# Abstract

Much has been made of the consequences of the social data revolution of the last decade. The shifting of human communication patterns towards online data sharing has created a reservoir of public data suitable for analysis and pattern recognition. This dissertation aims to explore the link between social network profile data and career paths over time.

# Acknowledgments

I would like to acknowledge my family, friends and supervisor Daniel Winterstein, for their help and encouragement throughout this project.

# Declaration

I declare that this thesis was composed by myself and any external sources herein will be appropriately declared and cited.

# Introduction

The question of predicting an individual’s life outcomes remains one of the great-unresolved mysteries of our time. Researchers and experts from fields as diverse as economics to zoology have all pursued the seemingly elusive holy grail of predicting the future **(citation 1)**. A common impediment encountered in each of these approaches is the inherent difficulty in modeling individual circumstances. Typically a person’s life is not static and is subject to sudden changes in environment or circumstance. The consequence is a fluid dynamic where each individual’s path is unique and nothing is set in stone. The recent resurgence of artificial intelligence techniques and methods, coupled with the advent of social networking platforms such as LinkedIn and Twitter has started a conversation on individual user profile data collection and statistical pattern matching. Most notably the emergence of social networks has created an ever increasing, reservoir of user profile data previously not available(Weigend)

The purpose of this M.Sc. project is to abstract away some of the intricacies associated with forecasting and build an artificial intelligence driven system which aims to predict a user’s future using pattern matching on previously collected sample data. More concretely the system will be aimed groups such as prospective university undergraduate students who would like to know what path may lie ahead of them. The professional social networking platform LinkedIn will act as the source of user profile data. The system will apply artificial intelligence methods to the user profile data in order to extract trends and insight into users career paths over a time period. These trends will be then used to infer pathways for users with similar criteria.

The motivation to undertake this project is two fold. Firstly, whilst in many ways while this project centers on AI methods and techniques, it is equally about taking advantage of the current burgeoning social network revolution. The emergence of social networking services such as Twitter, Facebook and LinkedIn has provided us with a plethora of information and potential insight into peoples day-to-day interactions. LinkedIn in particular provides a reservoir of pertinent, semi-structured data allowing for a smoother approach to analysis. To my best knowledge there is no open source application, which uses social network data and bespoke user parameters to try and predict an individual’s future career path.

Further motivation emanates from the relatively opened end nature of the task at hand. Whilst the core focus of the project is to build an artificial intelligence driven analysis system, the results could be useful to many related disciplines such as economics, which focus on labor movements and flexibility within the labor market or government policy where insight into career trends could be useful for planning.

Ultimately the end goal is to present a robust system grounded on AI principles and statistical methods, which potentially offers users with a glimpse into their potential futures.

## Thesis Outline

The remainder of this thesis document will be organized as follows.  
 Chapter 2 introduces more concretely the projects aims and objectives.

* Chapter 3 will focus on the background research and provide some insight into existing projects and techniques.
* Chapter 4 will provide some insight into the data set and relevant methods used in the project
* Chapter 5 will provide some insight on the system design and some of the choices made when deciding the implementation strategy.
* Chapter 6 presents the results of the system and user feedback.
* Chapter 7 evaluates the system and discusses potential future works and extensions. In addition, all code and user feedback forms will be included in a separate appendix section **(check)** separate from the main thesis body.

# 2 Aim and Objectives

## 2.1 High-level Aim

To develop a bespoke system using a user centered design approach, which enables prospective students to gain a glimpse into what their future careers might look like.

## 2.2 Objectives

1. Identify appropriate statistical methods and AI techniques to be included in system.

2. Examine existing literature to gain understanding of the current state of the art and discuss relevant approaches for project.

3. To identify the functional and data requirements for system based on results from requirements engineering

4. To iteratively build application prototype guided by user input.

5. Conduct final evaluation of effectiveness of tool and present suggestions for further work.

## 2.3 Objectives Outline

**Objective 1**: Given the plethora of statistical and machine learning techniques that exist, it is crucial that an appropriate subset of methods is selected. This choice will be based on the most popular sorting algorithms taught to students in introductory algorithms classes.

**Objective 2:** To research into AI methods such as classification and pattern matching to determine how these can be used in tandem with LinkedIn data to develop a reliable prediction system.

**Objective 3:** To implement a prototype which satisfies the aforementioned requirements as well as any further requirements, which are gathered from the user group. Functional requirements will detail the particular actions the tool will support, data requirements will focus on how the data will be handled (e.g. I/O interactions and cross compatibility issues).

**Objective 4**: Build a system, which satisfies the predefined requirements. The system will ideally adhere to the requests and constraints specified by a user group.

**Objective 5**: Conduct final evaluation to gauge how well application addresses the initial requirements.

# 3 Background Research

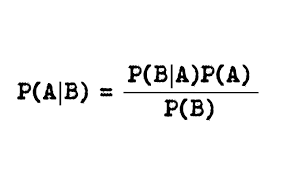
For the purpose of this project the research section will be subdivided into 5 categories with each category focusing on a specific element of the overall system.

## 3.1 Algorithms

The algorithms chosen will underpin the overall functionality of the system. Given the myriad of techniques available at our disposal it is essential to focus on algorithms that generate the best performance and are most appropriate to the problem space.

### 3.1.1 Naïve Bayes

The problem of predicting a person future can be viewed through a Bayesian lens. We can assume that we would like to know the probability of someone following a certain path A given that we have observed a certain number of paths B. Naturally it follows that this problem can be expressed and solved through the Bayes Theorem formula.



In machine learning there exists a family of Bayesian of probabilistic classifiers ranging from Bayesian Belief Networks to Averaged one-dependence estimators (AODE). We can formally refer to these methods as Bayesian networks. A Bayesian network encodes the joint probability distribution of a set of n variables, {X1, . . . , Xn}, as a directed acyclic graph and a set of conditional probability distributions (CPDs). Each node corresponds to a variable, and the CPD associated with it gives the probability of each state of the variable given every possible combination of states of its parents. The set of parents of Xi , denoted πi , is the set of nodes with an arc to Xi in the graph. The structure of the network encodes the assertion that each node is conditionally independent of its non-descendants given its parents. The joint distribution of the variables is thus given by Naive Bayes Models for Probability Estimation P(X1, . . . , Xn) = Qn i=1 P(Xi |πi)**(citation)**

Naïve Bayes is a particular instance of a Bayesian network, which has an independence assumption. Naive Bayes models are so named for their “naive” assumption that all variables Xi are mutually independent given a “special” variable C. The joint distribution is then given compactly by P(C, X1, . . . , Xn) = P(C) Qn i=1 P(Xi |C). The univariate conditional distributions P(Xi |C) can take any form (e.g., multinomial for discrete variables, Gaussian for continuous ones). When the variable C is observed in the training data, naive Bayes can be used for classification, by assigning test example (X1, . . . , Xn) to the class C with highest P(C|X1, . . . , Xn) (Domingos & Pazzani, 1997).

For the purposes of the system we are building we can imagine using a naïve bayes classifier to classify the probability of a LinkedIn profile matching a particular career path given observations of previous careers paths.

### 3.1.2 Support Vector Machines

Support Vector Machines (SVMs) are an idea which was pioneered by Cortes and Vapnik(**1995 cite)**. the main intuition behind them is that given a two-class training set they project its datapoints in a higher dimensional space and attempt to specify a maximum-margin separating hyperplane between the datapoints of two classes. This hyperplane is optimal in the sense that it generalizes well to unseen data (**CITE VLACHOS)**. A more detailed description of SVMs follows, based on Burges (1998) and Vapnik (1998). For the purposes of this project we will likely not use SVM’s due to their sensitivity to parameter settings. Inaddition SVMS will not provide use with a probabilistic distribution which could be used as a feature in our system.

### 3.1.3 Particle Filters

Sequential Monte Carlo methods are a set of methods which allow us to solve the “filtering problem” i.e estimate a reasonable value for a system given only some potential noisy observations of this systems. Particle filtering methodology uses a genetic type mutation-selection sampling approach, with a set of particles (also called individuals, or samples) to represent the [posterior distribution](https://en.wikipedia.org/wiki/Posterior_distribution) of some [stochastic process](https://en.wikipedia.org/wiki/Stochastic_process) given some noisy and/or partial observations. The state-space model can be nonlinear and the initial state and noise distributions can take any form required. Particle filter techniques provide a well-established methodology[[1]](https://en.wikipedia.org/wiki/Particle_filter#cite_note-dm962-1)[[14]](https://en.wikipedia.org/wiki/Particle_filter#cite_note-:22-14)[[15]](https://en.wikipedia.org/wiki/Particle_filter#cite_note-:1-15) for generating samples from the required distribution without requiring assumptions about the state-space model or the state distributions. However, these methods do not perform well when applied to very high-dimensional systems. **(reword AND CITE)**

The relative complexity coupled with the time constraints of the project mean that despite their considerable strengths particle filters are unlikely to be a suitable choice for this project.

### 3.1.4 Decision Trees

We can view a decision tree as a method for approximating a discrete valued target function, where the learned function represented as a decision tree **(Storkey DecisionTree informatics)**. Decision trees are a type of classification algorithm where each node tests for a particular attribute, while the branches provide the attribute values and the leaves output the classification. Decision trees are particularly effective when our data is potentially noisy and are relatively easy to implement. However like support vector machines, decision tress by default do not offer a probabilistic representation and are binary in their nature. Moreover they suffer from over fitting issues, which could potentially be complicate, any implementation. It is likely we will only consider decision tress should the timing permit.

### 3.1.5 Other AI algorithms

The algorithms we have described above barely begin to scratch the surface of an algorithmic approach to dealing with our problems. Other notable approaches include K-nearest neighbor, Boosting and Neural Networks.

## 3.2 LinkedIn Data

The rapid emergence of professional social networks such as LinkedIn has allowed us to obtain a rich information repository of semi-structured data from which we can carry out various experiments and analysis. We have seen several examples of LinkedIn user data being for analysis (Sharma 2010), Efstathiades 2014), (Sathick&Venkat 2015). Perhaps the closest example of career path analysis carried out on LinkedIn is by (Case et al 2013). The study concentrated on studying the entry-level jobs held by alumni, their subsequent career progress, and the long-term out- comes of the alumni study programs. In order to carry this out the users had access to a local university IS databases and a LinkedIn group with a 175 profiles. The results suggested that most graduates stay within their profession in the first 5 years and gradually move out to managerial professions. It is also worth stating that LinkedIn offers a feature called Linked In Alumni, which attempts to allow you to network with alumni from a similar background and guide you with your career path.

The unique selling point of our system will be its ability to predict via classification which sector a user will enter. To our knowledge at the moment there is no system, which offers this facility.

## 3.3 Implementation Languages and Libraries

The time constraint coupled with the relatively large size and scope of the project, mean that the implementation language will be a key decision in determining the overall performance of the system. Given the myriad number of programming languages on the market; each with their own strengths and drawbacks, it is critical to form a set of criteria in order to usefully differentiate between the respective languages. For the purpose of this project we propose using the following criteria.

1.Ease of use.

2. Functionality and extendibility.

3. Machine learning support network and documentation.

4 Performance.

### 3.3.1 C++

C++ is one of the most popular programming languages and is widely used in the software industry. It is particularly renowned for its speed and extensive documentation. In addition C++ is relatively “low level”, allowing developers to carry out memory management and utilize GPU resources. However C++ in spite of these significant advantages C++suffers from a number of major drawbacks. Whilst its low level nature undoubtedly empowers software developers it is also

adds an extra layer of complication making significantly harder to debug and reformat. Furthermore the restricted portability makes it less desirable when compared to a language such as Java. C++ implementation would likely mean that a lot of the implementation stage would be spent debugging, which is not the purpose of this task.

### 3.3.2 Python

Python is a widely used general-purpose, high-level programming language (tecosystems). Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. Python also has a strong natural language processing support network e.g. NLTK making it ideal for language processing and scripting. However, there two key drawbacks to a Python implementation. Firstly due to the fact that Python is an interpreted language and thus tends to suffers from slowness, which could be detrimental to the application. Secondly Python is not as well documented as older languages such as Java and C++.

### 3.3.3 C#

C# is a modern object-oriented programming language that has been strongly influenced by the Java

and C++ programming languages (Naugler). C# syntax is simpler than C++ allowing for easier

debugging. In addition it is optimized for the .Net Framework and is easily extendible. However, C# was designed for Windows environment and thus suffers from a lack of compatibility with other platforms such as UNIX. Furthermore there appears to not to be significant machine learning

support for C#.

### 3.3.5 Java

Java is a general-purpose computer programming language that is concurrent and object oriented in its nature (Gosling). In many ways Java mirrors C++ however unlike C++ Java provides several abstractions and safety mechanisms such as garbage collection and null pointer exception handling making the implementation more straightforward. Java provides Swing, which is a GUI development tool kit with supports graphical user interface design. In addition Java has a huge support network with over 6 million developers using it on daily basis. All in all a Java implementation seems like the most logical choice.

### 3.3.6 WEKA

Is a Java machine-learning suite, which offers visualization tools for algorithms and data analysis. Weka is particularly ideal, as it will offer several classification and clustering algorithms as well as Java Database Connectivity to interface with MySQL.

### 3.3.7 Rapid Miner

Is a similar machine-learning environment, which provides analytical solutions through template-based frameworks thus, reducing the computation time. Should we require more computationally arduous tasks we will likely turn to rapid miner as an alternative to WEKA.

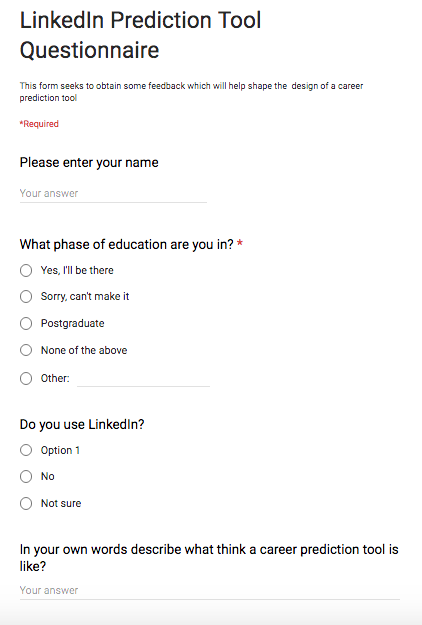
### 3.3.7 JavaML

# 4 System Design

In this section we will elaborate on our design approach and how we will approach constructing the system. The section will be divided into 4 distinct parts, each focusing on a particular aspect of the design process.

## 4.1 User Input

Since we will seek to incorporate aspects of a user centred design approach, we will need requirements engineering to formulate concrete functional requirements, which will shape the design of the application. In order to do this the following questionnaire was prepared and distributed to potential users who were interested in influencing the design of the tool. Given the time constraint it is unlikely that we will be able to implement most of the feature requests, however this likely to still prove a worthwhile exercise as it allows us to view the application from a non engineering perspective.



## 4.2 Functional Requirements

A functional requirement defines a function of a software system or its component. A function is described as a set of inputs, the behaviour, and outputs (Karthika). For the purposes of this project we will make a distinction between core functional requirements (minimal viable product) and ideal functional requirements. These requirements will form the basis of our overall evaluation.

Based on our initial assumptions and feedback obtained for the survey we will define our requirements as follows:

**Core**

**R1:** System is able to be given a set of HTML files and scrape the appropriate information for classification from LinkedIn profiles.

**R2:** System should be able to perform basic classification based on existing profiles (Naïve Bayes classification)

**R3:** The system should be able to perform some classification based on certain intervals 1,2,5,10 years.

**Bonus**

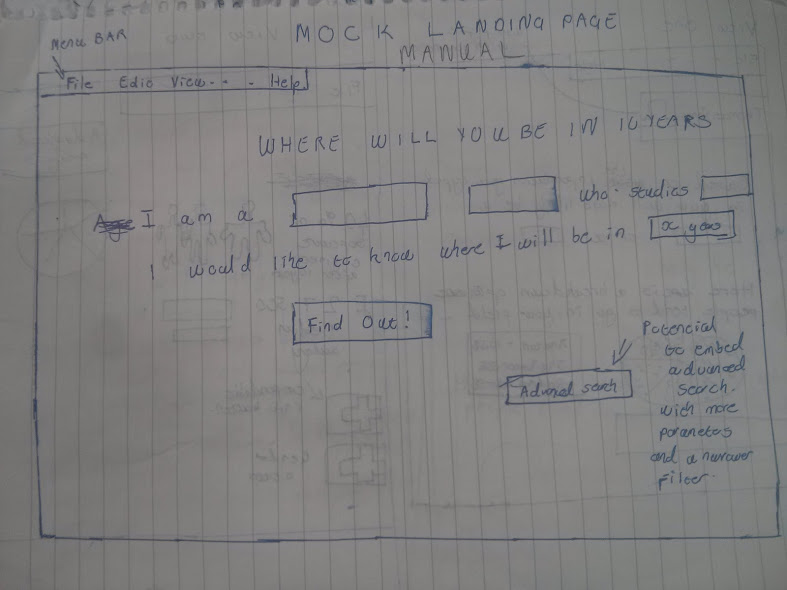
**R4:** System should have an interface, which users can interact with

**R5:** System should be able to compare various career paths based on different parameters

**R6:** Information about salaries and projected incomes will be included as part of the tool.

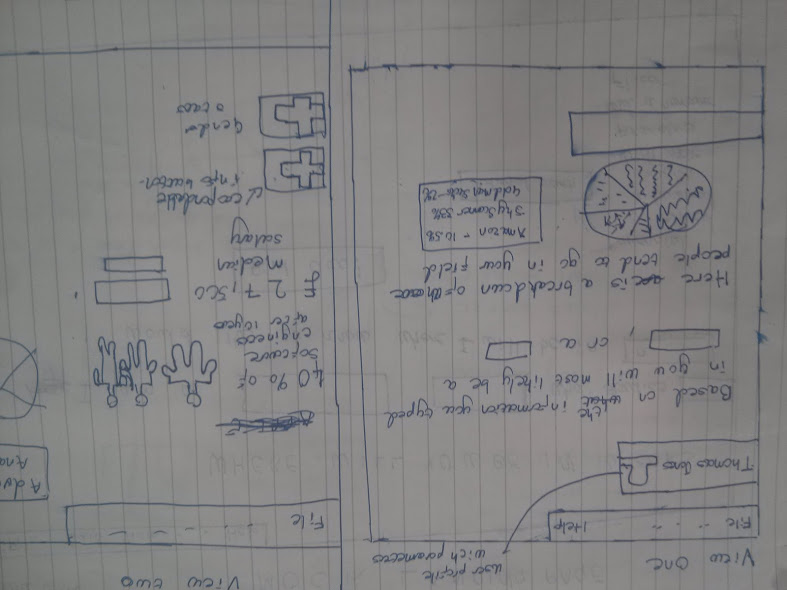
## Prototypes /Sketches

With the appropriate user functional requirements and a high level model of the application prototyping can now take place. A prototype can be defined as limited representation of a design that users or other key stakeholders can interact with in order to evaluate and provide feedback on the design (**Balaam)**. This is a crucial aspect of user centred design as it allows for exploration with users to create the best possible application. For the purposes of this application all prototypes will be “low fidelity” prototypes, which will be based on sketching, and paper prototyping. Below are screen grab ideas of low fidelity sketches. Each of the prototypes was evaluated based on how closely the prototype aligned with the functional requirements.

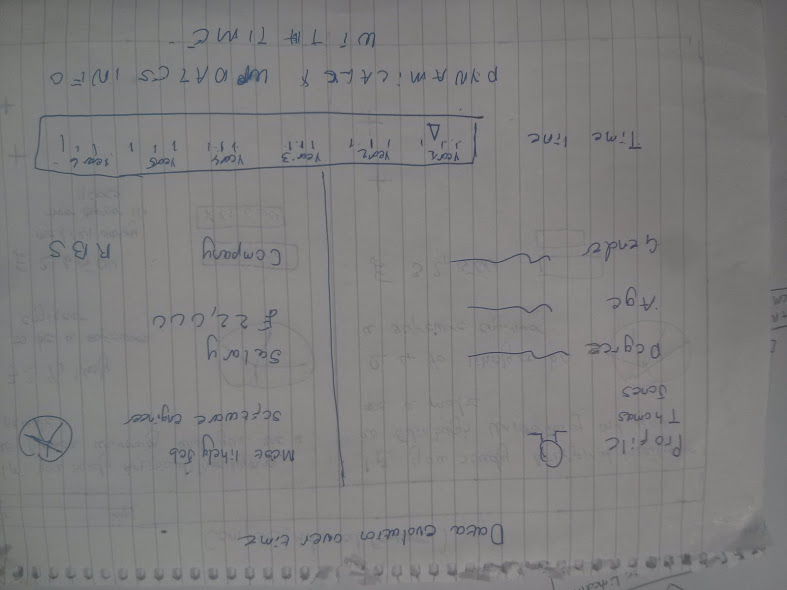


The above picture shows an example of the landing page a user might see when using the system. This assumes a user does not have a LinkedIn profile.

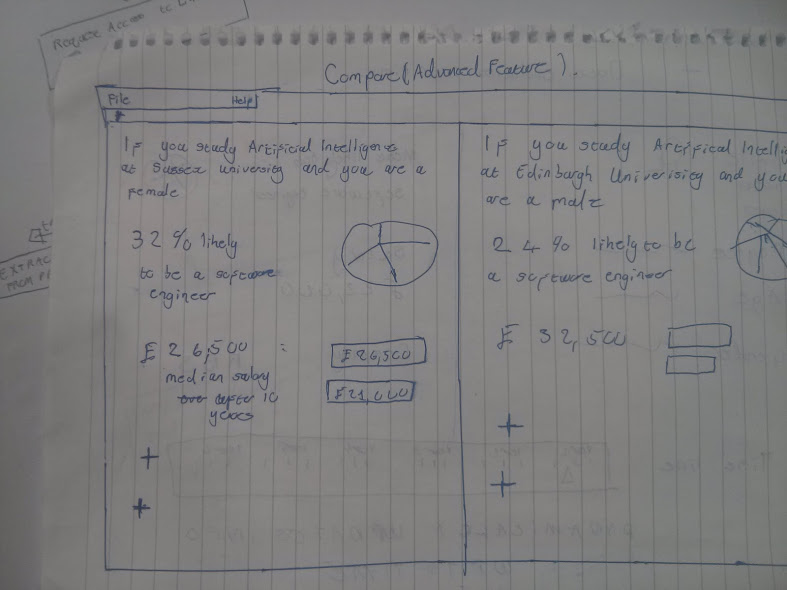
The sketch below shows the view a user may have when their profile has been classified.



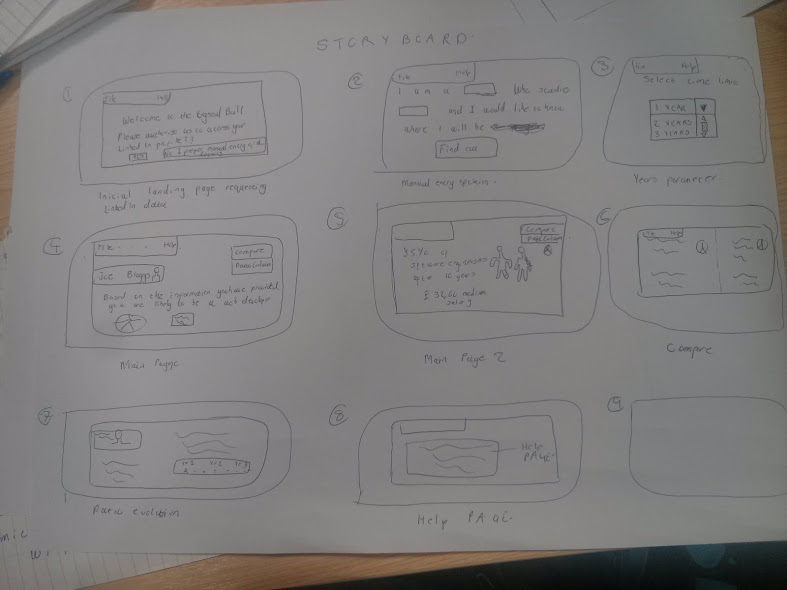
This sketch is below is an indication of the “time series” feature where the user is able to view the evolution of their profile data over time.



This final sketch below is a preview of what the comparison feature may look like. Here users can compare two separate career paths.



## 4.4 Storyboard



# Bibliography

Wolfers, J., & Zitzewitz, E. (2004). Prediction Markets. *Journal of Economic Perspectives*, *18*(2), 107–126. doi:10.1257/0895330041371321

Weigend, Andreas. "The Social Data Revolution(S)". *Harvard Business Review*. N.p., 2009. Web. 11 Aug. 2016.