Applying Machine Learning Techniques to Implement a Decision Support Mechanism for Human Resources Recruitment

Henrique Delgado
Nuno Horta
Instituto de Telecomunicações
Instituto Superior Técnico
Lisboa, Portugal

Abstract— This work describes a new approach based on machine learning techniques to automatically evaluate candidates to open positions in IT companies and its integration on the recruitment processes. This approach comprises four main layers: a processing layer, a clustering layer, a classification layer and an output layer. The processing layer receives CVs, extracting their features in order to be used in the latter layers. The clustering layer studies the underlying structure of the company while the classification layer is responsible for the evaluation of an incoming candidate, relying on the structured data obtained from the previous layer. The first implements the K-Means clustering algorithm while the latter implements the KNN classifier. At last, the final layer generates several outputs displayed in a web application, describing the evaluated candidate and its adequacy for the company. This work shows promising results regarding its usefulness as a tool to guide the recruitment process, standing out from similar systems by the deeper analysis performed on each candidate and simpler user interfaces.

Keywords—Clustering, Classification, Decision-Support, Machine Learning, Recruitment Process, Web Application

I. INTRODUCTION

For the past century, human resources management and the overall recruitment process have evolved significantly. In order to qualify companies to follow the fast-growing pace in HRM and stay competitive, there must be a significant investment in the integration of technology, alongside a more strategic and future-driven approach.

As far as the recruitment process evolution, the candidate selection process and human recruitment have been slowly changing; however, there is still huge potential and room for improvement. From jobs being only advertised in newspapers and recruiters relying mainly on personal connections, to the integration of personal computers and the internet, the process efficiency increased and significantly broadened the candidate pool. From the 2000s until today, almost everything is performed online and the whole recruitment process is digitalized, focusing on social media, company career sites, mobile apps and online job boards. In the past, the lack of resources and information was the main barrier between the candidates and the recruiters. Nowadays, the vast amount of information available is the issue. Companies are being increasingly overloaded with applications while most recruitment processes still rely on personal interviews, which have a cost and take time. Choosing to tackle this problem by allocating human resources to process each candidate application is no longer a viable option if a company wants to be competitive in recruiting the best candidates. This work studies the advantages of applying machine-learning techniques to the recruitment process of a company, by implementing a decision-support system. The software is projected to process a dataset of approximately 150 collaborators' CVs and, based on the information obtained, be able to synchronously find the best candidates, and how they compare themselves with the rest of the company. Besides the processing component of the proposed system (back-end), the system will provide a web application (front-end), displaying useful information for the recruiter. The system aims at greatly innovating the recruitment process but not at replacing human analysis. Although the proposed software is applied to the human resources at Polarising, this thesis aims to give an overall picture of the best solutions to this problem that can be applied to all companies.

This paper is organized as follows: Section II presents related work in this topic and background knowledge, Section III describes the system architecture chosen, Section IV presents the obtained results, metrics used to evaluate each component of the system and several tests performed, Section V presents the study conclusions.

II. STATE-OF-THE-ART

In this section, a considerable portion of previous research relative to this topic is described. The section is divided into three parts: inputs and pre-processing, outputs and system approaches. The inputs and pre-processing's subsection describes the multiple inputs that different systems consider and how they manage to organize the information. The outputs' subsection describes the most common outputs that current solutions give. At last, in the system approaches' subsection, is detailed how state-of-the-art systems are implemented and how they solve the problem of the candidate selection. The reference

selection criteria was based on previous works that relate the most to the problem in hand, trying to demonstrate all the possible strategies used in current systems.

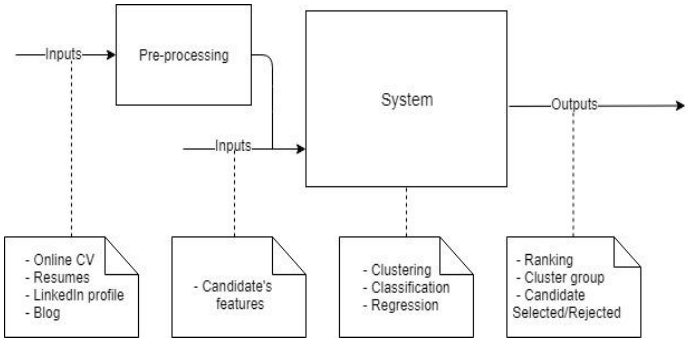

Fig. 1. State-of-the-art systems' structure.

A. Inputs and Pre-Processing

In order to characterize each candidate, some systems consider directly the candidate's features/attributes [1] and [5], whereas others consider inputs that require pre-processing before the main processing component [2], [3], [4] and [6]. Regarding direct inputs, related work selects a set of parameters, for example technical skills such as Java and C language or Database management and System design, each parameter having a rank/level of knowledge associated with the attribute. On the other hand, inputs requiring pre-processing mostly choose resumes/CVs as main entries to characterize each candidate. System [4] uses resumes collected from *indeed.com*, extracting a predefined set of keywords from the document. Based on those keywords extracted from all resumes, the system builds a term document matrix representing the frequency count of words among all resumes. Each row represents one resume, each column represents a word and each entry represents the frequency count of a word in a resume. Afterward, a pre-processing task removes punctuation marks and stop words, and converts upper-case letters to lower-case in order to standardize the matrix. Both systems [3] and [6] choose online forms/CVs alongside with the job position the candidate is applying to as inputs. Besides, [3] integrates a LinkedIn profile and the candidate's blog to perform a deeper analysis of each candidate, providing a user interface for the candidate's application process.

B. Outputs

Every related state-of-the-art system strives to provide the most valuable information to the recruiter by generating a set of outputs. Every output identified in previous works can belong to one of two categories: score/ranking or selection/rejection of a candidate regarding a certain job position. Systems [3] and [4] obtain score/ranking outputs, [3] providing a web application to the recruiter and [4] obtaining a cluster-based ranking (CBR). Based on the term document matrix obtained in subsection A, system [4] uses ReliefF algorithm [7], identifying top scoring features for feature selection. Besides, the scores obtained can be applied as feature weights to be used by the system. Then, each word count C(i) is multiplied by the corresponding weight W(i), and after the sum of all the products, the CBR is obtained (1).

$$CBR = \sum_{i=1}^{63} C(i) * W(i) \quad (1)$$

On the other hand, systems [1], [6], [3], [2] and [5] output if the candidate is suitable or not for a certain job position. Besides, [6] finds which class the candidate suits more.

C. System Approaches

This section describes how state-of-the-art systems process their inputs. From previous works, system approaches can be divided into two different learning methods: Supervised Learning and Unsupervised Learning.

1) Supervised Learning

In supervised learning the goal is, given a set of input variables and the corresponding output variables, to learn the function *f* that maps the input to the output (2).

$$Y = f(X) \quad (2)$$

Based on the mapping function and considering some new input data X, it is possible to predict the output variables Y for that data. Supervised learning is typically performed in the context of classification [2], [1], [5] and [6], mapping input to discrete output labels, or regression, mapping input to a continuous output. Systems [2] and [5] use decision tree (Fig. 2) classifiers [8], where internal tree nodes denote a test on an attribute, where each branch represents the outcomes of the test and the leaf node represents the class labels. From the generated tree, several classification rules can be deduced.

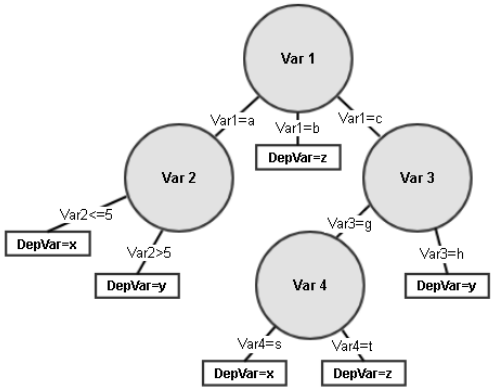

Fig. 2. Decision tree example.

In [2] the selected classifiers are Id3 and C4.5 whereas in [5] only C4.5 is tested. System [5] goal is to discover employees' performance patterns from the existing employees' performance data through the classifier. From the decision tree built, 43 classification rules are derived from 590 instances in order to bind each candidate to a rule. On the other hand, system [1] calculates frequencies and probabilities, applying Naïve Bayes theorem to compute the most probable class regarding each candidate. From the features extracted for each candidate, system [6] tests several classifiers such as KNN, LDA, J48, Naïve Bayes and SVM. K-Nearest Neighbors algorithm receives input vectors in a multidimensional feature space, each instance with a given label, and comprises two phases. The training phase consists only in the storage of the feature vectors with corresponding labels, while the classification phase requires a specified value *k* and a new un-labeled instance. First,

the algorithm calculates the distances between each point in the feature space and the new instance to be classified. Several distance metrics can be selected; however, the Euclidean distance is the most common (3).

$$d(p,q) = \sqrt{(a_1-b_1)^2 + (a_2-b_2)^2 + \cdots + (a_n-b_n)^2} \quad (3)$$

Then, based on the chosen parameter *k*, the algorithm obtains the k-shortest distances between every data point and the instance to be classified (Fig. 3). In order to make a prediction on the instance's class, the algorithm calculates the most common class concerning the k elements selected. Regarding the *k* parameter selection, usually larger values of k reduce the noise effect on the classification but make boundaries between classes less distinct.

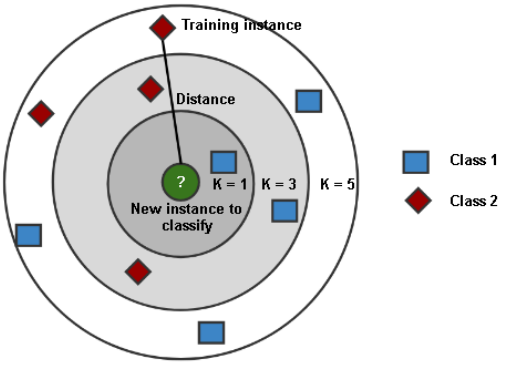

Fig. 3. KNN classification example.

System [3] studies three approaches of regression algorithms: linear regression, regression tree and support vector regression. The linear regression algorithm finds the optimal parameter vector w that minimizes the regression error (4). However, this approach is not appropriate when selection criteria interact in a complex and non-linear way.

$$y_i = w^T x_i + e \quad (4)$$

Regression trees can be described as decision trees having continuous instead of discrete variables. Lastly, support vector regression's goal is to find a function f that minimizes the expected error according to the unknown probability distribution of the data (5).

$$f(x) = \sum_{i=1}^{N} a_i K(x, x_i) + b, b \; and \; a_i \; \epsilon \; R \quad (5)$$

2) Unsupervised Learning

Unsupervised learning tries to model the underlying structure or distribution in the data, not having access to the output variables for each input data. This learning method is typically performed in the context of clustering.

Systems [2] and [4] both use K-Means clustering algorithm [21] whereas in [2] Fuzzy C-means is also tested. K-Means selects k centroids in a multidimensional feature space, each centroid representing each cluster mean or center. Then, the algorithm iteratively alternates between two steps: assignment and update. First, each object is assigned to the cluster they are most similar with, based on the distance between the object and the cluster mean. Then, the new mean is computed for each cluster, each mean being calculated as the average coordinates of all points in the cluster. The algorithm iterates until the criterion function converges, generally when the distance between the previous state centroid and the current state centroid is smaller than a predefined convergence epsilon, for all *k* clusters. On the other hand, Fuzzy C-Means is a method of clustering which allows one piece of data to belong to two or more clusters. This algorithm works by assigning membership to each data point based on the distance between the data point and each cluster center. The closest the point is to the cluster center; the more is its membership towards that cluster center. The sum of membership of each data point should be equal to one. In [4], all the resumes have been categorized into one of the K clusters, determined by the Elbow method [9], and each resume goes into the cluster with the nearest mean.

D. Conclusions

	Paper	Inputs	System	Outputs	Accuracy
Clustering Algorithms	[4]	Resumes	K-means	Ranking	*Good (no evidence)*
	[2]	Knowledge Base	Fuzzy C-means	Candidate's Cluster Group	52.1% - 63.1%
			K-means		53.5% - 69.6%
Classification Algorithms			Id3	Selected/Rejected	45.1% - 50.3%
			C4.5		76.7% - 79.1%
			CART		72.1% - 77.3%
	[1]	Candidate's features	Naïve Bayes	Selected/Rejected	*Good (no evidence)*
	[5]	Candidate's features	C4.5	Selected/Rejected	77.0%
	[6]	Online CV + Desired job position	LDA	Selected/Rejected	89.5% - 92.6%
			KNN		89.0% - 91.3%
			Naïve Bayes		82.6% - 87.8%
			TREE		87.6% - 89.0%
Regression Algorithms	[3]	LinkedIn profile + Blog + Online CV	LR	Ranking	~70.0%
			M5 Tree		~70.0%
			REP Tree		~65.0%
			SVR		~75.0%

Table. 1. Comparison between state-of-the-art systems and accuracies achieved.

Table 1 displays the results obtained from every system approach presented, providing an overview of the state-of-the-art implementations in order to decide which direction to take in the solution's architectural design. Regarding inputs, most systems consider CV's derived information. Regarding clustering algorithms, due to the nature of the data, state-of-the-art systems did not achieve good results. However, with the adequate data sets, K-Means is a very promising algorithm and important to further test in this context. Concerning classification and regression algorithms, trees-based algorithms are among all the most common and accurate, C4.5 being one of the best. Yet, Naïve Bayes, KNN, LDA and SVR also achieved significant accuracies. At last, most systems output either a ranking or a selected/rejected response regarding a candidate and a given job position.

III. System Architecture

This section describes the system's architecture by characterizing its different layers, how they communicate with each other and with external entities.

A. *Interaction with existing Functional Blocks*

The system within this thesis scope is mainly Brain, with CVs and Candidates being already implemented. However, some changes are required for the latter systems in order to enable the integration of all applications.

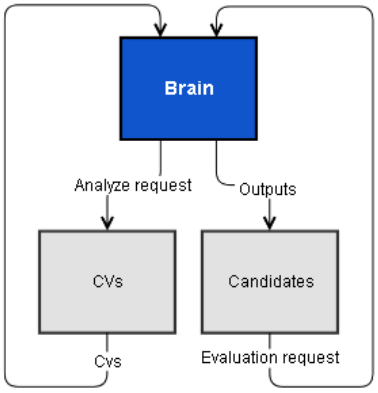

Fig. 4. Interaction between functional blocks.

1) *Authentication*

In order to securely communicate between applications, every functional block must have an authentication layer, distinguishing secure peers from insecure ones. Since the interaction between applications will be through requests and responses, each request must be sent with a valid access token. Then the system receiving the request must validate whether the requesting system has access to the intended resource. If it does, the system responds with the required information, if it doesn't, the system will receive a message informing it is unauthorized to access the resource. In order to provide the above behavior, Keycloak [10] was chosen, allowing single sign-on between applications with identity and access management.

2) *CVs*

The CVs block is where the data and CVs from all Polarising's collaborators are stored and organized. Regarding the company's context, it provides a web application where each collaborator can view and update their own CV.

3) *Candidates*

The Candidates' block works as the bridge between all the main recruitment processes and the candidates. This block receives applications from new candidates each day and for each application will trigger the evaluate function call in order to characterize/analyze an incoming candidate.
This block is the main communication block with the proposed system, displaying most Brain's outputs.

4) *Application Programming Interface (API)*

The project's API implements two different functionalities: analyze and evaluate. The analyze function will be triggered when Brain starts running or by user input at its web application, requesting all collaborators CVs to the CVs application, analyzing the received data. This function call should always be executed when the characterization methodology changes, for example, changing the importance of an attribute/feature.

On the other hand, the evaluate function call will be triggered by the candidates' block and, upon receiving the request, Brain's will evaluate the incoming candidate, producing a set of outputs and replying them to the candidates' web application. The implemented system compares the candidate to all collaborators, outputting the top individuals who match closer to the candidate, outputs several rankings based on the candidate's features, predicts the candidate's class, finding the candidate's and the predicted cluster main attributes.

5) *Technologies*

Based on the technologies used in Polarising and the application requirements, the system is implemented mainly in Java language using the Spring framework [11]. However, some components related to the system's front-end are implemented in TypeScript, CSS and HTML using Angular framework [12].

B. *Internal System Structure (Brain)*

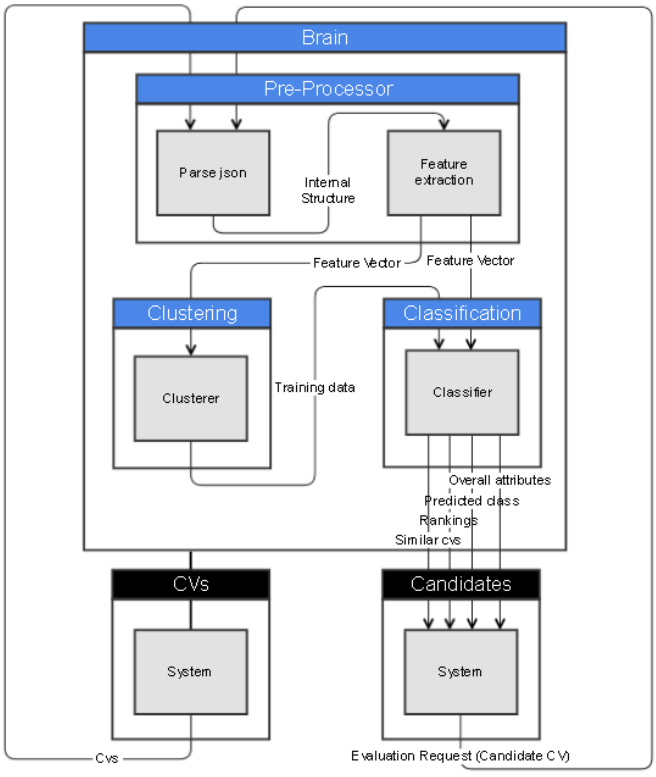

Fig. 5. Brain system architecture.

In this section is explained how each component of the system is structured and implemented. As described in section II, we can divide it into inputs and pre-processing, outputs and system approach.

1) *Inputs and Pre-Processing*

First, Brain requests all the available data to the CVs system (analyze request) before performing any other task. Then, based on the data received, each collaborator CV will be pre-processed and converted into a known format by the system, feeding the clustering stage. After the system initialization is performed, Brain is ready to receive evaluation requests. This happens by a Candidates' block evaluation request, sending to Brain a candidate's CV. The incoming CV will be pre-

processed the same way as any other company CV, feeding the classifier. Based on the obtained training data through the clustering stage along with the pre-processed candidate's CV, the classification stage is performed, obtaining several outputs and replying them to the Candidates application.

2) Outputs

The system will generate the following outputs:

1. Overall attributes
2. Similar collaborators
3. Rankings
4. Predicted cluster overall characteristics

The overall attributes represent the main characteristics the candidate possesses, concerning several predefined categories of evaluation. Based on those categories the system searches for relevant information in order to characterize the individual and identifying its strengths and weaknesses. Similar collaborators correspond to a list of the more resembling collaborators regarding an evaluated candidate. Based on the characteristics and attributes of each CV, the system finds the closest individuals from the whole company. In order to obtain the referred list, the system calculates the shortest distances (3) between the candidate and all collaborators during the KNN classification stage. For each incoming candidate, several rankings are calculated. These rankings evaluate how adequate a candidate is for a certain open job position. This ranking will be based on a chosen threshold and the distance between the candidate (i) and the points representing the requirements for the given positions (j). If the distance between the candidate and a job position is less or equal than the threshold, the ranking is max (ranking = 1), if the distance is greater than the threshold the ranking is equal to the division between the threshold and the given distance (7). The more attributes the candidate matches with the job position, the distance between the two points decreases and his ranking increases.

$$ranking = \min\left(1, \frac{threshold}{distance(i,j)}\right) \quad (7)$$

At last, the system outputs a class prediction for a certain candidate, representing the cluster where the candidate is more adequate to belong based on his characteristics. Also, for that class prediction, the cluster overall characteristics are obtained through the same strategy as the calculation of the candidate overall attributes. After calculating all the above outputs, a response will be sent to the Candidates' web application, displaying the data to the recruiter.

3) System Approaches

In terms of the system approach, we can specify two phases: a clustering phase and a classification phase.

First, we need to focus on the existing company's CVs, preparing them to be used as training data in the latter phase.

The **clustering stage** occurs at the system's boot and every time the web application's user intends to re-evaluate the company's distribution. Since no CV is already labeled, the first stage is to label each collaborator's CV by applying a clustering algorithm. From the study of state-of-the-art systems, K-means is chosen as the clustering algorithm. K-Means requires three parameters to be defined: the number of clusters k, initial centroids and a convergence epsilon (stop criteria). The k parameter determines how many clusters the algorithm finds, representing in how many groups the input data will be split. This work studies several choices of k, identifying which is more adequate given the dataset in hands. Based on that value, k initial centroids are chosen.

Regarding K-Means initial centroids there are two different strategies to consider: random centroids vs. chosen centroids. When considering random centroids, is possible to poorly generate initial points that result in a bad cluster distribution and increasing the clustering error. The clustering error can be defined as the sum of the distances between each element and the corresponding cluster centroid. Regarding chosen centroids, the quality of the approach depends on the choice of initial centroids. After the centroids are generated, K-Means starts the iterative process, checking each iteration if the algorithm converged. In each iteration, the algorithm clears the previous state before any computation. Then, each data point is added to their nearest mean. Afterward, is computed the new centroids for each mean based on the average coordinates of each point in the corresponding cluster. At last, the algorithm calculates the squared distance between each new centroid and the previous iteration centroid, converging if all the distances are smaller than a convergence epsilon. If one or more distances are higher than the stopping criteria value, the algorithm continues iterating until reaching convergence or the maximum number of iterations predefined.

Based on the feature vectors obtained from the inputs and pre-processing stage, K-Means finds the optimal distribution of Polarising's collaborators, labeling each instance to the identified cluster. After the algorithm converges, a method of organizing the clusters' information is performed. This method consists of identifying, for each cluster, the overall characteristics as referred in subsection 2). For each mean, a new cluster with the above information is created. Then each field is calculated based on its centroid coordinates' values. The arranged clusters' list and corresponding information will be later displayed on the system's web application.

After the clustering task is done and all CVs are labeled, the training data set is constructed and used to train the classifier.

The **classification stage** is triggered every time a candidate needs to be evaluated during the recruitment process. Since Polarising's main goal is to find out how each candidate compares himself with the existing company personnel and how well he fits for the open job positions, KNN will be used in order to find the nearest collaborators of a certain candidate and classify the candidate into a certain class/group.

First, a request will be received by the system at a specific endpoint, representing a pending classification request. The request will contain the candidate's CV in the request body. Then, the candidate's CV will be parsed into the defined internal structure the same way each collaborator CV is parsed and, based on the parsing result, the system will generate the corresponding feature vector as referred in subsection A).

From the candidate's feature vector and labeled training data obtained in the clustering stage, the system will perform the classification task, calculating all the required outputs. As referred in section II, K-Nearest Neighbors algorithm calculates the *k* closest points in the feature space to the instance to be tested, predicting its class based on the predominant class from the *k* neighbors. KNN algorithm considers only one user

configurable parameter: number of k neighbors. Based on the selection of k, the algorithm finds the k closest points to the instance being evaluated.

First, the system calculates the Euclidean distances between the candidate and each instance in the training data. Then, the list with the distances and the corresponding element is sorted ascendingly (smallest distances first). Based on the parameter *k*, the k nearest neighbors are selected. At last, the predicted class is calculated based on the most frequent class between the k elements.

C. Web Applications

This work comprises two web applications regarding the main communicating systems: Brain and Candidates.

1) Brain

The Brain web application serves the purpose of providing statistical information regarding Polarising's current workforce distribution, alongside with some adjustable parameters regarding the system's clustering task.

The user can adjust the weights assigned to each attribute/feature and to change the number of clusters to be found by the system. Furthermore, the clusters/groups found are displayed and characterized regarding:

1. Technical skills' categories
2. Work experiences' job positions
3. Group elements

Based on this data, the user will have a better understanding of the company's distribution by knowing which person is in each group and what characteristics defines them. Additionally, this information may be very valuable in identifying where the most skilled individuals are, and where the company lacks resources, guiding the recruitment process with the purpose of strengthening the company's workforce.

2) Candidates

On the other hand, the Candidates' web application displays the outputs generated by the proposed system. There are three main views regarding the application: candidates' view, candidate's edition and candidate's detail.

The candidates' view displays all current candidates enrolled in the recruitment process, along with some candidate specific information, such as personal information, status, area that he applied and the source of his application. The candidate's edition screen allows filling in the information about the candidate, such as personal information, technical skills and work experiences. At last, by clicking the view button on a candidate, the candidate's detail screen is rendered. After the click, if the candidate contains any technical skills or work experiences, an immediate request is sent to the proposed system, synchronously displaying brain's outputs regarding that candidate.

IV. System Validation and Results

A. Evaluation Metrics

In order to assess the quality of each strategy is necessary to define several evaluation metrics for the two system components: clustering and classification. Regarding the clustering component, each strategy is evaluated by the elbow method, silhouette analysis and runtime. On the other hand, the classification component is evaluated through its outputs, validating the results with several candidates against the company's reality and personnel.

1) Elbow Method

The elbow method is useful to validate the number of clusters chosen for the k-means clustering. This method runs k-means on the dataset for a range of k values and, for each value of k, calculates the sum of squared errors (SSE) between each point in a cluster and the corresponding cluster centroid. Then, with a line chart plot of the SSE for each k value we can identify the point that resembles an elbow. The goal is to find the smallest number of clusters that give us a low enough SSE. As we increase the number of clusters, the SSE decreases, reaching zero when the number of clusters equals the number of data points in the data set. However, the plot will not always show a clear elbow, showing more like a smooth curve. In this scenario, other methods may be needed for a better understanding of the problem, such as the computation of silhouette scores.

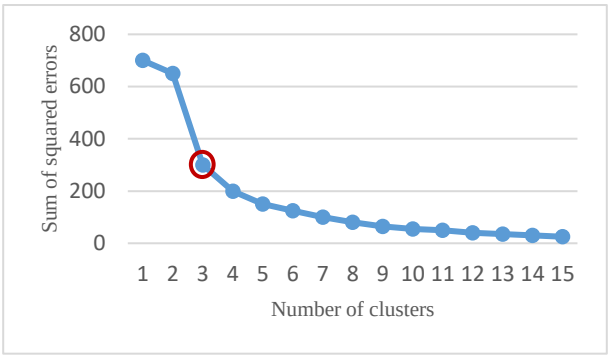

Fig. 6. Elbow method example.

Besides helping to decide the number of clusters (K), the SSE for the corresponding K value chosen, can be a metric to compare different implementations.

2) Silhouette Analysis

Silhouette is a method that provides a graphical representation of how well each object has been clustered. This method comprises two notions: cohesion and separation**.** Cohesion measures how similar an object is to its own cluster and separation measures how similar an object is to the other clusters. The silhouette score/value ranges from -1 to 1, where a higher value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

3) Runtime

At last, the clustering algorithm running time is evaluated, testing for different scenarios and input datasets size how the system responds.

4) Outputs Validation

Based on several candidates' scenarios created and the comparison against real company data, is possible to evaluate the classification stage correctness through the outputs obtained for each instance tested. This stage evaluation will be tested through the validity of the open position rankings, the similar

list of collaborators and the predicted cluster obtained, considering the overall attributes of the candidate. For the open position rankings, is expected that the candidate scores higher in positions where he matches the requirements better, and scores lower in positions where he possesses fewer skills required by the job offer. For the list of collaborators and the predicted cluster, is expected that, based on the candidate's attributes, it contains individuals with the same main characteristics.

B. Clustering Results

Every result obtained in this sub-section regarding the clustering stage considers all available Polarising collaborators' CVs (~130 CVs), testing the system with real data in order to evaluate the system in a real world scenario. In order to fully evaluate the clustering component and establish a baseline for the system to start, several parameters need to be evaluated and compared relative to the metrics used. The parameters are features, weights**,** centroids selection and number of clusters.

1) Features

The base selected features are age, technical skills, work experiences, educational level and languages. Several combinations of features selection were performed, varying the number of clusters chosen and selecting random and defined initial centroids.

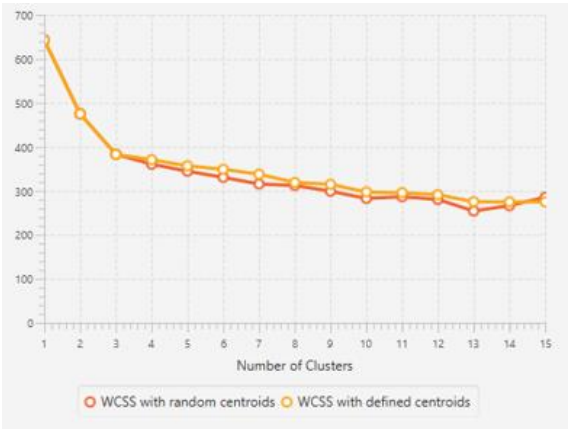

Fig. 7. Elbow method selecting all attributes.

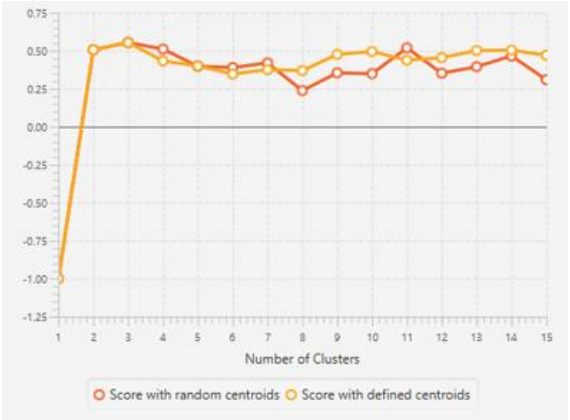

Fig. 8. Silhouette method selecting all attributes.

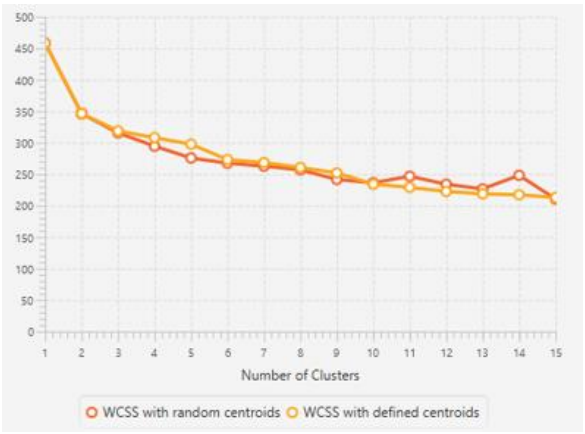

Fig. 9. Elbow method selecting technical skills and work experiences.

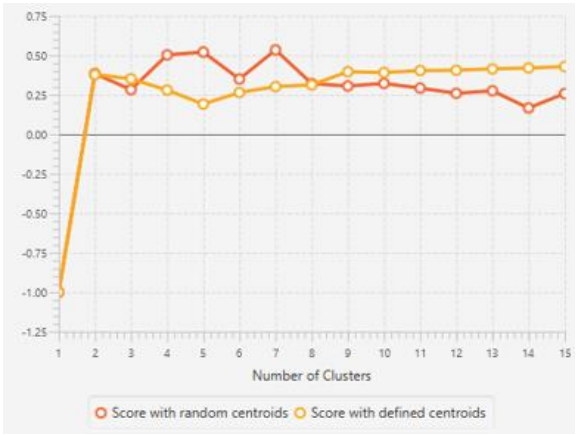

Fig. 10. Silhouette method selecting technical skills and work experiences.

The results obtained show that selecting fewer attributes, the within cluster sum of squared errors (WCSS) decreases and generally the silhouette score is higher due to the decrease of the complexity of the problem. Moreover, the selection of technical skills and work experiences seems to be the reason behind higher clustering errors. Even though selecting the age, educational level and languages attributes obtains the smallest clustering errors, it provides no useful information since the focus is to understand the company's distribution regarding skills and experience.

2) Weights

Regarding the weights' selection tests, all base attributes are considered, varying only the importance of each feature in order to extract useful information of each configuration.

Based on the results below, increasing the importance given to certain attributes increases the within cluster sum of squared errors, not having a significant effect on the silhouette scores. Specifically, increasing the weight of technical skills, dramatically increases the WCSS due to the high number of attributes (~70) associated with this feature, increasing the complexity of the problem.

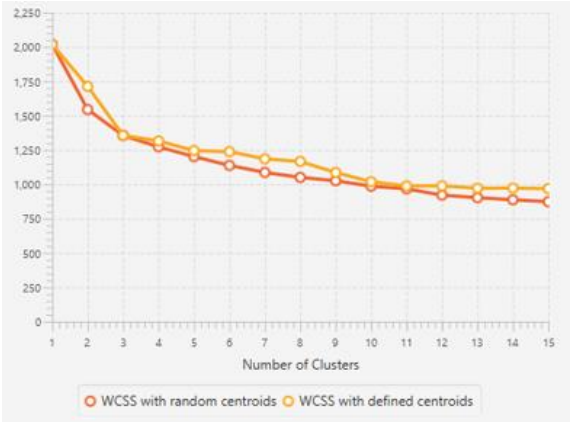

Fig. 11. Elbow method with technical skills and work experiences having twice the weight of the remaining attributes.

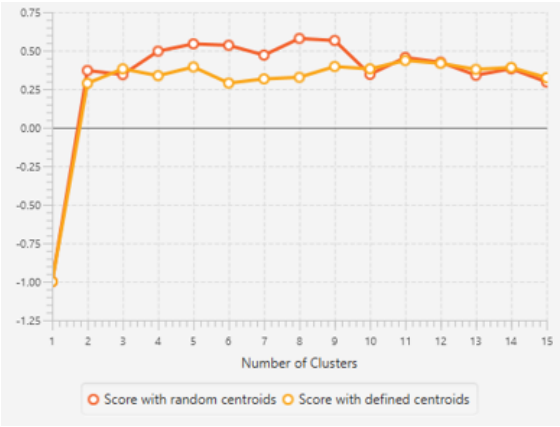

Fig. 12. Silhouette method with technical skills and work experiences having twice the weight of the remaining attributes.

3) Discussion

Overall, random initial centroids obtain lower WCSS values and higher silhouette scores since choosing adequate initial centroids is a hard problem. Regarding the selection of the number of clusters to consider in K-Means, based on both Elbow and Silhouette methods, K = 4 or 5 seems to be the best choice. However, it might be useful from a company's perspective to select a higher value of K in order to emphasize the distinction between collaborators in the corresponding K clusters. As observed by the results, increasing the weights of the attributes increases the within cluster sum of squared errors, however, it might be useful from a company's perspective to perform such evaluation.

C. Classification Results

In this sub-section is chosen the more adequate configuration for Polarising needs, considering the results obtained from the previous case study - selecting only the technical skills and work experiences' attributes, both having the same weights. Regarding the KNN algorithm chosen parameter K, is selected to return the five nearest neighbors of the evaluated instance. In order to validate the system's classification stage, two candidates were tested. The outputs will be evaluated regarding similar collaborators, open position rankings, predicted cluster technical skills and work experiences. Candidate I represents a real Polarising collaborator whereas Candidate II is a fabricated candidate based on Polarising open job positions. Every list of similar collaborators is hidden in order to prevent the disclosure of private data regarding Polarising's personnel.

1) Candidate I

Candidate I is mainly characterized for having many skills in software development and a vast experience as a developer and consultant. Regarding similar collaborators, all individuals are adequate since they belong to the same areas of expertise and have identical years of experience. Concerning the open positions, candidate I matches better with positions having mainly skills in software development, such as full-stack developer and software engineer, having lower scores in positions related to integration, such as integration consultant and senior integration consultant.

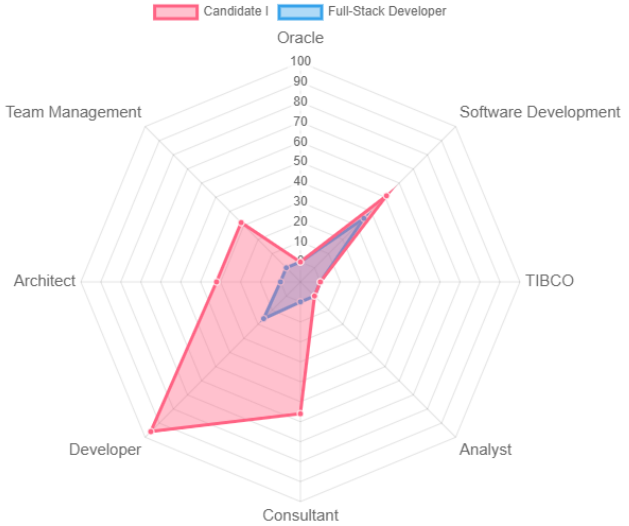

Fig. 13. Candidate I's comparison against Full-stack developer open position.

However, based on the radar charts' observation, is clear that Candidate I has significantly more skills than required for the Full-stack developer job position, achieving an average score on that position. The more accurate the match is between the candidate and the open position; more likely he will be adequate for that given position. At last, regarding the predicted cluster, are displayed both cluster technical skills and work experiences. Comparing the candidate's attributes to the cluster specific information, Candidate I is placed in a group with a higher skill in Software Development and a vast experience as a Consultant and Developer, matching his attributes.

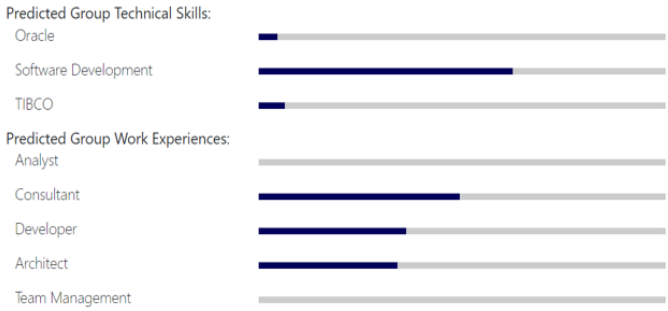

Fig. 14. Candidate I's predicted cluster overall characteristics.

2) Candidate II

Candidate II has fewer overall skills and work experiences than the previous candidate, standing out his skills in Oracle and

TIBCO technologies, along with some years of experience as a Consultant and Developer. Regarding similar collaborators, as said before, all individuals are adequate since they belong to the same areas of expertise and have identical years of experience. Regarding the open position rankings, Candidate II obtains high scores for multiple job positions, highlighting the Integration/BPM Consultant open position. Observing figure 14, we can see a close match, leading to a higher ranking.

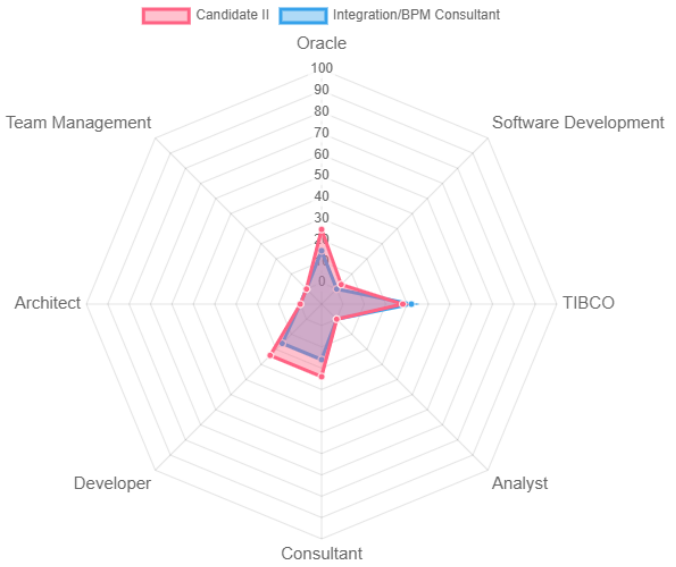

Fig. 14. Candidate II's comparison against integration/BPM consultant open position.

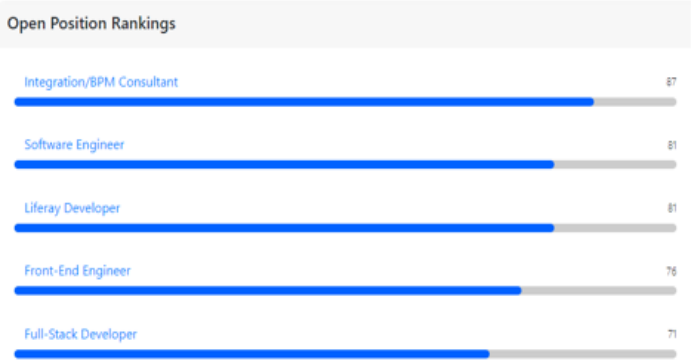

Fig. 15. Candidate II's open position rankings.

Lastly, Candidate II is placed in a cluster having some skill abroad all technical skills, highlighting TIBCO technology, and some experience as a Consultant, Developer and Analyst. This cluster doesn't match all the Candidate's attributes, however, possesses his main characteristics, such as some skills in Oracle and TIBCO, along with some experience as a Developer and a Consultant.

3) Discussion

Through the several tests performed, different system components were displayed and validated. First, the radar chart describing the main characteristics and attributes of a candidate provides a clear, quick and visual understanding of an individual. Then, the open position rankings explain how adequate a candidate is for a given job, which can be also validated through the radar charts comparison between the candidate and the job description. The similar collaborators give insight on how a candidate compares himself with the rest of the company and the predicted cluster overall characteristics details the company's group where the candidate is more adequate to fit in. With all this information the recruiters have at their disposal tools that can significantly boost their process and increase its efficiency.

D. System Scaling

The proposed system is projected to process around 130 CVs as its input, representing the approximate number of Polarising's collaborators. However, is important to test the system to a bigger scale, analyzing the effects on the clustering task with a larger CV dataset. The classification task will not be tested since there is no need to request evaluation for more than one candidate at each time. Regarding the classification task with random initial centroids, in order to prevent a bad initial state, the system runs the algorithm several times, each run with different randomly generated initial centroids, saving the scenario with the lowest clustering error. Doing so, obtains a better solution in regards to the company's distribution, however, has a significant time cost for larger input datasets (bigger companies). Several tests were performed varying the input dataset size and observing the resulting clustering runtimes.

1) Runtime variation

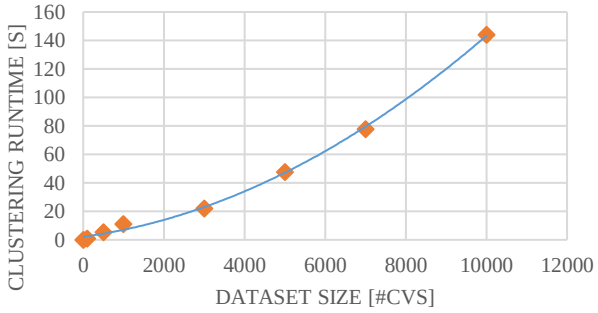

Fig. 15. Clustering runtime variation with random initial centroids.

Based on the results obtained, the clustering runtime displays a quadratic curve in relation to the dataset size. However, for a dataset of 10000 CVs, the clustering runtime is still acceptable with a period of approximately 2min20s. On the other hand, if the system considers defined initial centroids, there is no need to run the algorithm several times since the initial state will be the same. This way, the runtimes decrease drastically that, increasing the dataset size, the clustering runtime does not surpass 1s. Therefore, for a larger dataset, choosing the initial centroids is suggested.

2) Discussion

This case study shows that regarding Polarising's specifications and needs, the implementation with randomly generated initial centroids is better than the implementation with defined initial centroids because, as stated before, obtains lower clustering errors and similar clustering runtimes for a small scale. However, for larger input datasets, choosing random initial centroids may not be viable due to the quadratic

increase of the clustering runtimes. In this scenario, defined initial centroids would be a more adequate approach. At last, independently from the scale of the input dataset, the quality of classification task and the overall solution remains intact, considering valid input data.

V. Conclusions

Considering the emerging difficulties that HR departments are facing nowadays with the overload of data received, machine learning techniques seem to be the most promising approach in order to handle this phenomenon. Hereby, the proposed solution intends to handle the number of CV's reaching companies every day by building a decision support system based on machine learning techniques.

Based on the related work, a system was proposed and implemented comprising two stages: a clustering stage, and a classification stage. Several tests were performed, testing the different components of the system with different initial assumptions, evaluating runtime and correctness. The optimal solution regarding the clustering component was obtained by choosing random initial centroids to the K-Means algorithm and selecting only technical skills and work experiences as attributes, considering the size of the company involved. Regarding the classification component, the proposed solution is based on KNN outputs. This component uses K equal to five (five nearest neighbors), producing the class prediction for a candidate, alongside with three other outputs: a visual representation of the candidate's main characteristics, rankings relative to the company's open positions and the graphical comparisons between each candidate and each given position.

The system outputs proved that the recruitment process has much to gain from machine learning, as it improves the process efficiency and the overall candidates' analysis by performing a more deterministic procedure. Comparing the solution obtained to other similar systems, this work stands out by performing a deeper analysis on each candidate, having user-friendly interfaces and providing the tools to achieve a clearer idea of the company's underlying structure and personnel.

References

[1] M. Jannat, S. S. Chowdhury and M. Akther, "A probabilistic machine learning approach for eligible candidate selection," International Journal of Computer Applications, vol. 144, no. 10, Jun. 2016.

[2] N. Sivaram and K. Ramar, "Applicability of clustering and classification algorithms for recruitment data mining," International Journal of Computer Applications, vol 4, no. 05, pp. 23-28, July 2010.

[3] E. Faliagka, K. Ramantas and A. Tsakalidis, "Application of machine learning algorithms to an online recruitment system," presented at the seventh international conference on internet and web applications and services. 2012. pp. 215-220.

[4] Mayuri Verma, Cluster based ranking index for enhancing recruitment process using text mining and machine learning. International Journal of Computer Applications, vol. 157, no. 9, Jan 2017.

[5] H. Jantan, A. R. Hamdan and Z. A. Othman, "Human talent prediction in HRM using C4.5 classification algorithm", International Journal of Computer Applications, vol. 02, no. 08, pp. 2526-2534, 2010.

[6] A. Apatean, E. Szakacs and M. Tilca, "Machine-learning based application for staff recruiting," Acta technica napocensis, vol. 58, no. 04, 2017.

[7] R. J. Urbanowicz, M. Meeker, W. LaCava, R. S. Olson and J. H. Moore, "Relief-based feature selection: Introduction and review," Journal of Machine Learning Research, pp. 189-203, Nov. 2017.

[8] B. Hssina, A. Merbouha, H. Ezzikouri, and M. Erritali, "A comparative study of decision tree ID3 and C4.5," International Journal of Advanced Computer Science and Applications, pp. 13-19, 2014.

[9] P. Bholowalia and A. Kumar, "EBK-Means: A clustering technique based on elbow method and k-means in WSN," International Journal of Computer Applications, pp.17-24, Nov. 2014.

[10] JBoss, "Keycloak", [Online]. Available: https://www.keycloak.org/ [Accessed 12 September 2019].

[11] Pivotal Software, "Spring", [Online]. Available: https://spring.io/ [Accessed 12 September 2019].

[12] Google, "Angular", [Online]. Available: https://angular.io/ [Accessed 12 September 2019].