

# Incremental Learning Based Intelligent Job Search System

By

Fahmida Afrin

Irin Nahar

A thesis submitted to the faculty of BRAC University in partial fulfillment of the requirements  
for the degree of Bachelors of Science in Computer Science and Engineering

School of Engineering & Computer Science

Department of Computer Science & Engineering

BRAC University

## Declaration

We hereby declare that this thesis is based on the results we found. We have gone through numerous reference papers from national and international researchers. Materials of work found by other researchers are mentioned by reference. We have added all the necessary mathematical formulas, and figures in this paper. This thesis, neither in whole nor in part, has been previously submitted for any degree.

Signature of the Supervisor

---

Professor Dr. Md. Zahidur Rahman

Signature of the Authors

---

Fahmida Afrin

---

Irin Nahar

## **FINAL READING APPROVAL**

I have read the thesis on “Incremental Learning Based Intelligent Job Search System” by Fahmida Afrin and Irin Nahar in its final form and have found that its format, citations, and bibliography are acceptable. The materials including figures, tables, and charts are in place and the final result is satisfactory. The thesis has been prepared under my Supervision and is a record of bona-fide work carried out successfully.

Approved by

---

Supervisor

Professor Dr. Md. Zahidur Rahman

## **Acknowledgement**

We are glad to thank our thesis supervisor Professor Dr. Md. Zahidur Rahman who spent his valueable time to supervise our thesis work. We are thankful to him for his support and encouragement over the past year. We would also like to express our gratitude to our respected chairperson Dr. Md. Haider Ali for his support. We had major issues such as collecting original data of job seekers and job providers, the accepted and rejected CV for certain jobs and many more. Our Respected chairperson and our faculty Md. Moin Mustakim has showed us ways and helped us getting these confidential data by obeying all the official privacy rules and regulation of the organizations. We also would like to show our gratitude to our respected faculty Abu Mohammad Hammad Ali, for helping us in the mathematical and training part of our research.

We are grateful and humbled by the Department of Computer Science & Engineering, HR Department, and Office of Career Services and Alumni Relations (OCSAR), of BRAC University for providing us confidential data to train our system.

Lastly, we are thankful to our family, friends and everyone who helped for their endless support and love.

## **Abstract**

In today's world of smart phones and smart machines, our job search systems are still not smart enough. With the increasing population, and the rapid growth of internet usage, online job search is not a new issue anymore. From our very own bdjobs.com to the recent loosemonkies.com, there are many online based job sites. The sites still rely on the Boolean matching, whereas they have large amount of data of job seekers and recruiters. Our goal is to use this CV, Job and recruitment information to train the job search system incrementally, exactly the way, human beings learn. With every successful recruitment, and with every failed approach, the job search system will learn to predict, learn to behave how a recruiter does while recruiting an employee. Thus, not only the job seekers, the recruiters can rely on the job search system too for the best selection of employees.

# CONTENTS

ABSTRACT .....	v
LIST OF FIGURES .....	ix
CHAPTERS	
<b>1. INTRODUCTION .....</b>	<b>1</b>
1.1 Motivation.....	2
1.2 Goals.....	2
1.3 Methodology and Approaches.....	2
1.4 Literature review.....	3
<b>2. RELEVANT WORK</b>	
2.1 An expert system for job matching .....	4
2.1.1 Fuzzification and rule base evaluation .....	4
2.1.2. Training .....	5
2.2 Agent Based Job search system in android environment .....	5
<b>3. Job Search Theory and Machine Learning</b>	
3.1. Job Search Theory .....	6
3.2 Machine Learning .....	7
3.2.1 Supervised Machine Learning .....	7
3.2.1.1 Support Vector Machine (SVM) .....	8
3.2.2 Incremental Machine Learning	
3.2.2.1 Incremental Learning .....	10
3.2.2.2 Batch VS Online Incremental Learning .....	11

3.2.2.3 Schotastic Gradient Descent (SGD) .....	12
3.2.2.4 Average Schotastic Gradient Descent (ASGD) .....	13
3.2.2.5 Bipartite graph .....	14

#### **4. PROPOSED FRAMEWORK**

4.1 System Description .....	15
4.1.1 Data Flow of the System .....	15
4.1.2 Applicant Part .....	16
4.1.3 Recruiter Part .....	16
4.1.4 Parameters, Rule Base, and normalization of data .....	16
4.1.5 Job Matching system .....	18
4.1.5.1 Learning based Job matching system .....	19
4.1.5.2 Incremental Learning with ASGD .....	20
4.1.5.3 Bipartite Graph for optimal number of matching .....	20
4.1.6 Job Search Flow .....	21
4.2 Justification of Parameter Choices and Rule Base .....	22
4.2.1 Justification of Parameter choice .....	22
4.2.2 Justification of Normalization of Data .....	22
4.2.3 Justifying the machine learning algorithm ASGD .....	23

5. Implementation Using Django and Scikit Learn	
5.1 Interface using Django .....	23
5.2 Implementing Learning algorithm using Scikit-Learn .....	24
6. TRAINING AND TESTING	
6.1 Data Collection Methodology .....	25
6.1.1 Past history of job recruitment .....	25
6.1.2 Past history of rejected applicants by the recruiter .....	25
6.2 Training Dataset .....	25
6.2.1 Problem of over fitting .....	26
6.2.2 k-fold cross validation .....	26
6.3 PERFORMANCE	
6.3.1 PERFORMANCE ESTIMATION	
6.3.1.1 Performance with K-fold cross validation .....	28
6.3.3.2 Decreasing Error rate with increasing sample .....	29
6.3.2 Testing using interface .....	30
7. DISCUSSION .....	31
8. FUTURE WORK .....	32
8. REFERENCES .....	33



List of Figure:

Fig: 2.1.1a Converting input to Neuro-Fuzzy Network using Fuzzification rule .....	5
Fig: 3.2.1a Supervised Learning Model .....	8
Fig: 3.2.1.1a Support Vector Machine .....	9
Fig: 3.2.2.2a Batch Learning VS Online Learning .....	11
Fig: 3.2.2.5a Bipartite Graph .....	14
Fig: 4.1.1a Data Flow of the System .....	15
Fig: 4.1.4a Fuzzy Rules and Rule Based Evaluation .....	17
Fig: 4.1.4b Normalization of Data .....	18
Fig: 4.1.5.3a Maximum match for bipartite graph .....	20
Fig: 4.1.6a Job Search Flow .....	21
Fig: 4.2.3a Comparison of the Online incremental Algorithms .....	23
Fig: 6.3.1.1 a Performance estimation for with k=5 fold cross validation .....	28
Fig: 6.3.1.1b Performance estimation for with k=5 fold cross validation .....	29
Fig: 6.3.3.2a Decreasing Error Rate with Increasing Samples .....	30
Fig: 6.3.2a Modified pickled file after incremental learning .....	31

## 1. INTRODUCTION

In this era of robotics, from our TV to phone everything is smart. Our technology is developing and making progress in every second. Our systems are learning to think like us, humans and becoming even smarter. Machine learning is no more a new topic in the arena of technology. Supervised and unsupervised learning is now a very common topic in this world run by technologies based on Artificial Intelligence. Stochastic Gradient Descent (SGD) is an approach to discriminative learning of linear classifiers under convex loss functions such as Linear Support Vector Machines. Incremental learning using SGD is useful in many ways for large scale data learning. Incremental learning is the way how we learn, so training the machines to learn incrementally always brings excellent output.

On the other hand, finding a perfect job is still a complex task. There are many established online job search systems, but surely not intelligent. They merely suggest the matched jobs on the basis of criteria, whereas we can use incremental learning to make it intelligent and faster. With the help of bipartite graph we can make sure that most number of people gets the most suitable jobs for them. Keeping the track of who is getting the job, and who is not, will not only help the system, but also the recruiters, because in future they might not have to waste time on scanning CVs with their eyes. They will just give the criteria, and the system itself will choose the perfect candidate for the job. As the job search systems already have large amount of data, implementing incremental learning can make use of the data, and make a good use of them for the betterment of the system.

## **1.1 Motivation**

Supervised machine learning is widely used in natural language processing and image processing. However, in job search systems, learning or incremental learning is not much popular. In systems such as job search or job matching, incremental learning can add extraordinary dimensions. Moreover, matching perfect job for the seekers and suggesting them according to their chances of getting the job, actually can help the neural network to reach close to the optimal stage.

## **1.2 Goals**

Our goal is to make an intelligent Job Search System which will learn incrementally for the recruitment of applicants, and will be up to date with the recent job trend. There are many online job search systems but all they do is matching the criteria and suggest the job seekers to apply. It is more of short listing the jobs for them. The recruiters can have account on the site and can accept the CVs or download others potential CV's related to their choice. The process makes the job search and finding a little easier, but the system surely is not intelligent. Therefore, we want to develop a system based on incremental learning that can ensure optimal number of job matching.

## **1.3 Methodology and Approaches**

As there are already many popular job search websites, we had to learn about them first. We studied about those websites. Starting from our national bdjobs.com to the international job-

hunt.org, we studied all these websites. Social Networks such as Glassdoor or LinkedIn helps to search jobs as well. The study of all these systems was the first part of our research. After that, we started researching about job search theory, then machine learning, and some other areas, which will be mentioned in 1.4. Researching theoretically helped us to start our practical work. After starting to work practically we implemented the system with supervised machine learning using the Support Vector Machine (SVM) algorithm. However, to implement incremental learning, and to get better result we changed our algorithm and used Average Schotastic Gradient Algorithm. Implementing the total system with scikit-learn [39], we collected raw data, modified them according to the rule base, and trained out system. With training, we did the cross validation too, to estimate accuracy.

#### **1.4 Literature review**

Faggian(2014) in his “Job Search Theory”[1] discussed and summarized the overall important developments of job search theory and its basic models. He also elaborated matching function theory. Moreover, he explained the relation between job search theory and the migration theory. On the other hand, Gueven and Violante(2009) discussed about Joint search with multiple locations, which is based on how joint search of job can open up new opportunities and new friction by comparing it with single search agent[2]. Mortenson(1984) thoroughly discussed about job search and labor market analysis showing the wage search in real timeaccording to economics, which expresses how important the value of time is in job search[3]. Cornelißen(2008) showed an investigation of the relation between job search, job satisfaction and job changes[4]. Kumari(2012) has discussed about the selection and recruitment process in her paper[5].Syed ,Liu and Sung (1999), Ru’ping(2001) in their paper showed incremental learning with SVM[6][7]. Li, Liu, and Wang (2009) showed the incremental learning algorithm

with support vector machine based on hyper plane-distance [8]. Bottou(2010) has showed illustrated work based on stochastic learning, which is very effective for large scale machine learning or online learning[9]. Read, Bifet, Pfahringer, and Holmes(20 ) showed the practical comparison between Batch-Incremental and Instance-Incremental Learning in Dynamic and Evolving Data[10]. Bottou(2011) has showed many Stochastic Gradient Descent Tricks in [11]. Zhou, Shen, Li, Chen, Xie, and Wei (2014) have used bipartite-graph based approach for comparison [12].

## **2. RELEVANT Work**

### **2.1 An expert system for job matching**

This was an expert for the judgment of the jobseekers at vacant post. The system used Neuro-Fuzzy techniques for analyzing the parameters and to compare. Drigas, Kouremenos, Vrettos, Vrettaros, and Kouremenos(2004) in their paper explained the fuzzy logic and rule base they used[13]

#### **2.1.1 Fuzzification and rule base evaluation**

The system used Neuro-Fuzzy Techniques for the inductive training of complex fuzzy terms which are later used by be the rule base of the system and then evaluated for the job matching. The system had fixed six fields, Age, Education, Additional Education, Previous Employment (Experience), Foreign Language (English) and Computer Knowledge. There was a fuzzy rule for every field. They used binary Fuzzy Relations [14] which are the sets in  $X \times Y$  where every

element in the set is mapped to a membership grade between 0 and 1, where 0 means least significant and 1 means most significant.

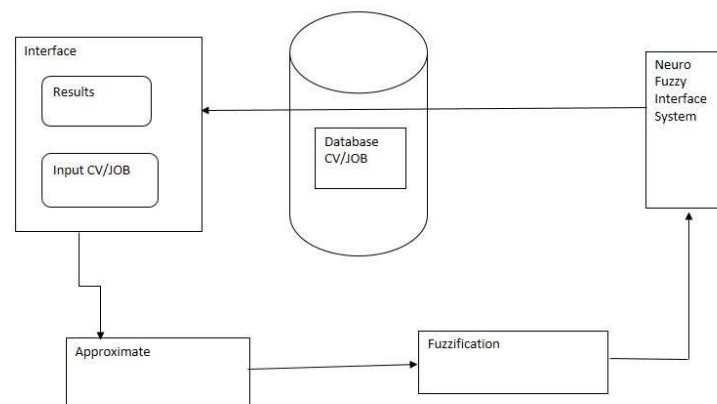


Figure: Converting input to Neuro Fuzzy Networks using fuzzification rules

### 2.1.2 Training

They trained the system with the training samples collected from the previous recruiting cases. If an applicant is accepted, then his sample is taken as positive sample, otherwise negative sample for the training. The samples are put into the training data samples using neuro Fuzzy Network. There is an input and a desirable output for each sample. After the training the system's performance is tested with test data set formed by the certain criteria.

### 2.2 Agent Based Job search system in android environment

This android based system developed by Bogle and Sankaranarayanan[2011], used fuzzy preferences rules which could dynamically find out the best suitable criteria for the job seeker.

The paper discussed the applicant part, the job search part, and the job search algorithm along with the job search theory[15].

### **3. Job Search Theories and Machine Learning**

#### **3.1. Job Search Theories**

Both in the basic job search model and Stigler's (1961, 1962) model, an applicant has more than one earning opportunities available and has to select the "best" one. However, the "strategy" to select the best job is different [1]. In job search models, the decision process is sequential and there is no "optimal" sample size because the jobs are randomly sampled one at a time and the individual stops when an acceptable job becomes available. (Mortensen 1986) said, the number of jobs sampled depends on their sequence and the sample size itself is a random variable. The basic job search model is simply an "optimal stopping rule" problem. In the basic job search models, most of the parameters are static, which means any job offer that is rejected today will be rejected tomorrow as well. [1]

The early job search models are biased where one of the main problems is to find an optimal stopping rule for job seekers. 2010 Nobel Prize winners for Economics changed the logic of allowing job seekers and recruiters to work in a single frame, they introduced matching function, which is:  $M=m(U,V)$  [1]

Here,  $m$  is a function which is unspecified,  $U$  is the number of unemployed workers,  $V$  is the number of vacant posts,  $M$  is the match between two sets. This  $m$  function is unspecified,

because it is like a black box. From [1] we get, an unemployed worker finds a job in a unit period length with probability,

$$\frac{m(U, V)}{V} \equiv m(1, \theta) \equiv \alpha(\theta)$$

And vacancy is filled with probability,

$$\frac{m(U, V)}{V} \equiv \frac{m(U, V) U}{U V} \equiv \frac{\alpha(\theta)}{\theta}$$

The parameter  $\theta = \frac{V}{U}$ . This parameter is called “labor market tightness”. According to it, with every match between an applicant and a job, there forms a “surplus”. The surplus is split through Nash bargaining process which is explained in Game Theory by John Nash(1953). According to him, in a two player contracts, they perform on the basis of their preference as their utility functions. Therefore, a high value of  $\theta$  means it will be easier for the applicant to find a job[1].

## **3.2 Machine Learning**

Machine Learning is one of the most exciting sectors in Artificial Intelligence of computer science. It teaches the machine to learn. It is about developing the learning algorithm which helps the computer to learn from data, and act according upon new data. The computer does not need to be explicitly programmed if we use machine learning.

### **3.2.1 Supervised Machine Learning**

Supervised Machine Learning is training a data set which is fixed, and the training data has set of input and output. It teaches the machine for each input we are supposed to get what response



value. It supervises the learning of the machine.  $\sum X$  is the input data set,  $\sum Y$  is the output data set. For each 'x' we have a 'y', and thus the machine is trained. Then for a new test data set, the machine will predict the output as it was trained with  $\sum X$  and  $\sum Y$ . From it, the supervised learning algorithm seeks to build a model that can make predictions of the response values for a new dataset. A test dataset is often used to validate the model.

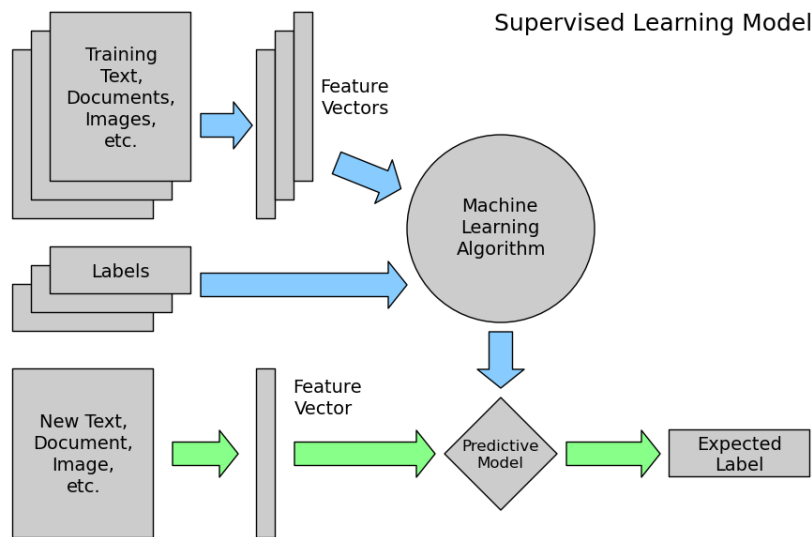


Figure: Supervised Learning Model

There are 2 types of supervised learning, classification and regression.

### 3.2.1.1 Support Vector Machine (SVM)

SVM is a kernel based technique which is used in machine learning algorithms. The kernel based technique supports the vector machines; SVM is a classifier method which can perform classification tasks by creating hyper planes in a multidimensional space that separates cases different cases of different class labels. "SVMs are a set of supervised learning methods used for classification, regression, and outlier's detection"[30].

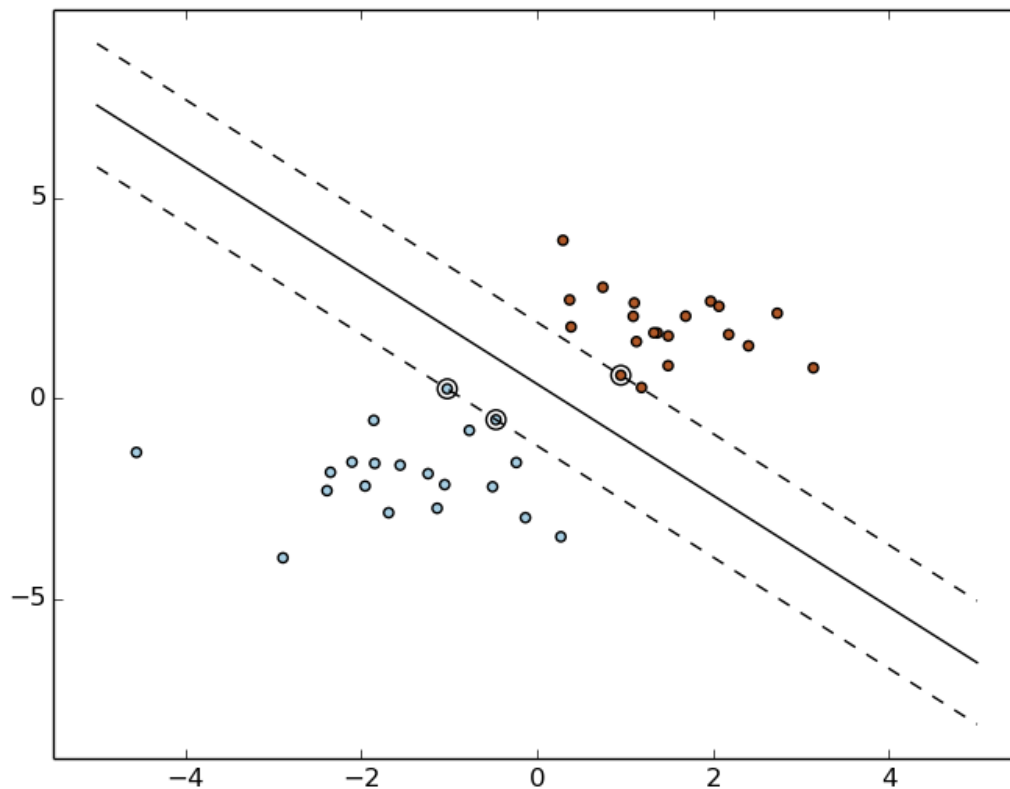


Figure: Support Vector Machine.

“A support vector machine constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space, which can be used for classification, regression or other tasks. Intuitively, a good separation is achieved by the hyper-plane that has the largest distance to the nearest training data points of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.”[30]

Ruping(2011) in his paper [7] explained how important Support Vector Machines (SVM) is. Vapnik(1995) when introduced SVM [17] said how this algorithm has been successfully used for machine learning with large and high dimensional data sets. The generalization property of an

SVM does not depend on the complete the training data, but only on the supportvectors. The number of support vectors typically is very small comparedto the number of training examples. When the data comes across the learning algorithm one by one in batches, the previous data can be compressed to their support vectors. For each new batch of data, a SVM is trained on the new data and the support vectors from the previous learning stage [7].

### **3.2.2 Incremental Machine Learning**

#### **3.2.2.1 Incremental Learning**

After Vapnik in [17,18] has introduced Support Vector Machine using Support Vectors, incremental learning has become more clear to researchers. Incremental learning is the learning process when new data are added to the system and needed to be adjusted. The main difference between traditional machine learning assumes the availability of a sufficient training set before the learning process, but Incremental machine learning learns about the training examples after they are added to the system. Incremental learning algorithm can be defined as one that meets the criteria [19]:

- “1. It will be able to learn and update with every new data-labeled or unlabeled
2. It will preserve previously acquired knowledge
3. It should not require access to the original data.
4. It will generate new class or cluster when required. It will divide or merge clusters as needed
5. It will be dynamic in nature with the changing environment.”

### 3.2.2.2 Batch Vs Online Incremental Learning

Batch Incremental Learning is when the data set learns incrementally after a batch of data is ready, then it trains the total set.

Online Incremental Learning is, data set is trained whenever a new data is arrived, and trains only with that recent training data only.

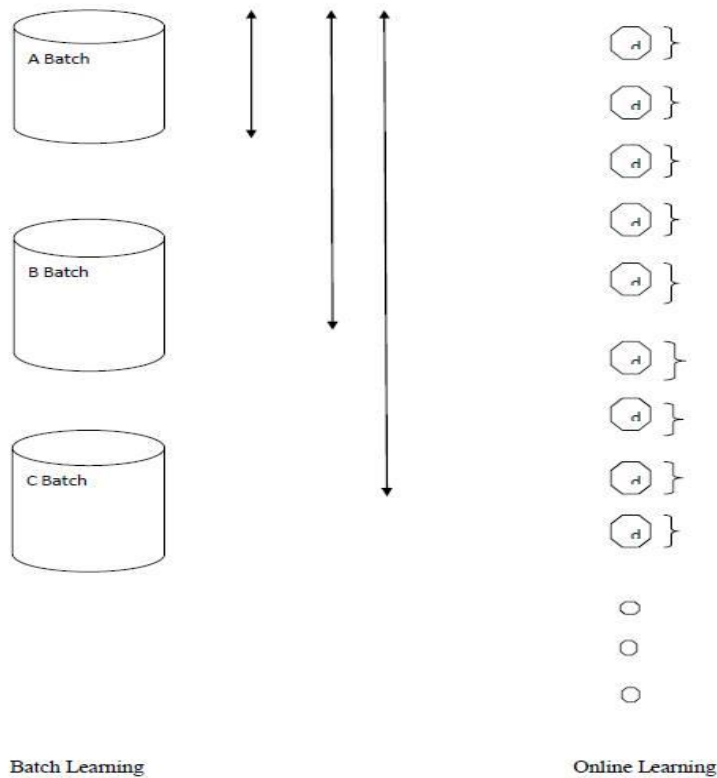


Figure: Batch Learning Vs Online Learning

### 3.2.2.3 Stochastic Gradient Descent (SGD)

Stochastic Gradient Descent (SGD) is a very efficient approach to discriminative learning of linear classifiers under convex loss functions such as linear SVM and logistic regression. SGD is successfully used to deal with large-scale of data and sparse machine learning problems. The main advantages of Stochastic Gradient Descent are efficiency and ease of implementation. It is related with the online algorithm which is very easy to implement and helps to find out the efficient perception. It is great for large-scale data. SGD picks random component of the sum above, and the number of steps required to reach given accuracy is proportional to the worst condition number (Lipshitz constant) over per-example losses.

SGD algorithms are used for a number of classic machine learning schemes. In [9] SVM was described with traditional optimization techniques. The stochastic gradient descent (SGD) algorithm is a drastic simplification. Instead of computing the gradient of  $E_n(f_w)$  exactly, each iteration estimates this gradient on the basis of a single randomly picked example  $z_t$ :  $w_{t+1} = w_t - \eta \nabla Q(z_t; w_t)$ . The stochastic process  $\{w_t\}_{t=1}^g$  depends on the examples randomly picked at each iteration. It is hoped that it behaves like its expectation despite the noise introduced by this sampled procedure [11]. The algorithm of standard SGD is given below [25]:

**Algorithm 2.1** (*standard SGD*)

Initialize  $\hat{w}_0$

**for**  $t = 1, 2, \dots$

Draw  $(X_t, Y_t)$  randomly from  $D$ .

Update  $\hat{w}_{t-1}$  as:

$\hat{w}_t = \hat{w}_{t-1} - \eta_t S^{-1} (\lambda \hat{w}_{t-1} + \phi'_1(w_{t-1}^T X_t, Y_t) X_t)$

**end**

### 3.2.2.4 Average Stochastic Gradient Descent (ASGD)

A promising fast probabilistic training method is the stochastic gradient method, for example, the SGD [9],[11]. In this stochastic gradient method, the true gradient is approximated more aggressively, by a single training or merely small amount of set. The parameter vector is updated based on the approximate gradients. The parameters of the model are updated much more frequently. Moreover, before the convergence it does not need much iteration. SGD is excellent training method compared to the batch gradient based training methods.

However, there are problems on the current SGD literature [24]:

“1) The SGD is sensitive to noise. The accuracy of the SGD training is limited when the data is noisy.

2) The SGD is not robust. It contains many hyper-parameters (not only regularization, but also decaying rate) and it is quite sensitive to them. Tuning the hyper-parameters for SGD is not a easy task.

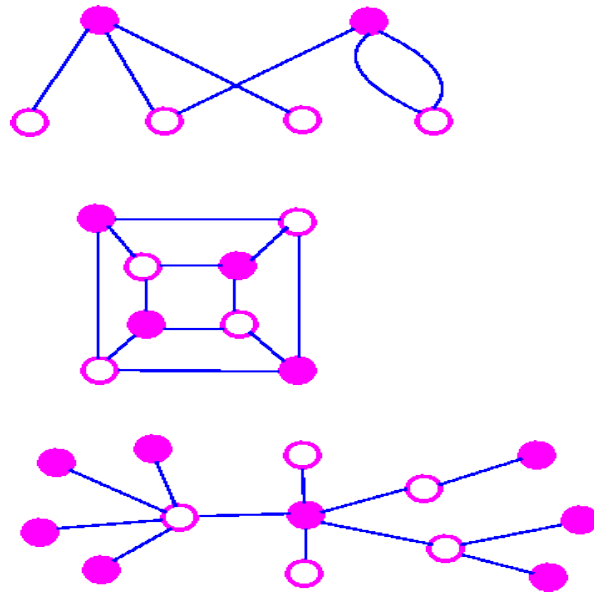
3) The speed is still not fast enough. It is time consuming to perform regularization with an online setting.”

Therefore traditional stochastic methods can be problematic, sun & others(2010), proposed a new approach, which is averaging the Stochastic Gradient Descent vectors across the iteration[24].

They performed ASGD with feedback, which they called ASF. However, ASGD itself a better version of SGD performs better than SGD, which we will get to see in the justification part [4.2.3] of this paper.

### 3.2.2.5 Bipartite graph

A bipartite graph is a graph whose vertex-set can be split into two sets in such a way that each edge of the graph joins a vertex in first set to a vertex in second set.



In bipartite graph the nodes are divided into two categories and each edge connects a node in one category to a node in the other category. For example: Administrators of a college dormitory are assigning rooms to students, each room for single student. Students and rooms are categories.

When there are an equal number of nodes on each side of a bipartite graph, a perfect matching is an assignment of nodes on the left to nodes on the right, in such a way that[27]

- I) each node is connected by an edge to the node it is assigned to, and
- II) No two nodes on the left are assigned to the same node on the right

## 4. PROPOSED FRAMEWORK

### 4.1 System Description

The system is an online program, which will have the user interface for applicants and recruiters, the databases and incremental learning based job matching part.

#### 4.1.1 Data flow of the System

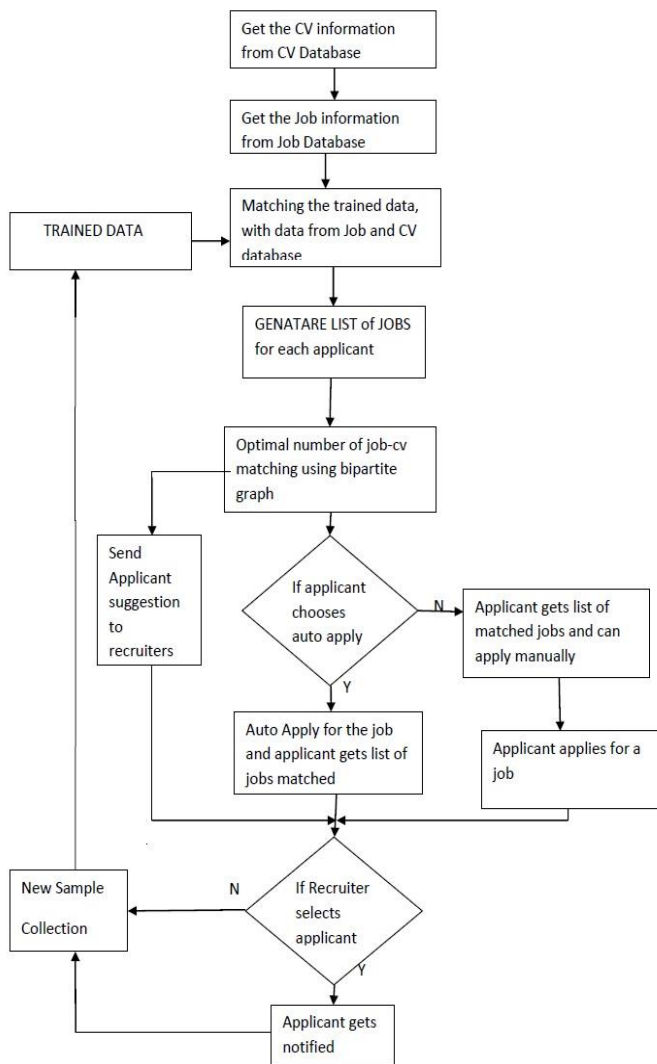


Figure: Flow chart



#### **4.1.2 Applicant Part**

The applicant Part has the Registration, login, CV edit/update facilities; they can also have the auto apply option. If the “auto apply” is on, then whenever any job matches with their job expectation, the system will apply to the job automatically on behalf of them. If they don't have the “auto apply” option on, the system will notify them, that they have suitable jobs at the site right now. After applying if the applicant gets the job, he will be notified about that.

#### **4.1.3 Recruiter Part**

The applicant Part has the Registration, login, JOB edit/update facilities. The system will notify them, whenever an employee applies for the job. He can check the CV, of the person who has applied. If he wants to give a job, he will click the button, give the job. Then the applicant will be notified. Sometimes there are perfect candidates for the job, and the recruiter will get this notification as well, that they can have this perfect employee, then they can contact the applicant, manually or through the system.

#### **4.1.4 Parameters, Rule Base, and normalization of data**

Rule Base of our system was very illustrated. We have total 14 parameters. They are:

Age, Gender, Result, Wage, Job Type, Department, Location, Experience, Foreign Language(English), Highest Degree of Education, Number of publication, Job Post, Job Level, Company Rating. For each parameter we had to choose a rule base. For example:

Job Type	Value Assignment
Part Time	0
Full Time	1
Internship	2

As mentioned above, there are other rule bases for the other parameters. After selecting the rule base, we will have to normalize the parameters and the values, the process is called fuzzification in [13].

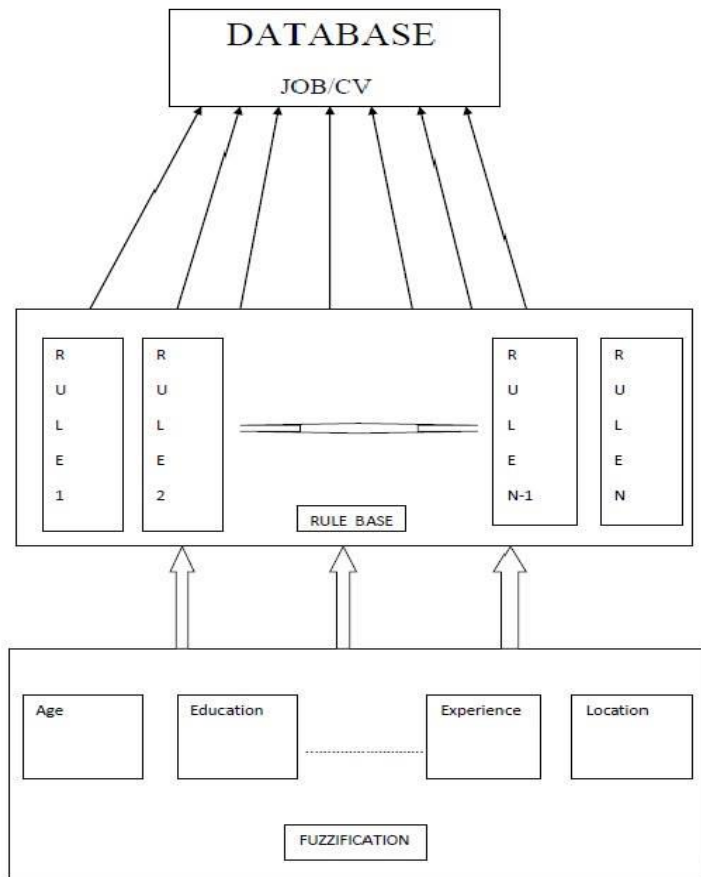


Figure: Fuzzy rules and Rule base Evaluation.

The fuzzification here in [13] is normalizing the data. Normalizing means to scale the value of a data to matrix [0, 1] range.

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

Where  $x = (x_1, \dots, x_n)$  and  $z_i$  is now your  $i$ th normalized data. As a proof of concept (although you did not ask for it) here is some R code and accompanying graphs to illustrate this point[31]:

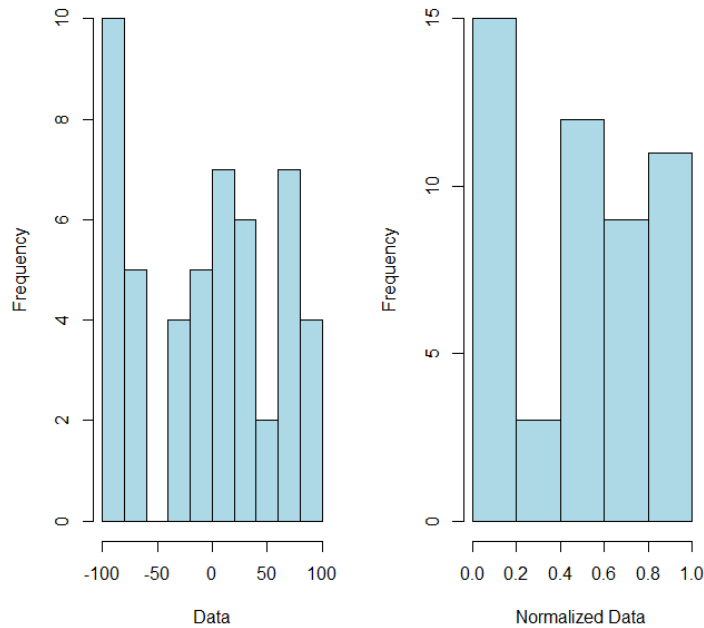


Figure: Normalization of data

#### 4.1.5 Job Matching system

This is the part where Applicant criteria will be matched with Job criteria. The matching is based on the learning of the system.

#### 4.1.5.1 Learning based Job matching

For the machine learning we have used scikit-learn [20]. We have trained our system and with their libraries. The class SGDClassifier helped implementing first order SGD learning routine. This algorithm is iterative based, and iterates over the training set. For each data set, it updates the model parameters. The update rule follows

$$w \leftarrow w - \eta \left( \alpha \frac{\partial R(w)}{\partial w} + \frac{\partial L(w^T x_i + b, y_i)}{\partial w} \right)$$

Where  $\eta$  is the learning rate which controls the step-size in the parameter space. The intercept  $b$  is updated similarly but without regularization [32].

The learning rate can be constant or gradually decaying. For classification, as we want the learning rate to be optimal we have used as it was in scikit-learn library for optimal:

$$\eta^{(t)} = \frac{1}{\alpha(t_0 + t)}$$

where  $t$  is the time step,  $t_0$  is determined based on a heuristic proposed by Léon Bottou[11] such that the expected initial updates are comparable with the expected size of the weights (this assuming that the norm of the training samples is approx. 1)[32].

From Learning, we get a set of trained data, and this trained data source is the matcher of our job requirement and cv data.

#### 4.1.5.2 Incremental Learning with ASGD

In scikitlearn library we have SGD classifier where there is ‘average’, which is a Boolean function, or int. In our program it is Boolean and set true, the function computes the average SGD weights and will store the result in the ‘coef’ attribute. Thus we will use the ASGD for incremental learning.

#### 4.1.5.3 Bipartite Graph for optimal number of matching

When the job matching part is trained by supervised learning, it can find the probable outcome; it predicts the job and CV match to be precise. Now while suggesting to applicants, the system intends to suggest job to maximum applicant. Therefore we have used bipartite graph for optimal number of matching.

While doing maximum matching in graph, the matching depends on the efficient funding of the augmenting paths. This approach fails when there are odd-cycles in the graph, but not in bipartite graphs this is not a problem because we visit each vertex only once while traversing the augmenting path [21]. Thus, the maximum matching in bipartite graph becomes easier. Here if U is the job set, and v is the cv set, we can present the matching as the figure below:

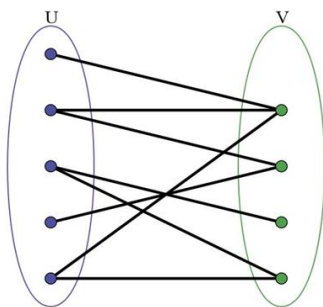


Fig: Maximum match for bipartite graph

### 4.1.6 Job Search Flow

After setting up the applicant part, the recruiter part, and the incremental learning based job matching part, we can merge the three part, and can establish the relationship as we stated in our flow chart, and the job search flow will look like:

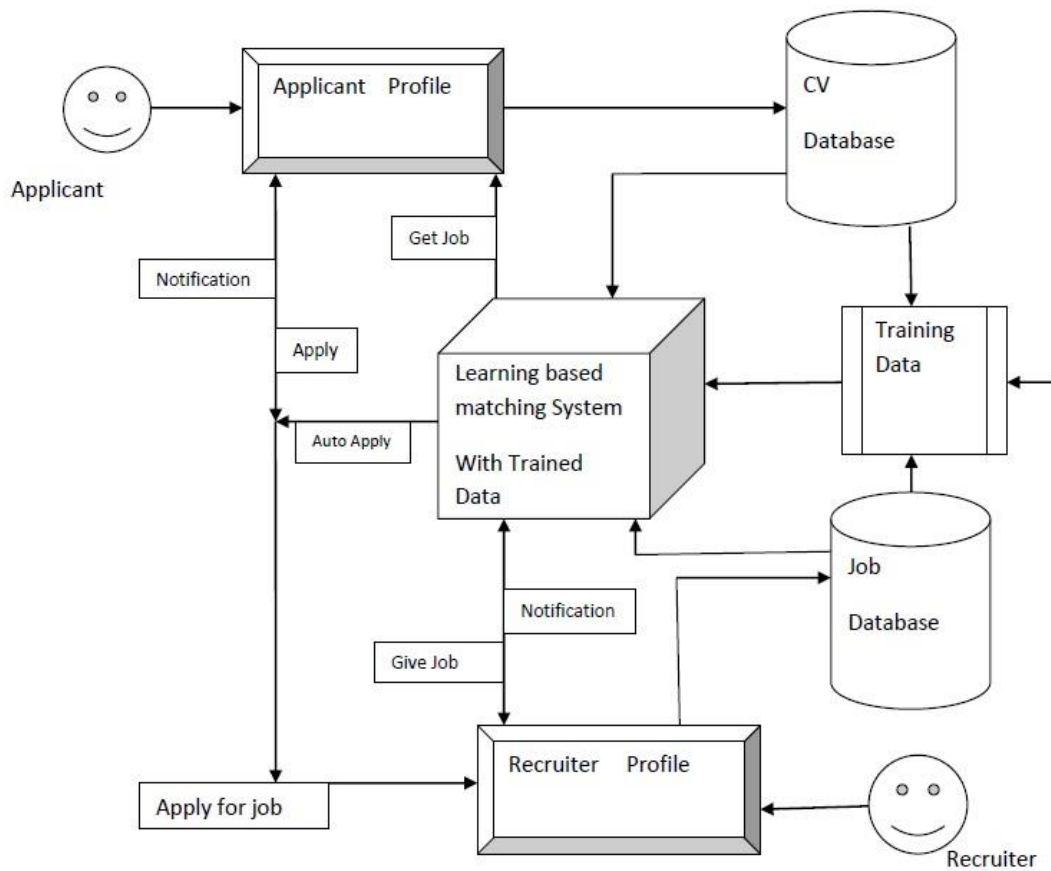


Figure: JOB SEARCH FLOW

## **4.2 Justification of Parameter and other Choices**

In this section we will discuss why chose such parameter and rule base for our system.

### **4.2.1 Justification of Parameter choice**

For our research we interviewed many recruiters, and many people who are related to job recruitment. While interviewing these people, we sorted out the common parameters they look for while recruiting. From age to location preference, there are many parameters, on which the recruiters select the applicant. While interviewing the recruiter, these were the most common and stronger basis of selecting employees of the recruiter. Therefore, we chose these 14 parameters.

On the other hand, when we were researching about different job search sites, we observed on what parameter people search their job. We even interviewed few numbers of job seekers and learnt about their process of job searching.

Thus, from real life, observation and interviews we selected our parameters.

### **4.2.2 Justification of Normalization of Data**

As the system is not tiny, and there are many parameters each having many options, we have lots of data. While comparing the data, and while training the data, we felt normalize might help us. From [13] we have seen fuzzification in neural networks, and their system gave great performance. Therefore being inspired by the system [13], and with the suggestion of scikit-learn [36] we normalized our data in the preprocessing part of the program.

### 4.2.3 Justifying the usage of machine learning algorithm ASGD

We used ASGD over SGD because[33] the error rate of ASGD is much less than normal SGD, as the figure below states:

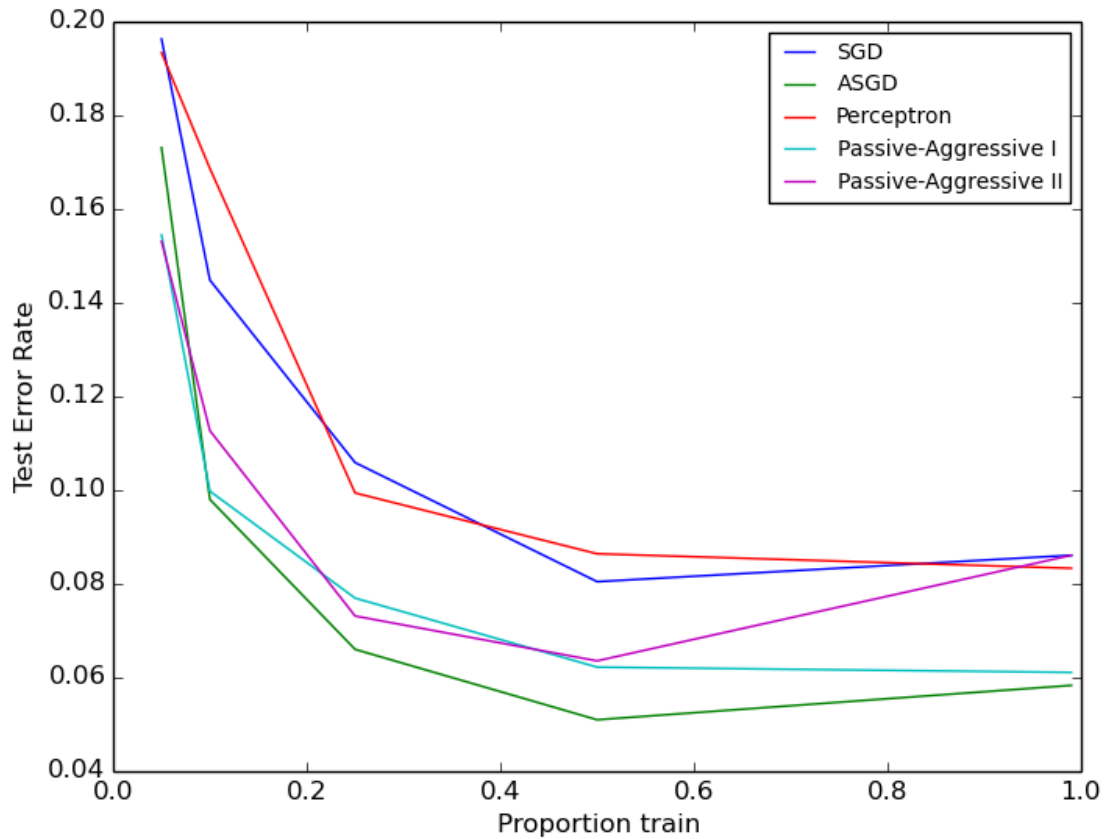


Figure a: Comparison of the online incremental Alorithm

## 5. Implementation Using Django and Scikit Learn

### 5.1 Interface using Django

Django is a high-level Python Web framework. For rapid development and clean, model based design Django is very helpful. It provides abstraction layers or models for structuring



and organizing the data of Web applications. The template layer of Django is designer-friendly and easy to use; the information to be presented to the user can be easily rendered. The manipulation of form is easier. Security is an important factor for the Web applications and Django provides multiple protection tools and mechanisms. There are many libraries which help to run a code more efficiently, faster and using a fewer resources. It is compatible with many versions of python. It offers multiple tools for the development of web applications such as authentication, caching, etc.

We have used Django models to develop our website, the job database, and the cv database. The overall system is basically developed under the framework of Django. We got mentionable help from the papers and documentation of django [22][23][36].

## **5.2 Implementing Learning algorithm using Scikit-Learn**

Scikit learn has contributed to the environment of the amazing language python library which can be used for implementing machine learning. It is different from the other language by toolbox of python. The Scikit learn depends only on numpy and scipy to facilitate easy distribution, it focuses on imperative programming. Scikit learn is very helpful for cross validation. It is accomplished by wrapping an estimator in a GridSearchCV object, where the “CV” stands for “cross-validated”. During the call to fit, it selects the parameters on a specified parameter grid, maximizing a score predict, score, or transform are then delegated.[20]

We have used the easy to use libraries, methods for normalizing, supervised learning, ASGD implementation, and cross validation. Their documentation from the website and github helped us implementing the package in our system.[30,32,36,38]

## **6. Training and Testing**

### **6.1 Data Collection Methodology**

As we had our own parameters, and data formation, we could not collect them from any research organizations. We had to collect the raw data of the cv and job circulars from te organizations.

#### **6.1.1 Past history of job recruitment**

As we mentioned before, we have interviewed many recruiters and went to organizations for the employee CV's they have employed in their organizations. We collected the CVs and took the data we needed. From the recruitment history of the organizations we took the data we needed.

#### **6.1.2 Past history of rejected applicants by the recruiter**

Finding the recruited employees were easier, but for the rejected ones, we had to go through numerous CV's, and find out who got rejected for which post. Accordingly, we had to match that, for that post who got recruited.

### **6.2 Training Dataset**

After implementing the system we had to train the system first for machine learning. In this part we will discuss the training method. Machine Learning is about learning properties of a data set

and applying them to new data set. For training, we will split the data in two sets, one is a training set, and the other is a testing set.

Usually scikit-learn saves models by using Python's built in persistence model named "pickle" file.

### **6.2.1 Problem of over fitting**

Learning the parameters of a prediction function and testing it on the same data can start a new methodological problem. A model that just repeats the labels of the samples that it has just seen is not useful at all. It would just predict anything that it has seen perfectly. However, it would not be able to predict anything it has not seen yet. This is called the problem of over fitting. To solve this problem, there is a way in machine learning experiments, to hold out part of the available as a X test set, and Y test set.

In scikit-learn `train_test_split` function is used to randomly split into training and test sets.[39]

### **6.2.2 k-fold cross validation**

When we split the training set, with the function and use estimators, still over fitting can occur because the parameters can be tweaked until the estimator performs optimally. Moreover, this can lead to generalized performance. For this problem, we can set another part of the data set as "validation set". This set will perform training on the training set. Then the evaluation will be done on the validation set. We need to continue this till we find a comparatively successful output. Then the final evaluation can be done on the test set.

As the validation set will cause the training set to be divided into three parts, it will reduce the number of samples to use for learning. Moreover the prediction outputs might depend on a particular random choice of the sets. To solve this, we have to use Cross Validation (CV).

In Cross Validation approaches, we will use k-fold Cross Validation. The training set will be split into k smaller sets. The steps of k-fold cross validation is for each k-folds:[40]

1. A model is trained using k-1 of the folds as training data.
2. The resulting model is validated on the remaining part of the data (for example: it is used as a test set to compute accuracy)
3. The accuracy reported by *k*-fold cross-validation is then the average of the values computed in the loop.

This k-fold cross validation can be taken as a computationally expensive approach. However, it will not waste data as it does in the normal cross validation due to splitting into three sets. As our data sample was comparatively small, this became a major advantage for our training.

KFold Function of scikitLearn divides all the samples into k groups of equal sizes. The prediction function is learned using k-1 folds, and the fold left out is used for test[40].

### **6.3 Performance**

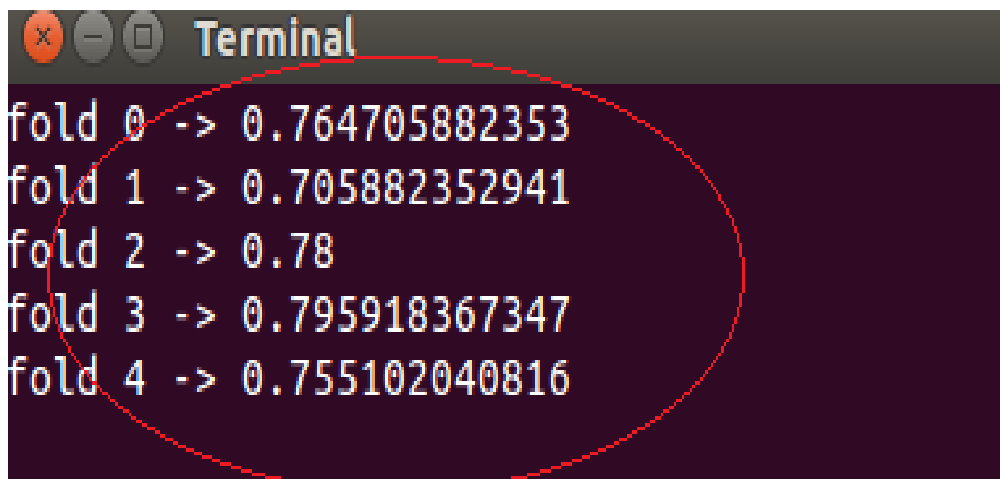
In this part we will show and discuss the test results with training data and Test result with interface.

### 6.3.1 Performance Estimation

#### 6.3.1.1 Performance with K-fold cross validation

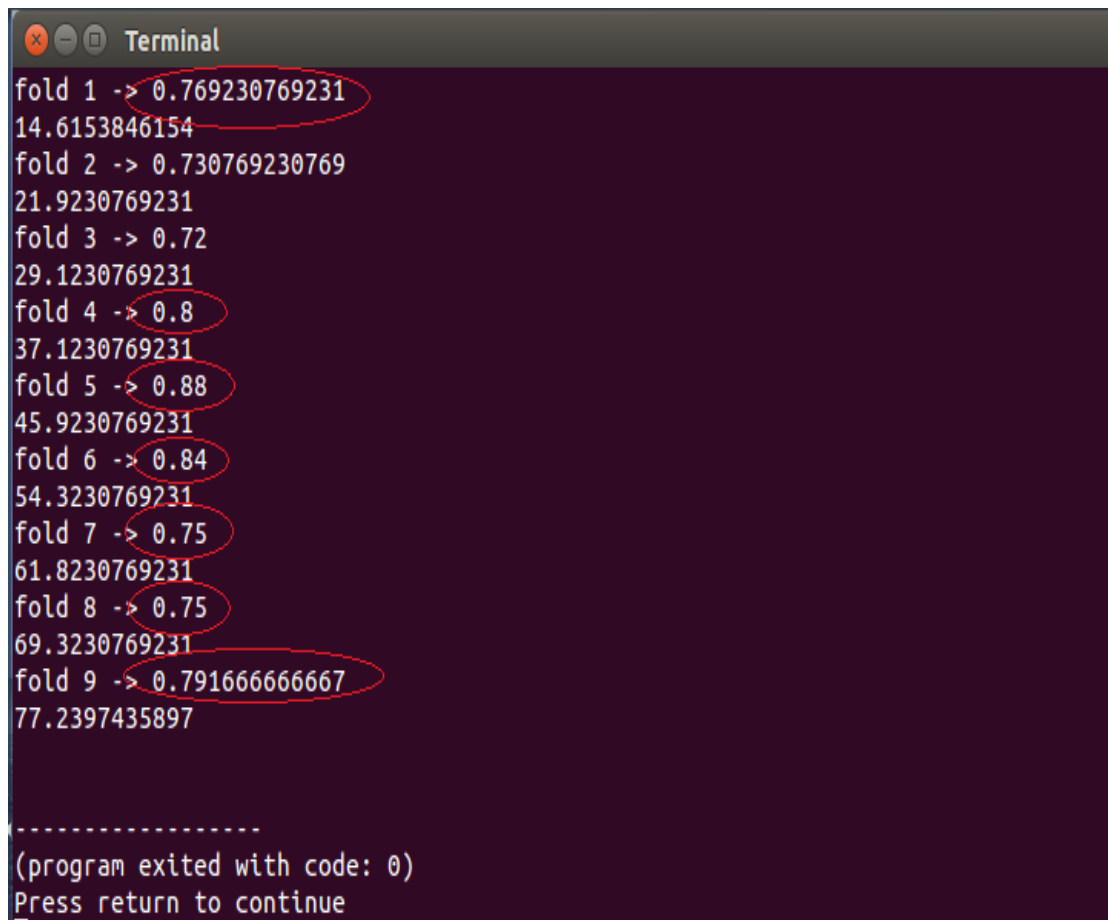
Cross-validation can be applied for performance estimation. Cross-validation allows for all the data to be used in obtaining an estimate. “Using 10-fold cross-validation one repeatedly uses 90% of the data to build a model and test its accuracy on the remaining 10%”[29]. On the other hand, Kohavi (1995) stated in his paper that, the induction algorithms are not stable they are approximately stable. ‘k’ fold cross validation with moderate k values(10-20) reduces the variance while increasing the bias. As ‘k’ decreases and the sample sizes get smaller, there is variance due to the instability of the training sets themselves leading to an increase in variance.[28]

Therefore we have selected k to be 10, for our 250 samples. After the k-fold cross validation we get the accuracy rate range of 75%-79%.

A terminal window titled "Terminal" with a dark background and light text. It displays five lines of output, each representing a fold of a k=5 cross-validation. The values are: fold 0 -> 0.764705882353, fold 1 -> 0.705882352941, fold 2 -> 0.78, fold 3 -> 0.795918367347, and fold 4 -> 0.755102040816. A red oval is drawn around the entire output text.

```
fold 0 -> 0.764705882353
fold 1 -> 0.705882352941
fold 2 -> 0.78
fold 3 -> 0.795918367347
fold 4 -> 0.755102040816
```

Figure a: Accuracy for k=5 fold



```
Terminal
fold 1 -> 0.769230769231
14.6153846154
fold 2 -> 0.730769230769
21.9230769231
fold 3 -> 0.72
29.1230769231
fold 4 -> 0.8
37.1230769231
fold 5 -> 0.88
45.9230769231
fold 6 -> 0.84
54.3230769231
fold 7 -> 0.75
61.8230769231
fold 8 -> 0.75
69.3230769231
fold 9 -> 0.791666666667
77.2397435897

-----
(program exited with code: 0)
Press return to continue
```

Figure b: Accuracy for k=10 fold

### 6.3.3.2 Decreasing Error rate with increasing sample:

We have use ASGD algorithm because with the increasing sample size it decreases its error rate.

Here we have shown in the graph, how the error rate of the prediction is decreasing with the increasing sample size. The error rate decreases from 30% to 20% for increasing sample size.

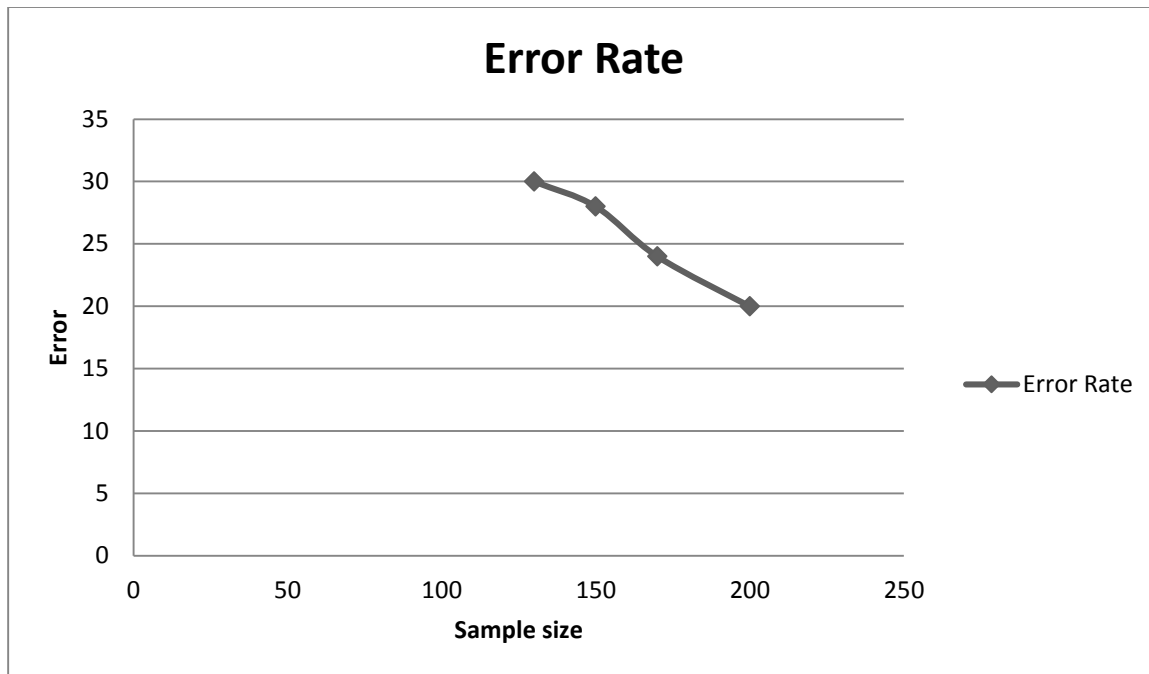


Fig: Decreasing Error Rate with Increasing Samples

### 6.3.2 Testing using interface

When we tested our system using the interface, it could do register, login, logout, add CV, Jobs, apply to jobs, and give jobs. However, our main concentration is the incremental learning part. When a recruiter, gave a job to an employee, our system automatically modified the training pickled files. As we can see below:

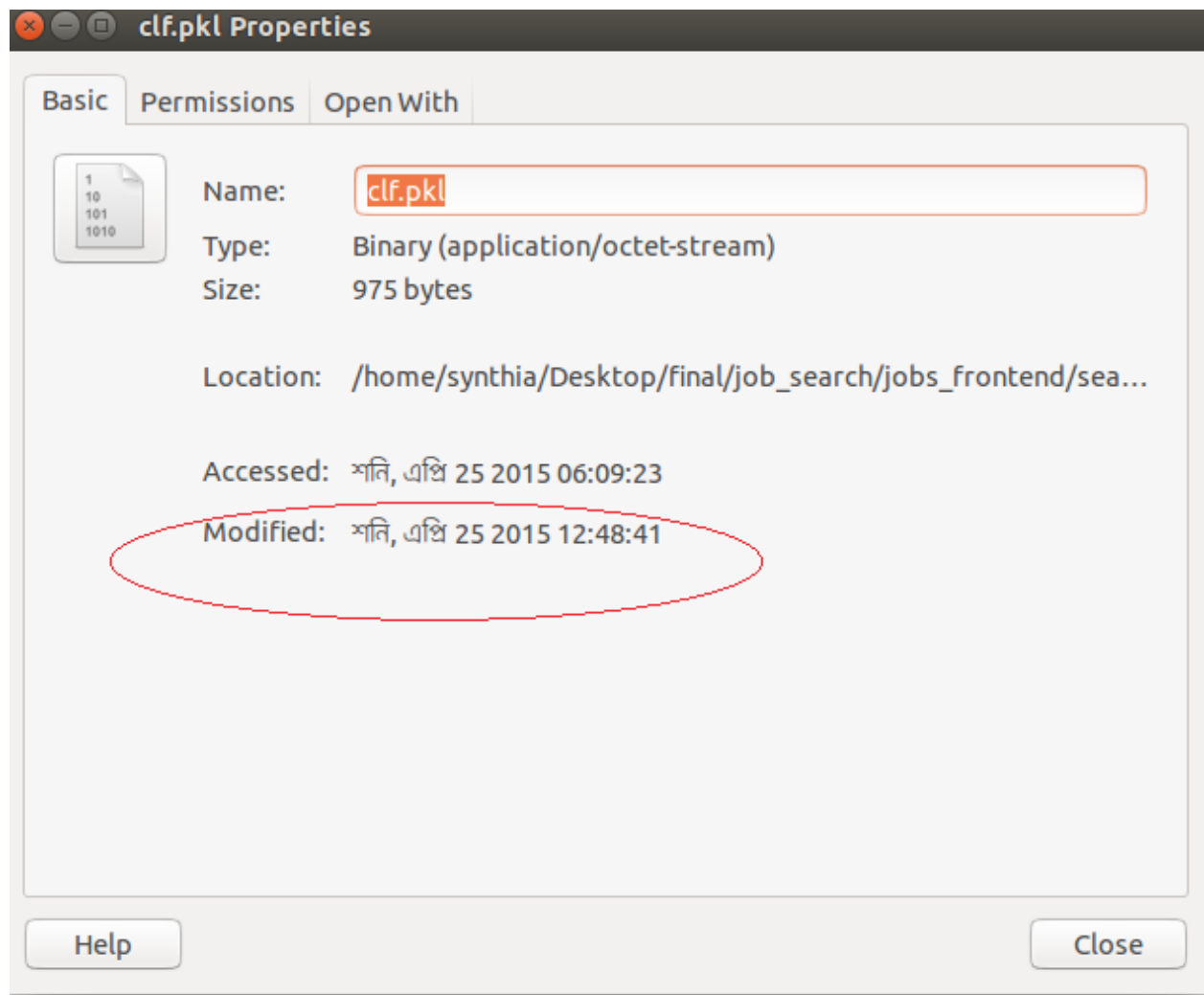


Figure : Modified the pickled files after incremental learning.

## 7. DISCUSSION

As we have discussed in our paper, we believe, incremental learning can add various dimensions to the area of digital job search system. Through the paper, we have discussed various job search theories, matching function, supervised machine learning algorithms, incremental learning and bipartite graph. We have also discussed about the cross validation method for efficiency



estimation of the learning. With the theoretical research, we have practically implemented the incremental learning in the job search. Moreover, we took many interviews and set the parameters, so that we could train our system properly. Finally, when we successfully trained the system, we ended up with decent percentage of accuracy. Using ASGD algorithm was another good decision. With our increasing data sample, the error rate was decreasing, and if we add more data the error rate will decrease more and more.

### **8. Future Work:**

We had few limitations in our research due to scarce amount of data. We have a plan to research on this with large scale data, and more detailed real life recruitment cases.

## **8. References:**

- [1] Faggian,A, Fischer.M.M&Nijkamp,P (2014), Job Search Theory Alessandra Faggian, *Handbook of Regional Science*, [p-59-73] DOI 10.1007/978-3-642-23430-9\_8;.
- [2] Joint-Search Theory: New Opportunities and New Frictions\*, *Federal Reserve Bank of Minneapolis Research Department Staff Report 426*,May 2009, Violante New York University, CPER, and NBER
- [3] Mortensen,T(1984), Job Search and Labor Market Analysis, Discussion Paper no 594.
- [4] Cornelißen,T,(2008).The Interaction of Job Satisfaction, Job Search, and Job Changes. *An Empirical Investigation with German Panel Data*, Published online: 18 March 2008 Springer Science+Business Media B.V. 2008.
- [5] Kumari,N(2012), A Study of the Recruitment and Selection process: *SMC Global, Industrial Engineering Letters*. (online) Vol 2, No.1, 2012,
- [6] Syed.A.N, Liu.H&Sung.K.K, Program For Research In Intelligent Systems (PRIS), *Incremental Learning with support vector machines*, Singapore 119260f ,nadeem,liuh,sungkkg@comp.nus.edu.sg.
- [7] Stefan.R, *Incremental Learning with Support Vector Machines*,Germany.
- [8] Cunhe.L, Kangwei.I&Hongxia.L.Wang(2009), The incremental learning algorithm with support vector machine based on hyperplane-distance, *Springer science+business media*

,Published online: 1 April 2009 ©, LLC 2009, ApplIntell (2011) 34: 19–27 DOI 10.1007/s10489-009-0176-9.

[9] Bottou,L, Large-Scale Machine Learning with Stochastic Gradient Descent, NEC Labs America, Princeton NJ 08542, USA [leon@bottou.org](mailto:leon@bottou.org).

[10] Read2,J, Bifet1,A, Pfahringer1,B, & Holmes1,G, Batch-Incremental versus Instance-Incremental Learning in Dynamic and Evolving Data, New Zealand {abifet,bernhard,geoff}@cs.waikato.ac.nz 2 Universidad Carlos III Madrid, Spain.

[11] Bottou.L, Stochastic Gradient Descent Tricks, *Microsoft Research, Redmond, WA* leon@bottou.org <http://leon.bottou.org>.

[12] Zhou.W, Chao.S, Tao.L, Shu-Ching.C&Ning.X, A Bipartite-Graph Based Approach for Disaster Susceptibility Comparisons among Cities, FL 33199, U.S.A. Email: {wzhou005,cshen001,taoli,chens,nxie,weijp}@cs.fiu.edu.

[13] Drigas,A, & Kouremnos.S(2004), An expert system for job matching of the unemployed.

[14] Jang, J.& S.R.( 1993), "ANFIS: adaptive-network-based fuzzy inference system," *Systems, Man and Cybernetics, IEEE Transactions on* , vol.23, no.3, pp.665,685, doi: 10.1109/21.256541.

[15] Intelligent Agent Based Job Search System in Android Environment.

- [16] R. G. McFadyen<sup>1</sup> & J. P. Thomas<sup>2,3</sup>(1997), Human Relations, *Economic and Psychological Models of Job Search Behavior of the Unemployed*, 16. Vol. 50, No. 12.
- [17] Wiley, J& Sons, (1995), Vapnik, V.: The Nature of Statistical Learning Theory, New York.
- [18] Wiley, J& Sons, (1998), Vapnik, V.: Statistical Learning Theory, New York.
- [19] Ms. R. R. Ade, P. R. Deshmukh (July 2013), Methods for incremental learning: a survey, *International Journal of Data Mining & Knowledge Management Process (IJDKP)* Vol.3, No.4, Pune rosh513@gmail.com, Amravati [pr\\_deshmukh@yahoo.com](mailto:pr_deshmukh@yahoo.com).
- [20] Pedregosa, *al*, (2011), Scikit-learn: Machine Learning in Python, , JMLR 12, pp. 2825-2830.
- [21] Chetan S, Nithin, M, V & Krishnan, M, K, (2011), A study of Matchings in Graphs, National Institute of Technology Calicut Kerala - 673601 April 2011.
- [22] Parkway.C(2010), Introduction to Django Chander Ganesan OSCON 2010, Suite 210 Morrisville, NC 27560 Phone: 919.463.0999 Fax: 866-229-3386 [www.opentechnologygroup.com](http://www.opentechnologygroup.com) Copyright ©2010 Open Technology Group, Inc.® All rights reserved.

- [23] Gendreau,B,T Using Python, Django and MySQL in a Database ,La Crosse La Crosse, WI 54601.
- [24] Sun,X, Kashima,H,Matsuzaki,K, &Ueda,N, Averaged Stochastic Gradient Descent with Feedback: *An Accurate, Robust, and Fast Training Method*, Kyoto, Japan {xusun, kashima}@mist.i.u-tokyo.ac.jp matuzaki@is.s.u-tokyo.ac.jp.
- [25] Zhang.T(2004), ICML 2004: Proceedings of the twenty-first international conference on machine learning. omnipress,*Solving Large Scale Linear Prediction Problems Using Stochastic Gradient Descent Algorithms* (2004)
- [26] Zheng,J ,Shen,F ,Fan,H&Zhao,J(2013), An online incremental learning support vector machine for large-scale data,NeuralComput&Applic (2013) 22:1023–1035,DOI 10.1007/s00521-011-0793-1
- [27] Easley,D&Kleinberg.J, Networks, Crowds, and Markets: *Reasoning About a Highly Connected World*, Page-251
- [28] Kohavi,R, Appears in the International Joint Conference on Artificial Intelligence IJCAI, *A Study of CrossValidation and Bootstrap for Accuracy Estimation and Model Selection*,ronnykCSStanfordEDU, <http://roboticsstanford.edu>.
- [29] Refaeilzadeh.P, Tang,L&Lui,H, (2009),Encyclopedia of Database Systems: *cross validation*, pp 532-53

### **Internet Reference**

- [30] <http://scikit-learn.org/stable/modules/svm.html#>
- [31] <http://stats.stackexchange.com/questions/70801/how-to-normalize-data-to-0-1-range>
- [32] <http://scikit-learn.org/stable/modules/sgd.html#mathematical-formulation>
- [33] [http://scikit-learn.org/stable/auto\\_examples/linear\\_model/plot\\_sgd\\_comparison.html](http://scikit-learn.org/stable/auto_examples/linear_model/plot_sgd_comparison.html)
- [34] <http://www.personal.kent.edu/~rmuhamma/GraphTheory/MyGraphTheory/defEx.htm>
- [35] <http://www.geeksforgeeks.org/bipartite-graph/>
- [36] <http://scikit-learn.org/stable/modules/preprocessing.html>
- [37] <https://docs.djangoproject.com/en/1.8/>
- [38] [https://github.com/scikitlearn/scikitlearn/blob/bb39b49/sklearn/linear\\_model/stochastic\\_gradient.py#L567](https://github.com/scikitlearn/scikitlearn/blob/bb39b49/sklearn/linear_model/stochastic_gradient.py#L567)
- [39] <http://scikit-learn.org/0.10/tutorial.html>
- [40] [http://scikit-learn.org/stable/modules/cross\\_validation.html#cross-validation](http://scikit-learn.org/stable/modules/cross_validation.html#cross-validation)