MACHINE-LEARNING BASED APPLICATION FOR STAFF RECRUITING

Anca APATEAN[1], Evelyn SZAKACS[1], Magnolia TILCA[2]
[1]*Technical University of Cluj-Napoca*
[2]*Vasile Goldiş Western University of Arad*

Abstract: The paper presents an application to classify candidates using an average of the results obtained from the use of several independent classifiers, like KNN, LDA, Tree J48 and Naïve Bayes. The main purpose of the classification is to help a company to select a large quantity of CVs by passing them through the application filter. The dataset was constructed with 172 instances (belonging to 17 classes), each characterized by 18 attributes. Information like Education, Experience, Foreign Languages, Programming Languages, Salary Range, Gender, Age and Personal Skills were considered for each candidate.

Keywords: *classifier, KNN, LDA, Tree, Naïve Bayes, voting, matching score.*

I. INTRODUCTION

In recent years, technology has witnessed an impressive evolution, taking control of many processes and companies had to adapt to market requirements, but also to technological changes. Business Magazine presents a study by Ericsson Consumer Lab on trends in 2017. This study shows that main tasks influenced by automation processes are: customer relationship management (34%), human resources departments (39%), financial-accounting departments (40%), suppliers (40%) and front-office services (55%). As stated in [1], "robots will relieve employees of repetitive, time-consuming processes and at the same time will help reduce operational costs."

With this in mind, the present application was developed with the intention to provide an optimal quality-time ratio in the staff selection process. The central idea is to use Data Mining (DM) algorithms for a staff recruitment system: on one hand, a candidate will have the possibility to complete an online Curriculum Vitae (CV), receiving real-time feedback, and the application will return the candidate's match score to the closest classes to his or her affinity. On the other hand, a company will be able to use the application to centralize the information about each candidate, with an updated list of their performance.

DM algorithms are used to classify a candidate on a particular position, called *the Class*. One sensitive aspect of the application is represented by its inputs: how to choose the classification problem, e.g. the number of possible classes, the number of attributes that represent an instance and the number of instances registered per each class, may greatly influence the performance of the entire system. As output, the presented system is able to return a predicted class or a score by which a certain class (or classes) is proposed for that input or candidate.

The application uses prediction and classification algorithms to deliver various results. For this, an initial set of data that the application will "learn" is required. This data set can be obtained from the company history and will represent the training data, thus different templates based on which the application will make future predictions.

The application has an interface through which candidates will be able to find out whether or not they are suitable for an existing position in a particular company, and the company will in turn be able to obtain information about potential candidates in real time. An important aspect is the personalization of the interface, with the company having the possibility to choose the information it needs from the candidate.

In turn, the candidate will be able to select the desired position in the company, then fill in the required personal data. As a result, it will receive a message confirming its compatibility with the desired post, or vice versa, the incompatibility. If the candidate meets the requirements for another position in the same company, the application will issue an information message to that effect.

The company will be able to select a large quantity of CVs and pass them through the application filter. As a result, they will get a list of the right people for different positions, in order of their matching score.

The purpose of this application is to classify candidates using an average of the results obtained from the use of several independent classifiers. The most important tasks DM, when preparing data are: discretization, cleaning, integration, transformation and reduction of data. The main processes used in DM are: data normalization, missing data analysis, data redundancy and unbalanced data. All these aspects made the creation of the data set to require a long time and the final version was chosen after multiple tests; it thus contains 172 instances (belonging to 17 classes), each characterized by 18 attributes.

II. SIMILAR SYSTEMS

As inspiration for the presented application, the most representative papers are next reviewed:

In [2] several classification algorithms, like Linear Regression, Regression Tree and Support Vector Regression were compared in order to find the most appropriate one for an online recruitment task. The application calculates some scores for each person to check their compatibility with a desired position in a company. The candidate must either complete an online form or log in with a Linkedin account or personal blog. For experimentation, three available

Manuscript received November 29, 2017; revised December 06, 2017

positions were used (Sales Engineer, Junior Programmer and Senior Programmer), these representing *the classes* and a total number of 100 candidates were registered. The authors obtained a high correlation number with a M5 Tree and a relative minimum error with a REP Tree scheme.

The application in [3] uses classification algorithms to predict whether an employee will be able to promote or not. As input the result of the work done, knowledge and skills, individual qualities, activities and contributions to the company were considered; the train set contains 590 data of which 10% were extracted for the test set. After refining the data set to an accuracy of 95%, the authors concluded that the C4.5 algorithm was a proper one for the chosen field.

The application in [4] is based on the same recruitment process, but with the use of the classifiers from Weka. A comparison of two methods, i.e. Naive Bayes and decision trees was made. The selection criteria were based on: age, gender, job, desired salary, studies, experience, performance and previous job. By comparing the two methods, the authors concluded that decision trees provided better accuracy. Also, after removing some attributes (like gender and age), accuracy was increased with about 5%.

In [5], a comparison between algorithms based on decision trees, fuzzy logic and clustering algorithms, i.e. Id3, C4.5, CART, Fuzzy K-means, K-means was accomplished for staff recruitment. Pruned and unpruned trees were also used, these terms referring to the preservation or removal of some redundant or unnecessary information in order to create a tree with a higher purity.

In [6], a decision support system designed to provide information to jobseekers allow users to log on to an online platform and enter different information about them, such as personal data, education, experience, but also the desired wage or location of an aimed company. On the other hand, employers have access to the created database and can select a person deemed appropriate by them, by comparison with other candidates. The main idea of this application has emerged from multi-criteria decisions: each information entered by the candidate was reaching a certain degree of importance, depending on the criteria decided by the employer.

III. OUR PROPOSED SYSTEM

The utility of the proposed application can be viewed from two perspectives: the candidate and the company. The candidate will be able to complete an online CV, receiving real-time response if it is compatible with the desired position. On the other hand, the company will be able to centralize the candidates who have completed the forms and are suitable for a particular position. In order to be able to exemplify the way the application works, it started from a fictitious company working in the IT field. The interface created can be identified with the online form for completing the data, and the response given to the candidate is the result of the classification process. The positions within the company represent the classes, while each candidate's data represents the attributes of the classification problem.

III. A. The company and available positions

Being an IT company, developing a proprietary software product has been chosen as main activity. For this, the company's activities were divided as follows:

The actual development of the software.
Programming language used: C #
Testing the product.
Programming language used:
Java Script and other adjacent products
Server Side and Plug-in.
Programming language used: Java
Development of internal tools.
Programming language used: Python

The positions derived from the above activities will be categorized into hierarchical levels: Internship, Junior, Middle, Senior, each of which having four possible programming languages: C #, Java, Java Script, and Python. Finally, there result a number of 16 positions, considering that they are all available. A 17-th class was added to the classification system to express a neither-position qualified candidate, as shown in Table 1.

Table 1. The 17 classes of the system.

Internship QA-**1**	Junior QA-**5**	Middle QA-**9**	Senior QA-**13**	
Internship Python-**2**	Junior Python-**6**	Middle Python-**10**	Senior Python-**14**	Not matched-**17**
Internship C#-**3**	Junior C#-**7**	Middle C#-**11**	Senior C#-**15**	
Internship Java-**4**	Junior Java -**8**	Middle Java-**12**	Senior Java-**16**	

III.B. Ideal candidate profile

As inspired by other state of the art systems, the criteria on which the candidate selection is made were chosen to be: Education (University and Faculty), Experience, Foreign Languages (English, French, German), Programming Languages (C#, Java Java Script, Python), Salary Range, Gender, Age and Personal Skills.

The process of creating the dataset was tedious, their correctness being very important. Each attribute is considered important, because having ambiguous, redundant or contradictory data will lead to negative results. The classification problem was based on an ideal candidate profile and this has a major impact on the way the dataset was constructed.

III.C. Creating the Data Set

The data set requires intensive processing in order to be used in the classification process. In the final version, it contains a number of 172 instances, representing the candidates for the above-mentioned jobs, each candidate being described by 18 attributes, more precisely the information extracted after completing the online CV.

A number of order was added to each instance just to identify the candidate in a list and keep in touch with the database containing the personal information of each person who came into contact with the application. Thus, the classification system does not receive it as an attribute. The classes, representing each job and position, are numbered from 1 to 17, as already presented in Table 1 (the last class was added to express an unqualified candidate). The number of 172 candidates were synthetic created, with the distribution per class as presented in Figure 1.

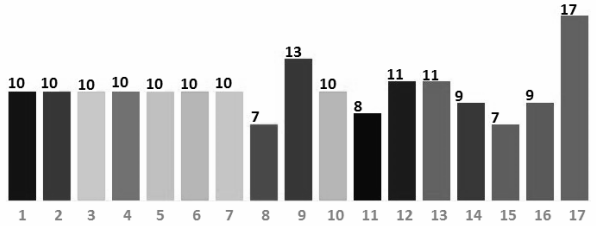

Figure 1. Graphical representation of the data set: number of instances per each of the 17 classes.

III.D. Data Encoding

Inspired by other systems from the literature, the attributes used to describe each instance in the database were chosen to be encoded in the following way:

A1: *Education (University)*: 1- Baccalaureate Diploma/ 2-Current or graduate students in a technical or scientific discipline/ 3-Current or graduate students in a non-technical discipline, thus encoded with {1, 2, 3};

A2: *Education (Faculty):* 1-Technical University-abroad, Engineering/ 2-Informatics-secundary specialization/ 3-Informatics- main specialization and 0 if A1 is 1 or 3; thus, A2 will be encoded in {0, 1, 2, 3};

A3: *Experience:* 0-6 months, 6-24 months, 24-60 months, over 60 months, encoded with: {6, 24, 60, 100}

A4-6: *Foreign languages:* English, French, German, with levels: 0-I don't speak this language at all/ 1-Independent user/ 2- I am fluent in this language, thus {0, 1, 2} for each foreign language;

A7-10: *Programming languages:* JavaScript, Java, C#, Python, with the levels: 0-I don't know this language/ 1- I am a beginner in this language/ 2- I have medium knowledge of this language/ 3- I have advanced knowledge of this language, thus encoded with: {0,1,2,3) – for each language;

A11: *Salary Range:* 400-450 EUR, 450-550 EUR, 550-1100 EUR, 1100-3500 EUR, 3500+ EUR: {0,1,2,3,4}

A12: *Gender:* M/F: {0, 1}

A13: Age: {Numeric}

A14-18: *Personal skills*

The encoding for the personal skills is presented in Figure 2 and it expresses how many qualities have been selected from each category, where I is for Internship, J is for Junior, M for Middle, S for Senior and U for Unqualified. The example illustrated in Figure 2 can be interpreted as follows: if a candidate has selected 3 qualities from the Internship category, one from Junior and Middle, and none from Senior and Unqualified, this information will be encoded as 3,1,1,0,0. In order to be considered suitable for a position, the candidate should have the highest value in the selected category (in this case Internship).

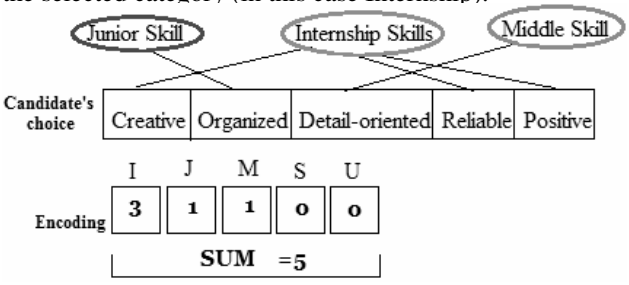

Figure 2. Example of Personal Skills encoding.

The values in Table 2 are the order in which the Personal Skills attributes are placed in the interface (intentionally not following any rule) and these have the following meaning: 1-Team player, 2-Organized, 3-Creative, 4-Positive, 5-Persuasive, 6-Enthusiast, 7-Independent, 8-Reliable, 9-Time manager, 10-Good explainer, 11-Public-speaker, 12-Multi-tasker, 13-Decision maker, 14-Problem-solver, 15-Detail-oriented, 16-Quick-learner, 17-Perseverant, 18-Leadership, 19-Debugging Skills, 20-Perfectionist, 21-Stress resistant, 22Goal-setter, 23-Pragmatic, 24-Diplomatic, 25-Adaptable.

The association of personal qualities to a specific category was based on common elements found in the description of real jobs:

Irrelevant skills: Persuasive, Public-speaker, Leadership, Goal-setter, Pragmatic: **{5,11,18,22,23}**

Skills for Internship: Creative, Positive, Enthusiast, Reliable, Perseverant: **{3,4,6,8,17}**

Skills for Junior: Team player, Organized, Multi-tasker, Quick-learner, Debugging Skills: **{1,2,12,16,19}**

Skills for Middle: Time manager, Problem-solver, Detail-oriented, Perfectionist, Stress resist.: **{9,14,15,20,21}**

Skills for Senior: Independent, Good explainer, Decision maker, Diplomatic, Adaptable: **{7,10,13,24,25}**

The ideal candidate profile emerged after numerous market studies. A number of 50 IT companies were selected both from Romania and abroad, synthesising the information on available jobs and the offer made by employers. Among the companies that we extracted information from are: Telenav, Arnia, Yardi, Ntt Data, Stefanini, Interactive Software, Ecrion Software Inc., etc. For each existing position, information shared between several companies on the market were extracted and a model profile was created.

Table 2. Attribute association for Personal Skills encoding

Internship	Junior	Middle	Senior	Unqual.
3	1	9	7	5
4	**2**	14	10	11
6	12	15	13	18
8	16	**20**	24	22
17	19	21	25	23
3 (creative, enthusiast, reliable)	***1 (organized)***	***1 (perfectionist)***	***0 (-)***	***0 (-)***

The sum must be 5

The encoding of the model profiles for each class is presented in Table 3 (with no cooperation with HR from companies). Several instances of a particular class can be seen in this table, being considered a model for each class. The other attributes, presented in Figure 3, represent the deviations from this template and have the role of data diversification. For example, the profile of the ideal candidate for the Internship QA class is highlighted in Table 3 on line 1, and in Figure 3 - Line 5. Deviations in the rest of Class 1 data are in the attributes like Education, Age, Gender, Personal Skills and very few at Experience and English. Starting from the idea that in a company the number of employees in different positions is not equal, the dataset was created as can be seen in Figure 1, so the number of instances for each class varies to be as close as possible to a real situation.

In order to be used in the classification process, class

profiles are required. An important aspect in creating the data set is precisely the idea of anchoring in reality. The operating principle is based on the training and testing of the system based on a data set. In fact, the training set would be the database of all the people employed in the company so far, and the test set will comprise those candidates who complete the online form. Being a fictitious company, the set of training data will be built manually, based on a detailed market research. Thus, the process of creating data is tedious, but their correctness is very important. Each attribute is considered important in a similar manner: in the creation of the data set, the importance of some attributes or others was not considered, leaving the classifier to automatically interpret the attributes as having the same level of importance. However, it is possible to modify the dataset by adding redundancy, so to increase the importance of some attributes versus others. The entire process of data creation and processing, as well as the steps taken until the final data set is obtained, will be presented below.

Table 3. The model profiles of each of the 17 considered classes.

Class	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18
1	1	10	6	1	0	0	2	0	0	0	0	0	22	5	0	0	0	0
2	1	10	24	1	0	0	0	2	0	0	1	0	24	0	5	0	0	0
3	1	7,5	60	1	0	0	3	0	0	0	3	1	28	0	0	5	0	0
4	1	7,5	100	1	0	0	3	2	0	0	4	0	36	0	0	0	5	0
5	1	10	6	1	0	0	0	2	0	0	0	0	22	5	0	0	0	0
6	1	10	24	1	0	1	1	2	0	0	2	0	24	0	5	0	0	0
7	1	7,5	60	1	0	1	0	3	0	0	3	0	28	0	0	5	0	0
8	1	10	100	1	0	1	1	3	2	0	4	1	36	0	0	0	5	0
9	1	10	6	1	0	0	0	0	2	0	0	0	22	5	0	0	0	0
10	1	10	24	1	1	0	0	1	2	0	2	0	24	0	5	0	0	0
11	1	7,5	60	1	1	0	0	0	3	0	3	0	28	0	0	5	0	0
12	1	10	100	1	1	0	0	2	3	0	4	0	36	0	0	0	5	0
13	1	10	6	1	0	0	0	0	0	2	0	1	22	5	0	0	0	0
14	1	7,5	24	1	0	1	0	0	1	2	2	1	24	0	5	0	0	0
15	1	7,5	60	1	0	1	0	0	0	3	3	0	28	0	0	5	0	0
16	1	7,5	100	1	0	1	0	0	2	3	4	0	36	0	0	0	5	0
17	0	0	6	0	0	0	0	0	0	0	4	0	18	0	0	0	0	5

	A1_ _	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	A16	A17	A18	CLS
No.	edu Numeric	univ Numeric	exp Numeric	engleza Numeric	franceza Numeric	germana Numeric	javascript Numeric	java Numeric	csharp Numeric	phyton Numeric	salary Numeric	gen Numeric	varsta Numeric	skill1 Numeric	skill2 Numeric	skill3 Numeric	skill4 Numeric	skill5 Numeric	**class** Nominal
1	1.0	2.5	6.0	1.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	1.0	21.0	3.0	1.0	0.0	0.0	1.0	1
2	1.0	5.0	6.0	1.0	0.0	1.0	1.0	2.0	0.0	0.0	0.0	0.0	24.0	4.0	0.0	1.0	0.0	0.0	1
3	1.0	7.5	24.0	2.0	0.0	0.0	1.0	1.0	0.0	1.0	0.0	0.0	22.0	3.0	2.0	0.0	0.0	0.0	1
4	1.0	10.0	6.0	1.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	1.0	23.0	5.0	0.0	0.0	0.0	0.0	1
5	1.0	10.0	6.0	1.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	22.0	5.0	0.0	0.0	0.0	0.0	1
6	2.0	0.0	24.0	1.0	1.0	1.0	2.0	0.0	1.0	0.0	0.0	1.0	26.0	3.0	1.0	0.0	0.0	1.0	1
7	1.0	5.0	6.0	1.0	0.0	0.0	0.0	3.0	0.0	0.0	0.0	0.0	22.0	4.0	0.0	1.0	0.0	0.0	1
8	1.0	7.5	6.0	1.0	1.0	1.0	2.0	0.0	1.0	0.0	2.0	1.0	23.0	5.0	0.0	0.0	0.0	0.0	1
9	1.0	10.0	24.0	1.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	22.0	0.0	0.0	3.0	1.0	1.0	1
10	2.0	0.0	6.0	2.0	2.0	1.0	2.0	1.0	0.0	0.0	0.0	1.0	26.0	3.0	1.0	0.0	0.0	1.0	1

Figure 3. Representation of the data: examples of instances from the class 1, with the profile highlighted (item 5).

IV. EXPERIMENTS

The results obtained with each of the classifiers depend very much on how the data set is created and how complex they are (the number of attributes, the number of classes).

Initially, the dataset was created with 5 classes and 15 attributes, and next grow up to 21 attributes; the number of instances was 100, with 20 candidates per class.

The classifiers used were some of those already implemented in Weka: KNN (with K=1 and K=3), LDA, TREE J48, NB (Naïve Bayes) and SMO (i.e. SVM).

Data were divided such as obtaining more testing schemes:
- 10 folds CV for performing crossvalidation,
- Train All-Test All,
- 90% train-10% test,
- 80% train-20% test and
- 70% train-30% test.

The first manual tests in Weka provided very good results (between 22.55% and 0% classification error), but without utility in real life. The way the data was created was without large variations from the model profile, and the small number of classes represented an advantage in the classification. Although these results could be used in the application, the central purpose was to test the classifiers on complex datasets to be used in real-life situations. For this reason, ***the second data set*** was created, with 17 classes and 18 attributes, as presented in previous sections. The first tests with the final dataset were also manually made in Weka, but the maximum error obtained was 55.77% (and the minimum 0%). It has also been observed that according to the *Train All-Test All* scheme, the lowest errors were reached. After performing these first attempt tests, the next step was the adaptation of the Matlab classifiers to the final data set. The "Stats" toolbox contains some of the previous classifiers, and multiple tests were run with a number of 16 classifiers.

Table 4 shows best results, i.e. with the errors below 10% and obtained with the KNN scheme (modifying one parameter at a time). The idea of these multiple tests was to find the most appropriate classifiers to be used in the final application. Being a complex dataset, it was decided to implement several algorithms, the final answer being based on the majority of the results these classifiers provided.

Table 4. Different KNN types providing error below 10%

The classifier with the error below 10%	k	Distance parameter	Rule parameter
KNN5	1	Euclidean	Nearest
KNN6	1	Cityblock	Nearest
KNN7	1	Cosine	Nearest
KNN8	1	Correlation	Nearest
KNN9	1	Euclidean	Consensus
KNN10	1	Cityblock	Consensus
KNN11	1	Cosine	Consensus
KNN12	1	Correlation	Consensus
KNN13	2	Euclidean	Nearest
KNN14	2	Cityblock	Nearest
KNN15	2	Cosine	Nearest
KNN16	2	Correlation	Nearest
KNN17	3	Cityblock	Nearest
KNN18	3	Cosine	Nearest
KNN19	3	Correlation	Nearest

Finally, after several tests were performed, it was concluded that all 16 classifiers will be kept with their permanent monitoring and the possibility of changing the code immediately to move to only 4 or 5 classifiers (or any other desired number, up to 16). Of the elected classifiers, the first 4 are LDA, KNN, TREE and NB, the remainder up to 16 representing KNN with varying parameters.

Table 5, Table 6 and Table 7 shows the results of the tests performed with: LDA, KNN, NB and TREE, with the data set divided in 3 different ways. Analyzing these results, one can notice that LDA offers the lowest error, while KNN offers the worst results.

Table 5. Error obtained with the Train All-Test All setup

Setup	Error (%)			
	LDA	KNN	NB	TREE
all 172 instances, but only based on first 15 attributes	9.30	11.05	17.44	11.63
all 172 instances, with all 18 attributes	10.47	8.72	12.21	11.00
only 121 instances: 70%, with all 18 attributes	7.44	9.09	14.05	12.40

Table 6. Error obtained with the x folds CV setup

Setup	Mean error per each folder (%)			
	LDA	KNN	NB	TREE
10 folds CV	**25.65**	54.70	43.72	35.55
5 folds CV	27.93	52.87	46.55	33.07
3 folds CV	29.05	55.20	47.67	43.64

Table 7. Error obtained at different disjunct data setup

Setup	Error (%)			
	LDA	KNN	NB	TREE
90% train, 10% test	29.4118	60.78	45.09	45.09
80% train, 20% test	23.5294	55.88	44.11	32.35
70% train, 30% test	**11.7647**	52.94	47.05	23.52

For the connection with the interface, it has been decided to choose several classifiers to ensure the correctness of the response.

From the entire number of classifiers selected for the final implementation of the system, the first 4 are LDA, KNN, TREE and NB, the rest up to 16 representing the KNN with various parameters (those who gave the lowest errors in previous tests). This method assures the possibility to compute a score, i.e. a percentage showing that a candidate fits or not in a particular class with a degree. The end user will receive a response based on the class selected considering this score and the decision of the classifiers.

Table 8 highlights the main advantages and disadvantages in the use of 16 classifiers: the main disadvantage is that the 13 KNNs are not among the most suitable for this dataset, as can also be noticed from the previous tests.

Concerning the data from line 1, the 16 classifiers only strengthen the final decision. On line 3, one can observe the problem that may arise in a large number of classifiers: with the same classifier at the base (KNN), the results obtained are not very varied, which in the present case is a disadvantage due to the wrong estimation of the class.

The input data was chosen so that the candidate would be suitable for class 4, but most classifiers decided a higher fit with class 12, with the final answer representing the class with the maximum number of votes.

Table 8. Results obtained with the combined 16 classifiers

No.	Dataset	Class	Class predicted by each classifier	Predicted Class
1	1,10,6,100,0,0,2,0,0,22,5,0,0,0,0	9	**9,9,9,9,9,9**,9,9,9,9,9,9,9,9,9,9	9
2	0,0,24,0,1,0,0,1,1,1,1,1,44,1,1,3,0	4	**17,3,17,9,3**,3,3,3,3,3,3,3,3,3,3,3	3
3	1,10,100,1,1,0,3,1,1,1,4,44,0,1,1,3,0	4	**4,12,4,4,12,**12,12,12,12,12,12,12,12,12,12	12
4	1,5,60,1,0,0,0,2,1,0,3,0,28,2,0,2,1,0	7	**7,7,15,6,3,**3,7,7,3,3,7,7,3,3,7	7
5	3,0,60,2,0,1,1,0,0,3,4,0,20,1,0,0,3,1	16	**14,16,3,15,15,**15,16,16,15,15,16,16,15,15,16,16	15 and 16

A first conclusion that can be drawn from line 3 is that there may be better results with fewer classifiers if they are chosen from those with minimal errors. However, lines 2 and 5 contradict the previous conclusion. If tests were performed only with the first 5 classifiers, for line 2 data, it would be considered that there is an equal match between class 3 and class 17 (which represents the wrong candidate). Line 5 supports the utility of the 16 classifiers, as follows: the first 5 classifiers considered that the inputs are suitable for class 15 (but only 2 classifiers have decided in this way) while with the 16 classifiers it is concluded that the candidate is suitable for the chosen position, albeit at a small percentage of voting.

Finally, after multiple tests, it was concluded that all 16 classifiers will be kept with their permanent monitoring and the possibility of changing the code immediately to pass to only 5 classifiers (or any other desired number, up to 16).

The type of messages received in the interface are:

Based on our application, you do not match your chosen position. We are sorry!

You are suitable for this position with a score of x%. Congratulations!

You are not suitable for your chosen position, but you match, however, with a score of x% at class y.

The final results obtained by creating the link between the classifiers and the interface are highlighted in Figure 4.

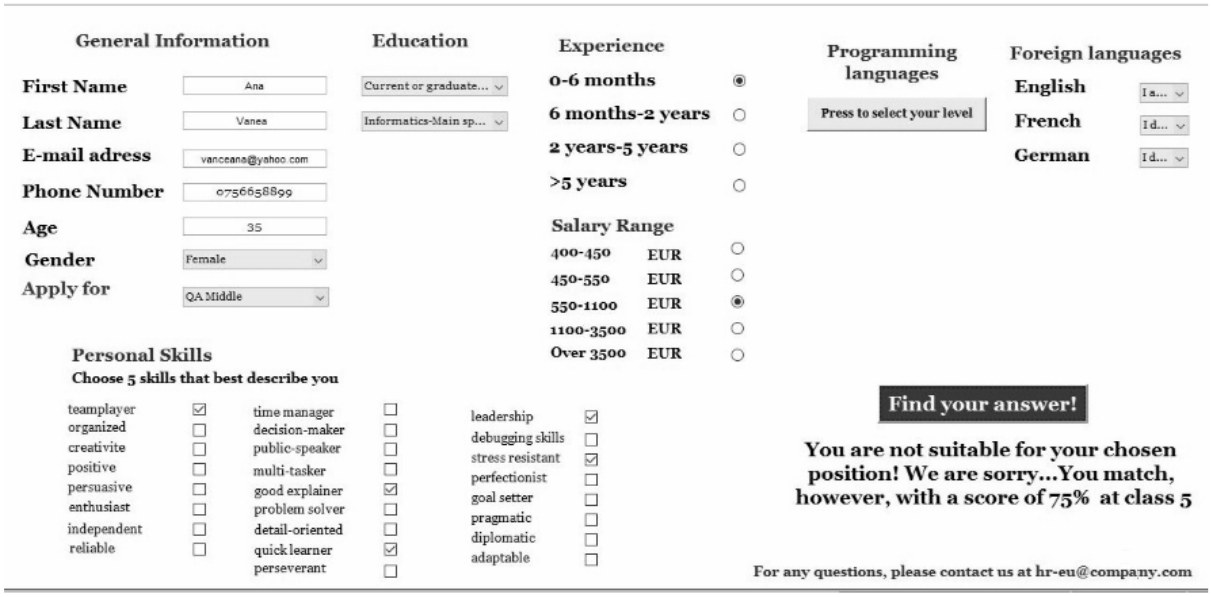

Figure 4. Application interface.

IV. CONCLUSION

The application is addressed to a particular market segment, namely companies that need support in the recruitment process, or that seek to link the candidates to their company. The ideal client for the application is large and medium size companies in terms of number of employees. Companies with large numbers of employees or companies pursuing the expansion could be interested by such an application. Another aspect that outlines the customer profile is its market notoriety. This could translate into a possible large number of applicants. Companies with Internship programs at different times of the year are also ideal customers, as there will be a large number of candidates in those times, and the application could be useful. Attention is also directed to companies that want to build closer ties with potential future employees. In terms of using the application for purposes other than personal recruitment, here, ideal customers are companies that have large inventory of information and who want certain automation processes.

To use the existing Weka algorithms, a file with an .arff extension is needed. This file must adhere to a specific template with exact and concise data, so specific formatting were required.

REFERENCES

[1] Business magazine, [Online] http://www.businessmagazin.ro/business-hi-tech/elementele-definitoriiale-anului-automatizarea-si-imbinarea-realitatilor-16056979 [Accessed: July 1, 2017].

[2] Faliagka, Evanthia, et al. "Application of machine learning algorithms to an online recruitment system." Proceeding of the International Conference on Internet and Web Applications and Services, pp.215-220, ISBN: 978-1-61208-200-4, 2012

[3] Jantan, Hamidah, Abdul Razak Hamdan, and Zulaiha Ali Othman. "Human talent prediction in HRM using C4. 5 classification algorithm.", International Journal on Computer Science and Engineering 2(8), pp.2526-2534, 2010

[4] Sarda, Vasudha, Prasham Sakaria, and Sindhu Nair. "Relevance Ranking Algorithm for Job Portals.", International Journal of Current Engineering and Technology 4.5, pp.3157-3160, 2014

[5] Sivaram, N., and K. Ramar. "Applicability of clustering and classification algorithms for recruitment data mining." International Journal of Computer Applications 4.5, pp. 23-28, 2010

[6] Keenan, Peter, et al. "Human resource management DSS." International Conference DSS2004.,pp.525-534, 2004