32(3):663–682, 1977.

S. Benartzi. Excessive extrapolation and the allocation of 401 (k) accounts to company stock. *The Journal of Finance*, 56(5):1747–1764, 2001.

L. Bottou. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91 (8):12, 1991.

M. M. Carhart. On persistence in mutual fund performance. *The Journal of finance*, 52(1): 57–82, 1997.

A. Cauchy. Compte rendu des s' eances de l'acad'emie des sciences. *Comptes Rendus Hebd. Seances Acad. Sci*, 21(25):536–538, 1847.

S. Çelik, G. Demirtas, and M. Isaksson. Corporate bond market trends, emerging risks and monetary policy, 2020.

H. Chen, P. De, Y. J. Hu, and B.-H. Hwang. Wisdom of crowds: The value of stock opinions transmitted through social media. *The Review of Financial Studies*, 27(5):1367–1403, 2014a.

J. Chen, G. Tang, J. Yao, and G. Zhou. Employee Sentiment and Stock Returns. *Available at SSRN 3612922*, 2020.

Z. Chen, A. A. Lookman, N. Schürhoff, and D. J. Seppi. Rating-based investment practices and bond market segmentation. *The Review of Asset Pricing Studies*, 4(2):162–205, 2014b.

K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

L. Choi. The Popular Asian Social Media Networks: Face-What? Insta-What?, 2020. Accessed: August 2020.

T. Chordia, A. Goyal, Y. Nozawa, A. Subrahmanyam, and Q. Tong. Are capital market anomalies common to equity and corporate bond markets? An empirical investigation. *Journal of Financial and Quantitative Analysis*, 52(4):1301–1342, 2017.

L. C. Chugh and J. W. Meador. The stock valuation process: the analysts' view. *Financial Analysts Journal*, 40(6):41–48, 1984.

L. Cohen. Loyalty-based portfolio choice. *The Review of Financial Studies*, 22(3):1213–1245, 2009.

M. Correia, S. Richardson, and İ. Tuna. Value investing in credit markets. *Review of Accounting Studies*, 17(3):572–609, 2012.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International conference on machine learning*, pages 647–655, 2014.

J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

N. Eccles and S. Viviers. The origins and meanings of names describing investment practices that integrate a consideration of ESG issues in the academic literature. *Journal of business ethics*, 104(3):389–402, 2011.

E. F. Fama. Efficient capital markets: A review of theory and empirical work. *The journal of Finance*, 25(2):383–417, 1970.

E. F. Fama and K. R. French. Common risk factors in the returns on stocks and bonds. *Journal of*, 1993.

A. Frazzini and L. H. Pedersen. Betting against beta. *Journal of Financial Economics*, 111 (1):1–25, 2014.

G. Friede, T. Busch, and A. Bassen. ESG and financial performance: aggregated evidence from more than 2000 empirical studies. *Journal of Sustainable Finance & Investment*, 5 (4):210–233, 2015.

Glassdoor. Glassdoor - About us, 2020. Accessed: July 2020.

I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

T. C. Green, R. Huang, Q. Wen, and D. Zhou. Crowdsourced employer reviews and stock returns. *Journal of Financial Economics*, 134(1):236–251, 2019.

J. B. Guerard. Additional evidence on the cost of being socially responsible in investing. *The Journal of Investing*, 6(4):31–36, 1997.

E. Haddi, X. Liu, and Y. Shi. The role of text pre-processing in sentiment analysis. *Procedia Computer Science*, 17:26–32, 2013.

Z. S. Harris. Methods in structural linguistics. 1951.

R. A. Haugen and A. J. Heins. On the evidence supporting the existence of risk premiums in the capital market. *Available at SSRN 1783797*, 1972.

K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

H. Henke, H. Kaufmann, P. Messow, and J. Fang-Klingler. Factor Investing in Credit. *The Journal of Index Investing*, 11(1):33–51, 2020.

J. Hirshleifer. Investment decision under uncertainty: Applications of the state-preference approach. *The Quarterly Journal of Economics*, 80(2):252–277, 1966.

S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

P. Houweling and J. Van Zundert. Factor investing in the corporate bond market. *Financial Analysts Journal*, 73(2):100–115, 2017.

J. Howard and S. Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, 2004.

J. Huang. The customer knows best: The investment value of consumer opinions. *Journal of Financial Economics*, 128(1):164–182, 2018.

A. Ilmanen. *Expected returns: An investor's guide to harvesting market rewards*. John Wiley & Sons, 2011.

S. Y. In, D. Rook, and A. Monk. Integrating Alternative Data (Also Known as ESG Data) in Investment Decision Making. *Global Economic Review*, 48(3):237–260, 2019.

R. Israel, D. Palhares, and S. A. Richardson. Common factors in corporate bond returns. *Forthcoming in the Journal of Investment Management*, 2017.

N. Jegadeesh and S. Titman. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of finance*, 48(1):65–91, 1993.

G. Jostova, S. Nikolova, A. Philipov, and C. W. Stahel. Momentum in corporate bond returns. *The Review of Financial Studies*, 26(7):1649–1693, 2013.

A. Karpathy. The Unreasonable Effectiveness of Recurrent Neural Networks. `http://karpathy.github.io/2015/05/21/rnn-effectiveness/`, 2015. Accessed: July 2020.

Y. Kim. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*, 2014.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

R. Lake. A Guide to the 11 Market Sectors, 2020. Accessed: July 2020.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

B. Liu. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167, 2012.

J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

C. A. Mallin, B. Saadouni, and R. J. Briston. The financial performance of ethical investment funds. *Journal of Business Finance and Accounting*, 22:483–483, 1995. doi: 10.1111/j.1468-5957.1995.tb00373.x.

C. Manning and H. Schutze. *Foundations of statistical natural language processing*. MIT press, 1999.

H. Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952. ISSN 00221082, 15406261. URL http://www.jstor.org/stable/2975974.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

S. Min, M. Seo, and H. Hajishirzi. Question answering through transfer learning from large fine-grained supervision data. *arXiv preprint arXiv:1702.02171*, 2017.

L. Mou, Z. Meng, R. Yan, G. Li, Y. Xu, L. Zhang, and Z. Jin. How transferable are neural networks in nlp applications? *arXiv preprint arXiv:1603.06111*, 2016.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. *arXiv preprint cs/0205070*, 2002.

R. Pascanu, T. Mikolov, and Y. Bengio. On the difficulty of training recurrent neural networks. In *International conference on machine learning*, pages 1310–1318, 2013.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014a.

J. Pennington, R. Socher, and C. D. Manning. GloVe: Global Vectors for Word Representation. https://nlp.stanford.edu/projects/glove/, 2014b. Accessed: July 2020.

H. Robbins and S. Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

S. A. Ross. The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, 13(3): 341–360, 1976.

S. A. Ross et al. *Return, risk and arbitrage*. Rodney L. White Center for Financial Research, The Wharton School . . . , 1973.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

R. Sennrich, B. Haddow, and A. Birch. Improving neural machine translation models with monolingual data. *arXiv preprint arXiv:1511.06709*, 2015.

G. Serafeim. Public Sentiment and the Price of Corporate Sustainability. *Financial Analysts Journal*, 76(2):26–46, 2020.

A. Severyn and A. Moschitti. Unitn: Training deep convolutional neural network for twitter sentiment classification. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*, pages 464–469, 2015.

J. Sheng. Asset pricing in the information age: Employee expectations and stock returns. *Available at SSRN 3321275*, 2019.

Sifma. The 2020 Outlook - Trends in US Capital Markets, 2020.

R. Sparkes. Ethical investment: whose ethics, which investment? *Business Ethics: A European Review*, 10(3):194–205, 2001.

T. Tieleman and G. Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

E. Van Duuren, A. Plantinga, and B. Scholtens. ESG integration and the investment management process: Fundamental investing reinvented. *Journal of Business Ethics*, 138(3): 525–533, 2016.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

Appendix

Appendix A - Descriptive Statistics - Ratings

1. Number of ratings/reviews across market sectors (for stock market indices)

S&P 500

Market sector	Mean	Std	10^{th}	Q1	Median	Q3	90^{th}
Basic Materials	139.5	218.69	38.1	48.0	64.0	100.5	256.2
Communications	897.8	1058.33	37.3	145.0	413.5	1275.3	2289.3
Consumer Cyclical	1391.4	3801.78	63.4	119.0	232.0	1139.0	2358.2
Consumer Defensive	945.08	2139.30	70.0	119.8	278.5	590.5	1558.5
Energy	215.96	285.08	26.0	50.0	92.0	241.0	644.6
Financial Services	655.43	938.28	60.9	160	328.5	610.5	1613.2
Healthcare	459.44	610.05	77.0	127	241.0	532.0	905.0
Industrials	493.25	653.78	61.0	117	268.0	646.0	1022.6
Real Estate	150.26	238.89	12.2	31.5	93.0	163.0	233.6
Technology	1198.45	2361.90	71.2	140.8	352.5	1014.8	3234.2
Utilities	68.72	37.15	26.4	34.0	71.0	90.0	110.0

FTSE 100

Market ector	Mean	Std	10^{th}	Q1	Median	Q3	90^{th}
Basic Materials	97.78	88.99	18.0	39.0	62.0	126.0	239.8
Communications	375.40	685.92	54.0	71.5	88.0	230.8	850.9
Consumer Cyclical	251.60	257.29	61.8	71.5	219.0	250.5	521.4
Consumer Defensive	726.36	1118.22	37.0	76.5	402.0	533.5	2042.0
Energy	722.00	830.14	252.4	428.5	722.0	1015.5	1191.6
Financial Services	633.41	1034.60	57.8	95.0	134.0	849.0	1958.4
Healthcare	566.00	538.44	112.1	187.3	471.5	850.3	1095.5
Industrials	188.30	153.49	16.5	41.0	194.0	301.3	338.1
Real Estate	15.00	4.24	12.6	13.5	15.0	16.5	17.4
Technology	41.50	2.12	40.3	40.8	41.5	42.3	42.7
Utilities	156.75	86.21	75.7	93.3	163.5	227.0	232.4

EURO STOXX 50

Market sector	Mean	Std	10^{th}	Q1	Median	Q3	90^{th}
Basic Materials	101.75	69.11	42.4	55.0	91.5	138.3	169.3
Communications	244.67	234.54	83.8	112.0	159.0	334.5	439.8
Consumer Cyclical	254.50	231.45	127.5	147.3	150.0	230.0	486.0
Consumer Defensive	529.00	427.76	120.6	237.0	567.0	639.0	951.0
Energy	165.50	116.67	99.5	124.3	165.5	206.8	231.5
Financial Services	315.78	309.61	53.0	73.0	225.0	498.0	640.6
Healthcare	400.25	300.88	104.7	221.3	458.0	637.0	649.6
Industrials	597.86	737.02	51.4	123.0	280.0	865.5	1592.2
Real Estate	66.00	N/A	66.0	66.0	66.0	66.0	66.0
Technology	2391.33	2115.38	633.6	1489.5	2916.0	3555.5	3939.2
Utilities	153.67	100.02	70.4	114.5	188.0	210.0	223.2

2. Ratings across market sectors (for stock market indices)

S&P 500

Market sector	Mean	Std	Q1	Median	Q3
Basic Materials	3.57	1.21	3.0	4.0	4.0
Communications	3.61	1.29	3.0	4.0	5.0
Consumer Cyclical	3.63	1.22	3.0	4.0	5.0
Consumer Defensive	3.34	1.27	3.0	3.0	4.0
Energy	3.58	1.18	3.0	4.0	4.0
Financial Services	3.58	1.22	3.0	4.0	5.0
Healthcare	3.44	1.31	3.0	4.0	5.0
Industrials	3.55	1.27	3.0	4.0	5.0
Real Estate	3.68	1.36	3.0	4.0	5.0
Technology	3.73	1.16	3.0	4.0	5.0
Utilities	3.58	1.32	3.0	4.0	5.0

FTSE 100

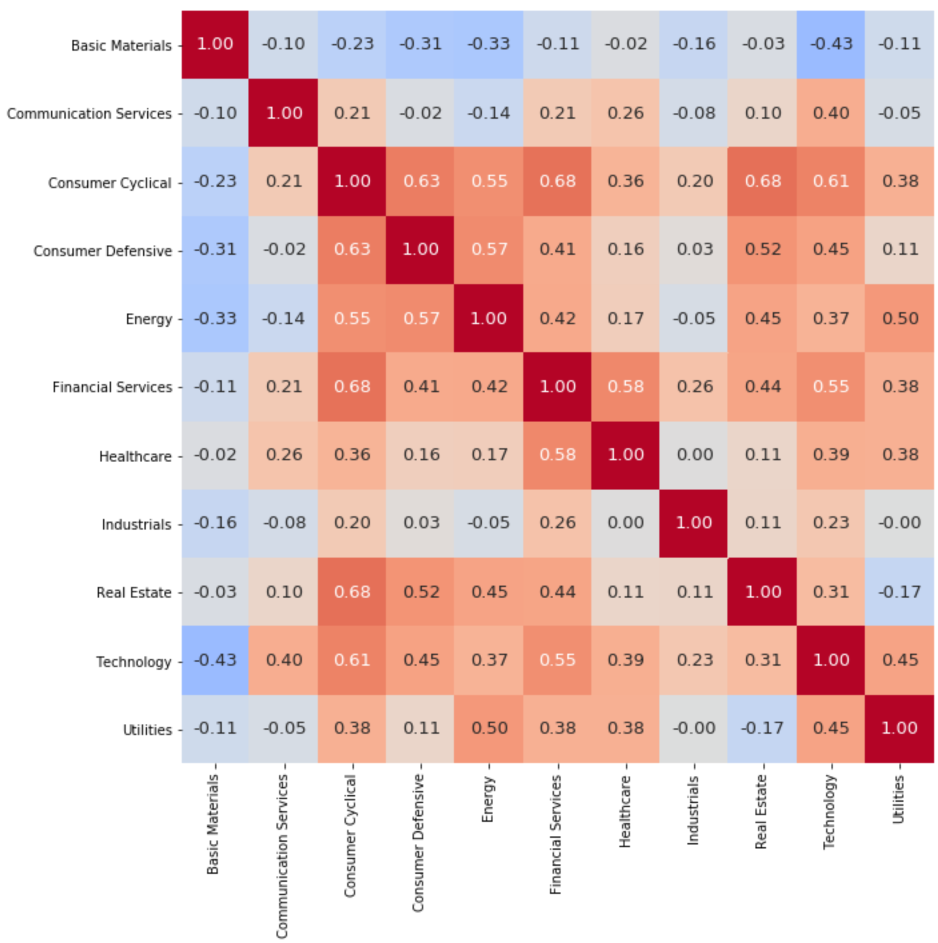
Market sector	Mean	Std	Q1	Median	Q3
Basic Materials	3.61	1.22	3.0	4.0	5.0
Communications	3.58	1.21	3.0	4.0	5.0
Consumer Cyclical	3.44	1.34	3.0	4.0	5.0
Consumer Defensive	3.55	1.18	3.0	4.0	4.0
Energy	3.93	1.12	3.0	4.0	5.0
Financial Services	3.65	1.12	3.0	4.0	4.0
Healthcare	3.76	1.16	3.0	4.0	5.0
Industrials	3.63	1.37	3.0	4.0	5.0
Real Estate	3.93	1.23	3.0	4.0	5.0
Technology	3.58	1.23	3.0	4.0	4.0
Utilities	3.48	1.31	3.0	4.0	5.0

EURO STOXX 50

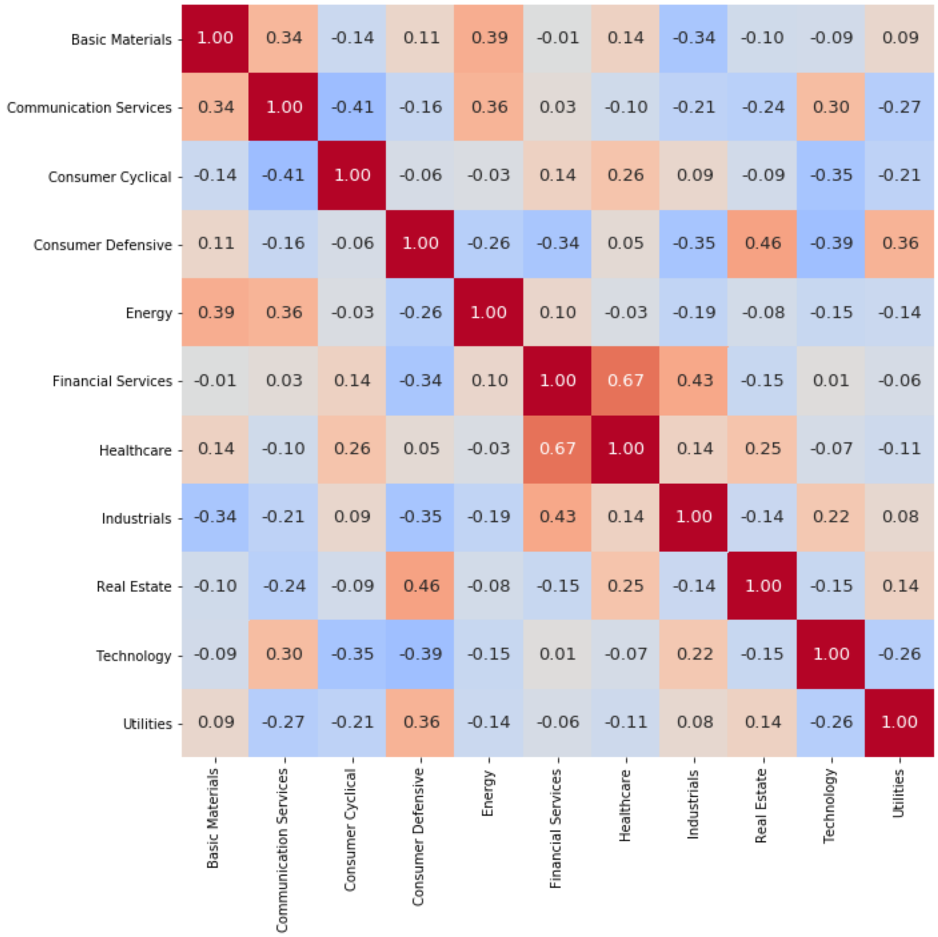
Market sector	Mean	Std	Q1	Median	Q3
Basic Materials	3.42	1.34	3.0	4.0	5.0
Communications	3.81	1.06	3.0	4.0	5.0
Consumer Cyclical	3.80	1.13	3.0	4.0	5.0
Consumer Defensive	3.71	1.22	3.0	4.0	5.0
Energy	3.66	1.21	3.0	4.0	5.0
Financial Services	3.51	1.20	3.0	4.0	4.0
Healthcare	3.69	1.16	3.0	4.0	5.0
Industrials	3.90	1.10	3.0	4.0	5.0
Real Estate	3.23	1.45	2.0	4.0	4.0
Technology	4.29	0.98	4.0	5.0	5.0
Utilities	3.49	1.25	3.0	4.0	4.0

3. Correlation of monthly changes in ratings across market sectors (for stock market indices)

S&P 500

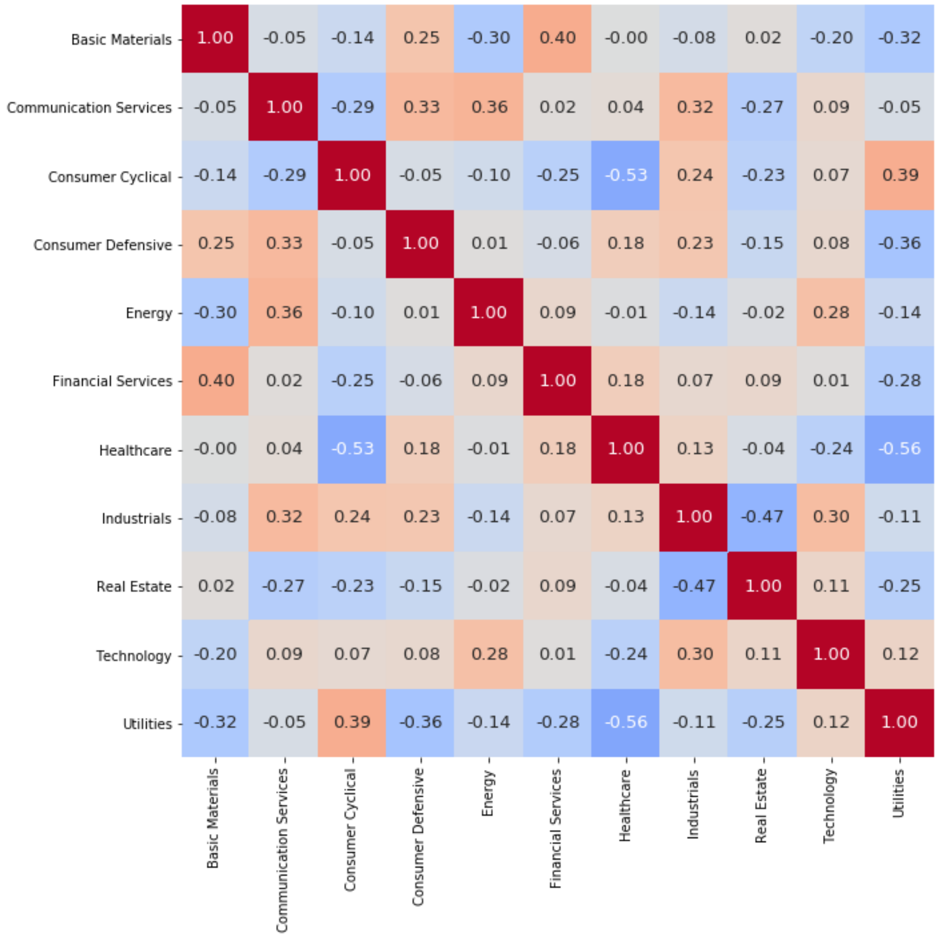


FTSE 100



Basic Materials
Communication Services
Consumer Cyclical
Consumer Defensive
Energy
Financial Services
Healthcare
Industrials
Real Estate
Technology
Utilities
1.00 0.34 -0.14 0.11 0.39 -0.01 0.14 -0.34 -0.10 -0.09 0.09
0.34 1.00 -0.41 -0.16 0.36 0.03 -0.10 -0.21 -0.24 0.30 -0.27
-0.14 -0.41 1.00 -0.06 -0.03 0.14 0.26 0.09 -0.09 -0.35 -0.21
0.11 -0.16 -0.06 1.00 -0.26 -0.34 0.05 -0.35 0.46 -0.39 0.36
0.39 0.36 -0.03 -0.26 1.00 0.10 -0.03 -0.19 -0.08 -0.15 -0.14
-0.01 0.03 0.14 -0.34 0.10 1.00 0.67 0.43 -0.15 0.01 -0.06
0.14 -0.10 0.26 0.05 -0.03 0.67 1.00 0.14 0.25 -0.07 -0.11
-0.34 -0.21 0.09 -0.35 -0.19 0.43 0.14 1.00 -0.14 0.22 0.08
-0.10 -0.24 -0.09 0.46 -0.08 -0.15 0.25 -0.14 1.00 -0.15 0.14
-0.09 0.30 -0.35 -0.39 -0.15 0.01 -0.07 0.22 -0.15 1.00 -0.26
0.09 -0.27 -0.21 0.36 -0.14 -0.06 -0.11 0.08 0.14 -0.26 1.00

EURO STOXX 50



Appendix B - Descriptive Statistics - Reviews

1. Length of reviews (in words) across stock market sectors

Market Sector	Mean	Std	10^{th}	Q1	Median	Q3	90^{th}
Masic Materials	34.60	50.25	11	13	19	35	70
Communications	35.13	55.93	11	13	18	34	72
Consumer Cyclical	32.32	48.93	11	13	18	32	64
Consumer Defensive	30.93	46.81	11	13	17	30	60
Energy	30.27	47.00	11	13	17	29	58
Financial Services	31.43	47.69	11	13	17	30	61
Healthcare	34.83	50.32	11	13	18	35	73
Industrials	35.88	56.41	11	13	19	35	74
Real Estate	46.84	66.06	11	14	23	52	104
Technology	31.99	50.60	11	13	17	31	63
Utilities	38.21	59.17	11	13	20	40	79

2. Length of reviews (in words) across stock market indices

Stock market	Mean	Std	10^{th}	Q1	Median	Q3	90^{th}
S&P 500	33.37	51.74	11	13	18	32	67
FTSE 100	29.70	42.88	11	12	17	29	58
EURO STOXX 50	29.66	40.92	11	13	17	30	58

3. Length of reviews across employee categories

Employee	Total reviews	Mean	Std	10^{th}	Q1	Median	Q3	90^{th}
Full-time	316,117	33.79	52.14	11	13	18	33	68
Part-time	58,240	28.76	41.83	11	12	17	28	54
Not specified	18,043	29.48	46.00	11	12	16	28	58
Current	220,135	31.91	46.63	11	13	18	31	64
Former	154,224	34.58	55.98	11	13	18	33	69
Not specified	18,041	29.48	46.00	11	12	16	28	58
Total	392,408	32.85	50.51	11	13	18	32	66

4. Wordcloud


care
change
benefit
make
think
department
work life
flexibility
Great company
good people
training
Good company
Great place
product
place
lot
still
good
benefit
really
everything
even
know
take
management
Great
customer
balance
growth
feel
home
opportunity
give
position
Good work
business
salary
will
lack
people
project
year
working
difficult
better
employee
store
best
way
well
pay
go
say
HR
job
great people
easy
time
nothing
need
bad
Great work
many
culture
culture
opportunitie
work
much
hour
promotion
everyone
come
staff
co worker
experience
life
grow
work environment
leadership
location
manager
support
work
one
help
team
thing
sometime
move
due
little
day
hard
worked
want
company
con
amp
keep
role
Good pay
long hour
learn
alway

Appendix C - Table scheme

```python
from django.db import models
# set max_lengths for texts of different lengths
short_text, mid_text, long_text = 25, 100, 500

class Company(models.Model):
    CompanyID = models.IntegerField(null=True)
    Company = models.CharField(max_length=mid_text, blank=True, unique=True)
    Symbol = models.CharField(max_length=short_text, blank=True)
    ListedOn = models.CharField(max_length=short_text, blank=True)
    Sector = models.CharField(max_length=mid_text, blank=True)
    Industry = models.CharField(max_length=mid_text, blank=True)
    Country = models.CharField(max_length=mid_text, blank=True)
    NoEmployees = models.IntegerField(null=True)
    Revenue = models.FloatField(null=True)
    Timestamp = models.TimeField(null=True)

    def __str__(self):
        return self.Company


class Review(models.Model):
    Company = models.ForeignKey(Company, on_delete=models.CASCADE)
    ReviewTitle = models.CharField(max_length=long_text, blank=True)
    Year = models.IntegerField(null=True)
    Month = models.IntegerField(null=True)
    Day = models.IntegerField(null=True)
    Rating = models.FloatField(null=True)
    JobTitle = models.CharField(max_length=long_text, blank=True)
    EmployeeRelationship = models.CharField(max_length=mid_text, blank=True)
    Location = models.CharField(max_length=mid_text, blank=True)
    Recommendation = models.CharField(max_length=short_text, blank=True)
    Outlook = models.CharField(max_length=short_text, blank=True)
    OpinionOfCEO = models.CharField(max_length=short_text, blank=True)
    Contract = models.CharField(max_length=mid_text, blank=True)
    ContractPeriod = models.CharField(max_length=mid_text, blank=True)
    Pros = models.CharField(max_length=long_text, blank=True)
    Cons = models.CharField(max_length=long_text, blank=True)
    AdviceToManagement = models.CharField(max_length=long_text, blank=True)
    Timestamp = models.TimeField(null=True)

    def __str__(self):
        return self.ReviewTitle
```