Automatic Age Detection Using Text Readability Features

Avar Pentel
Tallinn University,Tallinn, Estonia
+372 51 907 739
pentel@tlu.ee

ABSTRACT
In this paper, we present the results of automatic age detection based on very short texts as about 100 words per author. Instead of widely used n-grams, only text readability features are used in current study. Training datasets presented two age groups - children and teens up to age 16 and adults 20 years and older. Logistic Regression, Support Vector Machines, C4.5, k-Nearest Neighbor, Naïve Bayes, and Adaboost algorithms were used to build models. All together ten different models were evaluated and compared. Model generated by Support Vector Machine with Adaboost yield to f-score 0.94, Logistic regression to 0.93. A prototype age detection application was built using the best model.

Keywords
Automatic age detection, readability features, logistic regression, support vector machines, Weka.

1. INTRODUCTION
One important class of information in user modeling is related to user age. Any adaptive technology can use age prediction data. In educational context automatic tutoring systems and recommendation systems, can benefit on age detection.

Automatic age detection has also utilities in crime prevention. With widespread of social media, people can register accounts with false age information about themselves. Younger people might pretend to be older in order to get access to sites that are otherwise restricted to them. In the same time older people might pretend to be younger in order to communicate with youngster. As we can imagine, this kind of false information might lead to serious threats, as for instance pedophilia or other criminal activities.

But besides serious crime prevention, automatic age detection can by used by educators as indirect plagiarism detector. While there are effective plagiarism detection systems, they do not work when parents are doing pupils homework or students are using somebody else's original work, which is not published anywhere. There are closed communities where students can buy homework's for any topic.

Full scale authorship profiling is not an option here, because large amount of author texts is needed. Some authors [1] argue, that at least 10000 words per author is needed, other that 5000 [2]. But if we think about business purpose of this kind of age detector, especially when the purpose is to avoid some criminal acts, then there is no time to collect large amount of text written by particular user.

When automatic age detection studies fallow authorship profiling conventions then it is related to second problem – the features, widely used in authorship profiling, are semantic features. Probability that some sequence of words, even a single word, occur in short text is too low and particular word characterizes better the context [3] than author. Some authors use character n-grams frequencies to profile users, but again, if we speak about texts that are only about 100 words long, these features can also be very context dependent.

Semantic features are related to third problem - they are costly. Using part of speech tagging systems to categorize words and/or large feature sets for pattern matching, takes time and space. If our goal is to perform age detection fast and online then it is better to have few features that can be extracted instantly on client side.

In order to avoid all three previously mentioned shortcomings, we propose other set of features. We call them readability features, because they are previously used to evaluate texts readability. Texts readability indexes are developed already before computerized text processing, so for example Gunning Fog index [4] takes into account complex (or difficult) words, those containing 3 or more syllables and average number of words per sentence. If sentence is too long and there are many difficult words, the text is considered not easy to read and more education is needed to understand this kind of text. Gunning Fog index is calculated with a formula (1) below:

$$GunningFogIndex = 0.4 \times \left[\left(\frac{words}{sentences} \right) + 100 \times \left(\frac{complexwords}{words} \right) \right] \quad (1)$$

We suppose that authors reading skills and writing skills are correlated and by analyzing author's text readability, we can infer his/her education level, which at least to the particular age is correlated with actual age of an author. As readability indexes work reliably on texts with about 100 words, these are good candidates for our task with short texts.

As a baseline we used n-gram features in pre testing. Comparing readability features with n-gram features, we found that with wider age gap between young and adult groups, readability features making better classifiers if using short texts [5]. Now we continue this work with larger dataset and with readability features only.

Using best fitting model, we created an online prototype age detector.

Section 2 of this paper surveys the literature on age prediction. In Section 3 we present our data, features, used machine learning algorithms, and validation. In Section 4 we present our classification results and prototype application. We conclude this paper in Section 5 by summarizing and discussing our study.

2. RELATED WORKS
In this section we review related works on age- and other author-specific profiling. There are no studies that dealing particularly with effect of text sizes in context of age detection. In previous section we mentioned that by literature for authorship profiling 5000 to 10000 words per author is needed [1,2]. Luyckx and

Daelemans [6] reported a dramatic decrease of the performance of the text categorization, when reducing the number of words per text fragment to 100. As authorship profiling and authors age prediction is not the same task, we focus on works that dealing particularly with user age.

The best-known age based classification results are reported by Jenny Tam and Craig H. Martell [7]. They used age groups 13-19, 20-29, 30-39, 40-49 and 50-59. All age groups were in different size. As features word and character n-grams were used. Additionally they used emoticons, number of capital letters and number of tokens per post as features. SVM model trained on youngest age group against all others yield to f-score 0,996. Moreover this result seems remarkable, while no age gap between two classes was used.

However we have to address to some limitations of their work that might explain high f-scores. Namely they used unbalanced data set (465 versus 1263 in training data set and 116 versus 316 in test set). Unfortunately their report gave only one f-score value, but no confusion matrices, ROC or Kappa statistics. We argue, that with unbalanced data sets, single f-score value is not sufficient to characterize the models accuracy. In such test set – 116 teenagers versus 316 adults - the f-score 0.85 (or 0.42 depending of what is considered positive result) will simply be achieved by model that always classifies all cases as adults. Also, it is not clear if reported f-score is weighted average of two classes' f-scores or presenting only one class f-score. Secondly it is not clear if given f-score was result of averaging cross validation results.

It is worth of mentioning, that Jane Lin [8], used the same dataset two years earlier in her postgraduate thesis supervised by the Craig Martell, and she achieved more modest results. Her best average f-score in teens versus adult's classification with SVM model was 0.786 as compared to Tam's and Martell reported 0.996. But besides averaged f-scores, Jane Lin also reported lowest and highest f-scores, and some of her highest f-scores were indeed 0.996 as reported in Tam and Martell paper.

Peersman et al [9] used large sample 10,000 per class and extracted up to 50,000 features based on word and character n-grams. Report states, that they used posts average of 12,2 tokens. Unfortunately it is not clear if they combined several short posts from the same author, or used single short message as a unique instance in feature extraction. They tested three datasets with different age groups –11-15 versus 16+, 11-15 versus 18+ and 11-15 versus 25+. Also experimentations carried out with number of features, and training set sizes. Best SVM model and with largest age gap, largest dataset and largest number of features yield to f-score 0.88.

Santosh, et al [10,11] used word n-grams as content-based features and POS n-grams as style based features. They tested three age groups 13-17, 23-27, and 33-47. Using SVM and kNN models, best classifiers achieved 66% accuracy.

Marquart [12] tested five age groups 18-24, 25-34, 35-49, 50-64, and 65-xx. Used dataset was unbalanced and not stratified. He also used some of the text readability features as we did in current study. Besides of readability features, he used word n-grams, HTML tags, and emoticons. Additionally he used different tools for feature extraction like psycholinguistic database, sentiment strength tool, linguistic inquiry word count tool, and spelling and grammatical error checker. Combining all these features, his model yield to modest accuracy of 48,3%.

Dong Nguyen and Carolyn P. Rose [13] used linear regression to predict author age. They used large dataset with 17947 authors with average text length of 11101 words. They used as features word unigrams and POS unigrams and bigrams. Text was tagged using the Stanford POS tagger. Additionally they used linguistic inquiry word count tool to extract features. Their best regression model had r^2 value 0.551 with mean absolute error 6.7.

As we can see, most of previous studies are using similar features, word and character n-grams. Additionally special techniques were used like POS tagging, Spell Checker, and Linguistic inquiry word count tool to categorize words. While text features extracted by this equipment are important, they are costly to implement in real life online systems. Similarly large feature sets up to 50,000 features, most of which are word n-grams, means megabytes of data. Ideally this kind of detector could work using client browser resources (JavaScript), and all feature extraction routines and models have to be as small as possible.

Summarizing previous work in the following table (1), we don't list all possible features. So for example features that are generated using POS tagging or features generated some word databases are all listed here as word n-grams. Last column gives f-score or the accuracy (with %) according to what characteristic was given in paper. Most of papers reported many different results, and we list in this summary table only the best result.

Table 1. Summary of previous work

Authors	Used feature types				training dataset size	avg. words per author	separation gap (year)	result f-score or accuracy (%)
	readability	word n-grams	char n-grams	emoticons				
Nguyen (2011)		x			17947*	11101	0	55.1%
Marquardt (2014)	x	x		x	7746	N/a	0	47.3%
Peersman (2011)		x	x		20000	12.2**	9	0.917
Lin (2007)		x		x	1728*	343	0	0.786
Tam & Martell (2009)		x	x	x	1728*	343	0	0.996***
Santosh (2014)		x			236600*	335	5	66%
This Study	**x**				**500**	**93**	**4**	**0.94**

*unbalanced datasets

**12.2 words was reported average message length, but it is not clear if only one message per user was used or user text was composed form many messages.

***not enough data about this result

3. METHODOLOGY

3.1 Sample & Data

We collected short written texts in average 93 words long from different social media sources like Facebook, Blog comments, and Internet forums. Additionally we used short essay answers from school online feedback systems and e-learning systems, and e-mails. No topic specific categorization was made. All authors were identified and their age fall between 9 and 46 years. Most authors in our dataset were unique, but we used multiple texts from the same author only in case, when the texts were written in different age. All texts in the collections were written in the same language (Estonian). We chose balanced and stratified datasets with 500 records and with different 4-year age gaps.

3.2 Features

In current study we used in our training dataset different readability features of a text. Readability features are quantitative data about texts, as for instance an average number of characters in the word, syllables in the word, words in the sentences, commas in the sentence and the relative frequency of the words with 1, 2,.., n syllable. All together 14 different features were extracted from each text plus classification variable (to which age class text author belongs).

In all features we used only numeric data and normalized the values using other quantitative characteristics of the text.

Used Feature set with explanations is presented in Table 2:

Table 2. Used features with calculation formulas and explanations

Feature	Explanation
Average number of Characters in Word	$= \frac{NumberOfCharactersInText}{NumberOfWordsInText}$ We excluded all white space characters when counting number of all characters in text
Average number of Words in Sentence	$= \frac{NumberOfWordsInText}{NumberOfSentencesInText}$
Complex Words to all Words ratio	$= \frac{NumberOfComplexWordsInText}{NumberOfWordsInText}$ Complex word is loan from Cunning Fog Index, where it means words with 3 or more syllables. As Cunning Fog index was designed for English, and Estonian language has as average more syllables per word, we raised the number of syllables according to this difference to five. Additionally we count the word complex if it has 13 or more characters.
Average number of Complex Words in Sentence	$= \frac{NumberOfComplexWordsInText}{NumberOfSentencesInText}$
Average number of Syllables per Word	$= \frac{NumberOfSyllablesInText}{NumberOfWordsInText}$
Average number of Commas per Sentence	$= \frac{NumberOfCommasInText}{NumberOfSentencesInText}$
One Syllable Words to all Words ratio	$= \frac{NumberOfWordsWith1syllableInText}{NumberOfWordsInText}$
Similarly as previous feature, we extracted 7 features for words containing 2, 3, 4 to 8 and more syllables.	$= \frac{NumberOfWordsWith\_N-SyllableInText}{NumberOfWordsInText}$ Novel syllable counting algorithm was designed for Estonian language, which is only few lines length and does not include any word matching techniques

3.3 Data Preprocessing

We stored all the digitalized texts in the local machine as separate files for each example. A local program was created to extract all previously listed 14 features from each text file. It opened all files one by one; extracted features form each file, and stored these values in a row of a comma-separated file. In the end of every row it stored data about the age group. A new and simpler algorithm was created for syllable counting. Other analogues algorithms for Estonian language are intended to exact division of the word to syllables, but in our case we are only interested on exact number of syllables. As it turns out, syllable counting is possible without knowing exactly where one syllable begins or ends.

In order to illustrate our new syllable counting algorithm, we give some examples about syllables and related rules in Estonian language. For instance the word *rebane* (fox) has 3 syllables: *re – ba – ne*. In cases like this we can apply one general rule – when single consonant is between vowels, then new syllable begins with that consonant.

When in the middle of word two or more consecutive consonants occur, then usually the next syllable begins with last of those consonants. For instance the word *kärbes* (fly) – is split as *kär-bes*, and *kärbsed* (flies) is split as *kärb-sed*. The problem is that this and previous rule does not apply to compound words. So for example, the word *demokraatia* (democracy) is split before two consecutive consonants as *de-mo-kraa-tia*.

Our syllable counting algorithm deals with this problem by ignoring all consecutive consonants. We set syllable counter on zero and start comparing two consecutive characters in the word, first and second character, then second and third and so on. General rule is, that we count a new syllable, when the tested pair of characters is vowel fallowed by consonant. The exception to this rule is the last character. When the last character is vowel, then one more syllable is counted.

Implemented syllable counting algorithm as well as other automatic feature extraction procedures can be seen in section 4.3 and in the source code of the prototype application.

3.4 Machine Learning Algorithms and Tools

For classification we tested six popular machine-learning algorithms:

- Logistic regression
- Support Vector Machine
- C4.5
- k-nearest neighbor classifier
- Naive Bayes
- AdaBoost.

Motivation of choosing those algorithms is based on literature [14,15]. The suitability of listed algorithms for given data types and for given binary classification task was also taken in to account. Last algorithm in the list – Adaboost – is actually not classification algorithm itself, but an ensemble algorithm, which is intended for use with other classifying algorithms, in order to make a weak classifier stronger. In our task we used Java implementations of listed algorithms that are available in freeware data analysis package Weka [16].

3.5 Validation

For evaluation we used 10 fold cross validation on all models. It means that we partitioned our data to 10 even sized and random parts, and then using one part for validation and other 9 as training dataset. We did so 10 times and then averaged validation results.

3.6 Calculation of final f-scores

Our classification results are given as weighted average f-scores. F-score is a harmonic mean between precision and recall. Here is given an example how it is calculated. Let suppose we have a dataset presenting 100 teenagers and 100 adults. And our model classifies the results as in fallowing Table 3:

Table 3. Example illustrating calculation of f-scores

Classified as =>	teenagers	adults
teenagers	88	12
adults	30	70

When classifying teenagers, we have 88 true positives (teenagers classified as teenagers) and 30 false positives (adults classified as teenagers). We also have 12 false negatives (teenagers classified as not teenagers) and 70 true negatives (adults classified as not teenagers). In following calculations we use abbreviations: TP = true positive; FP = false positive; TN = true negative; FN = false negative.

Positive predictive value or precision for teenagers' class is calculated by formula 2.

$$precision = \frac{TP}{TP + FP} = \frac{88}{88+30} = 0.746 \quad (2)$$

Recall or sensitivity is the rate of correctly classified instances (true positives) to all actual instances in predicted class. Calculation of recall is given by formula 3.

$$recall = \frac{TP}{TP + FN} = \frac{88}{88+12} = 0.88 \quad (3)$$

F-score is harmonic mean between precision and recall and it is calculated by formula 4.

$$f-score = 2 \times \frac{precision \times recall}{precision + recall} = \frac{2TP}{2TP + FP + FN} \quad (4)$$

Using data in our example the f-score for teenager class will be 0.807, but if we do the same calculations for adult class then the f-score will be 0.769.

Presenting our results, we use a single f-score value, which is an average of both classes' f-score values.

4. RESULTS

4.1 Classification

Classification effect was related to placement of age separation gaps in our training datasets. We generated 8 different datasets by placing 4-year separation gap in eight different places. We generated models for all datasets, and present the best models' f-scores on figure 1. As we can see, our classification was most effective, when the age separation gap was placed to 16-19 years.

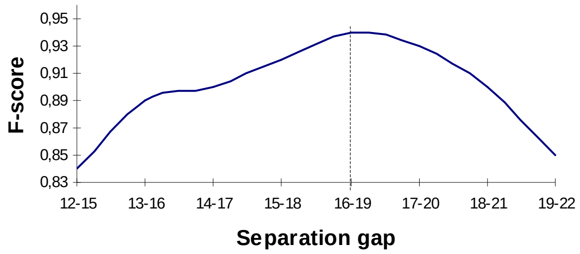

Figure 1. Effect of the position of separtion gap

With a best separation gap (16-19) between classes, Logistic regression model classified 93,12% of cases right, and Support Vector Machines generated model classified 91,74% of cases. Using Adaboost algorithm combined with classifier generated by Support Vector Machine yield to 94.03% correct classification and f-score 0.94. Classification models built by other algorithms performed less effectively as we can see in Table 4.

Results in fallowing table are divided in to two blocks. In the left side there are the results of the models generated by listed algorithms. In the right side there are the results of the models generated by Adaboost algorithm and the same algorithm listed in the row.

Table 4. Averaged F-scores of different models

	F-score	
		Using Adaboost
Logistic Regression	0.93	0.93
SVM (standardized)	0.92	0.94
KNN (k = 4)	0.86	0.86
Naïve Bayes	0.79	0.84
C4.5	0.75	0.84

As we can see in the table above, the best performers were classifiers generated by Logistic Regression algorithm and Support Vector Machine (with standardized data). In the right section of the table, where the effect of Adaboost algorithm is presented, we can see that Adaboost here cannot improve results with Logistic regression classifier, and kNN, but it improves results of SVM, Naïve Bayes and most significantly on C4.5. As Adaboost is intended to build strong classifiers out of weak classifiers, than the biggest effect on C4.5 is expectable. Two best performing classifiers remained still the same after using Adaboost, but now Support Vector Machine outperformed Logistic Regression by 0.91 percent points.

4.2 Features with highest impact

As there is relatively small set of readability features, we did not used any special feature selection techniques before generating models, and evaluating features on the basis of SVM model with standardized data. The strongest indicator of an age is the average number of words in sentence. Older people tend to write longer sentences. They also are using longer words. Average number of characters per word is in the second place in feature ranking. Best predictors of younger age group are frequent use of short words with one or two syllables.

In following Table (5), coefficients of standardized SVM model are presented.

Table 5. Features with highest impact in standardized SVM model

Coefficient	Feature
1.3639	Words in sentence
0.8399	Characters in word
0.258	Complex words in sentence
-0.2713	Ratio of words with 4 syllables
-0.3894	Commas per sentence
-0.7451	Ratio of words with 1 syllable
-0.762	Ratio of words with 2 syllables

4.3 Prototype Application

As the difference between performance of models generated by Adaboost with SVM and Logistic Regression is not significant, but as from the point of view of implementation, models without Adaboost are simpler, we decided to implement in our prototype application Logistic Regression model, which performed best without using Adaboost.[1] We implemented feature extraction routines and classification function in client-side JavaScript. Our prototype application uses written natural language text as an input, extracts features in exactly the same way we extracted features for our training dataset and predicts author's age class (Fig. 2.).

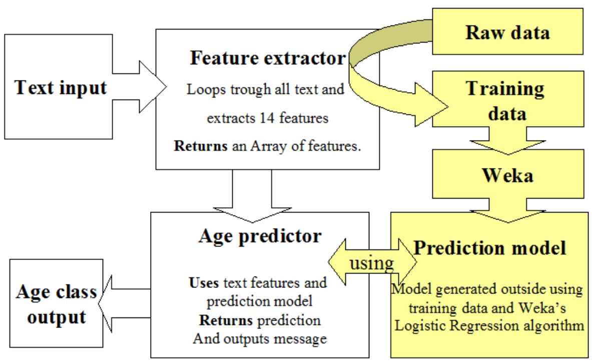

Figure 2. Application design

Our feature extraction procedure (Figure 3.) consists 3 stages:

1. Text input is split to sentences, and to words, and all excess white space chars are removed. Some simple features, number of characters, number of words, number of sentences, are also calculated in this stage.
2. In second stage syllables in words are counted.
3. All calculated characteristics are normalized using other characteristics of the same text. For example number of characters in text divided to number of words in text.

[1] http://www.tlu.ee/~pentel/age_detector/

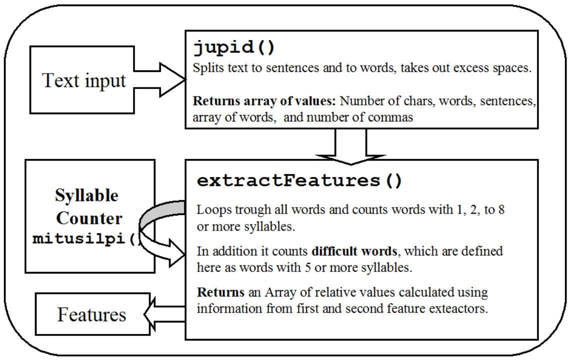

Figure 3. Feature Extractor

A new and simpler algorithm (5) was created for syllable counting. Other analogues algorithms for Estonian language are intended to exact division of the word to syllables, but in our case we are only interested on exact number of syllables. As it turns out, syllable counting is possible without knowing exactly where one syllable begins or ends. Unfortunately this is true only for Estonian (and maybe some other similar) language.

```
function number_of_syllables(w){              (5)

v="aeiouõäöü"; /* all vowels in Estonian lang. */

counter=0;

w=w.split('');/* creates char array of word */

wl=w.length; /* number of char's in word */

  for(i=0; i < wl - 1; i++){

   if(v.indexOf(w[i])!=-1 && v.indexOf(w[i+1])==-1)

      counter++;

 /*

if char is vowel and next char is not, then count a
syllable (there are some exceptions to this rule, which
are easy to program).

*/

   }

  if( v.indexOf(w[wl-1]) != -1) counter++;

// if last char in the word is vowel, count new syllable

  return counter;

}
```

Implemented syllable counting algorithm as well as other automatic feature extraction procedures can be seen in the source code of the prototype application.[2]

Finally we created simple web interface, where everybody can test prediction by his/her free input or by copy-paste. As our classifier was trained on Estonian language, sample Estonian texts are provided on website for both age groups (Fig. 4.).

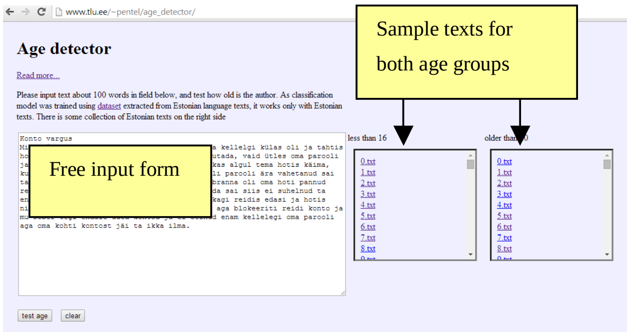

Figure 4. Prototype application at http://www.tlu.ee/~pentel/age_detector/

5. DISCUSSION & CONCLUSIONS

Automatic user age detection is a task of growing importance in cyber-safety and criminal investigations. One of the user profiling problems here is related to amount of text needed to perform reliable prediction. Usually large training data sets are used to make such classification models, and also longer texts are needed to make assumptions about author's age. In this paper we tested novel set of features for authors age based classification of very short texts. Used features, formerly known as text readability features, that are used by different readability formulas, as Gunning Fog, and others, proved to be suitable for automatic age detection procedure. Comparing different classification algorithms we found that Logistic Regression and Support Vector Machines created best models with our data and features, giving both over 90% classification accuracy.

While this study has generated encouraging results, it has some limitations. As different readability indexes measure how many years of education is needed to understand the text, we can not assume that peoples reading, or in our case writing, skills will continuously improve during the whole life. For most people, the writing skill level developed in high school will not improve further and therefore it is impossible to discriminate between 25 and 30 years old using only those features as we did in current study. But these readability features might be still very useful in discriminating between younger age groups, as for instance 7-9, 10-11, 12-13. The other possible utility of similar approach is to use it for predicting education level of an adult author.

In order to increase the reliability of results, future studies should also include a larger sample. The value of our work is to present suitability of a simple feature set for age based classification of short texts. And we anticipate a more systematic and in-depth study in the near future.

[2] http://www.tlu.ee/~pentel/age_detector/source_code.txt

6. REFERENCES

[1] Burrows, J. 2007. All the way through: testing for authorship in different frequency strata. Literary and Linguistic Computing. 22, 1, pp. 27–47. Oxford University Press.

[2] Sanderson, C., and Guenter, S. 2007. Short text authorship attribution via sequence kernels, Markov chains and author unmasking: an investigation. EMNLP'06. Association for Computational Linguistics. pp. 482–491. Stroudsburg, PA, USA.

[3] Rao, D. et al. 2010. Classifying latent user attributes in twitter, SMUC '10 Proceedings of the 2nd international workshop on Search and mining user-generated contents. pp. 37-44.

[4] Gunning, R. 1952. The Technique of Clear Writing. New York: McGraw–Hill

[5] Pentel, A. 2014. A Comparison of Different Feature Sets for Age-Based Classification of Short Texts. Technical report. Tallinn University, Estonia. www.tlu.ee/~pentel/age_detector/Pentel_AgeDetection2b.pdf

[6] Luyckx, K. and Daelemans, W. 2011. The Effect of Author Set Size and Data Size in Authorship Attribution. Literary and Linguistic Computing, Vol-26, 1.

[7] Tam, J., Martell, C. H. 2009. Age Detection in Chat. International Conference on Semantic Computing.

[8] Lin, J. 2007. Automatic Author profiling of online chat logs. Postgraduate Thesis.

[9] Peersman, C. et al. 2011. Predicting Age and Gender in Online Social Networks. SMUC '11 Proceedings of the 3rd international workshop on Search and mining user-generated contents, pp 37-44, ACM New York, USA.

[10] Santohs, K. et al. 2013. Author Profiling: Predicting Age and Gender from Blogs. CEUR Workshop Proceedings, Vol-1179.

[11] Santosh, K. et al. 2014. Exploiting Wikipedia Categorization for Predicting Age and Gender of Blog Authors. UMAP Workshops 2014.

[12] Marquart, J. et al. 2014. Age and Gender Identification in Social Media. CEUR Workshop Proceedings, Vol-1180.

[13] Nguyen, D. et al. 2011. Age Prediction from Text using Linear Regression. LaTeCH '11 Proceedings of the 5th ACL-HLT Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities. pp 115-123, Association for Computational Linguistics Stroudsburg, PA, USA.

[14] Wu, X. et al. 2008. Top 10 algorithms in data mining. Knowledge and Information Systems. vol 14, 1–37. Springer.

[15] Mihaescu, M. C. 2013. Applied Intelligent Data Analysis: Algorithms for Information Retrieval and Educational Data Mining, pp. 64-111. Zip publishing, Columbus, Ohio.

[16] Weka. Weka 3: Data Mining Software in Java. Machine Learning Group at the University of Waikato. http://www.cs.waikato.ac.nz/ml/weka/