INTEGRATING DYNAMIC PROFILES USING SENTIMENT ANALYSIS ON VAA’S

MASTER THESIS

Ulzhan Kakenova
from
Kazakhstan

University of Bern
July 2016

Prof. Dr. Andreas Meier, Information Systems, Department of Informatics, University of Fribourg, Supervisor
Dr. Luis Teran, Information Systems, Department of Informatics, University of Fribourg, Assistant
Abstract

In the latest electoral politics we can clearly see the growing widespread use of the internet-based tool - Voting Advice Applications (VAA’s). These online tools such as Smartvote in Switzerland, WhoGetsMyVoteUK in United Kingdom, Wahl-O-Mat in Germany, StemWijzer in the Netherlands and Electoral Compass in USA provide an assistance to the citizens in order to easily and quickly define the candidate or the party which matches their own point of view, moreover VAA’s are extremely attractive for the countries with the multi-party systems.

Voting advice applications (VAA’s) are interactive online tools that have become extremely popular in the electoral campaigns of European countries. The VAA’s are developed to assist voters by enhancing the basis on which they decide how to vote. Typical VAA’s do this by matching users’ policy-preferences with the positions of parties or candidates. Today VAA’s are able to attract huge number of respondents and to provide a significantly rich source of mass public opinion data.

However, not all the parties or candidates provide their positions regarding the statements revealed in the VAA to derive the recommendation for users. The widely used solution is to use expert analysis by extracting the position of parties or candidates from different data sources such as interviews, speeches, discussions and so on. This type of solution is time-consuming and sometimes arguable, which may lead to mismatching. Therefore, it is significant to find the dynamic approach for building candidate profile.

In this thesis, we propose a new VAA 2.0 which generates dynamic profiles of politicians by extracting their attitude towards policy issue statements from their official twitter accounts. The goal is to have a functional voting advice application where dynamic profiles generated more efficiently without expert analysis or when politicians reject to answer the policy issue statements.
Content

List of Acronyms ........................................................................................................... vii
List of Figures ................................................................................................................ ix

1 Introduction ................................................................................................................. 11
  1.1 Problem statement ................................................................................................. 11
  1.2 Objectives ................................................................................................................ 12
  1.3 Outline ..................................................................................................................... 12

2 Background knowledge ............................................................................................. 13
  2.1 VAA’s in general ..................................................................................................... 13
  2.2 Principles of VAA design ...................................................................................... 14
  2.3 Analysing VAA Data ............................................................................................ 22
  2.4 Recommender systems ......................................................................................... 23

3 Implementation of VAA 2.0 ..................................................................................... 26
  3.1 Scope ....................................................................................................................... 26
  3.1.1 VAA policy issue statements ............................................................................ 27
  3.1.2 VAA 2.0 architecture and design ..................................................................... 31
  3.2 Basic structure ....................................................................................................... 32
  3.2.1 Voter profile ...................................................................................................... 34
  3.2.2 Candidate profile ............................................................................................. 35
  3.2.3 Matching algorithm ......................................................................................... 36
  3.2.4 Admin panel ..................................................................................................... 38
  3.3 Sentiment analysis ................................................................................................ 45
  3.3.1 Representing emotion ....................................................................................... 46
  3.3.2 Sentiment dictionary ......................................................................................... 47
  3.3.3 Estimating sentiment ....................................................................................... 49
  3.3.4 Generating statements for dynamic profiles ................................................... 52

4 Evaluation .................................................................................................................. 58
  4.1 Dynamically generated data analysis ................................................................... 58
  4.2 Comparison of data (experts’ data vs. twitter data) ............................................. 63

5 Conclusion .................................................................................................................. 65

Bibliography .................................................................................................................. 67

Appendix ......................................................................................................................... 69
List of Acronyms

VAA Voting Advice Application.................................................................11
CA Completely Agree ...........................................................................21
A Agree.................................................................................................21
N Neutral..............................................................................................21
D Disagree...........................................................................................21
CD Completely Disagree.................................................................21
RS Recommender System.................................................................23
API Application Programming Interface.................................25
List of Figures

Figure 1: Best to avoid example of policy issue statement..........................15
Figure 2: Ideal distribution example for policy issue statement..................15
Figure 3: An outline of the Delphi method. Drawn from the Kostas Gemenis (2014) paper...........................................................................................................18
Figure 4: Main theories in science: matching voters to candidates/parties. Drawn from the Fernando Mendez and Jonathan Wheatley (2015) paper...........................................................................................................18
Figure 5: Directional and proximity model of issue voter. Drawn from the original Rabinowitz and Macdonald (1989) paper.................................19
Figure 6: “City block” metric for proximity model .....................................21
Figure 7: “Scalar product” metric for directional model ..............................21
Figure 8: Hybrid model................................................................................22
Figure 9: Initial policy issue statements .....................................................26
Figure 10: Modified policy issue statements and key words ......................28
Figure 11: Proposed VAA 2.0 Architecture ................................................30
Figure 12: Main page for VAA 2.0 ...............................................................31
Figure 13: Voters’ registration page for VAA 2.0 ........................................31
Figure 14: Candidates’ registration page for VAA 2.0 .................................32
Figure 15: Voters’ profile for VAA 2.0 ........................................................33
Figure 16: Candidate profile for VAA 2.0 ....................................................34
Figure 17: The recommendation results..................................................35
Figure 18: The admin panel .......................................................................36
Figure 19: Dynamically generated candidates’ position to statements ........43
Figure 20: Russell model of emotional affect. Drawn from the original Russell and James (1980) paper .................................................................44
Figure 21: Example of Russell models to be applied for visualizing the tweet’s estimated sentiment ........................................................................50
Figure 22: The negative and positive polar of the visualization map ..........51
Figure 23: The visualization map with CA, A, N, D, CD regions .................52
Figure 24: The example of estimated tweets ............................................53
Figure 25: Proposed Twitter API and Sentiment analysis usage ..................54
Figure 26: The raw data extracted using 100 tweets ..................................56
Figure 27: The modified data extracted using 100 tweets ..................................57
Figure 28: The data extracted using 200 tweets ........................................58
Figure 29: Periodical analysis of twitter accounts ..........................................62
Figure 30: The comparison of datasets (twitter vs. experts) .........................63
1

Introduction

1.1 Problem statement

Current VAA’s are widely used during political campaigns and significantly important as recommendation tool for citizens. However, not all the candidates across Europe and USA provide answers to the VAA statements, and usually experts run research analysis on number of issues by extracting the relevant data from random sources such as interviews, discussions or debates. The main disadvantage of this solution is that experts require enormous amount of time to gather, extract and process the data in order to summarize the answers to statements. Moreover, we should consider the human factor, so the data can be biased or manipulated for personal reasons. Therefore it is critically important to develop new solution for the cases when political candidates or parties refuse to answer VAA statements in order to provide well-functioning and trustful recommendations to VAA users.

Prior the implementation author aims to identify the most suitable approaches to build dynamic profiles of candidates, the most appropriate architecture for the recommender system with dynamic profiles, to define the
advantages and disadvantages of dynamic profiles and methods for building VAA with dynamic profiles of candidates and static profiles of citizens.

1.2 Objectives

The objective of this thesis is to research for the most appropriate solutions, and apply the most suitable approach for generating dynamic profiles in VAA’s as a new web-platform (VAA 2.0).

The author aims to find the solutions to the problems and questions mentioned in the previous section, such as finding an appropriate tools and implementing VAA 2.0 so that it can be modified or extended in the future. Moreover, author will provide enough background information so that readers can easily understand the main issues and the solution decisions.

The VAA 2.0 will be able to extract the data from candidates’ social account such as twitter, analyze the information using key words for each policy issue statement, process the extracted data using sentiment analysis and build candidates dynamic profile based on the twits of the candidates. These dynamic profiles allow matching with the users’ answers to the static statements of VAA, and finally generating reliable recommendation to users.

1.3 Outline

The next chapter reveals all necessary background information for the reader to understand the scope of the thesis and the main concepts of the following chapters. It provides the explanations of the following concepts: VAA’s in general, principles of VAA design, analysis of VAA data, recommender systems, and gives background information about some of the tools used for the implementation.

Afterwards the VAA 2.0 project is explained and the most significant code snippets are included and explained in that chapter along with the deployment information for the product. The sentiment analysis implementation and matching algorithm for the recommendation generation is also clearly explained in the following chapters. The features along with the approach and solutions are part of the appropriate chapters. Finally, this thesis as well as many other thesis’s, ends up with a conclusion about the overall work done.
2

Background knowledge

2.1 VAA’s in general

Voting Advice Application (VAA) is a generic term for the, freely available on the internet, tools that match the preferences of voters to that of candidates, political parties, or other stakeholders [1]. The overall mechanism of a general VAA is not complicated and differs between each other in terms design and architectural style. Generally, before an election, a team of researchers or special experts formulates a number of statements on policy issues that are politically salient and determines the positions of parties and/or candidates on each of these policy issues [2], [3], [4] either by making analysis on number of publically available sources such as interview, speeches, debates and so on or by directly gathering the attitude of each politician towards every policy issue presented on VAA. Then VAA users are then able to answer the same questionnaire of policy issues in order to compare their responses to those of political parties and/or candidates to receive a reliable recommendation. Current VAAs use variety of different matching algorithms and various visualization techniques to show how
correct the users’ policy preferences have been matched to those of different political parties and/or candidates [5].

VAAs have gained an increasing popularity across Europe and United States of America, where educational and media organizations, government-sponsored civic education institutes have developed over two dozen this kind of applications. Moreover, in some countries election campaigns widely use VAA tools with an estimated number of users reaching 40 per cent of the electorate (The Netherlands 2012), or as high as 6.7 million in (Germany 2009) [1] [6].

According to VAA designers, they assume that, VAAs can act as voter information tools by communicating information about the policy positions of political parties and/or candidates and by providing the matching output that could support citizens to make more informed decisions when casting a vote [7].

2.2 Principles of VAA design

The main principles of VAA design are statement selection and formulation, encoding political party and/or candidate answers and matching voters to political parties and/or candidates.

Statement selection is the basic step where the policy issue statements for VAA questionnaire identified and must be relevant to the political and social issues. First of all, VAAs should be prominent to the current political debate and cover as much as possible a variety of prominent issues, particularly from different policy topics such defense, foreign policy, taxation and spending, environment, role of religion in politics and so on. Different advices can be given to VAA users due to different combinations of policy issue statements. Therefore, a systematic bias in favour of one or other political party and/or candidate may occur due to the biased choice of statements given in the VAA questionnaire. Second, voters (VAA users) and political parties or candidates should be divided into two opposing groups by VAA statements. It is important to prevent the usage of so called “valence” issues in VAA questionnaire such as following policy issue: “Policies should be aimed at increasing economic growth”, the reason is that both political parties/candidates and voters are likely to be in the same camp and answer agree. Moreover, it is significant to avoid issues about which most or all voters
share a common preference, but where that preference is shared by only one political party and/or candidate as this will lead to bias in favour of that party/candidate.

The below given are the examples of policy issue statements which are best to avoid (figure 1) and good to implement (figure 2).

**Figure 1:** Best to avoid example of policy issue statement

**Figure 2:** Ideal distribution example for policy issue statement

*Statement formulation* is significant issue and could be a key to well-functioning and friendly VAA. Before all, policy issue statements should not refer to multiple attitude objects due to the fact that VAA users may have
different opinions on different objects, for instance: “The ownership of any kind of vehicle has to become cheaper and the use of the any kind of vehicles more expensive. Hence kilometer levy have to be integrated”. Second, policy issue statements should be free of any kind of technical or legal jargon. Most of the users could be not familiar with the technical or legal description or words; hence this may lead to misuse of VAA in general or misunderstanding of the question which will produce incorrect recommendation in the end. Third, policy issue statements should be as short as possible. It is well known that long and deep descriptions are usually partially skipped, skimmed or unread which may lead to misunderstanding of the question. Therefore, people may become less interested in using VAA which takes too much time to use and understand. Fourth, not all the policy issue statements should be directed to the same route, for example: all left-pointing, all right-pointing, or all pro-EU. Fifth, it is essential to present the usage of negative formulations in the statements such as using “not”, because double negatives, i.e. not disagree could be confusing for respondents and also may lead to misunderstanding of the statement. Sixth, it is better to avoid quantitative statements, for instance the user is given following statement: “The rate of income tax should be raised to 50%”. VAA users may disagree due to two different reasons such as 1) users do not want to raise income tax at all, or 2) because users want to raise it to 60%. So, this statement is confusing the users again and may give incorrect recommendation in the end.

When formulating policy issue statements it is significant to follow above described rules. This rules allows to create correct VAA questionnaire that users will find both useful and understandable. It is essential to avoid confusing statements which lead to misunderstanding and incorrect recommendations.

**Encoding political party and/or candidate answers** is another necessary point in the design of VAA. As soon as the questionnaire has been developed, beforehand identified political parties and/or candidates presented in the elections have to be coded on each policy issue statement of the questionnaire. In general, there are two different ways of doing this:

1) Political parties and/or candidates can be asked to code themselves. This means that they will provide their own answers for each policy issue statement of the VAA questionnaire. The main risks in using following method are those they either may lie while giving answers in order to provide
opinions that most users are likely to agree with, or they may simply reject to answer to the given statements of the VAA questionnaire due to the personal or political reasons.

2) Political parties and/or candidates can be coded by experts. This means that experts will make an analysis on different datasets extracted from different data sources such as interviews, speeches, discussions in order to define the correct position of political parties and/or candidates regarding to the VAA statements. However, there are also some particular danger occurs, for instance the codes may reflect the subjective views of the experts and can be unreliable due the data misuse or modification. In order to prevent previously mentioned danger the so-called Delphi method of coding is introduced. In an attempt to produce more reliable codes this method uses successive rounds of coding by experts [3].

The Delphi method of coding works in the following way:

i) in the beginning, a group of experts are assigned to code each political party and/or candidate. They do their work independently from each other, and then they have to provide estimates of the policy issue positions of the parties and/or candidates for each given VAA issue statement and prove their estimates by revealing information taken either from statements in the press, debates, party manifestos or any other relevant sources. Usually five different coders are selected to each political party and/or candidate.

ii) consequently, individual estimates and their proved cases are fed back to the panel anonymously for a second round of coding in which panelists modify and update their initial codes with the support of justifications from their so coders. As soon as the codes approximated to a sufficient extent to be considered consistent (van der Eijk’s (2001) measure of agreement ($A \geq 0.7$), the median response is taken as the final estimate. Overall only two rounds of coding are have to be done [8].
Choosing an algorithm to match users with political parties and/or candidates is the most significant and complex part of the VAA design that presents the key functionality issues of the tool.
According to the main theories in science the figure above shows the main issues considered during the process of identifying the preferences and making choices by voters. In general elections, voters either act using non-rational prejudices or rational instruments. In first case, voters consider charismatic leadership of candidates and the party identity, and then give a vote. In the second case, voters can be defined through clientalism or strategic vote due to the personal reasons, and programmatically by considering the shared policy issue statements, competence and the ideological beliefs of candidates/parties. However, VAA tool have to implement matching algorithm that would only consider the policy issue statements by implementing either directional or proximity behavioral model and avoiding non-rational aspects that may have negative effect on the correctness of decision making process.

The algorithm that developers have to choose is used to match users with political parties and/or candidates and mainly depends on the chosen model of voting behaviour. In general there are two different models of voting behaviour, such as:

1) The *proximity model* reveals that the most significant criteria which matters most in a voter's decision making is the distance between voter and the political party/candidate on a series of critical policies (Downs 1957) [9].

2) The *directional model* presumes that what really matters most is to be on the “same side” of the argument on as many policy issues as possible for both the voter and party / candidate (Rabinowitz 1989) [10].

![Figure 5: Directional and proximity model of issue voter. Drawn from the original Rabinowitz and Macdonald (1989) paper.](image)
Above given figure reveals the example of both directional and proximity models to define difference between them. In the figure, we can clearly see the example of the voter making choice regarding the state policy issue “Health insurance should be provided by the government or obtained through private companies”, where V stands for the voter, A for some candidate on the left side and B for another candidate on the right side. Here we can see that voters stand on the left side (-1), whereas A (-4) and B (1).

According to the proximity model, voter prefers candidate closest to him/her on this dimensional scale. So, according to the model, voter would prefer candidate “B” as his/her representative to any of the other candidates. Also, the farther away from the voters position, the less voter like the representative. Hence, after candidate “B”, voter would prefer candidate “A” if there is no any other candidate closer than candidate “A”.

In contrast, the directional model assumes that voters prefer candidates “on their side”, and the more voter on their side the better it is. A “side” can be defined as a politically meaningful grouping, for instance a political party or group of candidates who more or less share the same basic ideological views. In the given figure, “side” is defined as voters’ relative position to the zero point (marked by vertical line with 0 in Fig.4). Hence, according to the directional model the voter would prefer those candidates who are located to the left of zero point, because voter is on the left side (-1). Also, voter has the strongest intensity of preferences for the extreme candidates on the voters side due to the reason that those candidates on the left more likely members of the voters’ “team” or “grouping”. Therefore, voter located on the left side would prefer candidate “A” and all other who is on the left if there are any. Nevertheless, sides/teams/groupings are clearly defined, and the position of the zero point which is neutral to both sides is absolutely critical in the directional theory.

Above mentioned models are used to identify the matching algorithm. Hence, for proximity model of voting the most comprehensible choice of the matching algorithm is so called “city block” metric, which is shown in the following matrix (fig.6). In the figure below, the matrix headings in the columns and rows are based on a five-point scale with the following answer categories usually given on VAA questionnaires: “CA” stands for Completely Agree; “A” Agree; “N” Neutral or Neither Agree nor Disagree; “D” Disagree; “CD” Completely Disagree, and the numbers in the cell indicates the weight
given when a voter (in the rows) and a political party and/or candidate (in the columns) land in one of the possible cells for each policy statement.

\[
\begin{pmatrix}
CD & D & N & A & CA \\
CD & 1 & .5 & 0 & -.5 & -1 \\
D & .5 & 1 & .5 & 0 & -.5 \\
N & 0 & .5 & 1 & .5 & 0 \\
A & -.5 & 0 & .5 & 1 & .5 \\
CA & -1 & -.5 & 0 & .5 & 1 \\
\end{pmatrix}
\]

*Figure 6: “City block” metric for proximity model*

For a directional model of voting, the most obvious choice of matching algorithm is another metric, so called “scalar product” metric, where the weights given to each cell represent the product of the user’s and the party’s positions if Completely Agree as “CA” is assigned the value 1; Agree as “A” the value is 0.5; Neither Agree nor Disagree as “N” the value is 0; Disagree as “D” is -0.5; and Completely Disagree as “CD” the value is -1. In the development VAA 2.0 we decided to use directional model.

\[
\begin{pmatrix}
CD & D & N & A & CA \\
CD & 1 & .5 & 0 & -.5 & -1 \\
D & .5 & .25 & 0 & -.25 & -.5 \\
N & 0 & 0 & 0 & 0 & 0 \\
A & -.5 & -.25 & 0 & .25 & .5 \\
CA & -1 & -.5 & 0 & .5 & 1 \\
\end{pmatrix}
\]

*Figure 7: “Scalar product” metric for directional model*

However, there is an opportunity to apply a hybrid model that simply averages the weights given in the city block and scalar product matrices, as shown in the 'hybrid' matrix below. This kind of hybrid model is currently used by Preference Matcher. As it was mentioned above, the matrix can be described as “CA” that stands for Completely Agree; “A” Agree; “N” Neutral or Neither Agree nor Disagree; “D” Disagree; “CD” Completely Disagree, and the numbers in the cell indicates the averages calculated and then given when a
voter (in the rows) and a political party and/or candidate (in the columns) land in one of the possible cells for each policy statement.

![Figure 8: Hybrid model](image)

### 2.3 Analysing VAA Data

After the usage of VAA and huge number of generated recommendations there are still sufficient amount of datasets available. As we can remember, the main functionality of the VAA is to produce correct and reliable recommendations to voters and match them with the appropriate candidates. However, the datasets can be also efficiently used. For instance, some of the fundamental questions of political and social sciences can be taken from the VAA-generated datasets. Voting Advice Applications produce vast number of datasets which incorporate the opinions of number of voters on a wide range of policy issues that more or less traditional surveys isn't able to cover. Variety of experts and researchers have come to research using VAA datasets encompassing identifying latent ideological dimensions and mapping parties/candidates and part/candidate supporters. However, the disadvantage of using VAA-generated datasets as a source of information for researches is that the sample of users of VAA tool is self-selected and can relate to any age/social/gender group which is necessary to know in some cases, and it is not representative of the population at large.

However, if the VAA-generated datasets are decided to be used it is significant to clean out the datasets out of rogue data. Rogue data occur in the following cases, when: a) a VAA user answers the policy issue statements automatically without reading them and aims to see how the VAA tool works in general, b) if one same user completes the VAA questionnaire more than
once c) if a user from different country and outside the territory in question uses the VAA tool.

Typical VAA-generated dataset contains following information, such as responses to 21 policy issue statements or more, depends on the VAA; user responses to supplementary questions which includes age, gender, interest in politics, party affiliation and vote intention; timer data that includes the time taken for users to complete the questionnaire as a whole as well as time taken to complete each issue statement; and an encrypted code corresponding to the users’ IP number.

In order to clean out datasets from rogue data the following entries have to be removed from the VAA generated dataset, those in which: 1) there are more than ten successive identical answers, 2) there are 20 or more "no opinion" responses, 3) there is at least one policy issue statement answered in 1 second or less, 4) at least three policy issue statements are answered in 2 seconds or less, 5) the time taken to answer all thirty policy issue statements is less than two minutes or more than 5,000 seconds, 6) the user shares an IP number with another user and there is no added evidence that he/she is a different user (i.e. through responses to supplementary questions), 7) the user declared himself/herself ineligible to vote, 8) the user declares a probably spurious age (95 years old or more). If the dataset had been cleaned through using above mentioned ways, then the clean dataset is ready for further analysis and research, and we can strongly assume that the dataset does not contain any rogue data.

2.4 Recommender systems

Recommender system (RS) or sometimes also called a recommendation engine is online tool that allows algorithm developers to estimate future events on what a user may or may not like among a list of provided objects. Today we can clearly see to be recommendation engine a part of a well-known web sites and services which include Amazon, Facebook, YouTube and many others, so there is no reason to argue about its popularity. Moreover, recommendation engines widely used in Voting Advice Applications by matching political parties and/or candidates with the user by analyzing their preferences about policy issue statements. This means that like many other
information based problems, it is significant to choose correct algorithm that will be suitable to the addressed problem.

Basic approaches that mostly used in recommender systems are following: collaborative filtering or content-based filtering. Also, there exists a hybrid approach that combines above mentioned two approaches.

Collaborative filtering basically uses the model of prior user behaviour for recommendations. In order to build up the model the single user's behaviour exclusively or more efficiently the behaviour of other users with similar features are considered. When the collaborative filtering model considers other users' behaviour it mainly uses the group knowledge to set up a recommendation based on like users. Essentially, multiple users are automatically collaborated and the based on revealing the similar preferences or behaviors they are filtered out to build a recommendations. For instance, if there is a website which recommends blog, the system is able to gather information given from number of registered users who subscribed to and read blogs, and then can be wisely grouped according to their preferences. They can be grouped together if they read significant amount of similar blogs. And from this information, the most popular blog read among this grouping can be easily identified. After this, the system can recommend the most popular blog for a particular user in the identified group that the users neither read nor subscribe to.

Content-based filtering uses the information about user's behaviour to set up a recommendation. For instance, historical browsing information of the user such as which blogs is read by the user and the basic features of those blogs can be used for this approach. If a user mainly reads papers about Software Engineering or leaves comments on blogs related to this topic, content-based filtering approach is able to use this history information to define analogous content in order to recommend it to the user (papers on Software engineering or blog about it). In order to perform above mentioned steps there are two ways such as manually defining the content or automatically extracting content by using different similarity methods.

Hybrid approach basically combines above mentioned two approaches such as collaborative and content-based filtering where both of them improve the efficiency and increase the complexity of recommendation engine. Basically, by integrating the results of two approaches (collaborative and
content-based filtering) the hybrid approach produces the potential for more accurate and correct recommendation.
3

Implementation of VAA 2.0

3.1 Scope

The system offers the functionality of general VAA which allows users (i.e. voters) to find a candidate that stands closest to their preferences. General VAA uses expert analysis or directly asks candidates to answer policy issue statements to generate candidate profiles. The usage of these two methods and their drawbacks were discussed in previous sections, therefore it is essential to implement new method to build candidate profile. In VAA 2.0 we have implemented the twitter API, sentiment analysis in order to use official twitter accounts of candidates for data extractions and based on these data to build dynamic profiles of candidates which will be updated according to their tweets in their twitter accounts. Moreover, the gathered data from twitter accounts, which means the preferences of candidates to policy issue statements are evaluated and compared to the data proposed by experts through the analysis of number of official sources such as interviews, debates session and speeches of candidates. This datasets is taken from the “Electoral Compass” VAA for United States of America and Canada. The VAA policy issue
statements also taken from the following “Electoral Compass”, the main reason is that all USA politicians are absolutely active users of twitter in comparison with European candidates. Moreover, the VAA policy issues statements have to be used in real case campaign in order to evaluate the truthfulness and the real data have to be gathered and analyzed from real twitter accounts.

The VAA 2.0 allows different scenarios of usage. In first scenario, there are two types of users which are either voter or candidate. Voter registers into the system and creates users profiles where his/her recommendations will be stored. Candidate registers into the systems and creates candidate profile and fills in the VAA questionnaire. Moreover, candidate have to prove his identity by providing contact details and provident of identity such as passport details.

3.1.1 VAA policy issue statements

The questions provided in VAA 2.0 are taken from “Electoral Compass USA” [11]. There are twenty one policy issue statements which cover seven different areas such as gun control, environment, military, economy, income, national security and family. The available answers are following completely agree, agree (or tend to agree), neutral, disagree (tend to disagree), completely disagree and no opinion. In the table below you can see the 21 issue statements presented in VAA 2.0 which are relevant to the listed topics and has an appropriate size in terms of number of words and understandability.

<table>
<thead>
<tr>
<th>№</th>
<th>Topic</th>
<th>Policy issue statement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Gun control</td>
<td>People should have a background check and obtain a license before they can buy a gun</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Stricter gun control will not reduce crime</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>All semi-automatic weapons should be banned</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>The effects of global warming are grossly exaggerated</td>
</tr>
<tr>
<td>5</td>
<td>Environment</td>
<td>An additional carbon tax on fuel will effectively reduce pollution</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>6</td>
<td>The US should never sign international treaties on climate change that limit economic growth</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Military</td>
<td>The US had every right to invade Iraq</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td>The US is safer because of the invasion of Iraq</td>
</tr>
<tr>
<td>9</td>
<td></td>
<td>The new president should begin to bring home all US troops from Iraq immediately</td>
</tr>
<tr>
<td>10</td>
<td>Economy</td>
<td>The best way to reduce the federal deficit is to raise taxes</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>The tax cuts for people with a higher income should be reversed</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>The government has no responsibility to provide retirement funds</td>
</tr>
<tr>
<td>13</td>
<td>Income</td>
<td>Mortgage lenders should be more tightly controlled</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>People with higher incomes should receive less Medicare benefits</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>The federal government should reduce income inequality</td>
</tr>
<tr>
<td>16</td>
<td>National Security</td>
<td>The US should reduce its financial contribution to the UN</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>Iran is not an imminent threat to world peace</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>The US should decrease its spending on defense</td>
</tr>
<tr>
<td>19</td>
<td>Family</td>
<td>Same sex marriages should be made legal</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>Abortion should be made completely illegal</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>Using embryos for stem cell research is acceptable as it enables us to find cures for diseases</td>
</tr>
</tbody>
</table>

*Figure 9: Initial policy issue statements*

For generation of dynamic profiles, in order to extract relevant datasets from twitter accounts related to the above presented policy issue statements
it is necessary to properly define key words for each statement to run search and find the most related tweets for each policy issue statement. After searching the related tweets they are analyzed for sentiment identification. Hence, it is critically important to define key words for each statement. In the table below, the key words and statement to which they match are presented.

<table>
<thead>
<tr>
<th>№</th>
<th>Policy issue statement</th>
<th>Key words</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>People should have a background check and obtain a license before they can buy a gun</td>
<td>background check license buy gun</td>
</tr>
<tr>
<td>2</td>
<td>Stricter gun control will not reduce crime</td>
<td>gun control reduce crime</td>
</tr>
<tr>
<td>3</td>
<td>All semi-automatic weapons should be banned</td>
<td>semi-automatic/semiautomatic weapons ban</td>
</tr>
<tr>
<td>4</td>
<td>The effects of global warming are grossly exaggerated</td>
<td>global warming effects exaggerated</td>
</tr>
<tr>
<td>5</td>
<td>An additional carbon tax on fuel will effectively reduce pollution</td>
<td>carbon tax fuel reduce pollution</td>
</tr>
<tr>
<td>6</td>
<td>The US should never sign international treaties on climate change that limit economic growth</td>
<td>climate change international treaties limit economic growth sign</td>
</tr>
<tr>
<td>7</td>
<td>The US had every right to invade Iraq</td>
<td>invade/invasion Iraq every right</td>
</tr>
<tr>
<td>8</td>
<td>The US is safer because of the invasion of Iraq</td>
<td>safer invade/invasion Iraq</td>
</tr>
<tr>
<td>9</td>
<td>The new president should begin to bring home all US troops from Iraq immediately</td>
<td>new president bring home US troops Iraq immediately</td>
</tr>
<tr>
<td>10</td>
<td>The best way to reduce the federal deficit is to raise taxes</td>
<td>reduce federal deficit raise taxes best way</td>
</tr>
<tr>
<td></td>
<td>The tax cuts for people with a higher income should be reversed</td>
<td>tax cuts higher income reversed</td>
</tr>
<tr>
<td>---</td>
<td>---------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>12</td>
<td>The government has no responsibility to provide retirement funds</td>
<td>Government retirement funds no responsibility</td>
</tr>
<tr>
<td>13</td>
<td>Mortgage lenders should be more tightly controlled</td>
<td>mortgage lenders tightly controlled/control</td>
</tr>
<tr>
<td>14</td>
<td>People with higher incomes should receive less Medicare benefits</td>
<td>higher incomes people receive less medicare/medical care benefits</td>
</tr>
<tr>
<td>15</td>
<td>The federal government should reduce income inequality</td>
<td>Federal government reduce income inequality</td>
</tr>
<tr>
<td>16</td>
<td>The US should reduce its financial contribution to the UN</td>
<td>US reduce financial contribution UN</td>
</tr>
<tr>
<td>17</td>
<td>Iran is not an imminent threat to world peace</td>
<td>Iran imminent threat world peace dangerous</td>
</tr>
<tr>
<td>18</td>
<td>The US should decrease its spending on defense</td>
<td>US decrease spending defense</td>
</tr>
<tr>
<td>19</td>
<td>Same sex marriages should be made legal</td>
<td>same sex marriages made legal</td>
</tr>
<tr>
<td>20</td>
<td>Abortion should be made completely illegal</td>
<td>abortion completely illegal</td>
</tr>
<tr>
<td>21</td>
<td>Using embryos for stem cell research is acceptable as it enables us to find cures for diseases</td>
<td>using embryos stem cell research acceptable good find cures diseases</td>
</tr>
</tbody>
</table>

**Figure 10: Modified policy issue statements and key words**

In the table given above you can clearly see that statements number 2, 6, 12 and 17 were modified. The main modification is converting negative
statement into positive by excluding following words from the sentence - *no, not and never*. This allows using defined policy issue statements and running sentiment analysis on them avoiding twofold results.

3.1.2 VAA 2.0 architecture and design

![Figure 11: Proposed VAA 2.0 Architecture](image)

The conceptual model of VAA 2.0 which reveals general structure and behaviour of the system is given below (see figure 11). The following representation of the system gives an idea how the VAA 2.0 should work. First of all system identifies three types of users, these are adminstrator the only user who is able access admin panel, and candidate and voter. The profile difference of candidate and user is that the first one should provide his/her official twitter account in order to connect with it and extract relevant data. When candidate registers into the system he can either provide his twitter account and request for profile generation, or answer VAA 2.0 questionnaire himself. However, administrator is able to approve and remove requests, but also add another candidate into the system if his/her official twitter account is known.

In the admin panel administrator confirms the candidates id and their twitter accounts, after this system connects to the appropriate twitter accounts using twitter API. This allows extracting last 200 tweets from each
account and analyzing them. The extracted tweets analyzed using sentiment analysis (i.e. probability density function) and ANEW library.

### 3.2 Basic structure

According to the main goals of the project I have decided to use ASP.Net frameworks Web-Forms on C# programming language. The main reason of the following selection is the wide range of opportunities of code-behind programming and user friendly graphical interface implementation simultaneously. Moreover, the current web form supply number of implementation ways of administrator interface, user or candidate interfaces. Another reason is easy and advantageous access to the APIs of different services on the go.

![Figure 12: Main page for VAA 2.0](image)

In the figure above you can see the representation of the main page of the proposed VAA 2.0. New user can start using the system by pressing the button ‘start’ which will direct the user into the registration page. New users have to register by providing personal information, whereas old users who have already been registered can simply log into the system and use VAA several times.
Not registered users have to create their profiles through registration link. First of all, it should be clearly stated either user or candidate going to create profile. So system will sort out candidates and admin will receive requests on creation of candidates profile and will check them, whereas users will be able to go through questionnaire after registration. If a request was not send admin can himself add or remove candidates profile from the system. In the registration panel it is necessary to provide personal information and contact information.

The registration panels for voters and candidates do not have significant differences, both candidate and user (i.e. voter) have to provide personal information (full name, gender, date of birth, password, country of origin) and contact information (email address, phone number) related to them. However, only candidates have to provide twitter account username. This is very important information for candidate to provide because it will be used for further creation of candidates profile in case he/she reject to answer VAA questionnaire.
3.2.1 Voter profile

Voters profile generated after the registration of the user. The voters profile contains following personal information: full name, date of birth, gender, country, email address and phone number. The registration of users is implemented due to following reasons: first, is to track users using VAA 2.0, so the collected data can be used for further research. Second, system should know whether it is candidate or typical user who is going to use VAA 2.0. If candidate is registered in the system it will automatically send request for the generation of their profile, so system will connect to their twitter account after administrators’ review of the requests on the admin panel.

Registered voters can reuse the questionnaire for many times in case they change their point of view for some policy issue statement or in case they made a mistake while filling in the questionnaire. Also their data will be updated in the database when the questionnaire is reused.
3.2.2 Candidate profile

Candidates’ profile does not difference from voters profile even it is used to connect the profile with twitter account to extract data for defining the position of candidate. As typical voter the candidate has to provide personal and contact information as well so administrator can confirm the truthfulness of data and identify real candidate. The requests for creation of candidates profile are sent to admin panel automatically after the registration, where administrator either can confirm or decline it.

From the figure 15 given below it can be clearly seen that profiles representation for candidates and voters are designed in the same way. It contains the information given during registration.
3.2.3 Matching algorithm

After the data were extracted from twitter accounts of candidates their positions are stored in database. Hence, we can use their answers for 21 policy issue statement and match it with the answers of a user. After matching had been calculated, the system will reveal the results of VAA 2.0. The results contain the full name of candidates and the appropriate score for each candidate. The definition of score is also given in the results page, where 0* means that the following candidate is the most closest according to their preferences, then there is number *1, *2 and so on. There is no limit of numbers; it directly depends on number of candidates in database which positions can be used for matching. So here we can clearly see that the system contains information about four different candidates. In the figure 17 given below you can see the results interface given for users after they fill in the questionnaire.
Below you can see part of the code which is responsible for matching the answer of the user with answers of candidates that were extracted from twitter accounts:

```csharp
    for (int i = 0; i < reo; i++)
    {
        double q = 0;
        for (int j = 1; j < 22; j++)
        {
            if (res[i, j] != null)
            {
            }
            else if (res[i, j] == null && i < reo) i++;
            q += resuts[i, j];
        }
        compare[i] += q;
    }

    int indexAtMax = compare.ToList().IndexOf(compare.Max());
    fid = indexAtMax;
    string search = res[fid, 0];

    SqlDataAdapter sda = new SqlDataAdapter("select * from [user] where id = "+ search + "", dbconnection");
    DataTable dt = new DataTable();
    sda.Fill(dt);
    Repeater1.DataSource = dt;
    Repeater1.DataBind();
```
3.2.4 Admin panel

Admin panel is a key part of the VAA 2.0 functionality because it encompasses the whole process of dynamic generation of candidates’ profiles through twitter API usage and sentiment analysis. First of all, admin receives all the requests for the generation of candidate profiles and then approves those requests for further data extraction and profile generation, or admin himself can create new candidate profile without any request.

![Image of Admin Panel](image.png)

**Figure 18: The admin panel**

In the admin panel administrator approve requests, and request contains candidate name, twitter account id and other relevant personal information. The connected twitter account is used to extract the data for finding the position of candidate to the 21 questionnaire statements.

The below are given the most significant functionality of the system, where the system connects to the twitter account of the candidate and extracts data for building the dynamic profile and identifying the position of candidate to proposed 21 policy issue statements:

```csharp
using System;
using System.Linq;
using System.Data.SqlClient;
using System.Web.Configuration;
```
using System.Data;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Threading;
using LinqToTwitter;
using System.Web;

namespace VAA_webform
{
    public partial class admpanel : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            string id = "";
            if (HttpContext.Current.Request.IsAuthenticated)
            {
            }
            if (id != "ulzhan@mail.ru")
            {
                Response.Redirect("~/default");
            }
            if (!IsPostBack)
            {
                var con_string = WebConfigurationManager.ConnectionStrings["myconnection"];  
                using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString))
                {
                    dbconnection.Open();
                    SqlCommand command0 = new SqlCommand("select * from [isok]", dbconnection);
                    DataTable dt = new DataTable();
                    SqlDataAdapter sda = new SqlDataAdapter("select * from [isok]", dbconnection);
                    sda.Fill(dt);
                    Repeater1.DataSource = dt;
                    Repeater1.DataBind();
                }
            }
            protected void getit(object source, RepeaterCommandEventArgs e)
            {
                LinkButton confirm = e.Item.FindControl("confirm") as LinkButton;
            }
            protected void confirm_Click(object sender, EventArgs e)
            {
                string cid1 = ""
            }
        }
    }

    //start
    protected void twit_Click(object sender, EventArgs e)
    {
        foreach (RepeaterItem item in Repeater1.Items)
        {
            var cid = (Label)item.FindControl("cid");
            cid1 = cid.ToString();
        }
    }
}
**SECTION A: Initialise local variables**

```csharp
const string accessToken = "2671318080-eq0D0VdWGM0qjHHWyKnqrp0x59Mjvhfg7DQcb1d";
const string accessTokenSecret = "Lw6VFsaYOSiqv5DPan1HRtGcitq7XYoPMWogqjGfiQwu";
const string consumerKey = "p10YbJmlycuY18fewVkJv9nWN";
const string consumerSecret = "qlBkKI3jNQDllmAQe43imrTUxbpjFcM0wwL6qqlFWRloqh98g";
string aca = twac.Text.Replace(" ", "");
string accid = cid.Text.Replace(" ", "");
string twitterAccountToDisplay = aca;
```

**SECTION B: Setup Single User Authorisation**

```csharp
var authorizer = new SingleUserAuthorizer {
    CredentialStore = new InMemoryCredentialStore {
        ConsumerKey = consumerKey,
        ConsumerSecret = consumerSecret,
        OAuthToken = accessToken,
        OAuthTokenSecret = accessTokenSecret
    }
};
```

**SECTION C: Generate the Twitter Context**

```csharp
var twitterContext = new TwitterContext(authorizer);
```

**SECTION D: Get Tweets for user**

```csharp
var statusTweets = from tweet in twitterContext.Status
    where tweet.Type == StatusType.User &&
    tweet.ScreenName == twitterAccountToDisplay &&
    // tweet.TrimUser== true &&
    tweet.Count == 100
    select tweet;
PrintTweets(statusTweets);
```

`delim1(accid);`
if (data[i].Contains("carbon") || data[i].Contains("tax") || data[i].Contains("reduce") ||
data[i].Contains("pollution")) { temp[4]++; }
if (data[i].Contains("climate") || data[i].Contains("change") || data[i].Contains("international
treat") || data[i].Contains("limit") || data[i].Contains("economic growth");) { temp[5]++; }

//military
if (data[i].Contains("invade") || data[i].Contains("invasion") || data[i].Contains("Iraq");)
data[i].Contains("right") { temp[15]++; }
if (data[i].Contains("safer") || data[i].Contains("invade") || data[i].Contains("Iraq");)
data[i].Contains("invasion"); { temp[16]++; }
if (data[i].Contains("new president") || data[i].Contains("bring") || data[i].Contains("US
troops") || data[i].Contains("Iraq") || data[i].Contains("immediately");) { temp[17]++; }

//economy
if (data[i].Contains("reduce") || data[i].Contains("federal") || data[i].Contains("deficit");)
data[i].Contains("raise") || data[i].Contains("taxes");) { temp[0]++; }
if (data[i].Contains("tax") || data[i].Contains("cuts") || data[i].Contains("higher");)
data[i].Contains("income") || data[i].Contains("funds");) { temp[2]++; }
if (data[i].Contains("government") || data[i].Contains("retirement") || data[i].Contains("no responsibility");) { temp[2]++; }

//income
if (data[i].Contains("mortgage") || data[i].Contains("lenders") || data[i].Contains("lenders");)
data[i].Contains("control") || data[i].Contains("tightly");) { temp[12]++; }
if (data[i].Contains("higher income") || data[i].Contains("people");)
data[i].Contains("receive") || data[i].Contains("medical") || data[i].Contains("benefits");) { temp[13]++; }
if (data[i].Contains("Federal") || data[i].Contains("government") || data[i].Contains("reduce");)
data[i].Contains("income") || data[i].Contains("inequality");) { temp[14]++; }

//National
if (data[i].Contains("US") || data[i].Contains("financial") || data[i].Contains("contribution");)
data[i].Contains("UN") || data[i].Contains("contrib");) { temp[18]++; }
if (data[i].Contains("Iran") || data[i].Contains("imminent") || data[i].Contains("world peace");)
data[i].Contains("dangerous") || data[i].Contains("threat");) { temp[19]++; }
if (data[i].Contains("US") || data[i].Contains("decrease") || data[i].Contains("spending") ||
data[i].Contains("defense") || data[i].Contains("defense");) { temp[20]++; }

//Family
if (data[i].Contains("same sex") || data[i].Contains("marriage") || data[i].Contains("made");)
data[i].Contains("legal") || data[i].Contains("family") { temp[6]++; }
if (data[i].Contains("abortion") || data[i].Contains("completely") || data[i].Contains("illegal");)
temp[7]++; }
if (data[i].Contains("using embryos") || data[i].Contains("stem cell") ||
data[i].Contains("research") || data[i].Contains("diseases") || data[i].Contains("acceptable");) { temp[8]++; }

indexAtMax = temp.ToList().IndexOf(temp.Max());
if (finres[indexAtMax, 0] == null && finres[indexAtMax, 1] == null && finres[indexAtMax, 2] ==
null) { finres[indexAtMax, 0] = data[i]; i++; }
else if (finres[indexAtMax, 0] != null && finres[indexAtMax, 1] == null && finres[indexAtMax, 2] ==
nul;
else if (finres[indexAtMax, 1] == data[i]; i++; )
data[i]; i++; )
while (data[i] != null);

var con_string = WebConfigurationManager.ConnectionStrings["myconnection"]; using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString))
{
    int t = 0;
    for (int ii = 0; ii < 21; ii++)
    {
        int tt = t;
        for (int j = 0; j < 3; j++)

41
if (finres[ii, j] != null) {
    string[] check = finres[ii, j].Split();
    foreach (string q in check) {
        dbconnection.Close();
        dbconnection.Open();
        SqlCommand command1 = new SqlCommand("select * FROM [dictionary] where word = @word1", dbconnection);
        if (dbconnection.State == ConnectionState.Open) {
            command1.Parameters.Add("word1", SqlDbType.Char, 50).Value = q;
        }
        SqlDataReader reader1 = command1.ExecuteReader();
        while (reader1.Read()) {
            string w = (string)reader1["word"];
            double m1 = (double)reader1["mean1"];
            double d1 = (double)reader1["deviation1"];
            double m2 = (double)reader1["mean2"];
            double d2 = (double)reader1["deviation2"];
            ret[t, 0] = w;
            ret[t, 1] = m1.ToString();
            ret[t, 2] = d1.ToString();
            ret[t, 3] = m2.ToString();
            ret[t, 4] = d2.ToString();
            t++;
        }
        reader1.Close();
        dbconnection.Close();
    }
} else if (finres[ii, 0] != null && j > 0) {
    string res = "";
    double xcoor = 0;
    double ycoor = 0;
    double[,] end = new double[t - tt, 8];
    for (int a = tt; a < t; a++) {
        double m1_ = double.Parse(ret[a, 1]);
        double d1_ = double.Parse(ret[a, 2]);
        double m2_ = double.Parse(ret[a, 3]);
        double d2_ = double.Parse(ret[a, 4]);
        double pi = (1) / (Math.Sqrt(2 * 3.14 * d1_ * d1_));
        double pi2 = (1) / (Math.Sqrt(2 * 3.14 * d2_ * d2_));
        end[a, 1] = pi;
        end[a, 2] = pi2;
    }
    double pisum1 = 0;
    double pisum2 = 0;
    for (int a = tt; a < t; a++) {
        pisum1 += end[a, 1];
        pisum2 += end[a, 2];
        end[a, 3] = double.Parse(ret[a, 3]) / pisum1;
        end[a, 4] = double.Parse(ret[a, 4]) / pisum1;
        end[a, 5] = double.Parse(ret[a, 1]) * pisum1;
        end[a, 6] = double.Parse(ret[a, 3]) * pisum2;
xcoor += end[a, 5];
ycoor += end[a, 6];
}
if (xcoor > 5 && ycoor > 5) { res = "-0.5"; }
if (xcoor > 5 && ycoor < 5) { res = "-1"; }
if (xcoor < 5 && ycoor < 5) { res = "1"; }
if (xcoor < 5 && ycoor > 5) { res = "0.5"; }

string qn = (indexAtMax + 1).ToString();
dbconnection.Open();
SqlCommand command = new SqlCommand("UPDATE dAnswers SET question" + qn + ": @que where did=@id", dbconnection);
// SqlCommand command1 = new SqlCommand("UPDATE isok SET truly 'yes' where id=@id", dbconnection);
if (dbconnection.State == ConnectionState.Open)
{
    // command1.Parameters.Add("id", SqlDbType.Char, 50).Value = accid;
    command.Parameters.Add("que", SqlDbType.Char, 50).Value = res;
    command.Parameters.Add("id", SqlDbType.Char, 50).Value = accid;
}
command.ExecuteNonQuery();
// command1.ExecuteNonQuery();
dbconnection.Close();
else if (finres[ii, j] == null && ii < 21 && j<1) { j++; }
else if (finres[ii, j] == null && ii < 21 && j == 2) { j = -1; ii++; }
}
}
}

private static void PrintTweets(IQueryable>Status> statusTweets)
{
    try {
        int i = 0;
        foreach (var statusTweet in statusTweets)
        {
            Console.WriteLine(string.Format("\n\nTweet From [{0}] at [{1}]: \n-{2}\n", statusTweet.ScreenName, statusTweet.UserID, statusTweet.Text));
            Thread.Sleep(1000);
dataft[i] = statusTweet.Text.Replace("/n", " ");
i++;
        }
    }
    catch { }
}

static string[] delim = new string[21, 100];
static string[] dataft = new string[100];
static string[] finres = new string[21, 3];
//end
protected void confirm_Click1(object sender, EventArgs e)
{
}
protected void confirm_Click2(object sender, EventArgs e)
After the connection to the twitter account of each candidate the system extracts the answers to proposed 21 policy issue statements for each candidate. The answers of candidates are stored in the database for further usage in the system. In the figure below you can clearly see how the extracted data stored. The ‘Did’ stands for Database id which reveals the id of each registered candidate (the email address of the candidate). In the columns ‘question1, question2 ... question21’ the answers extracted from the twitter accounts are stored for every candidate appropriately. The numbers ‘-1, -05, 0, 0.5, 1’ are matched to policy issues statements answers and show their
weight for the directional model of matching algorithm. Completely agree equals to ‘1’, agree is ‘0.5’, neutral is ‘0’, disagree is ‘-0.5’, and completely disagree ‘-1’.

3.3 Sentiment analysis

Sentiment is referred to the number of words such as “an attitude, thought or judgment prompted by feeling” [12]. Thus Sentiment Analysis is the process which defines the attitude towards expression by identifying whether it is positive, negative or neutral. Moreover, Sentiment Analysis sometimes called Opinion Mining, which is intended to get the opinion of the speakers regarding the word or expression produced by them. Sentiment Analysis (or Opinion Mining) usually used to study how human beings feel about some particular themes.

In our project we are using the twitter as a main source of data, hence it is extremely important to analyze the sentiment of the tweets to identify the attitude of candidate towards policy issue statements (topics posted on twitter). The length of tweets posted on Twitter is usually comes up to 140 characters.

This length of tweets (140 characters per tweets) encourages users to write constructive, focused, timely posts. However, the number of tweets for particular topic could be enormous. Therefore, we are considering only tweets of specific candidate taken from his own twitter account to generate the statements for policy issues to generate candidates’ profile.
3.3.1 Representing emotion

There are some kinds of emotional models introduced in psychology which outline and compare different emotional states. In order to find the placement and the position of the emotions on two dimensional plane emotional models usually apply emotional dimensions. Generally the most understandable models reveal pleasure along a horizontal axis where the most unpleasant emotions lay on one end and the most pleasant on the other end, and there are number of different degrees of pleasure in between. This is how simple models recognized, whereas more complex models apply more than one dimension. For instance, Russell introduced new way to create an emotional circumflex of affect by using valence and arousal. Valence is so called pleasure, and arousal is activation. Russell’s’ model uses 28 emotional states which are positioned by multidimensional scaling. This model can be described in the following way: the valence (pleasure) given on the horizontal axes and arousal (activation) given on the vertical axes. The intermediate terms such as excited and depressed, distressed and relaxed are polar opposites created by intermediate states of valence and arousal (see on figure 12).

![Russell model of emotional affect](image)

*Figure 20: Russell model of emotional affect. Drawn from the original Russell and James (1980) paper.*
3.3.2 Sentiment dictionary

In order to run a concept-level analysis of natural language text common computational methods apply machine learning algorithms. Hence acceptable high-quality text that allows producing accurate NLP evaluations is the main requirement. Today we can clearly note the argues about not significant availability of above mentioned high-quality texts in short twitters texts so called tweets, or SMS (or instant) messages. Therefore, some researches proposed an alternative such as using dictionaries which describe the sentiment of group of words along 1 or more emotional dimensions. For instance there are few sentiment dictionaries available such as POMS that stands for Profile of Mood States and ANEW that stands for Affective Norms of English Words. From these two dictionaries we have decided to use last one because ANEW provides freely the word list as part of its instruction manual.

The instruction manual of ANEW dictionary includes all necessary information that we need for our sentiment analysis. This dictionary gives three types of measures such as valence, arousal and dominance for 1,034 English words, and each given word is rated from 1 to 9. Words in the ANEW dictionary were taken from previous research which defined to be appropriate for transmitting emotions. In general, for the formulation of dictionary number of different volunteers were given a text composition and a rating along each dimension for each raised ANEW recognizable word. Then volunteers have to read the given text and put an appropriate rating on (from 1 to 9). Ratings given for a common word were coupled to generate one mean rating and a standard deviation of the ratings for every dimension. For instance, the word “gun” is given on ANEW dictionary in the following way:

\[\text{gun } v = (\mu: 3.47, \sigma: 2.48), a = (\mu: 7.02, \sigma: 1.84), d = (\mu: 3.53, \sigma: 2.72), \text{fq} = 118.\]

This reveals that “gun” has a mean valence v of 3.47 and standard deviation of 2.48, a mean arousal a of 7.02 and a standard deviation of 1.84, a mean dominance d of 3.53 and a standard deviation of 2.72, and a frequency fq of 118 ratings.

The decision to use ANEW’S valence and arousal measurements to make estimation of the pleasure and arousal of tweet’s text were made. ANEW dictionary has been chosen upon POMS due to several advantages, such as its word list is publically available and it provides number of emotional
dimensions. Moreover, it gives opportunity to characterize sentiment in more practically significant ways instead of a using common positive–negative rating.

Consequently we have used these steps for each tweets to make estimation of valence and arousal:

1. Obtain the tweet’s body by using Twitter’s Search API.
2. For every word \( w_i \) captured in the tweet that is found in the ANEW dictionary, save the word’s appropriate values such as mean valence and arousal (\( \mu_{v,i} \) and \( \mu_{a,i} \)), standard deviation of valence and arousal (i.e. \( \sigma_{v,i} \) and \( \sigma_{a,i} \)).
3. In case tweet has less than \( n=2 \) ANEW words, ignore it as having not significant amount of ratings to estimate its sentiment.
4. Statistically average the \( n \) means and standard deviation to calculate the tweet’s overall mean valence and arousal values (i.e. \( M_v \) and \( M_a \)).

For instance, consider this tweet from the query of key words dedicated to find tweets related to the second policy issue statement (see figure 10) containing the four ANEW words such as “gun, control, save, achieve”. Words in bold italics are words that were found in the ANEW dictionary. The tweet found from twitter account of Hillary Clinton @HillaryClinton:

“Another devastating shooting. We need sensible **gun control** measures to **save** lives, and I will do everything I can to **achieve** that. –H”.

The ANEW’s measure of the \( n = 4 \) words’ means and standard deviations measurements of valence and arousal are following:

- **gun**, \( v = (\mu: 3.47, \sigma: 2.48), a = (\mu: 7.02, \sigma: 1.84), \text{fq} = 118. \)
- **controlling**, \( v = (\mu: 3.38, \sigma: 2.17), a = (\mu: 6.38, \sigma: 2.21), \text{fq} = 23 \)
- **save**, \( v = (\mu: 6.65, \sigma: 2.03), a = (\mu: 4.80, \sigma: 1.99), \text{fq} = 62 \)
- **achieve**, \( v = (\mu: 7.89, \sigma: 1.38), a = (\mu: 5.53, \sigma: 2.81), \text{fq} = 65 \)

In order to merge the mean values for above listed terms such as **gun**, **controlling**, **save** and **achieve**, we consider that the respective ratings given for every term form a normal distribution. Generally, in case of higher standard deviation of a term for valence \( \sigma_{v,i} \), the valence rating for the term would reach a wider range of values, whereas in case of lower standard deviation of a term for valence \( \sigma_{v,i} \), the valence ratings for the term would significantly reach the mean. According to this it is necessary to apply
probability density function to calculate the probability density of the term’s rating laying at the mean value precisely [12]. The probabilities \((p_i)\) are given as weights for every term when the mean values are summarized.

Using the probability density function we can calculate the overall weighted average mean valence and arousal: \(M_v = 5.66\) and \(M_a = 5.84\) appropriately.

### 3.3.3 Estimating sentiment

In order to estimate sentiment we consider analysis using ANEW dictionary. We use ANEW dictionary to run series of actions on a tweets’ text body in the consequent way:

First, break down the text body into components such as terms, then calculate the \(n\) number of ANEW terms \(\{t_1, \ldots, t_n\}\) that match entries in the ANEW dictionary.

Second, for every matched term \(t_i\) extract the average mean value and standard deviation for valence \((v_i, \mu_i, v_i, \sigma_i)\) and arousal \((a_i, \mu_i, a_i, \sigma_i)\) from the dictionary.

Third, in case a text body contains less than \(n=2\) ANEW terms, then eliminate this tweet because it has not enough measurements to evaluate an overall sentiment.

Fourth, in case a text body contains equal to or more than \(n \geq 2\) ANEW terms, then aggregate these individual term measurements to calculate an overall average and standard deviation for valence \((V_\mu, V_\sigma)\) and arousal \((A_\mu, A_\sigma)\).

In order to calculate an overall standard deviation from the group of term average and standard deviations pairs \((\mu_i, \sigma_i)\) we have applied a formula for averaging standard deviations. For instance, for \(V_\sigma\) it is:

\[
M = \frac{1}{n} \sum_{i=1}^{n} v_i, \quad V_\sigma^2 = \left(\frac{1}{n} \sum_{i=1}^{n} v_i^2 + M^2\right) - M^2
\]

Calculating the overall average of individual term averages is more complicated. A simple unweighted average can be used, but in this case every term’s standard deviation \(v_i, \sigma_i\) is not taken into account. A large deviation \(v_i, \sigma_i\) indicates the vague ratings of a term by respondents. The term can be pronounced the same as another, regardless of its meaning or spelling, for
example we can meet following homonym words: “lie” which means not
telling the truth or to recline. Also, the word can be used in different context
such as: when a word has direct view “I am sad” and when a word used as
contradiction “I am not sad”.

It seems that the larger valence deviation $v_{i\alpha}$, the less we aim to assign
mean valence in the overall average, because we are not completely sure
where exactly its true average lies. Therefore, to achieve this we compute a
word’s cumulative distribution function – $p$, then apply the probability at $p(v_{i\mu})$
to weight $v_{i\mu}$ in the overall average. This assigns a lower weights to words
with larger $v_{i\alpha}$.

The cumulative distribution function is a so called the normal curve, and
its height with $\mu = v_{i\mu}$ and $\sigma = v_{i\sigma}$ at $x = v_{i\mu}$ is:

$$
 p_i = \frac{1}{\sqrt{2\pi v_{i\sigma}^2}}
$$

Above presented $p_i$ is for every term $t_i$, we normalize the $p_i$'s and then
compute a final weighted average in the following way:

$$
 V_{i\mu} = \sum_{i=1}^{n} \frac{p_i}{\sum_{i=1}^{n} p_i} v_{i\mu}
$$

Consider the following tweet "Another devastating shooting. We need
sensible **gun control** measures to **save** lives, and I will do everything I can to
**achieve** that. –H" with four ANEW words such as “**gun**”($v_{\mu}$=3.47,
$v_{\sigma}$=2.48; $a_{\mu}$=7.02, $a_{\sigma}$=1.84), “**controlling**” ($v_{\mu}$=3.38, $v_{\sigma}$=2.17; $a_{\mu}$=6.38, $a_{\sigma}$=2.21),
“**save**” ($v_{\mu}$=6.65, $v_{\sigma}$=2.03; $a_{\mu}$=4.80, $a_{\sigma}$=1.99), “**achieve**”($v_{\mu}$ =7.89,
$v_{\sigma}$=1.38; $a_{\mu}$=5.53, $a_{\sigma}$=2.81).

First we calculate every term’s cumulative distribution function - $p_i$ (the
normal curve):
After calculating $p_i$ for every term $t_i$, we normalize the $p_i$ for calculating the final weighted average:

\[
p_{\text{gun}} = \frac{1}{\sqrt{2\pi \sigma_{\text{gun}}^2}} = 0.16
\]
\[
p_{\text{cont.}} = \frac{1}{\sqrt{2\pi \sigma_{\text{cont.}}^2}} = 0.18
\]
\[
p_{\text{save}} = \frac{1}{\sqrt{2\pi \sigma_{\text{save}}^2}} = 0.19
\]
\[
p_{\text{ach}} = \frac{1}{\sqrt{2\pi \sigma_{\text{ach}}^2}} = 0.28
\]

An unweighted average of the mean valence $\nu_i$'s gives $V_\mu = 5.34$, however due to fact that the standard deviation for the term "controlling" is higher than for the term "save", $\nu_{\text{cont.},\mu}$ assigned a weight of $0.18 / 0.82 = 0.21$, while $\nu_{\text{save},\mu}$ given a weight of $0.19 / 0.82 = 0.23$. The same tension with other standard deviation of terms can be seen as standard deviation is higher for term then the weight assigned is appropriately higher for the term. Hence, $\nu_{\text{gun},\mu}$ assigned a weight of $0.16 / 0.82 = 0.19$, $\nu_{\text{ach},\mu}$ given a weight of $0.28 / 0.82 = 0.34$. Finally, this gives us a weighted average $V_\mu = 5.66$ which is more accurate and falls.
Then the same calculations run for $A_\mu$ using an overall standard deviation from the group of term average and standard deviations pairs ($\mu_i, \sigma_i$) we have also applied a formula for averaging standard deviations. The calculations outcome is following: $a_{\text{gun},\mu}$ assigned a weight of $\frac{0.21}{0.73} = 0.28$, $a_{\text{cont},\mu}$ is given a weight of $\frac{0.18}{0.73} = 0.24$, $a_{\text{save},\mu}$ receives a weight of $\frac{0.20}{0.73} = 0.27$, and $a_{\text{ach},\mu}$ gets a weight of $\frac{0.14}{0.73} = 0.19$. From this we have come to the final weighted average $A_\mu = 5.84$, whereas unweighted average of the mean arousal $a_\mu$'s gives $V_\mu = 5.93$.

3.3.4 Generating statements for dynamic profiles

![Graph](image)

Figure 21: Example of Russell models to be applied for visualizing the tweet’s estimated sentiment

From the previous calculations we have derived the tweet’s sentiment estimation ($V_\mu = 5.66$, $A_\mu = 5.84$) which is placed on the graph given above. Following graph illustrates how Russell’s model had been applied to visualize tweet’s sentiment, also to be noted the ratings for the word estimation is given in range from 1 to 9.
In this case, after estimating every tweets sentiment related to the specified given policy issue statement separately, we can see the picture where different tweets lay on the visualization map. The negative tweets are labeled with red dot, because emotions on left side incorporate negative emotions such as bored, depressed, unhappy, sad, upset, stressed, nervous, and tense, whereas positive ones labeled with blue dot, because they are on the right side which represents positive emotions only such as calm, relaxed, serene, contended, happy, elated, excited, and alert, and neutral tweets are those which lay right on the middle of the visualization map and labeled region with green color (see figure 22). So, typically if the tweet given by the candidate describes the second issue (the tweet found due to the appropriately matched key words) and its sentiment values is following ($V_\mu = 5.66, A_\mu = 5.84$) (see figure 21), then the tweet is negative according to the proposed map.

![Visualization Map](image)

**Figure 22: The negative and positive polar of the visualization map**

However, we should define how this tweet describes the attitude of the candidate towards policy issue statement because the negative and positive attention cannot be matched with VAA answers (CA, A, N, D, CD). We should define whether he/she is completely agree (CA), agree (A), neutral (N),
disagree (D), or completely disagree (CD). Therefore, we decided to define the regions of the visualization map which reveals the tweets sentiment to match with the appropriate VAA answer given for each policy issue statement. And we have proposed following map divided into five different regions, where every region incorporates tweets which sentiments best match to the VAA answer types. So we can clearly see that the tweets sentiment given on figure 21 can be matched with (D), which stands for ‘disagree’ in VAA. Hence, we can assume that the candidate is revealing his attitude (D-disagree) to the second policy issue statement by his tweet which had been analyzed for sentiment identification. So, we can clearly admit that candidates “says” disagree to the VAA statement by his tweet. However, we should take into account that the number of tweets related to this topic could be significantly more than one or two. Hence, we should take into account other tweets related to the exact policy issue statement.

![Visualization Map with CA, A, N, D, CD Regions](image)

*Figure 23: The visualization map with CA, A, N, D, CD regions*

If the number of tweets related to the policy issue statement is more than one, which we can strongly assume, because today our examined candidates has enormous number of tweets posted. For instance, the twitter profile of Hillary Clinton shows over five thousand posted tweets, whereas Barack
Obama has posted almost fifteen thousand tweets and Bernie Sanders posted over fourteen thousand tweets till May 2016 and other candidates shows significant amount of tweets posted on their official accounts. This tells us that the probability of the number of tweets related to the policy issue statement more than one is huge. Therefore, we should summarize the 'tweets answers' and generate one answer per each policy issue statement.

For instance, we have filtered out 10 tweets related to the second policy issue statement which is described above. Assume, that the sentiments of these ten tweets had been estimated and given in the visualization graph (see figure 24). From the visualization graph given below we can clearly see that 1 tweet is for ‘completely agree’, 4 tweets stand for ‘agree’, 1 tweet tends to be ‘neutral’, 2 tweets are for ‘disagree’ and also 2 tweets are for ‘completely disagree.’

![Visualization Graph](image)

**Figure 24: The example of estimated tweets**

First calculated is following: \((A) = 4; (CA) = 1; (N) = 1; (D) = 2; (CD) = 2;\)

Second, the number of overall positive, negative and neutral tweets calculated: \((POS) = (CA) + (A) = 5; (NEG) = (CD) + (D) = 4; (NEUT) = (N) = 1;\)
Third, we compare the two different types of tweets which have the most number of tweets, so in this case the (N) is disregarded to be not sufficient amount. Hence, we compare (POS) and (NEG), and receive this: (POS) > (NEG), which tells us that there are more positive tweets than negative ones.

Fourth, therefore we consider the positive tweets only for the extraction of answer for VAA statement. As we can remember, there are (A) = 4; (CA) = 1; We compare the number of tweets standing for (A) and (CA). From this we can assume that (A) > (CA).

Finally, the attitude of the candidate to the specified policy issue statement is (A) according to the data extracted from the tweets related to the topic of statement. We can assign ‘agree’ to the statement number two for the appropriate candidate, in our example we have considered the twitter account of Hillary Clinton. So, typically we have defined the attitude of the candidate regarding the defined policy issue statement using her twitter account, related tweets and sentiment analysis.

![Figure 25: Proposed Twitter API and Sentiment analysis usage](image-url)
In the figure 25 you can see the structure of above described and proposed Twitter API and sentiment analysis usage for the generation of dynamic profiles of candidates for VAA 2.0. In general the data extraction process starts with the search of tweets related to each specified policy issue statement. The search is done using key words defined for every statement separately. After finding the related tweets and checking the number of terms found in ANEW library, which has to be more than \( n > 1 \), the individual term’s values aggregated in order to estimate an overall average and standard deviation for valence and arousal \( (V_{\mu}, V_{\sigma}; A_{\mu}, A_{\sigma}) \) to define the sentiment of tweets. Then the number of positive, negative and neutral tweets is calculated according to the previous estimations given on visualization graph. After this the attitude of candidates defined based on the estimated tweets and appropriate answer \( (CA, A, N, D, CD) \) is assigned per each statement for every candidate.
Evaluation

4.1 Dynamically generated data analysis

The main purpose of the system is to build an independent VAA with dynamic profiles of candidates. Usually the positions of candidate’s for policy issue statements are taken from variety of sources such as interviews, speeches, discussion and many other written data sources by experts. First of all, this kind of data extraction is time consuming, and second it can be biased by experts for personal reasons or attitudes towards specific candidate. Moreover, the data taken from official interviews may contain information that candidate wants to show, but which doesn’t really reveal their attitude or emotional relevance towards the issues given in VAA’s. Therefore, twitter accounts which is widely used by huge amount of people and political candidates also is a reach source of trustful information which show real emotional attitude about different issues because tweets are limited and many tweets always up to date, which also allows to get changed point of view of any candidate if he/she is registered and active twitter user. For today, we know that American politicians are active users of twitter, whereas European politicians are less active, this is mainly the reason we are aiming to evaluate
our application using real twitter accounts of real politicians, and up to date policy issue statements covering modern issues discussed and viewed by these politicians.

The only disadvantage of using twitter API for data extraction is that it can review and extract only last 200 tweets from every candidates' twitter profile. However, this also means that we can update the information as soon there are more new tweets posted by candidates.

In the figure 26 below are given results of extracting first 100 tweets for every candidate, then these 100 tweets went through sentiment analysis that were described in previous chapter of thesis which allowed us to extract the position of candidate towards 21 policy issue statement. The numbers in the table represent the values given for every type of answer such as ‘1’ stands for ‘completely agree’, ‘0,5’ is ‘agree (i.e. tend to agree)’, ‘0’ is ‘neutral’, ‘-0,5’ is ‘disagree (i.e. tend to disagree)’ and ‘-1’ is ‘completely disagree’. NULL means that tweets related for the following policy issue statement were not found and cannot be analyzed. This means that candidates did not posted tweets related to these topics and there is ‘no answer’ for the question(s).

<table>
<thead>
<tr>
<th>Question</th>
<th>@BarackObama</th>
<th>@SenJohnMcCain</th>
<th>@Hillary Clinton</th>
<th>@MittRomney</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Question 2</td>
<td>NULL</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Question 3</td>
<td>NULL</td>
<td>1</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 4</td>
<td>1</td>
<td>NULL</td>
<td>NULL</td>
<td>1</td>
</tr>
<tr>
<td>Question 5</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 6</td>
<td>1</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 7</td>
<td>NULL</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>Question 8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 9</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 10</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 11</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 12</td>
<td>NULL</td>
<td>1</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 13</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 14</td>
<td>1</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 15</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 16</td>
<td>NULL</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Question 17</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 18</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>NULL</td>
</tr>
<tr>
<td>Question 19</td>
<td>-0,5</td>
<td>-0,5</td>
<td>-0,5</td>
<td>-0,5</td>
</tr>
<tr>
<td>Question 20</td>
<td>NULL</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Question 21</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

*Figure 26: The raw data extracted using 100 tweets*
In the figure 27 the numerical results converted into appropriate type of answer. Now it can be clearly seen what are the positions of candidates to each VAA 2.0 question, are they completely agree, agree, neutral, disagree or completely disagree. Also, ‘no answer’ highlighted with yellow show the number of questions that doesn’t have answers from all four candidates. The amount of statements without appropriate answer is quite significant, which is 7 statements out of overall 21 statements. This means that one third of policy issue statements is lacking the positions of candidates. Therefore, I have decided to use maximum number of tweets that can be extracted using twitter API such 200 tweets from accounts of each candidate.

<table>
<thead>
<tr>
<th>Question</th>
<th>@BarackObama</th>
<th>@SenJohnMcCain</th>
<th>@Hillary Clinton</th>
<th>@MittRomney</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 2</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 3</td>
<td>No answer</td>
<td>CA</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 4</td>
<td>CA</td>
<td>No answer</td>
<td>No answer</td>
<td>CA</td>
</tr>
<tr>
<td>Question 5</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 6</td>
<td>CA</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 7</td>
<td>No answer</td>
<td>CD</td>
<td>CD</td>
<td>CD</td>
</tr>
<tr>
<td>Question 8</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
<td></td>
</tr>
<tr>
<td>Question 9</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 10</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 11</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 12</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 13</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 14</td>
<td>CA</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 15</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 16</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 17</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 18</td>
<td>N</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 19</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>Question 20</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 21</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
</tbody>
</table>

| 8 out of 21 | 11 out of 21 | 10 out of 21 | 7 out of 21 |

Figure 27: The modified data extracted using 100 tweets

The results of analyzed 200 tweets given in the table 3 differ from results that analyzed only 100 tweets. The number of policy issue statements without answers decreased essentially. So, if we had 7 statements absolutely free for all four candidates, now there is only 1 statement (question 13) that does not have an answer for all four candidates. There is at least an answer from one candidate for every statement. The overall number of statements with ‘no
answer’ is 36, in first trial with 100 tweets the number of statements with ‘no answer’ was 48. So, we can see that the number decreased.

This comparison show that if more tweets analyzed then more topics can be viewed and more data extracted. I believe that the limitation of 200 tweets makes is difficult to calculate all data posted on twitter. I assume if the system could compute all tweet from every account we would get appropriate answers for almost all policy issue statements.

The answers highlighted with red show that before there were ‘no answer’ for this statement, but now with increasing number of tweets the system extracted data related to this policy issue statement. So, we can see that 13 answers were computed from increasing number of tweets, and 4 answer were modified.

<table>
<thead>
<tr>
<th>Question</th>
<th>@BarackObama</th>
<th>@SenJohnMcCain</th>
<th>@Hillary Clinton</th>
<th>@MittRomney</th>
</tr>
</thead>
<tbody>
<tr>
<td>Question 1</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 2</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 3</td>
<td>No answer</td>
<td>CA</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 4</td>
<td>CA</td>
<td>No answer</td>
<td>No answer</td>
<td>CA</td>
</tr>
<tr>
<td>Question 5</td>
<td>CA (no answer b.)</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 6</td>
<td>CA</td>
<td>CA (no answer b.)</td>
<td>CA (no answer b.)</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 7</td>
<td>CA (no answer b.)</td>
<td>CA (CD b.)</td>
<td>CA (CD b.)</td>
<td>CA (CD b.)</td>
</tr>
<tr>
<td>Question 8</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 9</td>
<td>CA (no answer b.)</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 10</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 11</td>
<td>No answer</td>
<td>CA (no answer before)</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 12</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 13</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 14</td>
<td>CA</td>
<td>CA (no answer b.)</td>
<td>CA (no answer b.)</td>
<td>CA (no answer b.)</td>
</tr>
<tr>
<td>Question 15</td>
<td>CA (no answer b.)</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 16</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 17</td>
<td>CA (no answer b.)</td>
<td>No answer</td>
<td>No answer</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 18</td>
<td>NULL (N before)</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
</tr>
<tr>
<td>Question 19</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>Question 20</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>Question 21</td>
<td>No answer</td>
<td>CA (no answer b.)</td>
<td>No answer</td>
<td>CA (no answer b.)</td>
</tr>
<tr>
<td>12 out of 21</td>
<td>15 out of 21</td>
<td>12 out of 21</td>
<td>9 out of 21</td>
<td></td>
</tr>
</tbody>
</table>

Figure 28: The data extracted using 200 tweets

The assumption that analysis should be done on more than 200 tweets per account was correct due to the following results shown in the table below:
200 tweets per twitter account are not enough to build absolutely reliable profile of the candidate due to many reasons. First of all, candidate may post biased tweets with controversial viewpoints, or he/she can change their mind regarding some issues and post new information. Second, the twitter posts containing information about 13\textsuperscript{th} policy issue statement for instance might be discussed earlier (i.e. 200 tweets before) and this issue is no longer up-to-date or replaced by other newly posted tweets.

The results shown on tables 26-28 is done only once covering last 200 tweets posted on twitter accounts of candidates. In the table 29 you can clearly see the results of sentiment analysis and data retrieving done two weeks later than first data retrieving. From this it can be clearly assumed that during two weeks candidates updated their twitter profiles and added new tweets. However, the activity of candidates varies which is important because it effects on data analysis and development of VAA profiles of candidates; as it can be noticed from twitter profiles Barack Obama and Hillary Clinton is more active candidates on twitter than John McCain and Mitt Romney.
In the table 29 there are new 7 answers to policy issue statements generated using new twitter posts. This means that candidates posted at least two tweets related to this exact policy issue statements during this period of time. Moreover, they have also posted tweets controversial with those posted before or posts with negative/positive meaning, therefore the answers to 8 policy issue statements were changed. I believe, if the system will update the profiles of candidate every week for some period of time the answers for all policy issue statements will be generated. Moreover, new tweets containing controversial data will be also counted which will allow to build even more reliable profile. The dynamicity of the VAA allows to run data collection every week or two, but for now it cannot analyze data collected every weeks by calculating the average, it can update those data collected previously. However, these changes to the VAA system can be done in the future.

4.2 Comparison of data (experts’ data vs. twitter data)

Voting advice application ‘Electoral compass USA’ gives users the datasets that were analyzed by experts in order to extract candidate’s positions to policy issue statements. These datasets are issues and plans from candidate’s campaign website, debate transcripts, speech transcripts, website candidate’s former or current post/roll call behavior, candidate’s book, columns or other publications, and other websites and news sources. Whereas our VAA 2.0 uses datasets from official twitter accounts of candidate’s such as @BarackObama, @SenJohnMcCain, @HillaryClinton and @MittRomney.

In the table 30 the comparison of data extraction results given where is only few similar answers both from ‘expert’s analyses and ‘twitter analyses are stated. Significant number of answers is vice versa, and also the number of ‘no answer’ is considerable.

<table>
<thead>
<tr>
<th>Question</th>
<th>@BarackObama</th>
<th>@SenJohnMcCain</th>
<th>@Hillary Clinton</th>
<th>@MittRomney</th>
</tr>
</thead>
<tbody>
<tr>
<td>Twitter</td>
<td>Expert</td>
<td>Twitter</td>
<td>Expert</td>
<td>Twitter</td>
</tr>
<tr>
<td>1</td>
<td>CA</td>
<td>CA</td>
<td>CD</td>
<td>CA</td>
</tr>
<tr>
<td>2</td>
<td>No answer</td>
<td>CD</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>3</td>
<td>No answer</td>
<td>CA</td>
<td>CD</td>
<td>No answer</td>
</tr>
<tr>
<td>4</td>
<td>CA</td>
<td>CD</td>
<td>No answer</td>
<td>CD</td>
</tr>
<tr>
<td>5</td>
<td>CA</td>
<td>D</td>
<td>No answer</td>
<td>CD</td>
</tr>
<tr>
<td></td>
<td>CA</td>
<td>CD</td>
<td>CA</td>
<td>N</td>
</tr>
<tr>
<td>---</td>
<td>----</td>
<td>----</td>
<td>----</td>
<td>---</td>
</tr>
<tr>
<td>7</td>
<td>CA</td>
<td>CD</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>8</td>
<td>CA</td>
<td>CD</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>9</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
<td>CD</td>
</tr>
<tr>
<td>10</td>
<td>CA</td>
<td>A</td>
<td>CA</td>
<td>CD</td>
</tr>
<tr>
<td>11</td>
<td>No answer</td>
<td>A</td>
<td>CA</td>
<td>CD</td>
</tr>
<tr>
<td>12</td>
<td>No answer</td>
<td>CD</td>
<td>CA</td>
<td>A</td>
</tr>
<tr>
<td>13</td>
<td>No answer</td>
<td>A</td>
<td>No answer</td>
<td>A</td>
</tr>
<tr>
<td>14</td>
<td>CA</td>
<td>A</td>
<td>CA</td>
<td>A</td>
</tr>
<tr>
<td>15</td>
<td>CA</td>
<td>CA</td>
<td>No answer</td>
<td>D</td>
</tr>
<tr>
<td>16</td>
<td>No answer</td>
<td>D</td>
<td>CA</td>
<td>D</td>
</tr>
<tr>
<td>17</td>
<td>CA</td>
<td>D</td>
<td>No answer</td>
<td>D</td>
</tr>
<tr>
<td>18</td>
<td>No answer</td>
<td>CD</td>
<td>CA</td>
<td>A</td>
</tr>
<tr>
<td>19</td>
<td>D</td>
<td>A</td>
<td>D</td>
<td>CD</td>
</tr>
<tr>
<td>20</td>
<td>No answer</td>
<td>CD</td>
<td>CA</td>
<td>CA</td>
</tr>
<tr>
<td>21</td>
<td>No answer</td>
<td>CA</td>
<td>CA</td>
<td>CD</td>
</tr>
</tbody>
</table>

*Figure 30: The comparison of datasets (twitter vs. experts)*

The only disadvantage of using twitter data for generation of dynamic profile is the lack of answer for almost one third of policy issue statements. I believe that this could be solved if we add another social network profile of candidates such as Facebook which will allow to analyze more datasets than twitter API does. However, the answers generated from the tweets of candidate's are more reliable and show real emotional relation expressed in the tweet on specific topics. Hence, it can be said that obtained results reveal more emotionally precise, reliable and direct positions of candidates they have shown. Typical expert analysis does not make sentiment analysis on statements candidates propose, which may hide the internal attitude towards real ones in order to gain the winning positions in the elections.
Conclusion

Overall, the goals and objectives of the project are fulfilled appropriately. The presented VAA 2.0 can be used for further research and applied practically in the countries where VAAs are not implemented due to the lack of information about candidates’ positions for the VAA statements. In general the functionality of VAA 2.0 allows us to say that it is more reliable and easy to apply tool in comparison with current VAAs for many different reasons. First, the positions of candidates directly extracted and analyzed from their twitter account as it is one of the most reliable and secure sources of information because it cannot be biased and it is written directly by the candidates. Second, it is easy to use for users because it has traditional interface as most VAAs have.

Unfortunately the VAA 2.0 could not gather the answers for all policy issue statements from twitter accounts, therefore the number of questions had to be decreased or replaced by other questions that had been mentioned in the tweets of candidates. However, this only shows that policy issue statements had to be precisely developed before integration into the VAA 2.0. Or collecting answer can be done during some period of time with intervals in
length of weeks or days depending on the activity of candidate’s on twitter. This will allow us to cover more tweets than 200 which is the limit for twitter API.

Most VAA’s today use expert’s data about positions of candidates towards policy issue statements. The main disadvantage of this solution is that experts require enormous amount of time to gather, extract and process the data in order to summarize the answers to statements. Moreover, we should consider the human factor, so the data can be biased or manipulated for personal reasons. Therefore it is critically important to develop new solution for the cases when political candidates or parties refuse to answer VAA statements in order to provide well-functioning and trustful recommendations to VAA users. We propose new VAA 2.0 with integrated dynamic profiles of candidates where the data for position defining will be generated from the tweets posted on the official twitter accounts of candidates. In order to solve this isse, we have proposed new VAA 2.0 with dynamic profile generation. For the generation of dynamic profiles we have used twitter API to connect with twitter accounts of candidates, ANEW dictionary, probability density function and cumulative distribution function for sentiment analysis that had already been used in ‘twitalyzer’ project. After deep research, the implemented sentiment analysis in ‘twitalyzer’ could be applied in VAA 2.0 for defining the sentiments of tweets and extracting the positions of candidates to the statements presented in VAA. Sentiment analysis had never been used before for profile generation and position extraction in VAA’s. Sentiment analysis of microblogs or tweets such as Twitter has recently obtained a fair amount of attention. The functionality of one of the sentiment analysis approaches compares the words of a posting against a labeled word list, where each word has been scored for valence, a "sentiment lexicon" or "affective word lists". There exist several affective word lists, but ANEW (Affective Norms for English Words) that had been used in our project was developed before the advent of microblogging and sentiment analysis, and freely available online. Therefore, we have decided to use ANEW library instead of others. However, I assume that for further development we can increase the list of affective words in ANEW library by repeating the experiment with politically related words and terms, that had been done when the first list of ANEW library were developed.
Bibliography


Appendix

A. Admin Panel

using System;
using System.Linq;
using System.Data.SqlClient;
using System.Web.Configuration;
using System.Data;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Threading;
using LinqToTwitter;

namespace VAA_webform
{
    public partial class admpanel : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                var con_string = WebConfigurationManager.ConnectionStrings["myconnection"].ConnectionString;
                using (SqlConnection dbconnection = new SqlConnection(con_string))
                {
                    dbconnection.Open();
                    SqlCommand command0 = new SqlCommand("select * from [isok]", dbconnection);
                    DataTable dt = new DataTable();
                    SqlDataAdapter sda = new SqlDataAdapter("select * from [isok]", dbconnection);
                    sda.Fill(dt);
                    Repeater1.DataSource = dt;
                    Repeater1.DataBind();
                }
            }
        }

        protected void getit(object source, RepeaterCommandEventArgs e)
        {
            LinkButton confirm = e.Item.FindControl("confirm") as LinkButton;
        }

        protected void confirm_Click(object sender, EventArgs e)
        {
            string cid1 = ";
            foreach (RepeaterItem item in Repeater1.Items)
            {
                var cid = (Label)item.FindControl("cid");
                cid1 = cid.ToString();
            }
        }

        //start
        protected void twit_Click(object sender, EventArgs e)
        {
            foreach (RepeaterItem item in Repeater1.Items)
            {
                var cid = (Label)item.FindControl("cid");
                var twac = (TextBox)item.FindControl("twac");
                Console.WriteLine("SECTION A: Initialise local variables");
```
const string accessToken = "2671318080-eqO0D0VdWGM0qjHWHyKnqtp0x59Mjvhfg7DQcb1d";
const string accessTokenSecret = "Lw6VfSaYOSiqw5DPan1HRtGcitq7YXoPMWogqjL6fI4fuWn";
const string consumerKey = "p10YbJmlycuY18fewVkJv9nWN";
const string consumerSecret = "qlBkKI3jNQDillmAE43imrTUXbpjFcm0ww6qjWF5Roq98g";
string aca = twac.Text.Replace(" ", ");
string accid = cid.Text.Replace(" ", ");
string twitterAccountToDisplay = aca;

Console.WriteLine("SECTION B: Setup Single User Authorisation");
var authorizer = new SingleUserAuthorizer
{
    CredentialStore = new InMemoryCredentialStore
    {
        ConsumerKey = consumerKey,
        ConsumerSecret = consumerSecret,
        OAuthToken = accessToken,
        OAuthTokenSecret = accessTokenSecret
    }
};

Console.WriteLine("SECTION C: Generate the Twitter Context");
var twitterContext = new TwitterContext(authorizer);

Console.WriteLine("SECTION D: Get Tweets for user");
var statusTweets = from tweet in twitterContext.Status
    where tweet.Type == StatusType.User &&
    tweet.ScreenName == twitterAccountToDisplay &&
    // tweet.TrimUser == true & &
    tweet.Count == 100
    select tweet;
PrintTweets(statusTweets);
delim1(accid);
```
if (dataft[i].Contains("semi-automatic") || dataft[i].Contains("semiautomatic") || dataft[i].Contains("weapons") || dataft[i].Contains("ban") { temp[11]++; }
  //Environment
  if (dataft[i].Contains("global") || dataft[i].Contains("warming") || dataft[i].Contains("effects") || dataft[i].Contains("exaggerated") { temp[3]++; }
  if (dataft[i].Contains("carbon") || dataft[i].Contains("tax") || dataft[i].Contains("reduce") || dataft[i].Contains("pollution") { temp[4]++; }
  if (dataft[i].Contains("climate") || dataft[i].Contains("change") || dataft[i].Contains("international treat") || dataft[i].Contains("limit") || dataft[i].Contains("economic growth") { temp[5]++; }

  //military
  if (dataft[i].Contains("invoke") || dataft[i].Contains("invasion") || dataft[i].Contains("Iraq") || dataft[i].Contains("right") { temp[15]++; }
  if (dataft[i].Contains("safer") || dataft[i].Contains("invasion") || dataft[i].Contains("Iraq") || dataft[i].Contains("invasion") { temp[16]++; }
  if (dataft[i].Contains("new president") || dataft[i].Contains("bring") || dataft[i].Contains("US troops") || dataft[i].Contains("Iraq") || dataft[i].Contains("immediately") { temp[17]++; }

  //economy
  if (dataft[i].Contains("reduce") || dataft[i].Contains("federal") || dataft[i].Contains("deficit") || dataft[i].Contains("raise") || dataft[i].Contains("taxes") { temp[0]++; }
  if (dataft[i].Contains("tax") || dataft[i].Contains("cuts") || dataft[i].Contains("higher") || dataft[i].Contains("income") || dataft[i].Contains("reversed") { temp[1]++; }
  if (dataft[i].Contains("Government") || dataft[i].Contains("retirement") || dataft[i].Contains("funds") || dataft[i].Contains("no responsibility") { temp[2]++; }

  //Income
  if (dataft[i].Contains("mortgage") || dataft[i].Contains("lenders") || dataft[i].Contains("lenders") || dataft[i].Contains("control") || dataft[i].Contains("tightly") { temp[12]++; }
  if (dataft[i].Contains("higher income") || dataft[i].Contains("people") || dataft[i].Contains("receive") || dataft[i].Contains("medical") || dataft[i].Contains("benefits") { temp[13]++; }
  if (dataft[i].Contains("Federal") || dataft[i].Contains("government") || dataft[i].Contains("reduce") || dataft[i].Contains("income") || dataft[i].Contains("inequality") { temp[14]++; }

  //National
  if (dataft[i].Contains("US") || dataft[i].Contains("financial") || dataft[i].Contains("contribution") || dataft[i].Contains("UN") || dataft[i].Contains("reduce") { temp[18]++; }
  if (dataft[i].Contains("Iran") || dataft[i].Contains("imminent") || dataft[i].Contains("world peace") || dataft[i].Contains("dangerous") || dataft[i].Contains("threat") { temp[19]++; }
  if (dataft[i].Contains("US") || dataft[i].Contains("decrease") || dataft[i].Contains("spending") || dataft[i].Contains("defense") { temp[20]++; }

  //Family
  if (dataft[i].Contains("same sex") || dataft[i].Contains("marriage") || dataft[i].Contains("made") || dataft[i].Contains("legal") || dataft[i].Contains("family") { temp[6]++; }
  if (dataft[i].Contains("abortion") || dataft[i].Contains("completely") || dataft[i].Contains("using embryos") || dataft[i].Contains("stem cell") || dataft[i].Contains("research") || dataft[i].Contains("diseases") || dataft[i].Contains("acceptable") { temp[8]++; }

  indexAtMax = temp.ToList().IndexOf(temp.Max());
  if (finres[indexAtMax, 0] == null && finres[indexAtMax, 1] == null && finres[indexAtMax, 2] == null) { finres[indexAtMax, 0] = dataft[i]; i++; }
  else if (finres[indexAtMax, 0] != null && finres[indexAtMax, 1] == null && finres[indexAtMax, 2] == null) { finres[indexAtMax, 1] = dataft[i]; i++; }
  else if (finres[indexAtMax, 0] != null && finres[indexAtMax, 1] != null) { finres[indexAtMax, 2] = dataft[i]; i++; }
  else i++;

  if (finres[indexAtMax, 0] != null)
  {
    string[] check = finres[indexAtMax, 0].Split();
    foreach (string q in check)
```csharp
{  
    dbconnection.Close();
    dbconnection.Open();
    SqlCommand command1 = new SqlCommand("select * FROM [dictionary] where word = @word1", dbconnection);
    if (dbconnection.State == ConnectionState.Open)  
    {  
        command1.Parameters.Add("word1", SqlDbType.Char, 50).Value = q;
    }
    SqlDataReader reader1 = command1.ExecuteReader();
    while (reader1.Read())  
    {  
        string w = (string)reader1["word"];  
        double m1 = (double)reader1["mean1"];  
        double d1 = (double)reader1["deviation1"];  
        double m2 = (double)reader1["mean2"];  
        double d2 = (double)reader1["deviation2"];  
        ret[t, 0] = w;  
        ret[t, 1] = m1.ToString();  
        ret[t, 2] = d1.ToString();  
        ret[t, 3] = m2.ToString();  
        ret[t, 4] = d2.ToString(); t++;
    }
    reader1.Close();
    dbconnection.Close();
}

string res = "";
    double xcoor = 0;
    double ycoor = 0;
    double[,] end = new double[150, 8];
    for (int a = tt; a < t; a++)  
    {  
        double m1_ = double.Parse(ret[a, 1]);  
        double d1_ = double.Parse(ret[a, 2]);  
        double m2_ = double.Parse(ret[a, 3]);  
        double d2_ = double.Parse(ret[a, 4]);  
        double pi = (1) / (Math.Sqrt(2 * 3.14 * d1_ * d1_));  
        double pi2 = (1) / (Math.Sqrt(2 * 3.14 * d2_ * d2_));  
        end[a, 1] = pi;  
        end[a, 2] = pi2;
    }
    double pisum1 = 0;
    double pisum2 = 0;
    for (int a = tt; a < t; a++)  
    {  
        pisum1 += end[a, 1];  
        pisum2 += end[a, 2];  
        end[a, 3] = double.Parse(ret[a, 3]) / pisum1;  
        end[a, 4] = double.Parse(ret[a, 4]) / pisum1;  
        end[a, 5] = double.Parse(ret[a, 1]) * pisum1;  
        end[a, 6] = double.Parse(ret[a, 3]) * pisum2;  
        xcoor += end[a, 5];  
        ycoor += end[a, 6];
    }
    if (xcoor > 5 && ycoor > 5) { res = "-0.5"; }
    if (xcoor > 5 && ycoor < 5) { res = "-1"; }
    if (xcoor < 5 && ycoor < 5) { res = "1"; }
    return res;
}
```
if (xcoor < 5 && ycoor > 5) { res = "0.5"; }

    string qn = (indexAtMax + 1).ToString();
dbconnection.Open();
SqlCommand command = new SqlCommand("UPDATE dAnswers SET question" + qn + "+@que where did=@id", dbconnection);

    if (dbconnection.State == ConnectionState.Open)
    {
        command.Parameters.Add("que", SqlDbType.Char, 50).Value = res;
        command.Parameters.Add("id", SqlDbType.Char, 50).Value = accid;
    }
    command.ExecuteNonQuery();
    qn = "";
t = 0;
indexAtMax = 0;
dbconnection.Close();

}

private static void PrintTweets(IQueryable<Status> statusTweets)
{
    try
    {
        int i = 0;
        foreach (var statusTweet in statusTweets)
        {
            Console.WriteLine(string.Format("\n\nTweet From [{0}] at [{1}]\n-{2}",
                statusTweet.ScreenName,
                statusTweet.UserID,
                statusTweet.Text));
            Thread.Sleep(1000);
            dataft[i] = statusTweet.Text.Replace("/n", " ");
            i++;
        }
    }
    catch { }
}

static string[,] delim = new string[21, 100];
static string[] dataft = new string[300];
static string[,] finres = new string[21, 3];

//end

protected void confirm_Click1(Object sender, EventArgs e)
{
}

protected void confirm_Click2(Object sender, EventArgs e)
{
    foreach (RepeaterItem item in Repeater1.Items)
    {
        var cid = (Label)item.FindControl("cid");
        var twac = (TextBox)item.FindControl("twac");

}
Console.WriteLine("SECTION A: Initialise local variables");
const string accessToken = "2671318080-eq0D0VdWGM0qjHHWyKnrqp0x59Mjvhf7DQcb1d";
const string accessTokenSecret = "Lw6VfsaYOSiqw5Dcn1HRtGcitq7YXoPMWogqjIgfiQwu";
const string consumerKey = "p10YbJmlycuY18fewVkJv9nWN";
const string consumerSecret = "qlBkKI3jNQDlImAq43imrTUxbpFjCm0wwL6qIFWSRloq98g";
string aca = twac.Text.Replace(" ", "");
string accid = cid.Text.Replace(“, “);
string twitterAccountToDisplay = aca;

Console.WriteLine("SECTION B: Setup Single User Authorisation");
var authorizer = new SingleUserAuthorizer
{
    CredentialStore = new InMemoryCredentialStore
    {
        consumerKey, consumerSecret, onSuccess = completion =>
        {
            print("Authorisation completed successfully.");
        }, onFailure = exception =>
        {
            print("Authorisation failed: ", exception.Message);
        },
    },
};
Console.WriteLine("SECTION C: Generate the Twitter Context");
var twitterContext = new TwitterContext(authorizer);

Console.WriteLine("SECTION D: Get Tweets for user");
var statusTweets = from tweet in twitterContext.Status
    where tweet.Type == StatusType.User
        && tweet.ScreenName == twitterAccountToDisplay
        && tweet.Count == 300
    select tweet;
PrintTweets(statusTweets);
delim1(accid);

B. Candidate Profile

using System;
using System.Web;
using System.Data.SqlClient;
using System.Web.Configuration;
using System.Data;
using System.Web.UI;
using LinqToTwitter;
using System.Threading;
using System.Linq;

namespace VAA_webform.main
{
    public partial class CandidateProf : System.Web.UI.Page
    {
        string cid = "";
    }
}
protected void Page_Load(object sender, EventArgs e)
{
    cid = MyBaseC._doresp_id.requiredid;
    var con_string = WebConfigurationManager.ConnectionStrings["myconnection"]; using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString))
    {
        SqlCommand command = new SqlCommand("select * from [user] where id = @id", dbconnection);
        if (dbconnection.State == ConnectionState.Open)
        {
            command.Parameters.AddWithValue("@id", id);
        }
        SqlDataReader reader = command.ExecuteReader();
        while (reader.Read())
        {
            surname = (string)reader["usurname"]; name = (string)reader["uname"]; dob = (string)reader["udob"]; country = (string)reader["ucountry"]; mail = (string)reader["umail"]; mobile = (string)reader["umobile"]; // mobile1 = (string)reader["umobile1"]; gender = (string)reader["ugender"]; dbconnection.Close();
        }
    
    public string name1 = "";
    public string surname = "";
    public string id = "1";
    public string name = "";
    public string age = "";
    public string dob = "";
    public string country = "";
    public string mail = "";
    public string mobile = "";
    public string mobile1 = "";
    public string gender = "";
}

C. Voter Profile

namespace VAA_webform
{

    public partial class About : Page
    {
    
        protected void Page_Load(object sender, EventArgs e)
        {
        
            if (HttpContext.Current.Request.IsAuthenticated)
            {
            }

            var con_string = WebConfigurationManager.ConnectionStrings["myconnection"]; using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString)) {
            
                SqlCommand command = new SqlCommand("select * from [user] where id = @id", dbconnection); if (dbconnection.State == ConnectionState.Open) {
                    command.Parameters.AddWithValue("@id", id);
                }

                SqlDataReader reader = command.ExecuteReader(); while (reader.Read()) {
                    surname = (string)reader["usurname"];
                    name = (string)reader["uname"];
                    dob = (string)reader["udob"];
                    country = (string)reader["ucountry"];
                    mail = (string)reader["umail"];
                    mobile = (string)reader["umobile"];
                    // mobile1 = (string)reader["umobile1"];
                    gender = (string)reader["ugender"];
                }

                //
                // catch
                // {
                //     Literal1.Text = "fail_connection";
                //     // }
                // finally {
                //     dbconnection.Close();
                //     dbconnection.Dispose();
                //     LbName.Text = surname;
                // }

                public string name1 = "";
                public string surname = "";
                public string id = "1";
                public string name = "";
                public string age = "";
                public string dob = "";
                public string country = "";
                public string mail = "";
                public string mobile = "";
                public string mobile1 = "";
                public string gender = "";
                public void form_profile() {
                }
        }
    }
}
D. VAA questions

using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Data.SqlClient;
using System.Web.Configuration;
using System.Data;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;

namespace VAA_webform
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        string[] answers = new string[22];
        public int eer { get; set; }
        public int m = 0;
        public string que1 = "";
        public string que2 = "";
        public string que3 = "";
        public string question_title = "";
        string id = "";
        public string[,] data = new string[7, 4];
        public string[,] cq = new string[40, 11];
        int ii = 0;

        protected void Page_Load(object sender, EventArgs e)
        {
            if (HttpContext.Current.Request.IsAuthenticated)
            {
            }
            var con_string = WebConfigurationManager.ConnectionStrings["myconnection"]; using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString))
            {
                // try
                // { command.Connection = dbconnection;
                dbconnection.Open();
                SqlCommand command = new SqlCommand("select * from [questions]", dbconnection);

                SqlDataReader reader = command.ExecuteReader();

                while (reader.Read())
                {
                    int jj = 0;
                    data[ii, jj] = (string)reader["topic"]; jj++;
                    data[ii, jj] = (string)reader["q1"]; jj++;
                    data[ii, jj] = (string)reader["q2"]; jj++;
                    data[ii, jj] = (string)reader["q3"]; jj++;
                    ii++;
                    m = ii;
                    //que = (string)reader["q1"];    
                }
            }
        }
    }
}
    dbconnection.Close();

    question_title = data[0, 0];
    que1 = data[0, 1];
    que2 = data[0, 2];
    que3 = data[0, 3];

}

public void newquest(int n)
{
    // n = n + 1;
    question_title = data[n, 0];
    que1 = data[n, 1];
    que2 = data[n, 2];
    que3 = data[n, 3];
    RadioButtonList1.ClearSelection();
    RadioButtonList2.ClearSelection();
    res.ClearSelection();
}

protected void Button1_Click(object sender, EventArgs e)
{
    int nn = MyBaseC_doresp_id.td;
    //answers[nn] =
    int qwe = 0;
    if (res.SelectedValue == "" || res.SelectedValue == null) res.SelectedValue = "0";
    MyBaseC_doresp_id.uq[nn] = res.SelectedValue;
    if (RadioButtonList1.SelectedValue == "" || res.SelectedValue == null) RadioButtonList1.SelectedValue = "0";
    MyBaseC_doresp_id.uq[nn + 1] = RadioButtonList1.SelectedValue;
    if (RadioButtonList2.SelectedValue == "" || res.SelectedValue == null) RadioButtonList2.SelectedValue = "0";
    MyBaseC_doresp_id.uq[nn + 2] = RadioButtonList2.SelectedValue;

    MyBaseC_doresp_id.td = MyBaseC_doresp_id.td + 3;
    MyBaseC_doresp_id.td1++;;
    qwe = MyBaseC_doresp_id.td1;

    if (MyBaseC_doresp_id.td1 == 6)
    {
        Button1.Visible = false;
        Button2.Visible = true;
        //try
        //({
        //    writedo();
        //})
        //catch
        //({
        //    Response.Redirect("../default");
        //})
        newquest(qwe);
    }

    public void writedo()
    {
    }
if (res.SelectedValue == "" || res.SelectedValue == null) res.SelectedValue = "0";

MyBaseC._doresp_id.uq[18] = res.SelectedValue;
if ((RadioButtonList1.SelectedItem.Value == "" || res.SelectedValue == null) radioButton1.SelectedValue = "0";

if ((RadioButtonList2.SelectedItem.Value == "" || res.SelectedValue == null) radioButton2.SelectedValue = "0";


try
{
    var con_string1 = WebConfigurationManager.ConnectionStrings["myconnection"];
    using (SqlConnection dbconnection = new SqlConnection(con_string1.ConnectionString))
    {
        dbconnection.Open();
        SqlCommand command = new SqlCommand("INSERT INTO dAnswers(did,question1,question2,question3,question4,question5,question6,question7,question8,question9,question10,question11,question12,question13,question14,question15,question16,question17,question18,question19,question20,question21) + "
            values(@id,@q1,@q2,@q3,@q4,@q5,@q6,@q7,@q8,@q9,@q10,@q11,@q12,@q13,@q14,@q15,@q16,@q17,@q18," + @q19,@q20,@q21)", dbconnection);
        SqlCommand command1 = new SqlCommand("INSERT INTO dAnswers(did) values(@did)", dbconnection);
        if (con_string1 == null)
        {
            command.ExecuteNonQuery();
            if (dbconnection.State == ConnectionState.Open)
            {
                command1.Parameters.AddWithValue("id", SqlDbType.Char, 50).Value = "k1oko";
                command1.Parameters.AddWithValue("id", SqlDbType.Char, 50).Value = id;
                command1.Parameters.AddWithValue("q1", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[0];
                command1.Parameters.AddWithValue("q2", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[1];
                command1.Parameters.AddWithValue("q3", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[2];
                command1.Parameters.AddWithValue("q4", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[3];
                command1.Parameters.AddWithValue("q5", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[4];
                command1.Parameters.AddWithValue("q6", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[5];
                command1.Parameters.AddWithValue("q7", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[6];
                command1.Parameters.AddWithValue("q8", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[7];
                command1.Parameters.AddWithValue("q9", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[8];
                command1.Parameters.AddWithValue("q10", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[9];
                command1.Parameters.AddWithValue("q11", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[10];
                command1.Parameters.AddWithValue("q13", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[12];
                command1.Parameters.AddWithValue("q14", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[13];
                command1.Parameters.AddWithValue("q15", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[14];
                command1.Parameters.AddWithValue("q16", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[15];
                command1.Parameters.AddWithValue("q17", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[16];
                command1.Parameters.AddWithValue("q18", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[17];
                command1.Parameters.AddWithValue("q19", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[18];
                command1.Parameters.AddWithValue("q20", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[19];
                command1.Parameters.AddWithValue("q21", SqlDbType.Char, 50).Value = MyBaseC._doresp_id.uq[20];
                command1.ExecuteNonQuery();
                // Response.Redirect("default");
                dbconnection.Close();
                Response.Redirect("results");
            }
        }
        catch
        {
        }
    }
}

79
E. VAA results

```csharp
using System;
using System.Web;
using System.Data.SqlClient;
using System.Web.Configuration;
using System.Data;
using System.Web.UI;
using System.Threading;
using System.Linq;
using System.Web.UI.WebControls;

namespace VAA_webform.main
{
    public partial class Results : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            int reo = 0;
            if (HttpContext.Current.Request.IsAuthenticated)
            {
            }
            int fid = 0;
            try
            {
                var con_string = WebConfigurationManager.ConnectionStrings["myconnection"];  
                using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString))
                {
                    // try
                    // { command.Connection = dbconnection;
                }
            }
        }
    }
}
```
dbconnection.Open();
SqlCommand command1 = new SqlCommand("select * FROM [danswers] where did = @id", dbconnection);
SqlCommand command0 = new SqlCommand("SELECT COUNT(*) FROM [auth] where who='Candidate'", dbconnection);
SqlCommand command = new SqlCommand("select did,question1,question2,question3,question4,question5,question6,question7,question8,question9,question10,question11,question12,question13,question14,question15,question16,question17,question18,question19,question20,question21 FROM [danswers] JOIN [auth] ON did=idd where who='Candidate'", dbconnection);
reo = (Int32)command0.ExecuteScalar();
double[,] results = new double[reo, 22];
double[] compare = new double[reo];
string[,] res = new string[reo, 22];
SqlDataReader reader = command.ExecuteReader();
int ii = 0;
while (reader.Read())
{
    int jj = 0;
    res[ii, jj] = (string)reader["did"]; jj++;
    res[ii, jj] = (string)reader["question1"]; jj++;
    res[ii, jj] = (string)reader["question2"]; jj++;
    res[ii, jj] = (string)reader["question3"]; jj++;
    res[ii, jj] = (string)reader["question4"]; jj++;
    res[ii, jj] = (string)reader["question5"]; jj++;
    res[ii, jj] = (string)reader["question6"]; jj++;
    res[ii, jj] = (string)reader["question7"]; jj++;
    res[ii, jj] = (string)reader["question8"]; jj++;
    res[ii, jj] = (string)reader["question9"]; jj++;
    res[ii, jj] = (string)reader["question10"]; jj++;
    res[ii, jj] = (string)reader["question11"]; jj++;
    res[ii, jj] = (string)reader["question12"]; jj++;
    res[ii, jj] = (string)reader["question13"]; jj++;
    res[ii, jj] = (string)reader["question14"]; jj++;
    res[ii, jj] = (string)reader["question15"]; jj++;
    res[ii, jj] = (string)reader["question16"]; jj++;
    res[ii, jj] = (string)reader["question17"]; jj++;
    res[ii, jj] = (string)reader["question18"]; jj++;
    res[ii, jj] = (string)reader["question19"]; jj++;
    res[ii, jj] = (string)reader["question20"]; jj++;
    res[ii, jj] = (string)reader["question21"]; jj++;
    ii++;
}
reader.Close();
if (dbconnection.State == ConnectionState.Open)
{
    command1.Parameters.Add("id", SqlDbType.Char, 50).Value = id;
}
SqlDataReader reader1 = command1.ExecuteReader();
if (reader1.HasRows) { }
while (reader1.Read())
{
    int jj = 0;
    rr[jj] = (string)reader1["did"]; jj++;
    rr[jj] = (string)reader1["question1"]; jj++;
    rr[jj] = (string)reader1["question2"]; jj++;
    rr[jj] = (string)reader1["question3"]; jj++;
    rr[jj] = (string)reader1["question4"]; jj++;
    rr[jj] = (string)reader1["question5"]; jj++;
    rr[jj] = (string)reader1["question6"]; jj++;
}
for (int i = 0; i < reo; i++)
{
    double q = 0;
    for (int j = 1; j < 22; j++)
    {
        if (res[i, j] != null)
        {
        }
        else if (res[i, j] == null && i < reo) i++;

        q += results[i, j];
    }
    compare[i] += q;
}

reader.Close();
reader1.Close();
int indexAtMax = compare.ToList().IndexOf(compare.Max());

// double[] doubleArray = new double[5] { 8.1, 10.2, 2.5, 6.7, 3.3 }; // Write array
// Array.Sort(compare);
// for (int t = 0; t < compare.Length; t++)
// { // for (int r = 0; r < compare.Length; r++)
//     if (compare[t] < compare[r])
//         temp = compare[t];
//     compare[t] = compare[r];
//     compare[r] = temp;
// }

string[,] c1 = new string[reo, 2]; // foreach (int d in compare)
for (int y = 0; y < reo; y++)
{
    fid = y;
    string search = res[fid, 0];
SqlCommand readfromdb = new SqlCommand("select * from [user] where id = " + search + ",
  dbconnection");
SqlDataReader reader11 = readfromdb.ExecuteReader();
if (reader11.HasRows) {
    while (reader11.Read()) {
        score1 = reo.ToString();
        c1[y, 0] = (string)reader11["uname"];
        c1[y, 1] = (string)reader11["usurname"];
        TableRow row = new TableRow();
        TableCell cell0 = new TableCell();
        TableCell cell1 = new TableCell();
        TableCell cell2 = new TableCell();
        cell0.Text = y.ToString();
        cell1.Text = c1[y, 0];
        cell2.Text = c1[y, 1];
        row.Cells.Add(cell1);
        row.Cells.Add(cell2);
        row.Cells.Add(cell0);
        myTable.Rows.Add(row);
    }
    reader11.Close();
}
}
}
}
catch {}
}
public void print()
{
}
public double score = 0;
public void ua() {
    var con_string = WebConfigurationManager.ConnectionStrings["myconnection"];  
    using (SqlConnection dbconnection1 = new SqlConnection(con_string.ConnectionString))
    {
        SqlCommand command1 = new SqlCommand("select * FROM [danswers] where did = @id",
            dbconnection1);
            if (dbconnection1.State == ConnectionState.Open)
            {
                command1.Parameters.Add("id", SqlDbType.Char, 50).Value = id;
            }
        SqlDataReader reader1 = command1.ExecuteReader();
        if (reader1.HasRows) {
            while (reader1.Read())
            {
                int jj = 0;
                rr[jj] = (string)reader1["did"]; jj++;
                rr[jj] = (string)reader1["question1"]; jj++;
                rr[jj] = (string)reader1["question2"]; jj++;
                rr[jj] = (string)reader1["question3"]; jj++;
                rr[jj] = (string)reader1["question4"]; jj++;
                rr[jj] = (string)reader1["question5"]; jj++;
                rr[jj] = (string)reader1["question6"]; jj++;
                rr[jj] = (string)reader1["question7"]; jj++;
                rr[jj] = (string)reader1["question8"]; jj++;
                rr[jj] = (string)reader1["question9"]; jj++;
            }
    }
rr[jj] = (string)reader1["question10"]; jj++; 
rr[jj] = (string)reader1["question11"]; jj++; 
rr[jj] = (string)reader1["question12"]; jj++; 
rr[jj] = (string)reader1["question13"]; jj++; 
rr[jj] = (string)reader1["question14"]; jj++; 
rr[jj] = (string)reader1["question15"]; jj++; 
rr[jj] = (string)reader1["question16"]; jj++; 
rr[jj] = (string)reader1["question17"]; jj++; 
rr[jj] = (string)reader1["question18"]; jj++; 
rr[jj] = (string)reader1["question19"]; jj++; 
rr[jj] = (string)reader1["question20"]; jj++; 
rr[jj] = (string)reader1["question21"]; }
reader1.Close();
dbconnection1.Close();
} }

public void ca()
{
    var con_string = WebConfigurationManager.ConnectionStrings["myconnection"];
    using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString))
    {
        SqlCommand command0 = new SqlCommand("SELECT COUNT(*) FROM [auth] where who='Candidate'", dbconnection);
        SqlCommand command = new SqlCommand("select did,question1,question2,question3,question4,question5,question6,question7,question8,question9,question10 ,question11,question12,question13,question14,question15,question16,question17,question18,question19,question20,question21 FROM [danswers] JOIN [auth] ON did=idd where who='Candidate'", dbconnection);

        int reo = (Int32)command0.ExecuteScalar();
        string[,] res = new string[reo, 22];
        SqlDataReader reader = command.ExecuteReader();
        int ii = 0;
        while (reader.Read())
        {
            int jj = 0;
            res[ii, jj] = (string)reader["did"]; jj++; 
            res[ii, jj] = (string)reader["question1"]; jj++; 
            res[ii, jj] = (string)reader["question2"]; jj++; 
            res[ii, jj] = (string)reader["question3"]; jj++; 
            res[ii, jj] = (string)reader["question4"]; jj++; 
            res[ii, jj] = (string)reader["question5"]; jj++; 
            res[ii, jj] = (string)reader["question6"]; jj++; 
            res[ii, jj] = (string)reader["question7"]; jj++; 
            res[ii, jj] = (string)reader["question8"]; jj++; 
            res[ii, jj] = (string)reader["question9"]; jj++; 
            res[ii, jj] = (string)reader["question10"]; jj++; 
            res[ii, jj] = (string)reader["question11"]; jj++; 
            res[ii, jj] = (string)reader["question12"]; jj++; 
            res[ii, jj] = (string)reader["question13"]; jj++; 
            res[ii, jj] = (string)reader["question14"]; jj++; 
            res[ii, jj] = (string)reader["question15"]; jj++; 
            res[ii, jj] = (string)reader["question16"]; jj++; 
            res[ii, jj] = (string)reader["question17"]; jj++; 
            res[ii, jj] = (string)reader["question18"]; jj++; 
            res[ii, jj] = (string)reader["question19"]; jj++; 
            res[ii, jj] = (string)reader["question20"]; jj++; 
        }
}
res[ii, jj] = (string)reader["question21"]; ii++; } reader.Close(); } public void cp() {
    var con_string = WebConfigurationManager.ConnectionStrings["myconnection"]; using (SqlConnection dbconnection = new SqlConnection(con_string.ConnectionString)) {
        // try
        // { command.Connection = dbconnection;
        //     dbconnection.Open();
        SqlCommand command = new SqlCommand("select * from [user] where id = (select * from [auth] where who='candidate')", dbconnection);
        if (dbconnection.State == ConnectionState.Open)
        {
            command.Parameters.AddWithValue("@id", id);
        }
        SqlDataReader reader = command.ExecuteReader();
        while (reader.Read())
        {
            surname = (string)reader["surname"]; name = (string)reader["name"]; dob = (string)reader["dob"]; country = (string)reader["country"]; mail = (string)reader["mail"]; mobile = (string)reader["mobile"]; // mobile1 = (string)reader["mobile1"]; gender = (string)reader["gender"]; 

        }
        // catch
        // { 
        //Literal1.Text = "fail_connection";
        // }
        // finally {
        //   dbconnection.Close();
        // }
    } public string score1=""; public string id = "";
    public string name1 = "";
    public string surname = "";
    public string[,] caa = new string[10,22];
    public string name = "";
    public string age = "";
    public string dob = "";
    public string country = "";
    public string mail = "";
    public string mobile = "";
    public string mobile1 = "";
    public string gender = "";
    public string[,] rr = new string[22];
}