Project Report - 2II36

Name	Student Number	Email address
Alexander Nieuwenhuijse	0744933	a.p.h.nieuwenhuijse@student.tue.nl
Valcho Dimitrov	0826904	v.dimitrov@student.tue.nl
Evans Owusu	0825824	e.b.owusu@student.tue.nl
Youcef Mammar	0873972	y.mammar@student.tue.nl

Part I: Used approaches

Approach 1: from OkCupid to Twitter

The concept

OkCupid.com is a dating website that contains a lot of personal information about its users and most of it is publically available. Our first approach consisted in matching the usernames used on this dating website with the twitter with the assumption that users tend to keep the same username across the social networks they use. There were quite a few other dating websites we could choose from, and some could provide more direct link with twitter profiles such as attwaction.com (a dating website based on twitter profiles, though apparently not very popular), but we were not satisfied with the amount of profiles we were able to collect, nor with the quality of the data provided by the website.

The implementation

Step 1: Collect the profiles on the OkCupid

By default profiles are public and registration for this website requires their users to provide their age, gender and location. The algorithm used to retrieve these profiles is pretty straightforward. First we run google search requests to collect profiles from users who are from the United Kingdom. Then we run a crawler to collect the user information from the profile. Last we use the usernames of the collected profiles and search Twitter for account with the same username. If there is one, we use the Twitter API to collect the tweets.

Before the google search, we collected a list of all the towns and cities in the United Kingdom by parsing wikipedia. For this we created a script to gather the name of the city with the corresponding county. The list is used later for labeling the user location (`city > county >`

LocationID). For the google search we had to use the Google API. It helps us to fetch the information which is returned after the request in an useful way. We create a search request in the following way: we get a city from the list as specified above, then create a request which includes the website, the gender ((**M**)ale or (**F**)emale) and the city (the format of the request can be seen in the code of the scripts). We collect the URLs and store the user profiles in a database for easy access at a later stage.

Step 2: Try to match their usernames to Twitter

Next step is to process the list of usernames and URLs. We check if for each username in the list, there exists a Twitter account. We just check if the **"https://twitter.com/*<username>*"** returns an account. If it does we store the URL of the twitter account, if not we discard the username from the list. From the remaining list which now contains OkCupid and Twitter account, we run a script to visit the OkCupid account URL and parse the dom-tree for information about gender, location, age and education. We run a script that uses the Twitter API to fetch tweets from the twitter accounts and adds the retrieved tweets to the OkCupid profile.

Step 3: Build a database of the twitter profiles collected with their demographic information

Gender, **Age** and **Location** are stored in the title tag of the OkCupid website, and are visible in the Google results, making them really easy to collect. For the **Education** though, we had to write an extra script that parses the DOM of the OkCupid profile page to retrieve this information. More specifically, the website allows their users to enter their education using two dropdown menus, each with the following choice:

- 1st dropdown
  - Graduated from
  - Working On
  - Dropped out
  - <Empty> (this field can be left empty)
- 2nd dropdown
  - two-year college
  - college/university
  - masters program
  - law school
  - med school
  - Ph.D program
  - High School
  - Space Camp

So the combination could look like this:

- graduated from law school
- dropped out of High school
- Ph.D program
- … (their number is limited)

Noticing this restriction made it very easy to get the Education of our users, with a simple rule based classifier. If no education was specified in the profile we presume that they at least went to high school, and label the profile as such.

Step 4: Analyse the sentiment of the tweets

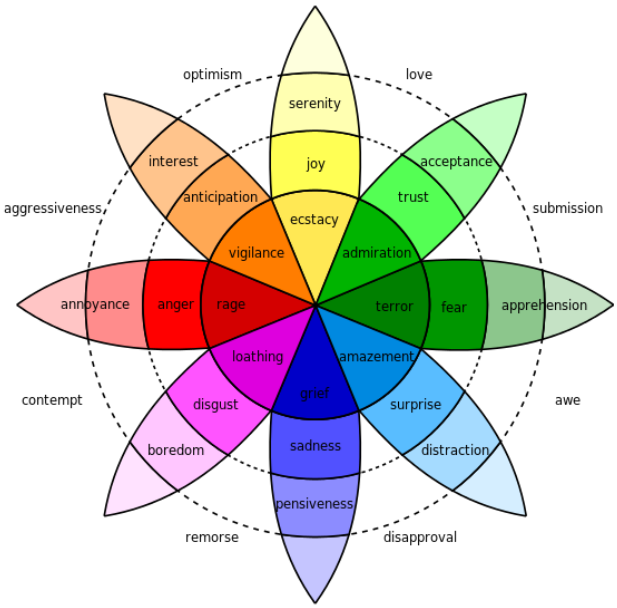
Once the tweets were fetched with the twitter api, we used the API provided by *lymbix.com* to measure the polarity and the emotion. It was really hard to find an API that returns emotions while being sufficiently “generous”. In fact even Lymbix was not a perfect choice because their API renders emotions following theses features:

- affection/friendliness
- enjoyment/elation
- contentment/gratitude
- amusement/excitement
- sadness/grief
- anger/loathing
- fear/uneasiness
- fear/uneasiness

Here is an example of output answered by the API.

```
{
  "article":"He was happy and surprised instead of being an angry guy.
             Although he wasn't too happy about it he said yes anyways!
             What do you think of this decision?",
  "ignored_terms":["He"],
  "affection_friendliness":2.6,
  "enjoyment_elation":4.55,
  "amusement_excitement":3.58,
  "contentment_gratitude":3.78,
  "sadness_grief":-0.24,
  "anger_loathing":-2.46,
  "fear_uneasiness":-1.28,
  "humiliation_shame":-0.12,
  "dominant_emotion":"enjoyment_elation",
  "article_sentiment": {
                          "sentiment":"Positive",
                          "score":7.8},
  "coverage":53,
  "intense_sentence": {
                             "sentence":"was happy and surprised instead of being an angry
guy.",
                         "dominant_emotion":"enjoyment_elation",
                         "intensity":0.15},
  "reference_id":1243122,
  "clarity":53.91
}
```


Therefore, we had to map theses results to the expected emotions, we tried to use the the following Plutchik Model of Emotions



We found this tree structure[1] very helpful in that regard. We end up with the following mapping:

```
"neutral" : [NEUTRAL],
"joy" : [ENJOYMENT_ELATION],
"sadness" : [SADNESS_GRIEF],
"anger" : [ANGER_LOATHING],
"fear" : [FEAR_UNEASINESS],
"surprise" : [AMUSEMENT_EXCITEMENT],
"trust" : [CONTENTMENT_GRATITUDE, AFFECTION_FRIENDLINES],
"disgust" : [ANGER_LOATHING, HUMILIATION_SHAME],
"anticipation" : [ENJOYMENT_ELATION, FEAR_UNEASINESS, AMUSEMENT_EXCITEMENT]
```

Neutral, joy, sadness, anger, fear and surprise were the easiest to map. The task was more delicate for trust, disgust and anticipation.
The wikipedia page for anticipation defines it[2] as a mix of happiness, excitement and fear (anxiety) . Disgust[3] is closer to anger and sadness as we can see it on the wheel and trust is a mix of love and respect, which we found was very close to affection and gratitude.

[1] http://en.wikipedia.org/wiki/List_of_emotions#Parrott.27s_emotions_by_groups
[2] http://en.wikipedia.org/wiki/Anticipation_(emotion)
[3] http://en.wikipedia.org/wiki/Disgust

This is obviously a lot of assumptions. Redefining such basic emotions with more elaborated emotions can't be easy. Unfortunately, this was our best solution.

Approach 2: from CoderWall to Twitter, Linkedin, Wordpress, …

The concept

Coderwall.com is a website targetting a professional audience, allowing its users (people or companies) to have a profile page with their social links. It is also easy to search for companies (or "teams" according to their jargon) by location, and then access the profile pages of their employees.
Linkedin is a great source of information for demographics. Using Coderwall as a starting point allows us to collect a number of twitter usernames along with their Linkedin profile.
Blogs can also be a great source of information about their author. We'll explain how we used them to guess the user information in the next part

The implementation

Step 1: Collect Coderwall profiles and the socials links associated

At this time, Coderwall does not provide a search API. We has to use the search engine Bing to query for companies and people that have a profile on coderwall. We used the Bing API with these queries for companies pages:

- "United Kingdom" site:http://coderwall.com/teams
- "UK" site:http://coderwall.com/teams
- "England" site:http://coderwall.com/teams

and theses for users pages:

- "United Kingdom" site:http://coderwall.com/
- "UK" site:http://coderwall.com
- "England" site:http://coderwall.com

The company pages were parsed to find the profiles of their members. Theses profiles were merged the to the rest of the profiles found with Bing.
And finally each profile page's DOM is parsed to retrieve the social links of their users. Since Coderwall doesn't provide social links through their Profile API, DOM parsing was the only solution. We drop the profiles which do not have links to at least Twitter and Linkedin.

Step 2: Crawling the Linkedin pages for Location, Education, Age and Gender

The public information available on Linkedin by default:

- location,
- academic background with a start date and an end date for each school, university, etc.
- firstname

Getting the location: we noticed that Linkedin pages had the most consistent way of expressing the location. It followed one of theses three main patterns:

- City, County, Country
- City, Country
- County, Country

This made it very to find to locate our users.

Getting the education: we used a rule based classifier that loops through the different institutions the user attended and looks keywords.

```
 HIGH_IND = [
     "university", "politec", "polytech", "Institute", "academy", 'U. of ', “University
of”,  "business",  "management",  "marketing",  "Univers",  "science",  "economics",
"political" ]

  MID_IND = [
    "high school", "high", "Grammar", "boys", "girls", "college"
  ]

  LOW_IND = [
    "elementary"
  ]
```

We also use a list of university names and high school names in England. We obtained this list by searching the internet with Google. But these lists are used only if we cannot determine the level of education by probing with the keywords. Experience proved that classification is less restrictive and more efficient with smaller carefully selected keywords.

Getting the age: this uses two different classifiers that complement each other. The first is a simple rule based classifier, that checks:

- the end date of high school
- the start date of the first university (and other “high” education institutes)

and makes the assumption that this dates corresponds to the user’s 18th birthday. The education is mandatory in United Kingdom until the age of 18 and for almost all of the people this happens to be the year they graduate from highschool. For that reason first we check the finishing year of the high school and if not available then the starting year of university.

This worked nicely for most of the profiles, except for those whose academic background do not include start and end date. This is where the second classifier comes in: it is available through *ageanalyzer.com* and allows to guess the age of user based on his blog content. This classifier returns a range for the age and we store the median of that range.

Getting the gender: for this we used the first name provided on Linkedin with a list of masculin, feminin and unisex first names. The case of a unisex firstname is fairly rare, that’s why we chose to simply ignore theses profiles. The list of names was found on the internet.[4]

[4] http://www.codeproject.com/Articles/2169/Name-genderization

Step 3: Sentiment analysis on the tweets

This follows the exact same procedure as explained in approach 1.

Approach 3: from About.me to Twitter, Linkedin, Wordpress, …

The concept

About.me is another website much like *Coderwall.com* in a sense that it serves as a public page for their users, containing a small description, location, social links, etc. The procedure was exactly like the one used for Coderwall.

The implementation

Step 1: Collect About.me profiles and the socials links associated

About.me does provide with a search API, only though tons of paperwork. The easiest and most realistic solution for us was to use our Bing crawler again, and to look for *about.me* profiles in the UK. We first started with the usual:

- "United Kingdom" site:http://about.me
- "UK" site:http://about.me
- "England" site:http://about.me

and then we deepened the search by trying each county, for example

- "West Yorkshire" site:http://about.me

and finally, we tried each city and town found on wikipedia:

- "London" site:http://about.me

This gathered a lot of data but some wasn't necessarily relevant (for example, we gathered many profiles from "London, Ontario, Canada". Fortunately the Linkedin location will later allow to filter irrelevant profiles.
On all the *about.me* profiles that were collected, we parsed the page to retrieve the social links (including the personal website/blog links)

Step 2: Crawling the Linkedin pages for Location, Education, Age and Gender

This step is the same as for Coderwall. Linkedin and personal websites/blogs were our main source of data and the same classifiers were used.

Step 3: Sentiment analysis on the tweets

This follows the exact same procedure as for the other approaches, using lymbix API.

Part II: Technical issues

Throughout the project we faced several challenges, the main of which were: collecting enough user profiles and finding the relevant Twitter profile, gathering accurate demographic information for the users.

In the first approach we had a issue with the number of search requests which you can do with the Google API per day (for free). The number is limited to 100 per day per account, so to be able to find profiles of people from every town in the list we had to run search requests for several days.

Another problem was that in the OkCupid profiles, education field is not mandatory and therefore some of the users did not fill it in. Another issue which we had with the education was the way it was represented. There users can choose from a predefined list of options. Most of them were clear (like “High School”, “Graduated from Master program”, “Graduated from two year college”), but some like “Working on College/University” were problematic. In some countries college and university give you the same level of education but in UK they do not. We think that this option has to be chosen to indicate that the user is still in university but some UK users might choose it if they are still in college and thus we take their level of education as `Mid` while it is still `Low`.

The way to check for twitter account is easy and straightforward but **during the manual check and labeling it turned out that some of the twitter accounts which we found do not correspond to the same OkCupid user**. For some of these users we found out that on twitter they use slightly different username than on OkCupid (for example: “`AmandaTree`” on OkCupid and “`_amandatree`” on Twitter). We had an idea to use a distance measure to search over the Twitter API for the accounts with similar usernames and later check which of the accounts was the correct one based on common topics mined from the profiles, but we did not succeed in this task. This will have some reflection on the accuracy of the training dataset.

In the second and third approach, the collection of user profiles was not a big problem. there were no limitations on the number of search requests made with Bing. We had no problems connecting the users with their twitter accounts (as we directly get the URL to it). The problem here was collecting the correct the demographic information form Linkedin. As mentioned above, for determining the gender we used dictionaries with masculin, feminin and unisex first names. To increase the accuracy of the dataset we just dropped the people with unisex names. During the process of collecting profiles, we gathered many users who live in a city which happens to have the same name as some city in the United Kingdom but in fact be in another country. Therefore in the Linkedin profile we did a check if “United Kingdom” is listed in the location field.

The big issue in this approaches was to determine the age of the user. None of the websites which we crawled or searched here was storing information about age (or birthdate). For that we decided to check the years in which the user graduated high school or started university. As stated before, being mandatory, most of the people in UK finish high school at the age of 18, while it is not sure that he will start university immediately after high school graduation and hence the user might be older than 18-19 at that point. And so knowing the age at some point in time we can calculate the current age. For the users which do not provide enough information in the education field we used the *ageanalyzer.com*. While it is less accurate it still provides some age detection. We also had another idea for age detection from the name: to use the name trends over the years. Therefore we would be able to determine the generation of the person. But this method is not that accurate so we focused on implementing the the other two and at the end we did not have time to return to the name trends.

Detecting the level of education is described in the first part of the report, but the problem of detecting education if it is not filled in Linkedin stays. For that we just decided to classify the education of these people as “2”. Some of the universities or schools are given with their abbreviation and therefore we put also the abbreviation of each educational institution in the lists of schools and universities which we made. Therefore at the end if we have to check the list we search both name and abbreviation for a match.

Part III: Data quality

For each of our 3 approaches, we make different assumptions that can more or less influence the quality of the data

Accuracy of the information gathered

For the first approach, the assumption is that a users stick to the same username across social networks, which can be true but only to some extent. As already mentioned above, if the username doesn't contain only common words or names, such as `@aksowihp78` this is likely to be true. On the other hand, some username are as simple as a first name like: '@paul'.
Regardless of this assumption, the amount of matches found between twitter and OkCupid usernames was high enough for us to provide half of our total data. The bright side of it is that when the usernames actually match, we have very accurate demographic information about users thanks to the dating website, assuming they don't lie about their profile information. We think this is unlikely since people sign up to a dating site to be matched to people of the same age and who live in the same location, and it would not benefit them to lie about their information.
We noticed that male profiles were more likely to have a matching twitter username than the female profiles, this caused a slight skew in our dataset towards male profiles.

For the second and the third approach, *about.me* and *coderwall.com* provide very accurate links between the social networks. The assumptions are then made at the classification level. While the location and the education is very close to accurate thanks to linkedin, the age remains less certain. Our assumptions on these are that the user is around 18 during his first year of university or last year of college (ie. british for high school).
Sometimes it is difficult to get the education (because it's not mentioned in the linkedin profile) and also it's hard to guess the gender based on the firstname: since this occurs rarely, we just release that profile from our database, in order to keep the information as certain as possible.
Having very accurate information was always our main goal, that's why we preferred to have less data but of better quality.

Usefulness of the data gathered for classification

If we look only at *Coderwall.com* and *About.me*, some might argue that they target a very specific audience, IT people for the first (highly educated males, often young), mediatized/public personnalities for youtube. People who maintain a controlled and cultivated online identity; not the guys/girl next door. And some might argue that this can introduce asymmetrical behaviour of the classifier.

However, the fact that we added the dating website information introduce a bigger variety in our profile database, especially for a website as popular as OkCupid[5], that sees registration from really all ages and all backgrounds.

[5] according to wikipedia, 1.3 million unique visitors on February 2011 (*unique*, meaning it's can't be another group's crawler).

Part IV: Free-Form task

Goal

The goal of our free-form task was to do **topic mining on the dating profiles of people** to determine what important topics are for the different genders at different ages.To determine these topics we revisisted the dating website and gathered the summary people gave about themselves. Via the use of a third-party api we extracted the topics from the summaries gathered and categorised these topics by gender and age.

The end goal was to use again different profile descriptions (ex: Okupid description and twitter bio) to construct a distance that could possibly help identify twitter usernames that were wrongly assigned to Okcupid accounts. *Unfortunately we were not able to finish this task.* Nevertheless, we have gathered a few interesting results.

Results

In the table below some of the most frequently mentioned topics are displayed per gender and age class. Overall music was one of the most mentioned topics by both genders and across different ages. This might suggest that people find taste in music a very important feature to be matched on, and look for a partner with the same tastes.
Another easy to spot trend is the number of mentions of children that increases with with age. It is mentioned more frequently on female profiles and the mentions start at an earlier age for females.

We also noticed that females with an age between 31 and 40 mention their job three times more often than males of the same age.

	Male topics	**Female topics**
age <= 20	football, sports, music	music, university, work
21 <= age <= 30	work, job, music	music, film, job, first mention of being a mother
31 <= age <= 40	music, food, working day, first mention of a ‘Son’	work (A lot of mentions), food, child

41 <= age <= 50	Commitment, work, son	work (Less mentions), children, music
51 <= age <= 60	Health, old, son, work	Very few profiles, but they to mention “Glass of wine” relatively frequently