Data Science project

Profile Recommandation System

Realised by :

- Ben Salah Slim
- Benhaj Minyaoui Mustapha
- Boukhayatia Oumayma
- Chebbi Dhia
- Kitar Chifa
- Slatnia Wassim

Table of contents

Illustration table

I. General Introduction

With the increasing volume of information available online, recommender systems have become a daily tool for Internet users, providing them with desirable help in finding information.

The recommender systems used to determine the interested items for a specific user by employing a variety of information resources that related to users and items.

In the mid-1990s, the term recommender system was published for the first time in information system literature. Recommender systems are being broadly accepted in various applications to suggest products, services, and information items to latent customers.

Many e-commerce applications joint recommender systems in order to expand customer services, increase selling rates and decrease customers search time. For example Microsoft that provides users many recommendations such as the free download products, bug fixes... Netflix and YouTube that nowadays have a very important place in our lives. All these companies have successfully set up commercial recommender systems and have increased web sales and improved customer fidelity.

For many years, information system supports in human resource management have been mainly restricted in storing and tracking applicants' data through the applicant management systems. These systems support the internal workflows and communication processes between the human resource management department and the other departments.

The Internet-based online recruiting platform or e-recruitment platform is one of the most successful e-business changes, which changed the way companies employ candidates. These platforms spread in the recent years because the recruiting of the appropriate person is a challenge faced most companies, as well as the unavailability of certain candidates in some skill areas has long been identified as a major obstacle to companies success. The online channels like Internet job portal, social media on these portals , job-seeker uses them to publish their profiles.

Applications or a firm’s career website have driven this development. While the companies established job positions

For each posted job, thousands of resumes received by companies. Consequently, a huge volume of job descriptions and candidate resumes are becoming available online.

This vast volume of information gives a great opportunity for enhancing the matching quality, this potential is unused since search functionality in recruiting applications is mainly restricted to Boolean search method.

The need increases for applying the recommender system technologies that can help recruiters to handle this information efficiently. Many researches have been conducted to discuss different issues related to the recruiting problem as well as, the applying of recommender system technologies. However, job recommendation is still a

challenging domain and a growing area of research.

In this particular context, our project is defined between these few concepts, as data scientists, it was our job to develop a solution for Wevioo in order to provide them with an efficient solution that may facilitate the research for the best profiles in the best place which is a profile recommender system that will help them increase their turnover.

II. Business understanding

1. Introduction

During this chapter, we will be going through the business comprehension by detailing the problematic, rising the business questions and ultimately turning them into data science goals then we are going to present the methodology of the project , and finally the big data architecture.

2. Business objectives

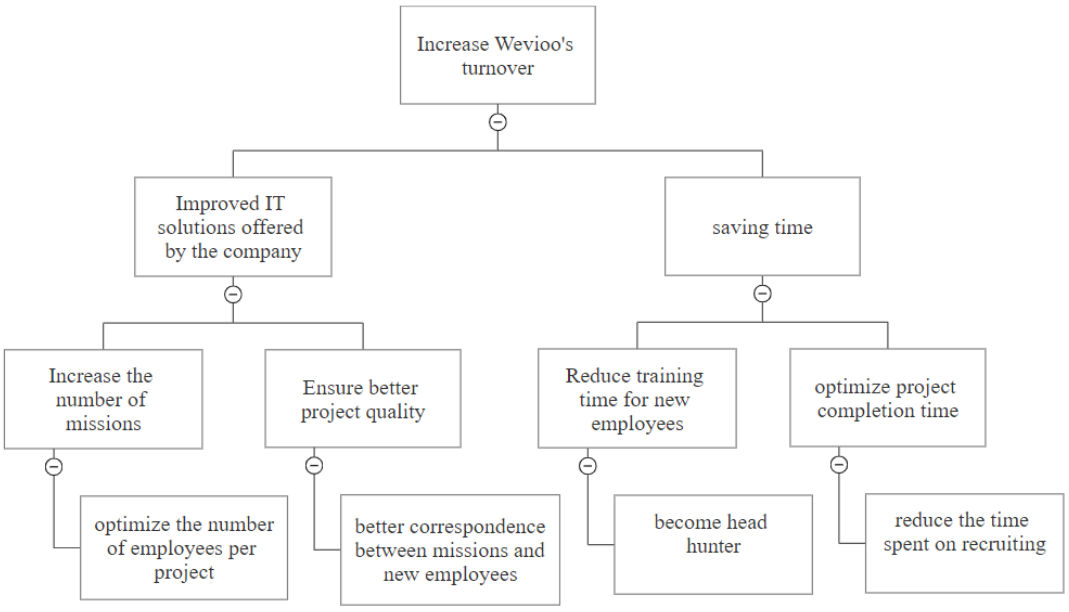

Increase Wevioo's turnover
Improved IT solutions offered by the company
saving time
Increase the number of missions
Ensure better project quality
Reduce training time for new employees
optimize project completion time
optimize the number of employees per project
better correspondence between missions and new employees
become head hunter
reduce the time spent on recruiting

Figure 1 - Business objectives

description of the specific objective selected :

create a recommendation platform dedicated to the human resources department in order to give the 10 best profiles corresponding to a search

3. Data Science Objectives

Feature engineering	Data understanding
	Data cleaning
	Create new features for the database
External Data	Web scrapping: getting other information of the profile from Linkedin or personal web site
Data Modeling	Create a recommender system
	Classification

	Scoring
Deployment	Create a web platform for this recommender system and create dashboards

Figure 2 - Data science objectives

4. IBM Master Plan methodology

During the realization of our project we are going to follow the IBM Master Plan methodology which is based on a problem-solving logic presented as follows :

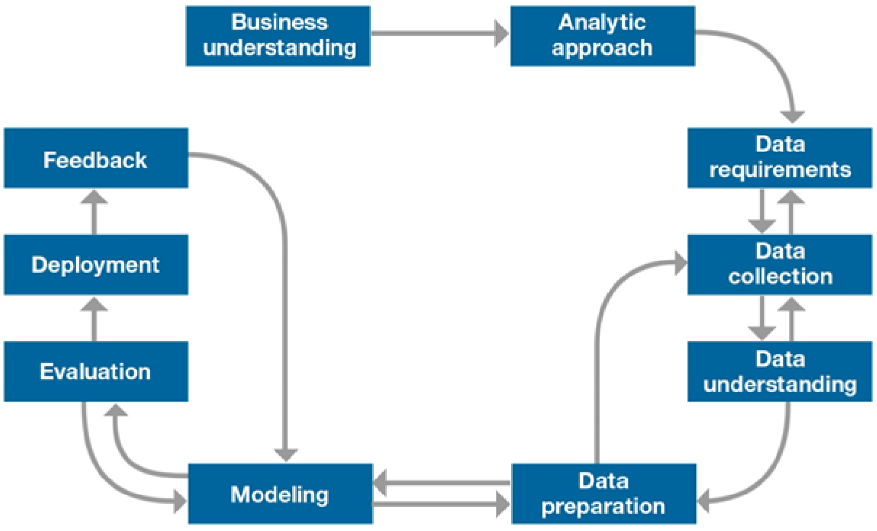


Figure 3 - IBM master plan

a) From problem to approach:

What's the problem you're trying to solve? **(Business Understanding)**

How can you use data to answer the question? (**Analytic Approach)**

b) Working with data

What data do you need to answer? **(Data Requirements)**

Where is the data coming from and how will you get it? **(Data Collection)**

Is the data that you collected representative to the problem? **(Data Understanding)**

What additional work required to work with data? **(Data preparation)**

c) Deriving the answer

In what way can the data be virtualized to get an answer? **(Modeling)**

Does the model used really answer the initial question? **(Evaluation)**

Can you put the model into practice? **(Deployment)**

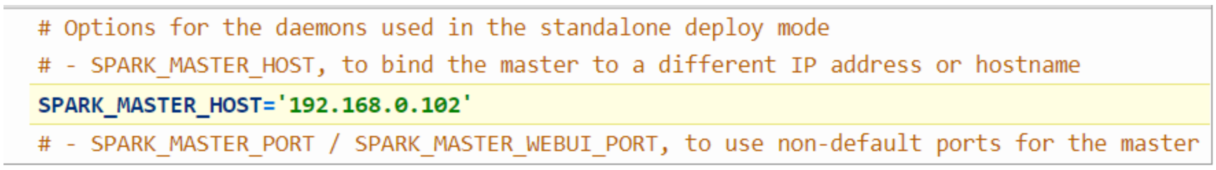
Can you get constructive feedback into answering the question? **(Feedback)**

5. Big data Architecture

During this project we are going to install a big data architecture. We are going to work with Spark , pySpark exactly . in this part we are going to put screenshots of the steps of installing this architecture (Master and slaves) and our computers .

a) Master device

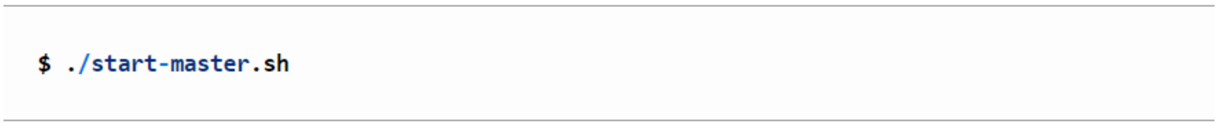
We must go to the file spark-env.sh (/opt/spark/conf/spark-env.sh) and we add the ip address of the master device like this :



```
# Options for the daemons used in the standalone deploy mode
# - SPARK_MASTER_HOST, to bind the master to a different IP address or hostname
SPARK_MASTER_HOST='192.168.0.102'
# - SPARK_MASTER_PORT / SPARK_MASTER_WEBUI_PORT, to use non-default ports for the master
```

Figure 4 - Ip adress of the master device

And then we start the device as master :

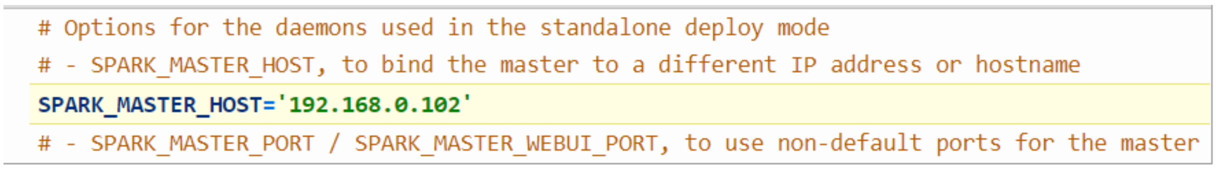


```
$ ./start-master.sh
```

Figure 5 - start master

b) Slave devices

In the same file we add also the ip address of the master device



```
# Options for the daemons used in the standalone deploy mode
# - SPARK_MASTER_HOST, to bind the master to a different IP address or hostname
SPARK_MASTER_HOST='192.168.0.102'
# - SPARK_MASTER_PORT / SPARK_MASTER_WEBUI_PORT, to use non-default ports for the master
```

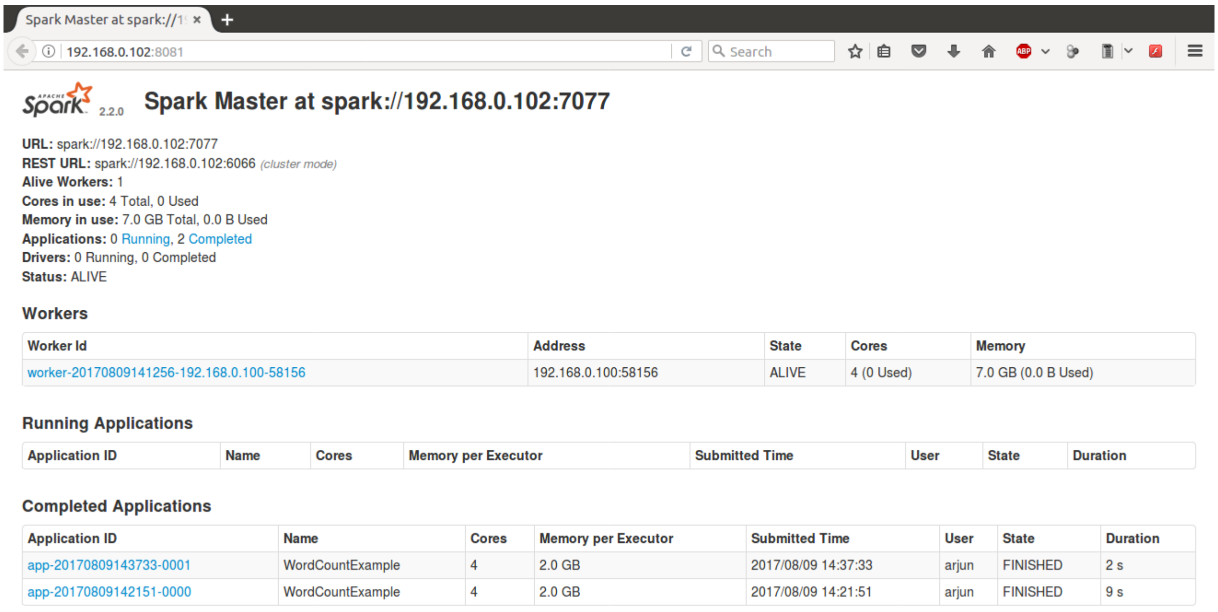
Figure 6 - ip adress on the slave device

And the we start the file as a slave:

```
$ ./start-slave.sh spark://<your.master.ip.address>:7077
```

Figure 7 - starting the slave

And we can check it in the spark Web UI :



Spark Master at spark://192.168.0.102:7077

URL: spark://192.168.0.102:7077
REST URL: spark://192.168.0.102:6066 *(cluster mode)*
Alive Workers: 1
Cores in use: 4 Total, 0 Used
Memory in use: 7.0 GB Total, 0.0 B Used
Applications: 0 Running, 2 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers

Worker Id	Address	State	Cores	Memory
worker-20170809141256-192.168.0.100-58156	192.168.0.100:58156	ALIVE	4 (0 Used)	7.0 GB (0.0 B Used)

Running Applications

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Completed Applications

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
app-20170809143733-0001	WordCountExample	4	2.0 GB	2017/08/09 14:37:33	arjun	FINISHED	2 s
app-20170809142151-0000	WordCountExample	4	2.0 GB	2017/08/09 14:21:51	arjun	FINISHED	9 s

Figure 8 - verifying the architecture in spark UI

6. Conclusion

During this chapter we realized the first phase of the project which consists of business understanding and installing BigData architecture.

III. Data Manipulating

1. Introduction

During this chapter we are going to create new features and collect data from an external source which is LinkedIn.

2. Natural language processing

Since we have textual data, we are in need of specific processing to be able to extract meanings and learn from this data in order to make it more useful. Thus, we proceeded by using the Natural Language Processing which is known as NLP. The techniques of the preparation of NLP data covers several stages, the result of which is cleaned and formatted text data.

a) Tokenization

Tokenization consists of dividing a string into atomic elements called tokens. This process is provided by the NLTK library which offers tokenizers already trained on a set of documents or corpus. In this purpose we did apply the tokenization on the the jobs description of each candidate (Figure 2).

b) Stop words

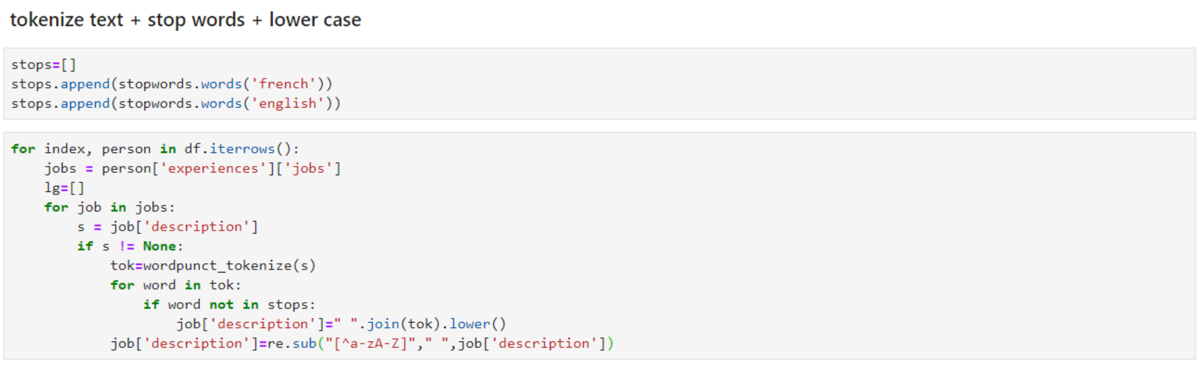
Many words have little semantic interest, this is the reason for which the stop words must be deleted. Indeed, stopwords are all the most common words in a language. NLTK library has a list of stopwords for many languages. By downloading the list of the French language and the English one we became able to eliminate the stopwords (Figure 2).

c) Lower case

One of the initial steps in preprocessing text data is to eliminate the capital letter. This step is used to normalize the data. So we have proceed by eliminating the capital of the 2 variables "skills" and "description" to be able to compare both of them in order to extract the experienced skills of each candidate (Figure 2).

d) Stemming words

The rooting is the method which allows to represent in the same word several derived words from it keeping only the radical. We will need this treatment in our case because the consultants write in several ways the same expression semantically speaking. Take the example of these 3 expressions: "Java development", "java developer" and "develop java". Thanks to rooting provided by the NLTK library, they all become the same expression which is "develop java". Now the skills of each candidate is ready to be compared correctly with the description of the job (Figure 3).

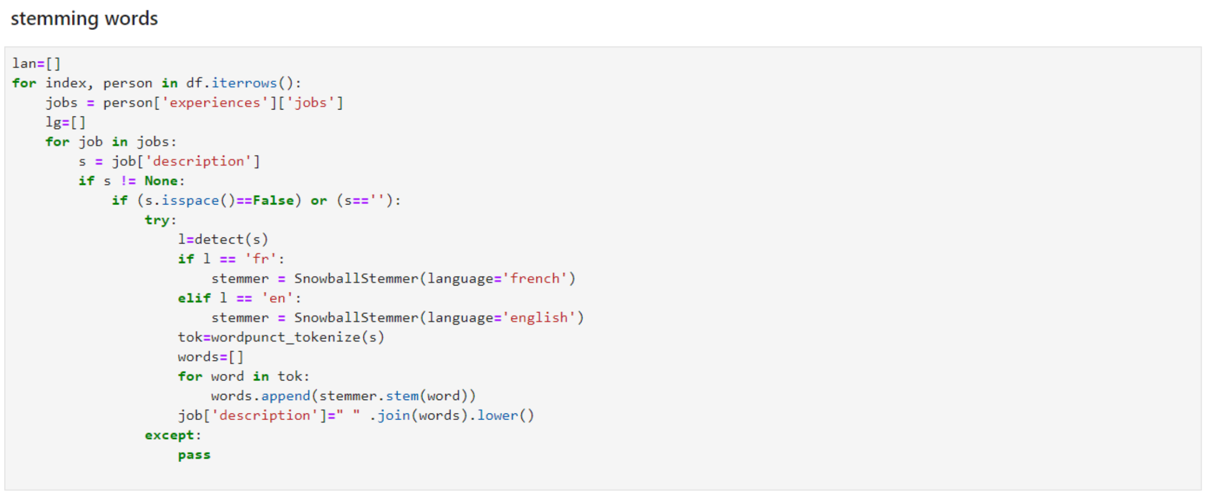


tokenize text + stop words + lower case

```
stops=[]
stops.append(stopwords.words('french'))
stops.append(stopwords.words('english'))
```

```
for index, person in df.iterrows():
    jobs = person['experiences']['jobs']
    lg=[]
    for job in jobs:
        s = job['description']
        if s != None:
            tok=wordpunct_tokenize(s)
            for word in tok:
                if word not in stops:
                    job['description']=" ".join(tok).lower()
            job['description']=re.sub("[^a-zA-Z]"," ",job['description'])
```

Figure 9 - tokenize + stop words + lower case



stemming words

```
lan=[]
for index, person in df.iterrows():
    jobs = person['experiences']['jobs']
    lg=[]
    for job in jobs:
        s = job['description']
        if s != None:
            if (s.isspace()==False) or (s==''):
                try:
                    l=detect(s)
                    if l == 'fr':
                        stemmer = SnowballStemmer(language='french')
                    elif l == 'en':
                        stemmer = SnowballStemmer(language='english')
                    tok=wordpunct_tokenize(s)
                    words=[]
                    for word in tok:
                        words.append(stemmer.stem(word))
                    job['description']=" " .join(words).lower()
                except:
                    pass
```

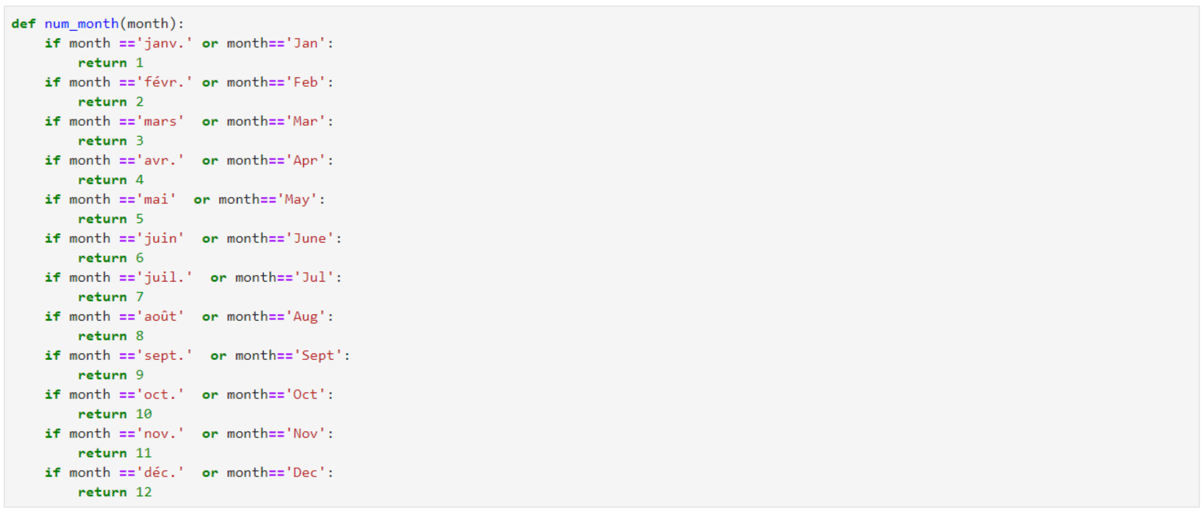
Figure 3 – Stemming words

3. New features

a) Months of experience

This feature allows as to know the number of months of experience of each candidate. So first we implemented a function that gives as the number of month given by letter

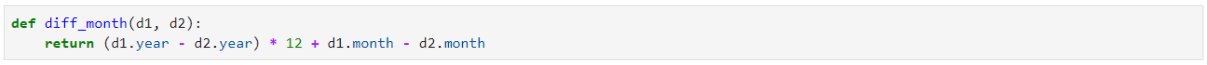
Number of months of experience



```python
def num_month(month):
    if month =='janv.' or month=='Jan':
        return 1
    if month =='févr.' or month=='Feb':
        return 2
    if month =='mars'  or month=='Mar':
        return 3
    if month =='avr.'  or month=='Apr':
        return 4
    if month =='mai'  or month=='May':
        return 5
    if month =='juin'  or month=='June':
        return 6
    if month =='juil.'  or month=='Jul':
        return 7
    if month =='août'  or month=='Aug':
        return 8
    if month =='sept.'  or month=='Sept':
        return 9
    if month =='oct.'  or month=='Oct':
        return 10
    if month =='nov.'  or month=='Nov':
        return 11
    if month =='déc.'  or month=='Dec':
        return 12
```

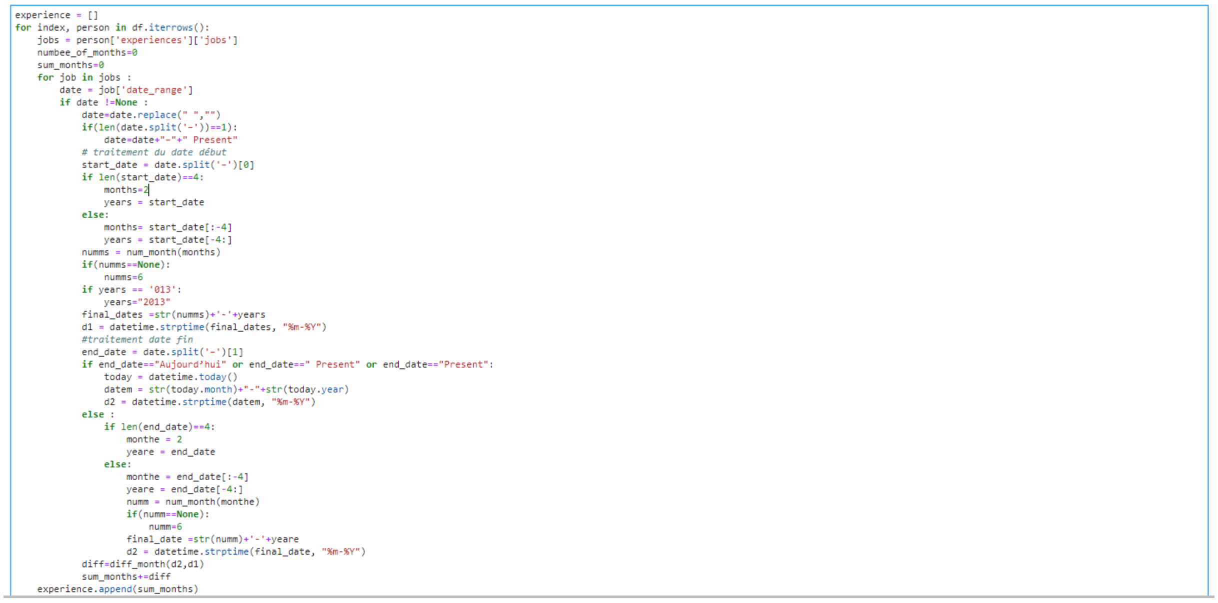
Figure 10 - num month

Then we implemented a function that calculate the difference in months between two dates



```python
def diff_month(d1, d2):
    return (d1.year - d2.year) * 12 + d1.month - d2.month
```

Figure 11 - diff month

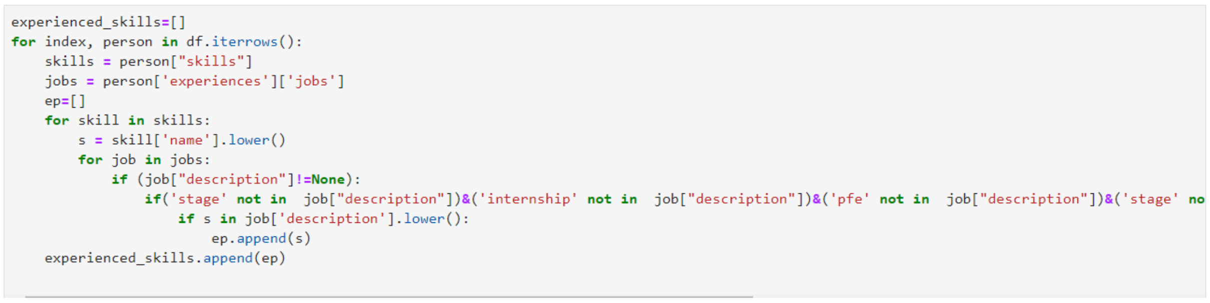


```python
experience = []
for index, person in df.iterrows():
    jobs = person['experiences']['jobs']
    numbee_of_months=0
    sum_months=0
    for job in jobs :
        date = job['date_range']
        if date !=None :
            date=date.replace(" ","")
            if(len(date.split('-'))==1):
                date=date+"-"+" Present"
            # traitement du date début
            start_date = date.split('-')[0]
            if len(start_date)==4:
                months=2
                years = start_date
            else:
                months= start_date[:-4]
                years = start_date[-4:]
            numms = num_month(months)
            if(numms==None):
                numms=6
            if years == '013':
                years="2013"
            final_dates =str(numms)+'-'+years
            d1 = datetime.strptime(final_dates, "%m-%Y")
            #traitement date fin
            end_date = date.split('-')[1]
            if end_date=="Aujourd’hui" or end_date==" Present" or end_date=="Present":
                today = datetime.today()
                datem = str(today.month)+"-"+str(today.year)
                d2 = datetime.strptime(datem, "%m-%Y")
            else :
                if len(end_date)==4:
                    monthe = 2
                    yeare = end_date
                else:
                    monthe = end_date[:-4]
                    yeare = end_date[-4:]
                    numm = num_month(monthe)
                    if(numm==None):
                        numm=6
                    final_date =str(numm)+'-'+yeare
                    d2 = datetime.strptime(final_date, "%m-%Y")
            diff=diff_month(d2,d1)
            sum_months+=diff
    experience.append(sum_months)
```

Figure 12 - months of experience

b) Skills used in professional context

To make sure our candidate masters the skills mentioned in the skills section we have compared them with those indicated in the experience section. Then we have extracted the common skills between both sections. each skill in this new feature will have a superior weight when calculating the score of the candidate.

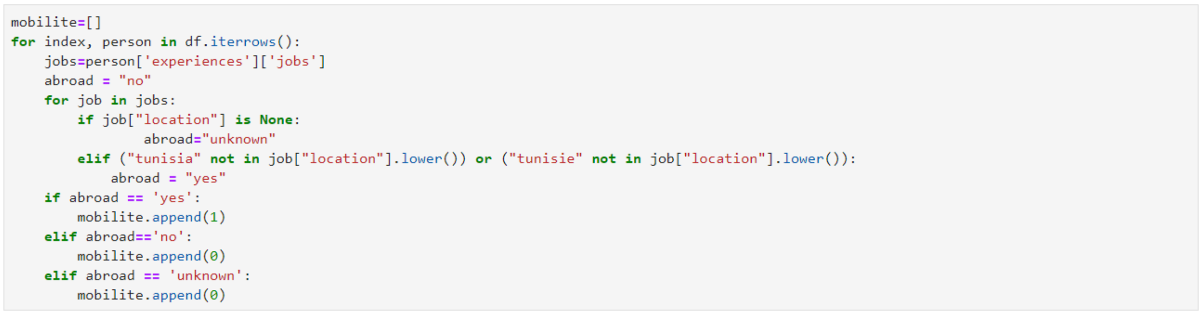


```
experienced_skills=[]
for index, person in df.iterrows():
    skills = person["skills"]
    jobs = person['experiences']['jobs']
    ep=[]
    for skill in skills:
        s = skill['name'].lower()
        for job in jobs:
            if (job["description"]!=None):
                if('stage' not in  job["description"])&('internship' not in  job["description"])&('pfe' not in  job["description"])&('stage' no
                    if s in job['description'].lower():
                        ep.append(s)
    experienced_skills.append(ep)
```

Figure 13 - extraction of experienced skills

c) Mobility

This feature will allow us to check if the candidate had an experience abroad. The human resource department will favor the ones who have done experience with mobility in case they are recruiting candidates for a mission abroad.

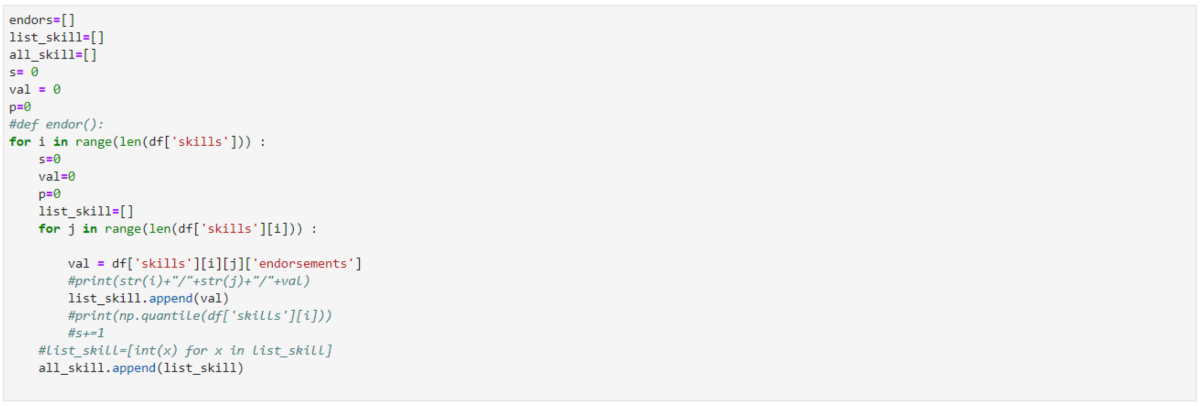


```
mobilite=[]
for index, person in df.iterrows():
    jobs=person['experiences']['jobs']
    abroad = "no"
    for job in jobs:
        if job["location"] is None:
                abroad="unknown"
        elif ("tunisia" not in job["location"].lower()) or ("tunisie" not in job["location"].lower()):
            abroad = "yes"
    if abroad == 'yes':
        mobilite.append(1)
    elif abroad=='no':
        mobilite.append(0)
    elif abroad == 'unknown':
        mobilite.append(0)
```

Figure 14 - getting mobility information from experience

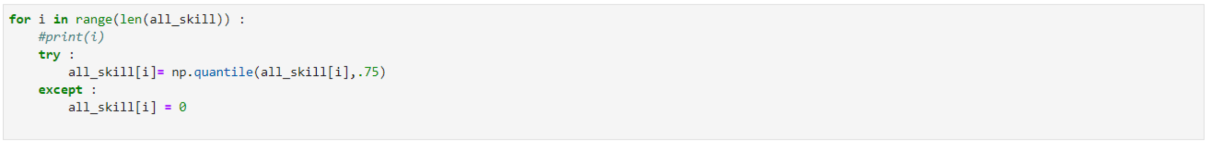
d) Skills with endorsement

This feature will allow us to extract the most endorsed skills of each candidate. So we implemented a function that takes out the most endorsed skills. We will need this feature to recommend the best skilled candidates .



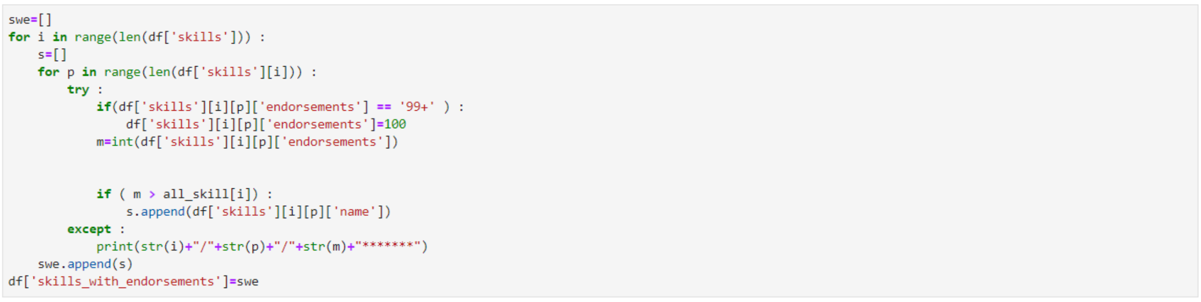
```python
endors=[]
list_skill=[]
all_skill=[]
s= 0
val = 0
p=0
#def endor():
for i in range(len(df['skills'])) :
    s=0
    val=0
    p=0
    list_skill=[]
    for j in range(len(df['skills'][i])) :

        val = df['skills'][i][j]['endorsements']
        #print(str(i)+"/"+str(j)+"/"+val)
        list_skill.append(val)
        #print(np.quantile(df['skills'][i]))
        #s+=1
    #list_skill=[int(x) for x in list_skill]
    all_skill.append(list_skill)
```

Figure 15 - extract the endorsements of each candidate's skills



```python
for i in range(len(all_skill)) :
    #print(i)
    try :
        all_skill[i]= np.quantile(all_skill[i],.75)
    except :
        all_skill[i] = 0
```

Figure 16 - calculate the third quartile

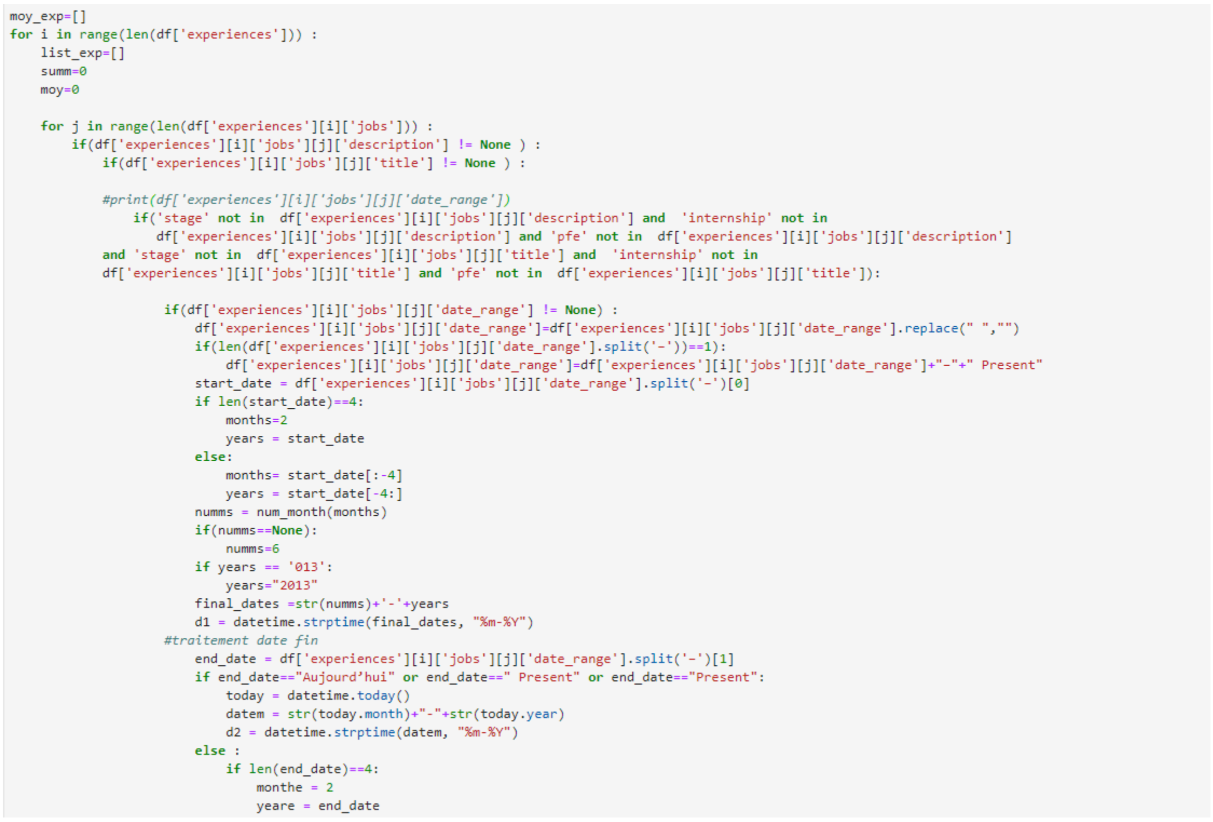


```python
swe=[]
for i in range(len(df['skills'])) :
    s=[]
    for p in range(len(df['skills'][i])) :
        try :
            if(df['skills'][i][p]['endorsements'] == '99+' ) :
                df['skills'][i][p]['endorsements']=100
            m=int(df['skills'][i][p]['endorsements'])


            if ( m > all_skill[i]) :
                s.append(df['skills'][i][p]['name'])
        except :
            print(str(i)+"/"+str(p)+"/"+str(m)+"*******")
    swe.append(s)
df['skills_with_endorsements']=swe
```

Figure 17 - get most endorsed skills

e) Average duration

We chose to develop a function that returns the average duration of experience in each job for each candidate. This feature will help us studying the behavior of the candidates: either the company can rely on him on long term projects or susceptible to left the job before the end of the contract.

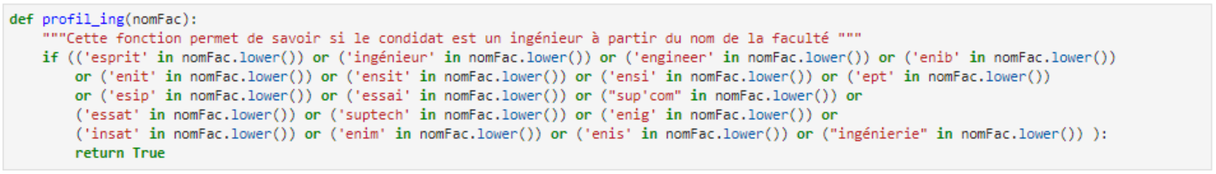


```
moy_exp=[]
for i in range(len(df['experiences'])) :
    list_exp=[]
    summ=0
    moy=0

    for j in range(len(df['experiences'][i]['jobs'])) :
        if(df['experiences'][i]['jobs'][j]['description'] != None ) :
            if(df['experiences'][i]['jobs'][j]['title'] != None ) :

            #print(df['experiences'][i]['jobs'][j]['date_range'])
                if('stage' not in  df['experiences'][i]['jobs'][j]['description'] and  'internship' not in
                   df['experiences'][i]['jobs'][j]['description'] and 'pfe' not in  df['experiences'][i]['jobs'][j]['description']
            and 'stage' not in  df['experiences'][i]['jobs'][j]['title'] and  'internship' not in
            df['experiences'][i]['jobs'][j]['title'] and 'pfe' not in  df['experiences'][i]['jobs'][j]['title']):

                    if(df['experiences'][i]['jobs'][j]['date_range'] != None) :
                        df['experiences'][i]['jobs'][j]['date_range']=df['experiences'][i]['jobs'][j]['date_range'].replace(" ","")
                        if(len(df['experiences'][i]['jobs'][j]['date_range'].split('-'))==1):
                            df['experiences'][i]['jobs'][j]['date_range']=df['experiences'][i]['jobs'][j]['date_range']+"-"+" Present"
                        start_date = df['experiences'][i]['jobs'][j]['date_range'].split('-')[0]
                        if len(start_date)==4:
                            months=2
                            years = start_date
                        else:
                            months= start_date[:-4]
                            years = start_date[-4:]
                        numms = num_month(months)
                        if(numms==None):
                            numms=6
                        if years == '013':
                            years="2013"
                        final_dates =str(numms)+'-'+years
                        d1 = datetime.strptime(final_dates, "%m-%Y")
                    #traitement date fin
                        end_date = df['experiences'][i]['jobs'][j]['date_range'].split('-')[1]
                        if end_date=="Aujourd'hui" or end_date==" Present" or end_date=="Present":
                            today = datetime.today()
                            datem = str(today.month)+"-"+str(today.year)
                            d2 = datetime.strptime(datem, "%m-%Y")
                        else :
                            if len(end_date)==4:
                                monthe = 2
                                yeare = end_date
```

Figure 18 - average duration function

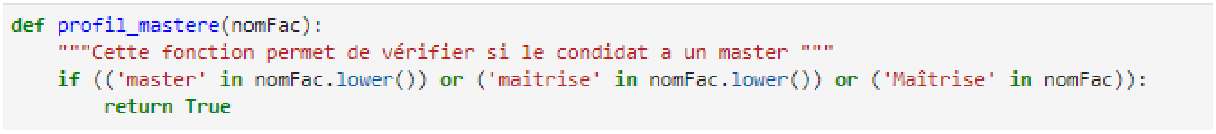
f) High degree

In order to define the level of education of each candidate we have created a new feature that indicates the highest degree obtained using the function below. This feature will surely help the human resources department finding the right person for the right job, and attributing the adequate salary.



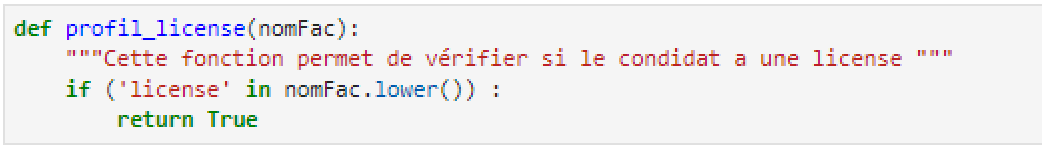
```
def profil_ing(nomFac):
    """Cette fonction permet de savoir si le condidat est un ingénieur à partir du nom de la faculté """
    if (('esprit' in nomFac.lower()) or ('ingénieur' in nomFac.lower()) or ('engineer' in nomFac.lower()) or ('enib' in nomFac.lower())
        or ('enit' in nomFac.lower()) or ('ensit' in nomFac.lower()) or ('ensi' in nomFac.lower()) or ('ept' in nomFac.lower())
        or ('esip' in nomFac.lower()) or ('essai' in nomFac.lower()) or ("sup'com" in nomFac.lower()) or
        ('essat' in nomFac.lower()) or ('suptech' in nomFac.lower()) or ('enig' in nomFac.lower()) or
        ('insat' in nomFac.lower()) or ('enim' in nomFac.lower()) or ('enis' in nomFac.lower()) or ("ingénierie" in nomFac.lower()) ):
        return True
```

Figure 19 - engineering degree



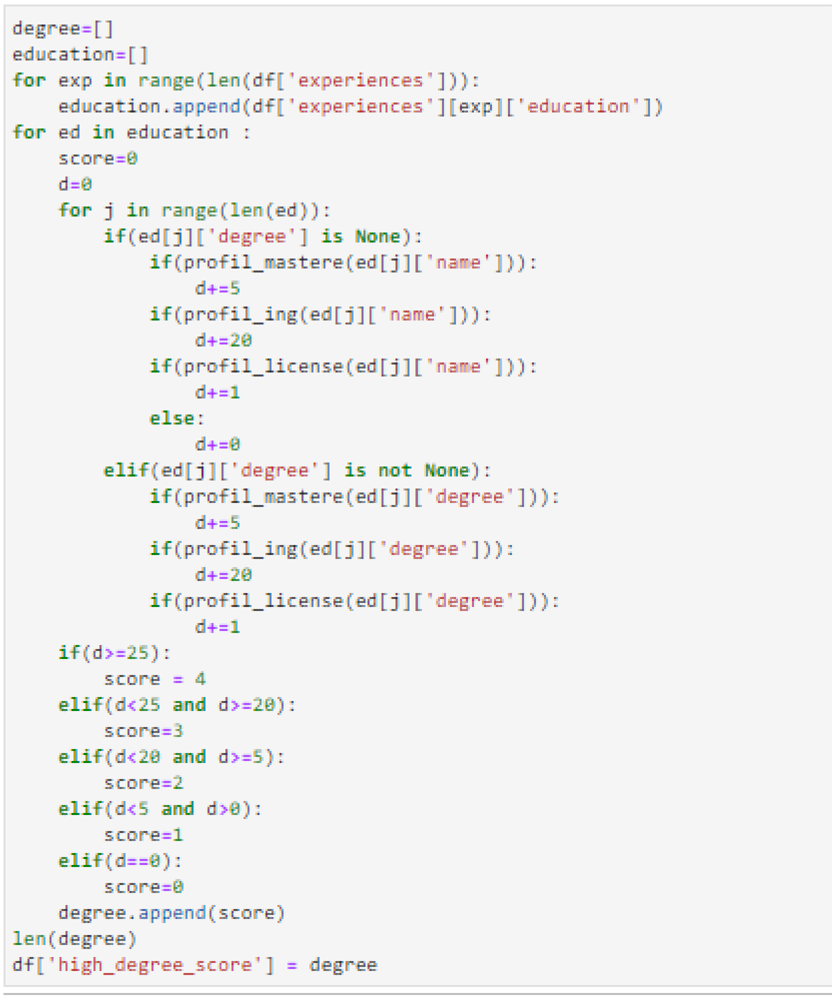
```
def profil_mastere(nomFac):
    """Cette fonction permet de vérifier si le condidat a un master """
    if (('master' in nomFac.lower()) or ('maitrise' in nomFac.lower()) or ('Maîtrise' in nomFac)):
        return True
```

Figure 20 - master's degree



```
def profil_license(nomFac):
    """Cette fonction permet de vérifier si le condidat a une license """
    if ('license' in nomFac.lower()) :
        return True
```

Figure 21 - bachelor's degree



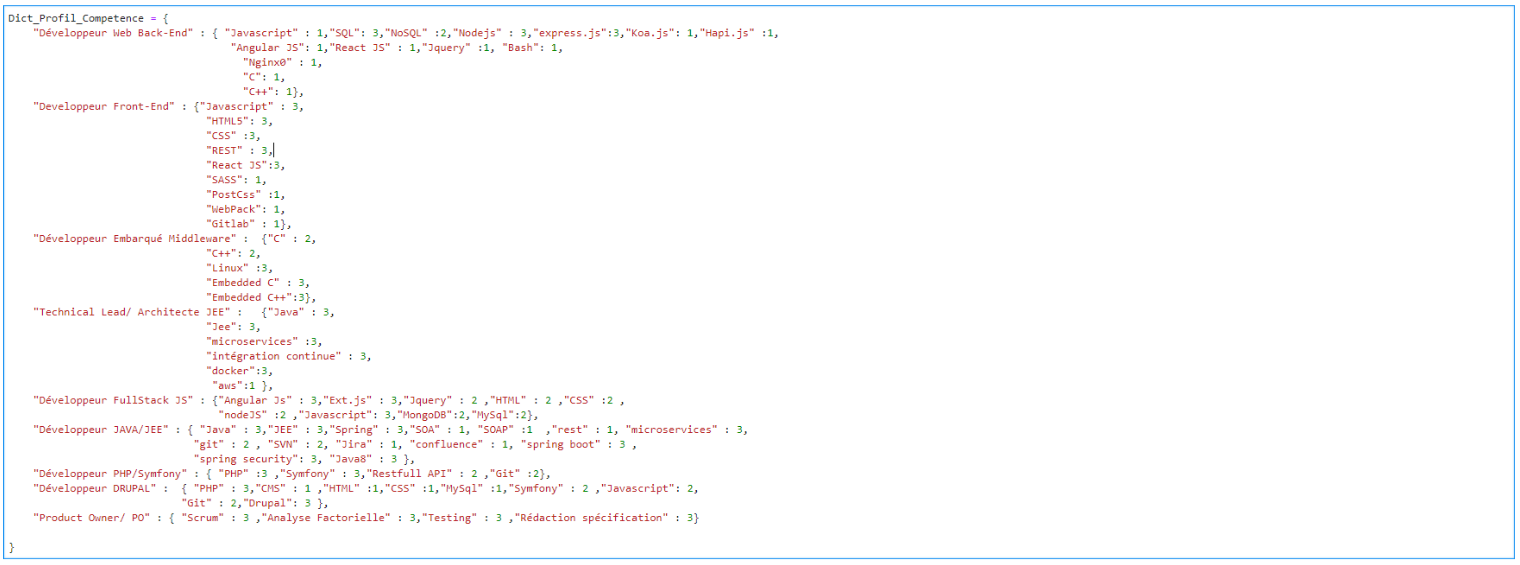
```
degree=[]
education=[]
for exp in range(len(df['experiences'])):
    education.append(df['experiences'][exp]['education'])
for ed in education :
    score=0
    d=0
    for j in range(len(ed)):
        if(ed[j]['degree'] is None):
            if(profil_mastere(ed[j]['name'])):
                d+=5
            if(profil_ing(ed[j]['name'])):
                d+=20
            if(profil_license(ed[j]['name'])):
                d+=1
            else:
                d+=0
        elif(ed[j]['degree'] is not None):
            if(profil_mastere(ed[j]['degree'])):
                d+=5
            if(profil_ing(ed[j]['degree'])):
                d+=20
            if(profil_license(ed[j]['degree'])):
                d+=1
    if(d>=25):
        score = 4
    elif(d<25 and d>=20):
        score=3
    elif(d<20 and d>=5):
        score=2
    elif(d<5 and d>0):
        score=1
    elif(d==0):
        score=0
    degree.append(score)
len(degree)
df['high_degree_score'] = degree
```

Figure 22 - final high degree

g) Score per profile

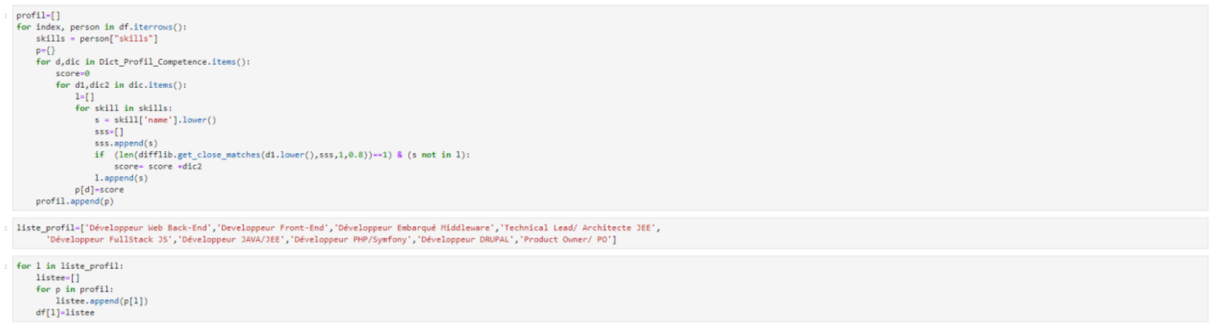
According to the profiles and the weight of each skill given by “Wevioo” , we managed to create a new feature in which we have attributed a score for each candidate per profile. This feature will help us recommend the best searched profile according to the attributed mission.

So first we created a dictionary with the skills of each profile and weight of each skill :



```
Dict_Profil_Competence = {
    "Développeur Web Back-End" : { "Javascript" : 1,"SQL": 3,"NoSQL" :2,"Nodejs" : 3,"express.js":3,"Koa.js": 1,"Hapi.js" :1,
                                "Angular JS": 1,"React JS" : 1,"Jquery" :1, "Bash": 1,
                                  "Nginx0" : 1,
                                  "C": 1,
                                  "C++": 1},
    "Developpeur Front-End" : {"Javascript" : 3,
                            "HTML5": 3,
                            "CSS" :3,
                            "REST" : 3,
                            "React JS":3,
                            "SASS": 1,
                            "PostCss" :1,
                            "WebPack": 1,
                            "Gitlab" : 1},
    "Développeur Embarqué Middleware" :  {"C" : 2,
                            "C++": 2,
                            "Linux" :3,
                            "Embedded C" : 3,
                            "Embedded C++":3},
    "Technical Lead/ Architecte JEE" :   {"Java" : 3,
                            "Jee": 3,
                            "microservices" :3,
                            "intégration continue" : 3,
                            "docker":3,
                             "aws":1 },
    "Développeur FullStack JS" : {"Angular Js" : 3,"Ext.js" : 3,"Jquery" : 2 ,"HTML" : 2 ,"CSS" :2 ,
                              "nodeJS" :2 ,"Javascript": 3,"MongoDB":2,"MySql":2},
    "Développeur JAVA/JEE" : { "Java" : 3,"JEE" : 3,"Spring" : 3,"SOA" : 1, "SOAP" :1  ,"rest" : 1, "microservices" : 3,
                          "git" : 2 , "SVN" : 2, "Jira" : 1, "confluence" : 1, "spring boot" : 3 ,
                          "spring security": 3, "Java8" : 3 },
    "Développeur PHP/Symfony" : { "PHP" :3 ,"Symfony" : 3,"Restfull API" : 2 ,"Git" :2},
    "Développeur DRUPAL" :  { "PHP" : 3,"CMS" : 1 ,"HTML" :1,"CSS" :1,"MySql" :1,"Symfony" : 2 ,"Javascript": 2,
                         "Git" : 2,"Drupal": 3 },
    "Product Owner/ PO" : { "Scrum" : 3 ,"Analyse Factorielle" : 3,"Testing" : 3 ,"Rédaction spécification" : 3}

}
```

Figure 23 - Skills and weights dictionnary



```
profil=[]
for index, person in df.iterrows():
    skills = person["skills"]
    p={}
    for d,dic in Dict_Profil_Competence.items():
        score=0
        for d1,dic2 in dic.items():
            l=[]
            for skill in skills:
                s = skill['name'].lower()
                sss=[]
                sss.append(s)
                if  (len(difflib.get_close_matches(d1.lower(),sss,1,0.8))==1) & (s not in l):
                    score= score +dic2
                l.append(s)
            p[d]=score
    profil.append(p)
```

```
liste_profil=['Développeur Web Back-End','Developpeur Front-End','Développeur Embarqué Middleware','Technical Lead/ Architecte JEE',
      'Développeur FullStack JS','Développeur JAVA/JEE','Développeur PHP/Symfony','Développeur DRUPAL','Product Owner/ PO']
```

```
for l in liste_profil:
    listee=[]
    for p in profil:
        listee.append(p[l])
    df[l]=listee
```

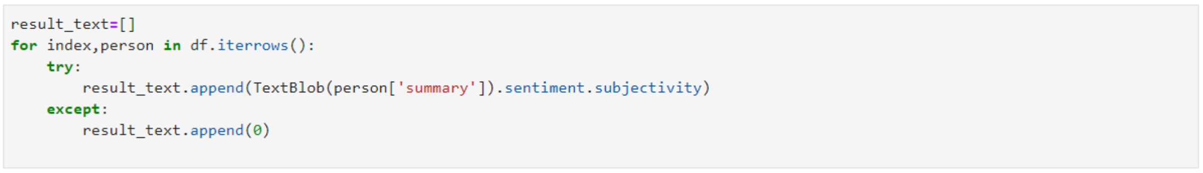
Figure 24 - adding the score of each profile to the DataFrame

h) Language

From accomplishments section we extracted the feature language which contains the number of languages that the candidate masters. (0: three languages, 1: four languages, 2: five languages and 3: more than five languages). This feature will help “Wevioo” choosing the appropriate person in case they have projects that require languages other than English and French.

i) Sentimental study

This feature will allow us to make sentimental study about the summary section of each candidate. We extracted the subjectivity and polarity of the text describing the candidate.



```python
result_text=[]
for index,person in df.iterrows():
    try:
        result_text.append(TextBlob(person['summary']).sentiment.subjectivity)
    except:
        result_text.append(0)
```

Figure 25 - Sentimental analysis

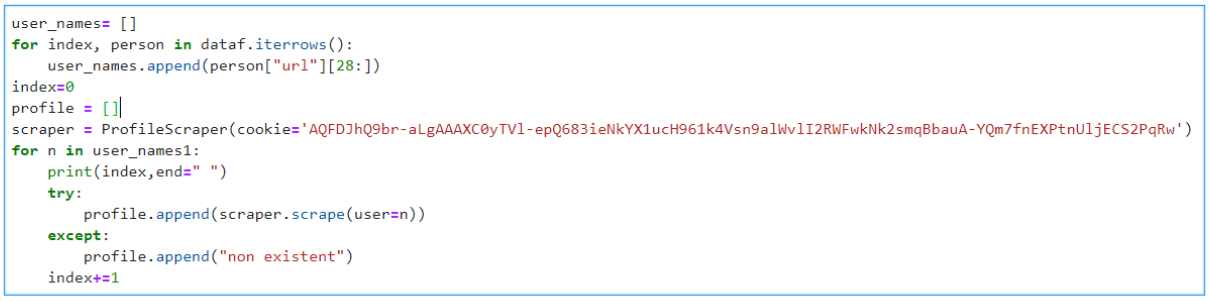
4. External data (web scraping)

a) Definition

Web scraping (sometimes called harvesting) is a technique for extracting content from websites, via a script or a program, with the aim of transforming it to allow its use in another context

b) Database update

In order to make our data up to date we used the library “LinkedIn-Scraper” that uses “Selenium” to crawl data from LinkedIn. In fact, we swept all the profiles given in the data base and extracted the needed information from each candidate’s profile using “url” column from the old database.



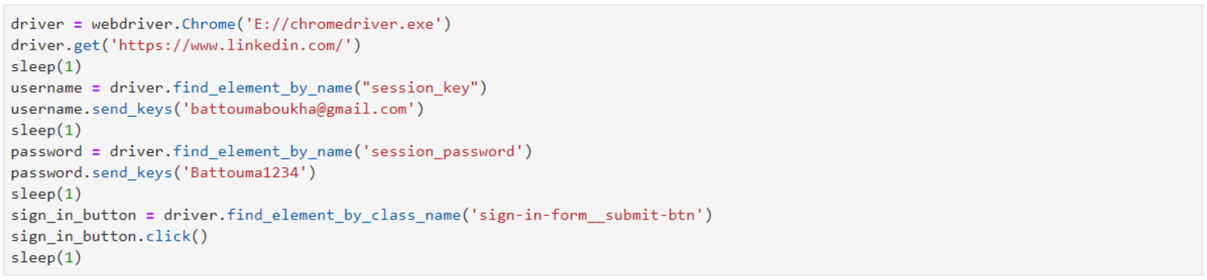
```python
user_names= []
for index, person in dataf.iterrows():
    user_names.append(person["url"][28:])
index=0
profile = []
scraper = ProfileScraper(cookie='AQFDJhQ9br-aLgAAAXC0yTVl-epQ683ieNkYX1ucH961k4Vsn9alWvlI2RWFwkNk2smqBbauA-YQm7fnEXPtnUljECS2PqRw')
for n in user_names1:
    print(index,end=" ")
    try:
        profile.append(scraper.scrape(user=n))
    except:
        profile.append("non existent")
    index+=1
```

Figure 26 - scraping script

The data scrapped has the same layout as the data given by “ Wevioo “ so we compared it with the old one keeping the unchanged data and updating the others.

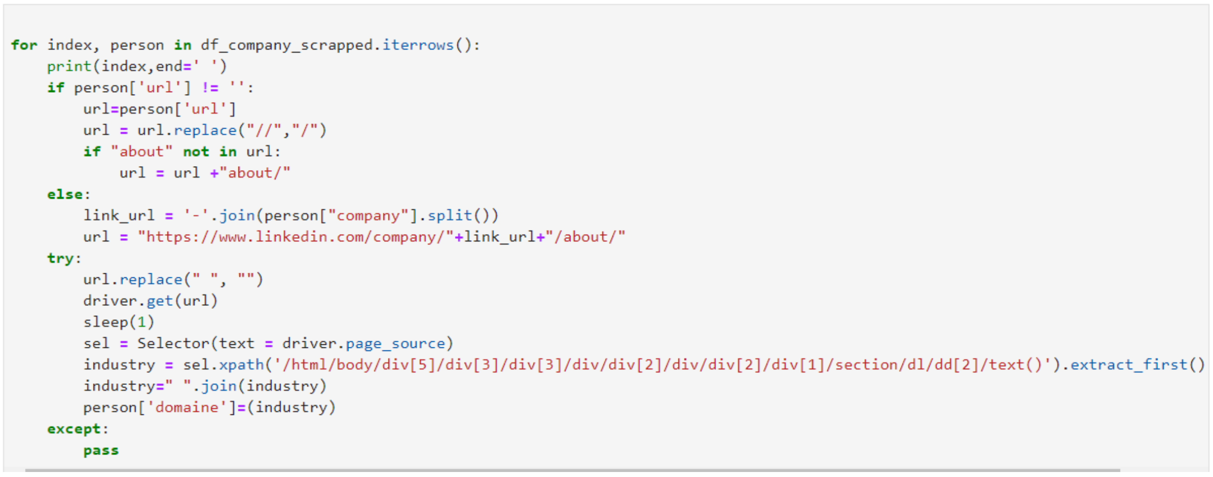
c) Business line

Determining the business line in which worked each one of the candidates seems to be important in the study of the candidate’s behavior since it can give an information that may improve the results of the predictions later, that’s why we managed to scrap this new feature from LinkedIn using “Selenium”.



```
driver = webdriver.Chrome('E://chromedriver.exe')
driver.get('https://www.linkedin.com/')
sleep(1)
username = driver.find_element_by_name("session_key")
username.send_keys('battoumaboukha@gmail.com')
sleep(1)
password = driver.find_element_by_name('session_password')
password.send_keys('Battouma1234')
sleep(1)
sign_in_button = driver.find_element_by_class_name('sign-in-form__submit-btn')
sign_in_button.click()
sleep(1)
```

Figure 27 - opening the browser and go to linkedin

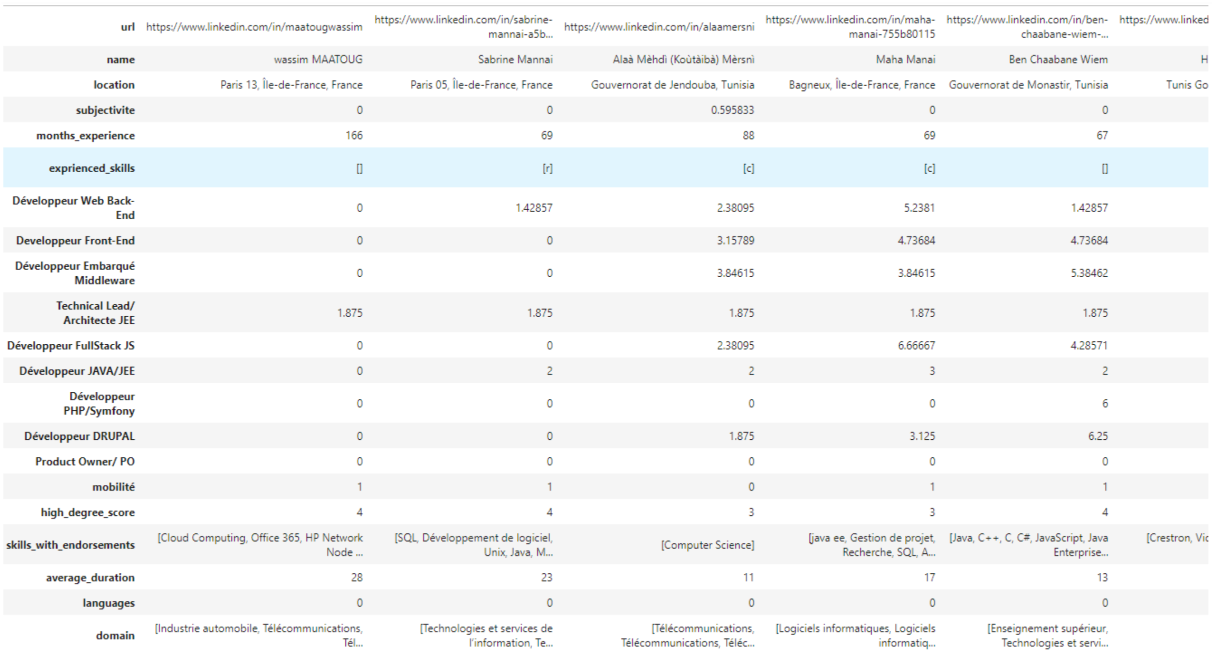


```
for index, person in df_company_scrapped.iterrows():
    print(index,end=' ')
    if person['url'] != '':
        url=person['url']
        url = url.replace("//","/")
        if "about" not in url:
            url = url +"about/"
    else:
        link_url = '-'.join(person["company"].split())
        url = "https://www.linkedin.com/company/"+link_url+"/about/"
    try:
        url.replace(" ", "")
        driver.get(url)
        sleep(1)
        sel = Selector(text = driver.page_source)
        industry = sel.xpath('/html/body/div[5]/div[3]/div[3]/div/div[2]/div/div[2]/div[1]/section/dl/dd[2]/text()').extract_first()
        industry=" ".join(industry)
        person['domaine']=(industry)
    except:
        pass
```

Figure 28 - getting the industry of each company

5. The final Dataframe

The figure in below shows the final data frame which contains the features that we found the most necessary and important for the analysis after passing by the internal and external data preparation.



url	https://www.linkedin.com/in/maatougwassim	https://www.linkedin.com/in/sabrine-mannai-a5b...	https://www.linkedin.com/in/alaamersni	https://www.linkedin.com/in/maha-manai-755b80115	https://www.linkedin.com/in/ben-chaabane-wiem-...	https://www.linked
name	wassim MAATOUG	Sabrine Mannai	Alaà Mèhdì (Koùtàìbà) Mèrsnì	Maha Manai	Ben Chaabane Wiem	H
location	Paris 13, Île-de-France, France	Paris 05, Île-de-France, France	Gouvernorat de Jendouba, Tunisia	Bagneux, Île-de-France, France	Gouvernorat de Monastir, Tunisia	Tunis Go
subjectivite	0	0	0.595833	0	0	
months_experience	166	69	88	69	67	
exprienced_skills	[]	[r]	[c]	[c]	[]	
Développeur Web Back-End	0	1.42857	2.38095	5.2381	1.42857	
Developpeur Front-End	0	0	3.15789	4.73684	4.73684	
Développeur Embarqué Middleware	0	0	3.84615	3.84615	5.38462	
Technical Lead/ Architecte JEE	1.875	1.875	1.875	1.875	1.875	
Développeur FullStack JS	0	0	2.38095	6.66667	4.28571	
Développeur JAVA/JEE	0	2	2	3	2	
Développeur PHP/Symfony	0	0	0	0	6	
Développeur DRUPAL	0	0	1.875	3.125	6.25	
Product Owner/ PO	0	0	0	0	0	
mobilité	1	1	0	1	1	
high_degree_score	4	4	3	3	4	
skills_with_endorsements	[Cloud Computing, Office 365, HP Network Node ...	[SQL, Développement de logiciel, Unix, Java, M...	[Computer Science]	[java ee, Gestion de projet, Recherche, SQL, A...	[Java, C++, C, C#, JavaScript, Java Enterprise...	[Crestron, Vic
average_duration	28	23	11	17	13	
languages	0	0	0	0	0	
domain	[Industrie automobile, Télécommunications, Tél...	[Technologies et services de l'information, Te...	[Télécommunications, Télécommunications, Téléc...	[Logiciels informatiques, Logiciels informatiq...	[Enseignement supérieur, Technologies et servi...	

Figure 29 - the final dataframe

6. Conclusion

This chapter allowed us to prepare our data and determine the final dataset that we will use for the rest of the project. And with some encoding methods for the qualitative features, the dataset will be ready so that it can be processed by models and therefore go to the next step which is modeling.

IV. Modeling

1. Introduction

Modeling is the phase in which our work begins to be more clear. After collecting, cleaning and building all the necessary data, the last part of this project is to create the most suitable model that will meet our business objectives.

In this section, the development of different techniques used will be discussed and explained thoroughly.

2. Cosine Similarity

a) Concept

To measure the similarity between two vectors, the cosine similarity is frequently used especially in recommender systems. In fact, the idea is to calculate the cosine of the angle between two vectors representing the values of the features by the formula below. the obtained results will be between 0 and 1. The closer the value is to 1 the more similar are the vectors, the closer the value is to 0 the less similar are the two vectors.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

Figure 30 - cosine similarity formula

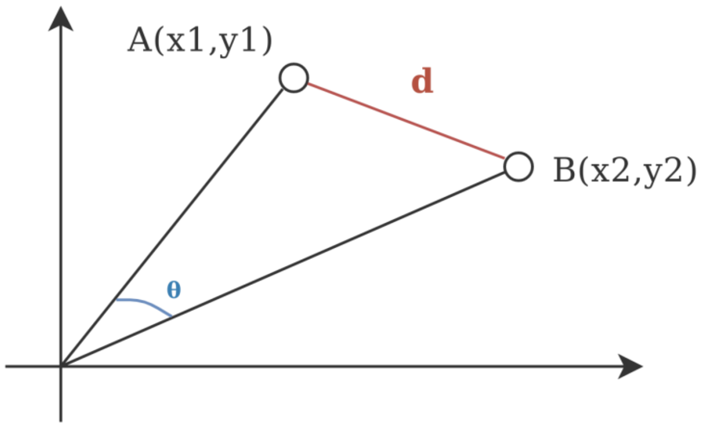
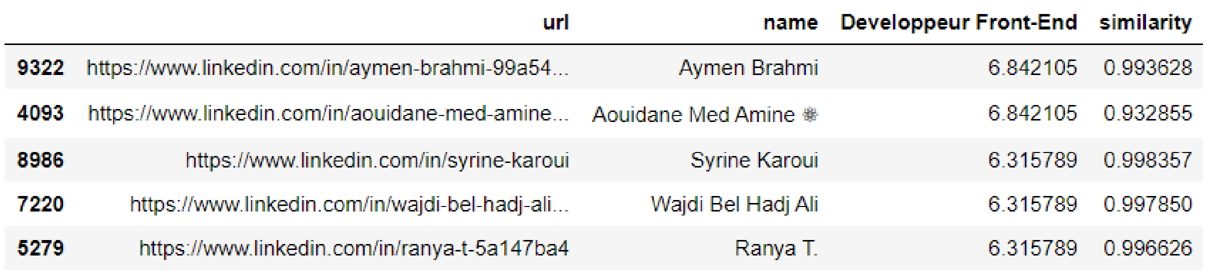


Figure 31- cosine similarity graphically explained

In our case we will compare the similarity between an ideal profile (that contains desirable values) for exemple : Ideal Back-end developper profile … , and the candidates profiles. We will recommend Top n similar candidates to the ideal profile.

b) Results

This dataframe below shows top 5 Front-end developers. This result is obtained by calculating the similarity between the “ideal Front end developer” and others candidates’ profiles. We display then the most similar ones to the ideal profile.



	url	name	Developpeur Front-End	similarity
9322	https://www.linkedin.com/in/aymen-brahmi-99a54...	Aymen Brahmi	6.842105	0.993628
4093	https://www.linkedin.com/in/aouidane-med-amine...	Aouidane Med Amine ❅	6.842105	0.932855
8986	https://www.linkedin.com/in/syrine-karoui	Syrine Karoui	6.315789	0.998357
7220	https://www.linkedin.com/in/wajdi-bel-hadj-ali...	Wajdi Bel Hadj Ali	6.315789	0.997850
5279	https://www.linkedin.com/in/ranya-t-5a147ba4	Ranya T.	6.315789	0.996626

Figure 32 - Top 5 Front end developers

As we can see in this data frame the recommended candidates have a high score in « front-end developer score » and the similarity to the ideal profile is very close to one.

c) Evaluation

Since we are in the case of unsupervised learning, the usual evaluation methods are not valid in our case. So we will take the recommended candidates by the cosine similarity to the ideal profile and make an analogy between the real results and the recommended one.

So our method consists at plotting the recommended results and see if really are similar to the ideal profile. The figure below will explain more.

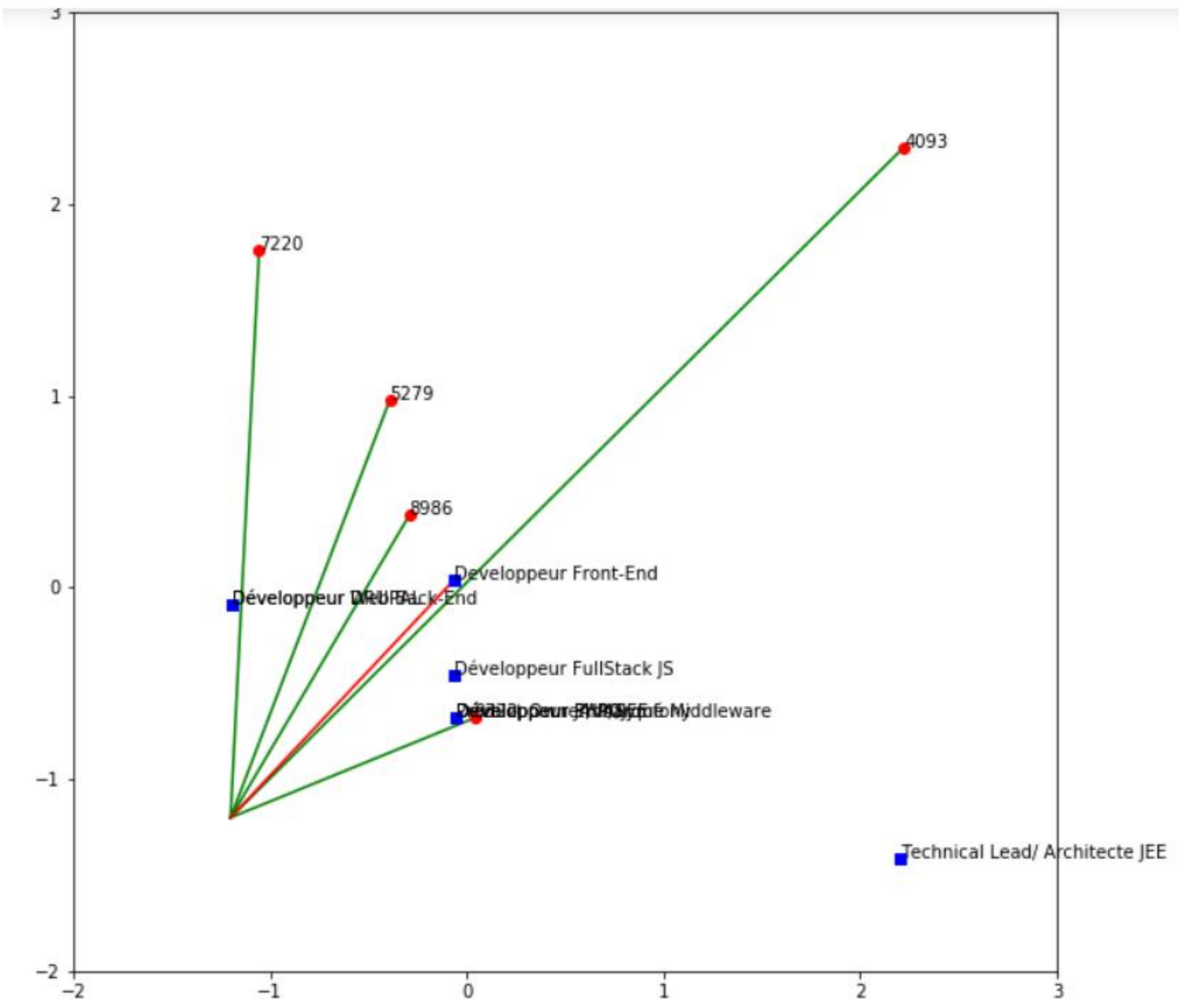


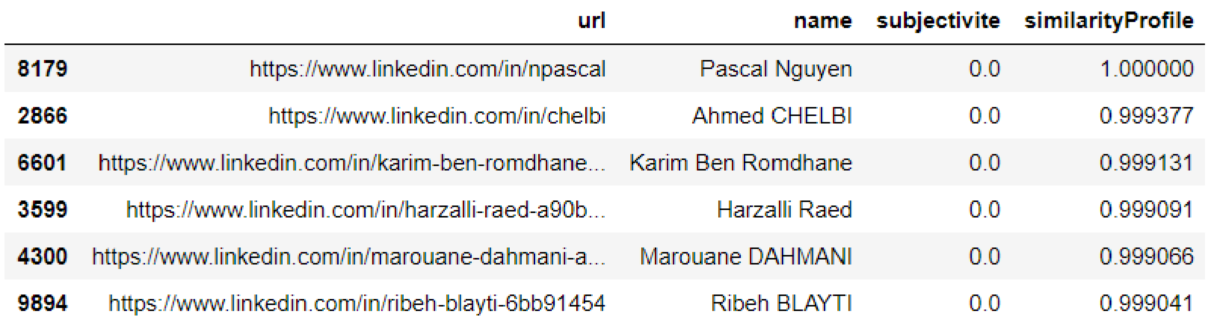
Figure 33 - model evaluation by plotting real results

As you can see in the plot: the red line represents the ideal profile vector and the green lines represent the recommended candidates. The angle between the red line and the greens is small that means that the cosine is close to 1. So the plot confirmed the results obtained.

d) Top n similar to a candidate profile

This functionality allows us to determine the most similar profiles to a given candidate. The concept is the same: we will calculate the cosine similarity between the given profile and the other candidates and we will recommend top n similar profiles.

The dataframe below shows as top 5 similar profiles to “Pascal Nguyen”



	url	name	subjectivite	similarityProfile
8179	https://www.linkedin.com/in/npascal	Pascal Nguyen	0.0	1.000000
2866	https://www.linkedin.com/in/chelbi	Ahmed CHELBI	0.0	0.999377
6601	https://www.linkedin.com/in/karim-ben-romdhane...	Karim Ben Romdhane	0.0	0.999131
3599	https://www.linkedin.com/in/harzalli-raed-a90b...	Harzalli Raed	0.0	0.999091
4300	https://www.linkedin.com/in/marouane-dahmani-a...	Marouane DAHMANI	0.0	0.999066
9894	https://www.linkedin.com/in/ribeh-blayti-6bb91454	Ribeh BLAYTI	0.0	0.999041

Figure 34 - Top 5 similar profiles to Pascal Nguyen

As you can see "Pascal Nguyen" is the most similar to himself (similarity=1) and then there are the top 5 similar profile to this given one.

The evaluation of this recommendation consists also at plotting the real profiles and verify if they are close to this given profile?

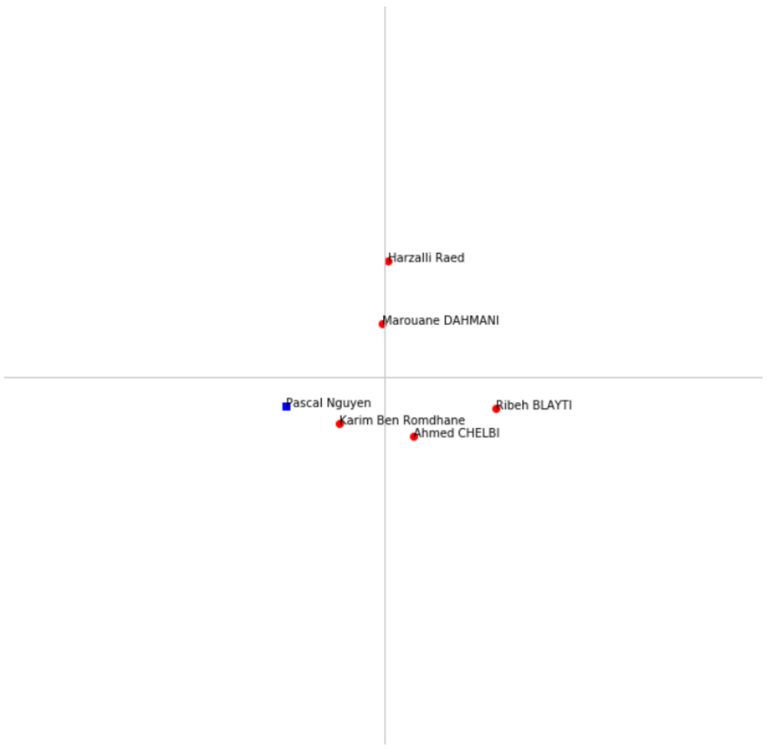
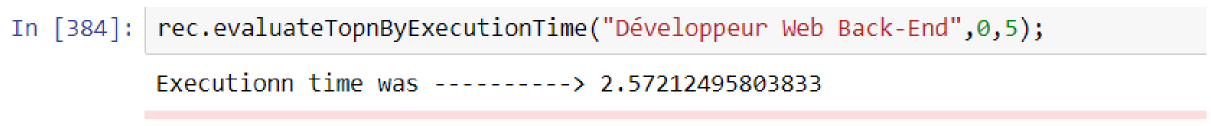


Figure 35 - Plotting similar profiles to Pascal Nguyen

The blue square is the given profile and the red dots are the similar this one as we can see the profiles are close to each other.

e) Execution time :

As you know one of the methods to evaluate a model is execution time, that's why we created a method that calculates execution time for our model



```
In [384]: rec.evaluateTopnByExecutionTime("Développeur Web Back-End",0,5);
          Executionn time was ----------> 2.57212495803833
```

Figure 36 - Execution time for cosine model

As you can see in the figure our model has a good execution time 2 seconds. Which is good by taking into consideration the volume of data that we have and the number of features

3. Vectors difference method

a) Concept

The objective of this method is to give a rating from 1 to 10 based on the initial score which is calculated from the candidate's skills and the other features created on the data preparation steps such as months of experience, mobility, subjectivity, score of high degree. to compute the candidate final score and select the best scores related to each profile searching.

The concept of this method is represented by the following steps:

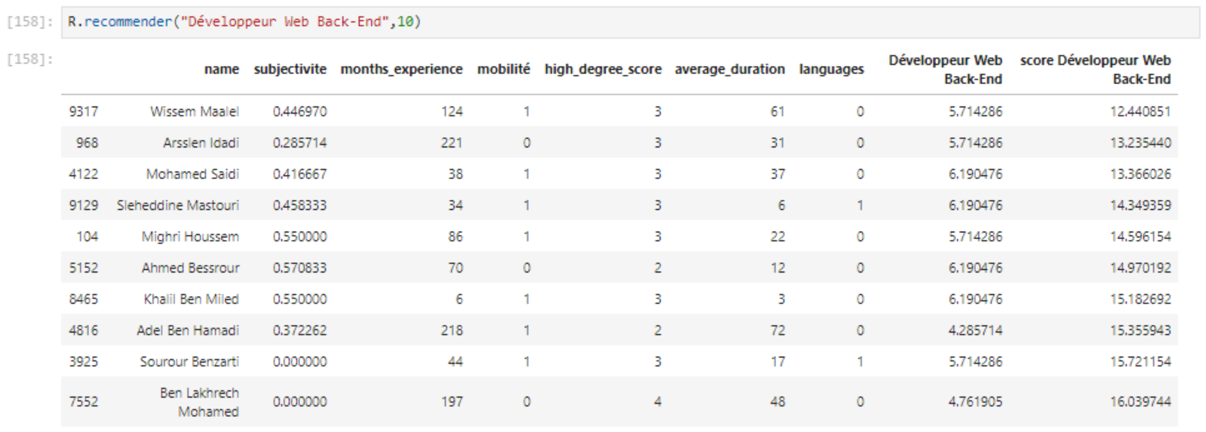
- Create a dataframe with ideal profiles.
- Create a class Recommendation which contains all the methods to be called for:

* Final scores calculation: This method computes the similarity of each candidate with ideal profiles followed by few functions with which our method calculate the similarity between two

vectors representing the values of the features. The idea in this functions is to calculate the difference between those vectors in every feature values. The obtained result will be represented in new columns in our dataframe.

b) Results

Recommendation: This method idea is to select the input number of the most similar candidates (having the most closer score to the ideal input profile) to the given profile.



```
[158]: R.recommender("Développeur Web Back-End",10)
```

[158]:

	name	subjectivite	months_experience	mobilité	high_degree_score	average_duration	languages	Développeur Web Back-End	score Développeur Web Back-End
9317	Wissem Maalel	0.446970	124	1	3	61	0	5.714286	12.440851
968	Arsslen Idadi	0.285714	221	0	3	31	0	5.714286	13.235440
4122	Mohamed Saidi	0.416667	38	1	3	37	0	6.190476	13.366026
9129	Sleheddine Mastouri	0.458333	34	1	3	6	1	6.190476	14.349359
104	Mighri Houssem	0.550000	86	1	3	22	0	5.714286	14.596154
5152	Ahmed Bessrour	0.570833	70	0	2	12	0	6.190476	14.970192
8465	Khalil Ben Miled	0.550000	6	1	3	3	0	6.190476	15.182692
4816	Adel Ben Hamadi	0.372262	218	1	2	72	0	4.285714	15.355943
3925	Sourour Benzarti	0.000000	44	1	3	17	1	5.714286	15.721154
7552	Ben Lakhrech Mohamed	0.000000	197	0	4	48	0	4.761905	16.039744

Figure 37 - Top 10 recommended pofiles Web Back-END

c) Evaluation

As a first idea of model's evaluation, we choose to give the hand to "Wevioo" to give the profile and the number of the most closer candidate to this ideal profile, to plot them in the scatter below to make sure that those candidates were the same recommended persons.

```
[160]: R.evaluation("Développeur Web Back-End",4)
```

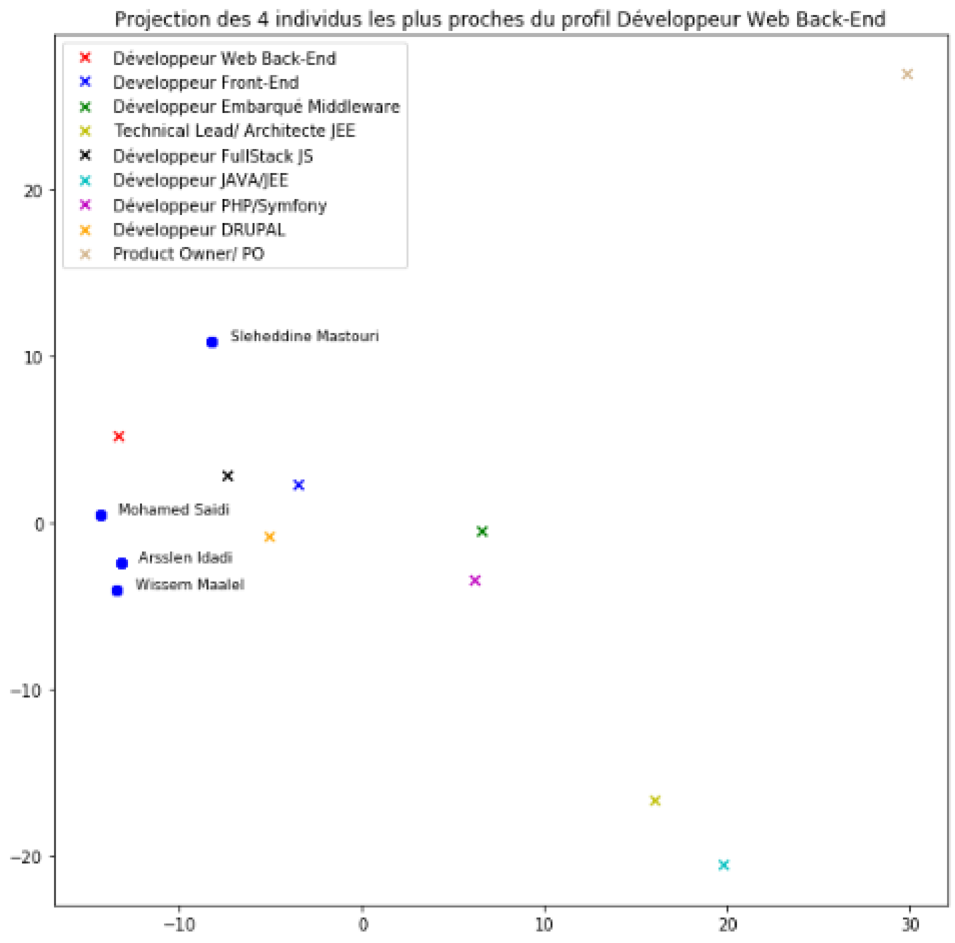
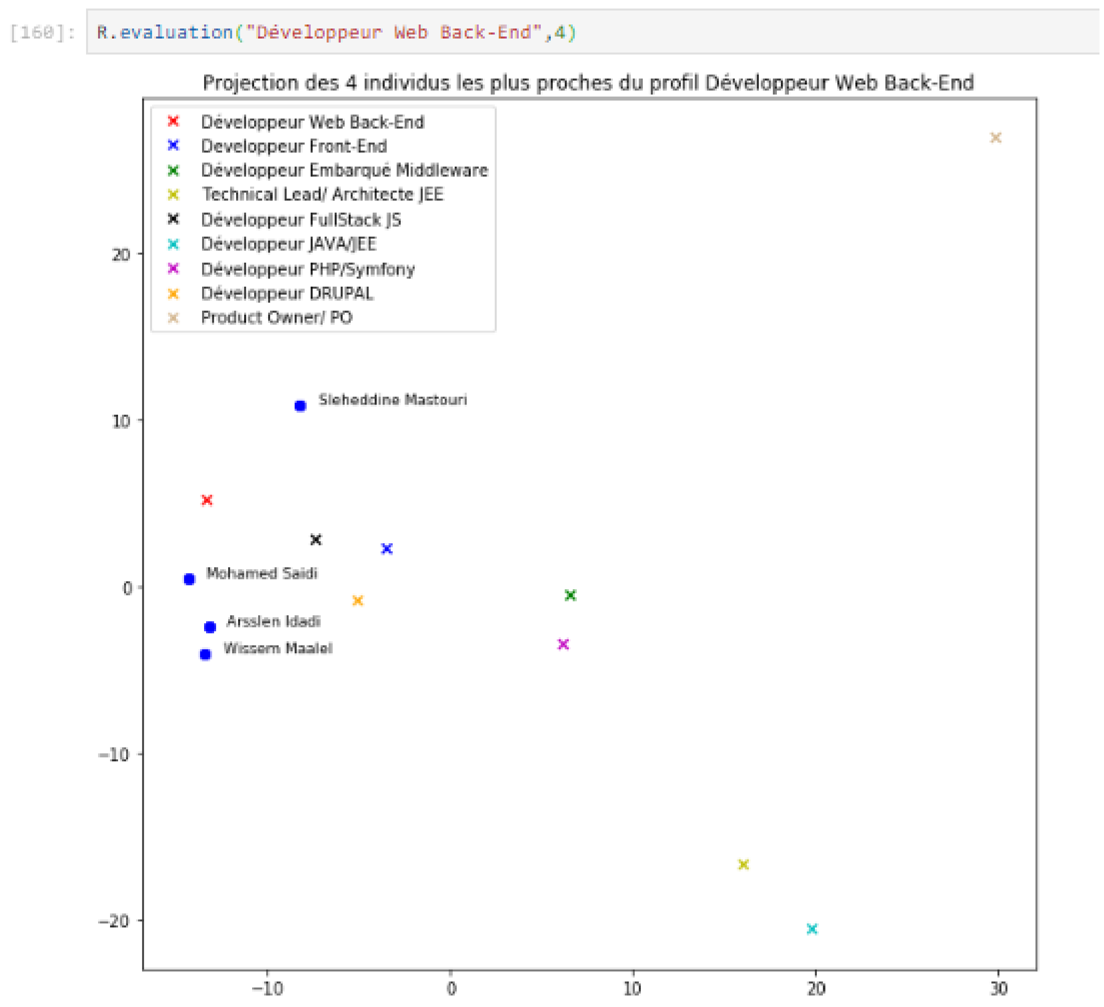


Figure 38 plotting similar candidates to back-end profile

This method allows us to display the Top n similar candidates to the ideal profiles (which contains desirable values) in order to compare every ideal profile with the most n closer candidate to him. The graph bellow will explain more.



```
[160]: R.evaluation("Développeur Web Back-End",4)
```

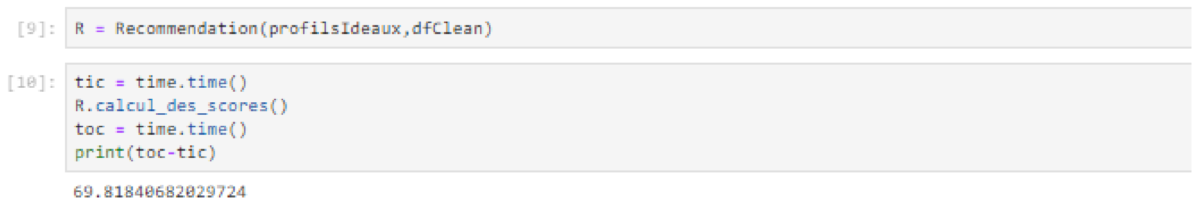
Figure 39 - plotting top n similar candidates to each ideal profile

As you can see in the plot: Every color represent the ideal profile and the top n similar candidates to this last one.

As you can see in the plot: Every color represent the ideal profile and the top n similar candidates to this last one.

- Model run time :

We use the time module witch required to determine the run time of model. So we have worked on our class's functionality to make this time the more less as possible.



```
[9]:  R = Recommendation(profilsIdeaux,dfClean)

[10]: tic = time.time()
      R.calcul_des_scores()
      toc = time.time()
      print(toc-tic)

      69.81840682029724
```

Figure 40 - execution time for this model

4. Linear function by pieces method

a) Concept

Based on the importance of experience helping each individual gaining more knowledge, going deep into technologies and tools used in his domain and having more constructive point of view that's why experience was considered as the most important feature when calculating a score for each candidate.

The goal was to calculate a score for each ideal profile then calculate the score of each profile in order to compare them and to do that we opted to calculate the score based on a function that calculates the growth of skills based on experience.

$$f_p(moisExperience) = \text{a * moisExperience +b}$$

$$a = \frac{MaxSalary - MinSalary}{MaxPerid - MinPeriod}$$

Figure 41profile function formula

<u>from where comes the function fi(MonthsExperience)?</u>

trying to find a relation between months of experience and skills and knowledge of candidates in such a way that "Wevioo" enterprise finds the appropriate person having the most adequate profile.so as said before and according to many studies experience is the most decisive element when recruiting a new person due to its effects on helping candidates approaching the ideal profile.

on our way searching for that relation between months of experience and that ideal profile we found that the relation between the salary and years of experience is the most appropriate

one giving us significant results and helping us to build a clear idea to explain the evolution of a profile depending to experience.

how the function fi(MonthsExperience) works ?

to calculate the fi(MonthsExperience) function we have used linear function by pieces by dividing our experience into intervals (example between 2 and 5 years, then between 5 and 8 and so on ..)

and for each interval (based on a salary barometer by position website values) we have extracted the formula of a linear line having an intercept B and a coefficient A.

$$\text{Score des skills} = \sum_{i=0}^{n} scoreProfile_i * f_p(MoisExperience_i) * corr_i$$

Figure 42skills score

where n represents all scores

f_p represents fonction for the profil p

$coor_i$ represents correlation between ideal profil and i score .

combination between fi(MonthsExperience) function and scores on each job

trying to make our score as logical as possible the output of fi(MonthsExperience) function was multiplied by the score of each profile from our candidates dataframe and then multiplied by the score of each profile from the ideal profile dataframe in other words this score is a translation of the evolution of the skills of a person based on time and its dependence to ideal score on that profile.

$$\text{Score total} = \text{Score des skills} + (3 - \text{Score\_languages}) * (-1) + (4 - \text{Score\_high\_degree}) * (-1) + 100 * \text{indiceExperience}$$

Figure 43total score

Combination between fi(MonthsExperience) function and language score

In the feature “languages” the ideal score 3 is assigned when the profile practice at least 5 languages, then in the score calculation we will subtract the score assigned to each profile in the feature “languages” from 3 and multiply it by -1, in other words the profile with less than 5 languages will be penalized and it depends on the difference

combination between fi(MonthsExperience) function and high_degree_score

In the feature “high_degree_score” the ideal score 4 is assigned when the profile has at least engineering degree and master degree, then in the score calculation we will subtract the score assigned to each profile in the feature “high_degree_score” from 4 and multiply it by -1, in other words the profiles which have highest degree will be less penalized.

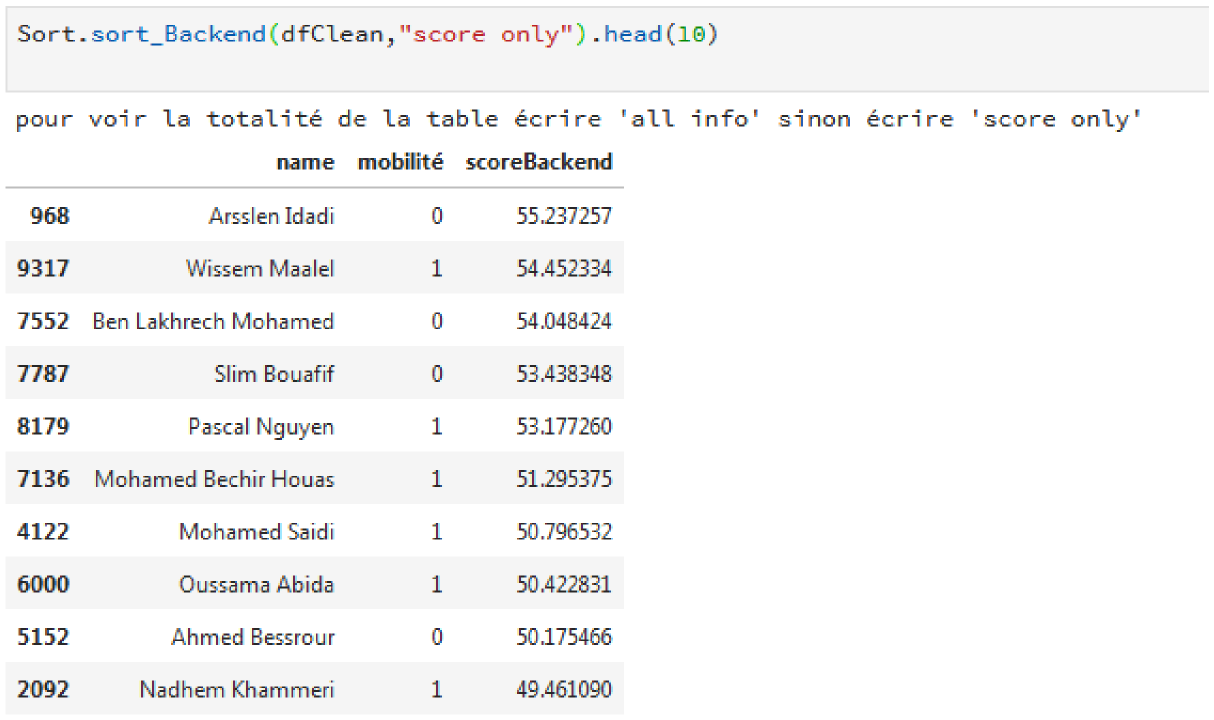
Experience index

Experience index is a binary value (0 or 1) which indicates if a person possesses an average duration of experience in the optimal range defined or not.

The optimal range was defined between 3 and 10 years , so candidates having an average duration in this interval will get 1 and if not they will get 0

We have also treated cases where candidates still did not turn their first three years of experience yet, and in such cases we have assigned 1 to these candidates .

b) Results



```
Sort.sort_Backend(dfClean,"score only").head(10)
```

```
pour voir la totalité de la table écrire 'all info' sinon écrire 'score only'
```

	name	mobilité	scoreBackend
968	Arsslen Idadi	0	55.237257
9317	Wissem Maalel	1	54.452334
7552	Ben Lakhrech Mohamed	0	54.048424
7787	Slim Bouafif	0	53.438348
8179	Pascal Nguyen	1	53.177260
7136	Mohamed Bechir Houas	1	51.295375
4122	Mohamed Saidi	1	50.796532
6000	Oussama Abida	1	50.422831
5152	Ahmed Bessrour	0	50.175466
2092	Nadhem Khammeri	1	49.461090

Figure 44 - Scoring results

In this example and proceeding with the steps explained before we have extracted the top 10 candidates having the most adequate profile for a backend developer we have also mentioned for each one if he had worked aboard before or not and thus wevioo recruitment department will select the one responding to their needs.

c) Evaluation

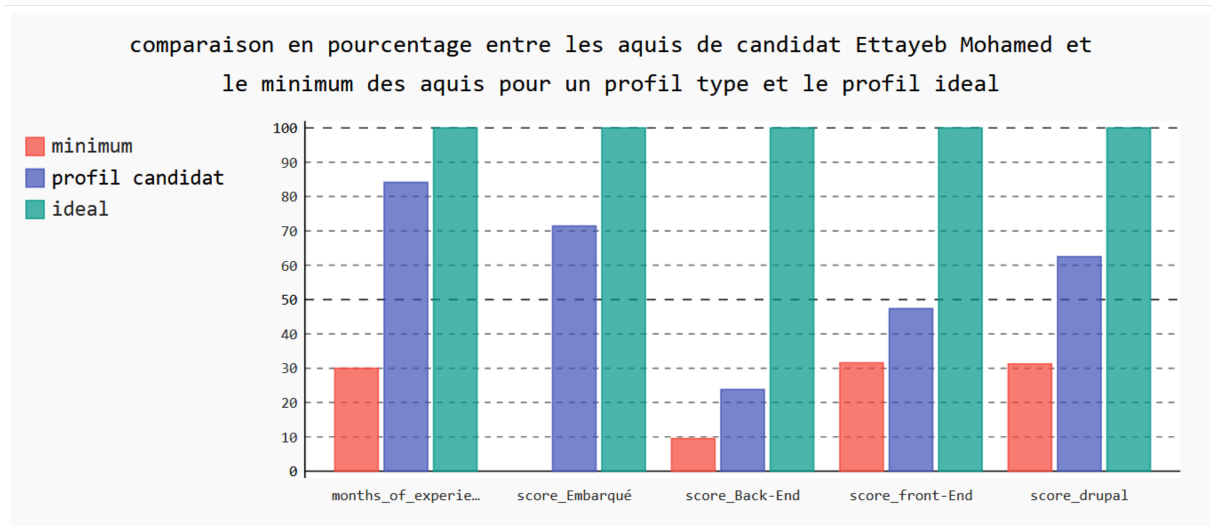


Figure 45-Plotting profile between minimum and ideal

To evaluate our results we have chosen to plot our candidate months of experience and skills between the minimum demanded in that post and the maximum score representing the ideal profile

5. Euclidean smilarity

a) Concept

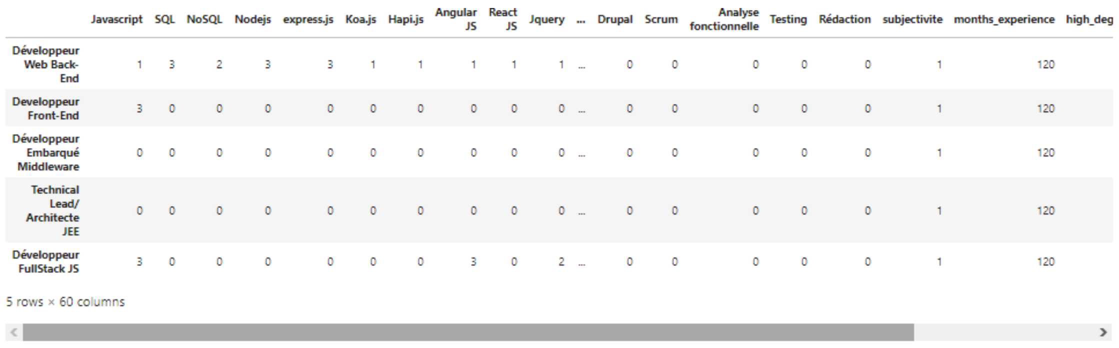
Euclidean distance is one of the most used distance metrics in machine learning algorithms. It is calculated using the Murkowski Distance formula. It provides a relationship metric between each element in the dataset. The distance ‘d’ formula is calculated a below:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Figure 46 - euclidean distance formula

So how did we use the Euclidean distance to make our prediction?

First of all we have created a typical profile in which we have concentrated in the first place on the skills that each candidates masters while taking into consideration the weightings of skills for each profile given by Wevioo putting them in the columns with the other features that we have extracted in the previous phase. The dataFrame has the following structure:

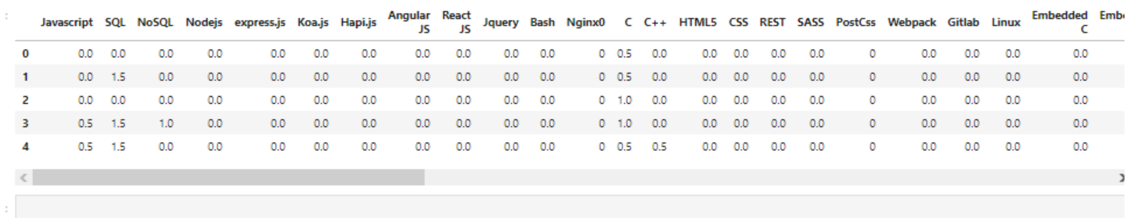


	Javascript	SQL	NoSQL	Nodejs	express.js	Koa.js	Hapi.js	Angular JS	React JS	Jquery	...	Drupal	Scrum	Analyse fonctionnelle	Testing	Rédaction	subjectivite	months_experience	high_deg
Développeur Web Back-End	1	3	2	3	3	1	1	1	1	1	...	0	0	0	0	0	1	120	
Developpeur Front-End	3	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	1	120	
Développeur Embarqué Middleware	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	1	120	
Technical Lead/ Architecte JEE	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	1	120	
Développeur FullStack JS	3	0	0	0	0	0	0	3	0	2	...	0	0	0	0	0	1	120	

5 rows × 60 columns

Figure 47 - the new dataframe

Then we have created a new DataFrame which has the same structure of the previous one but is indexed with the candidates given by “Wevioo”. In this DataFrame each candidate was given a score for each skill according to the weightings proposed by the society. The DataFrame has the following structure:



	Javascript	SQL	NoSQL	Nodejs	express.js	Koa.js	Hapi.js	Angular JS	React JS	Jquery	Bash	Nginx0	C	C++	HTML5	CSS	REST	SASS	PostCss	Webpack	Gitlab	Linux	Embedded C	Emb
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.5	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	
1	0.0	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.5	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	1.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	
3	0.5	1.5	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	1.0	0.0	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	
4	0.5	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0	0.5	0.5	0.0	0.0	0.0	0.0	0	0.0	0.0	0.0	0.0	

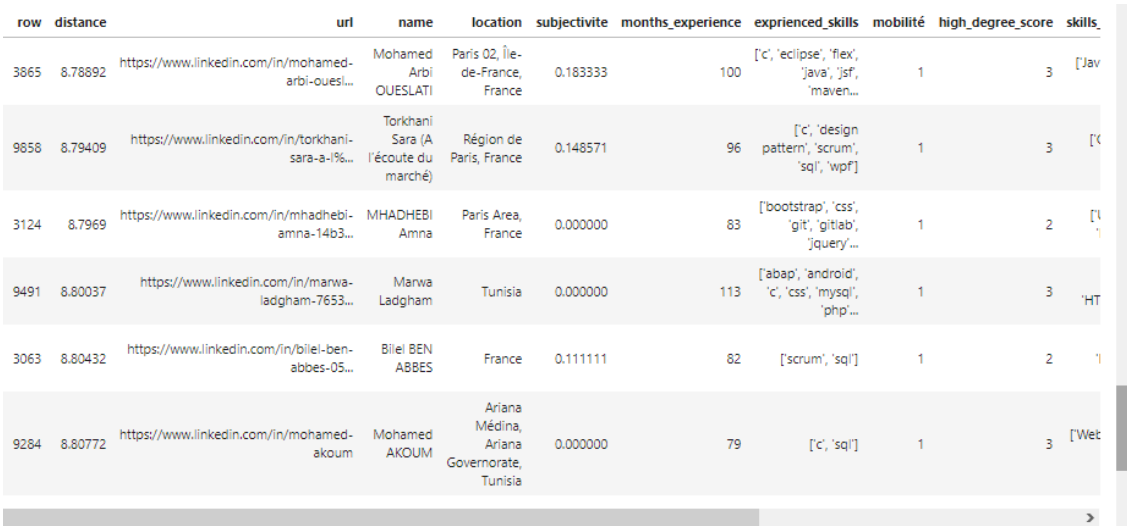
Figure 48 - ideal profiles dataFrame

The idea is to calculate the Euclidean distance between each candidate and the desired profile (In each iteration a distance will be calculated between a specific row from the typical profile DataFrame and a row from the candidates DataFrame).

b) Results

Once we have all the Euclidean distances calculated we have sorted them in order to find the k-nearest neighbors to a profile and then suggest the most k profiles that are suitable for a specific job.

In this figure we have represented the most 6 profiles which according to this method are more susceptible to occupy the job of "Développeur Web Back-End":



row	distance	url	name	location	subjectivite	months_experience	exprienced_skills	mobilité	high_degree_score	skills_
3865	8.78892	https://www.linkedin.com/in/mohamed-arbi-ouesl...	Mohamed Arbi OUESLATI	Paris 02, Île-de-France, France	0.183333	100	['c', 'eclipse', 'flex', 'java', 'jsf', 'maven...	1	3	['Jav
9858	8.79409	https://www.linkedin.com/in/torkhani-sara-a-l%...	Torkhani Sara (A l'écoute du marché)	Région de Paris, France	0.148571	96	['c', 'design pattern', 'scrum', 'sql', 'wpf']	1	3	['(
3124	8.7969	https://www.linkedin.com/in/mhadhebi-amna-14b3...	MHADHEBI Amna	Paris Area, France	0.000000	83	['bootstrap', 'css', 'git', 'gitlab', 'jquery'...	1	2	[' '
9491	8.80037	https://www.linkedin.com/in/marwa-ladgham-7653...	Marwa Ladgham	Tunisia	0.000000	113	['abap', 'android', 'c', 'css', 'mysql', 'php'...	1	3	'HT
3063	8.80432	https://www.linkedin.com/in/bilel-ben-abbes-05...	Bilel BEN ABBES	France	0.111111	82	['scrum', 'sql']	1	2	'
9284	8.80772	https://www.linkedin.com/in/mohamed-akoum	Mohamed AKOUM	Ariana Médina, Ariana Governorate, Tunisia	0.000000	79	['c', 'sql']	1	3	['Web

Figure 49 - Recomended candidates in Euclidean similarity

6. Decision making

After building, testing and evaluating our 4 models which allowed us to recommend the appropriate candidates from different perspectives (all criteria perspective, score perspective, experience perspective and skills perspective), and to be in harmony with our client demands we have developed a decision making system which will make a precise selection from the results of our four models to return the best result for the recruiter. The decision consists of calculating a recommendation score for the recommended candidates and the figure below will explain more

For example, if the recruiter chooses to make his decision based on “All criteria” for only two candidates he will be following this scenario

If the candidate belongs to all criteria model a value of two multiplied by a coefficient describing his rank will be assigned and if a candidate shows up in another model, he will have as a recommendation score one multiplied by the value describing his rank.

All criteria Recommendation

Rank	candidate	Recommendation score
0	X	2 * (2-0) = 4
1	Y	2 *(2-1) = 2

Score Recommendation

Rank	candidate	Recommendation score
0	X	1 * (2 -0) =2
1	Z	1 * (2 -1) =1

Experience Recommendation

Rank	candidate	Recommendation score
0	Y	1 * (2 -0) =2
1	W	1 * (2 -1) =1

Skills Recommendation

Rank	candidate	Recommendation score
0	A	1 * (2 -0) =2
1	W	1* (2 -1) =1

After having calculated the score of each candidate according to the order of their appearance in each recommendation model ,the result will be as follows

candidate	Recommendation score
X	6
Y	4
W	2
A	2
Z	1

And this will be the final result displayed for the recruiter

candidate
X
Y

7. Conclusion

This phase is one of the most important steps of the IBM MASTER PLAN method, where from which we were able to test different techniques in order to have an optimal model. After this phase we will pass to the deployment part in which we will create a web application for "Wevioo" human resources department where they will be able to get recommendation about candidates when recruiting new employers.

V. Deployment

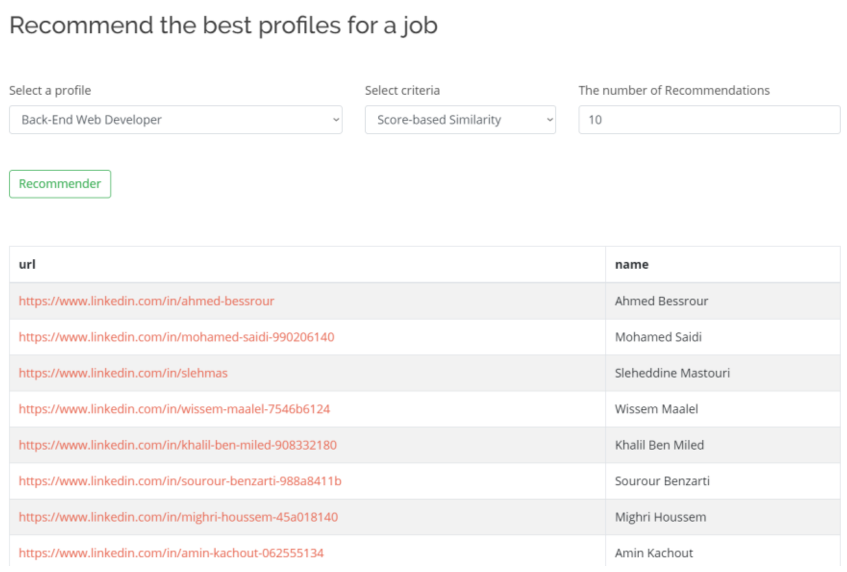
1. Introduction

After we have a set of models that perform well, we can operationalize them for other applications to consume in easy way. In this final part of the project we will deploy our model in a format of a practical solution that enables the model to be easily consumed from a website application which is realized with Django.

2. Our web application

a) Front office

- Recommend top N: In this page the recruiter can select the desired profile title and the number of the candidates he needs to recommend, and then after all calculations are done a final list of candidates will be displayed.



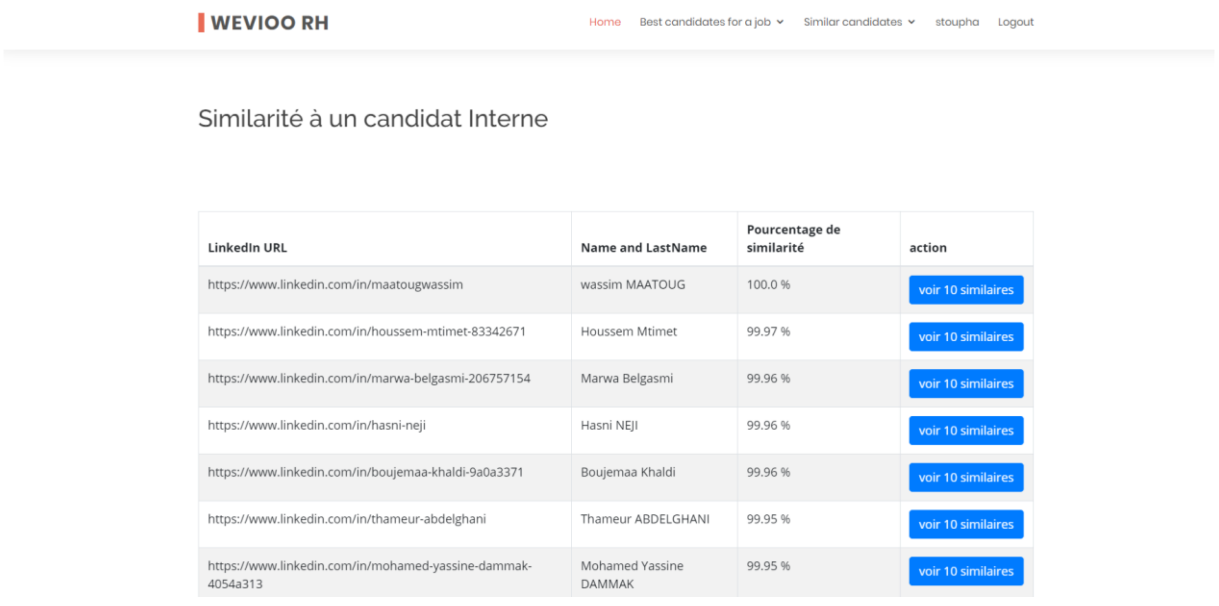
Recommend the best profiles for a job

Select a profile: Back-End Web Developer
Select criteria: Score-based Similarity
The number of Recommendations: 10

url	name
https://www.linkedin.com/in/ahmed-bessrour	Ahmed Bessrour
https://www.linkedin.com/in/mohamed-saidi-990206140	Mohamed Saidi
https://www.linkedin.com/in/slehmas	Sleheddine Mastouri
https://www.linkedin.com/in/wissem-maalel-7546b6124	Wissem Maalel
https://www.linkedin.com/in/khalil-ben-miled-908332180	Khalil Ben Miled
https://www.linkedin.com/in/sourour-benzarti-988a8411b	Sourour Benzarti
https://www.linkedin.com/in/mighri-houssem-45a018140	Mighri Houssem
https://www.linkedin.com/in/amin-kachout-062555134	Amin Kachout

Figure 50 - Recommend Top n

- Internal similarity: In this page there will be the list of all the candidates, where it's possible to choose one of the candidates in order to find the 10 most similar profiles to the selected one.

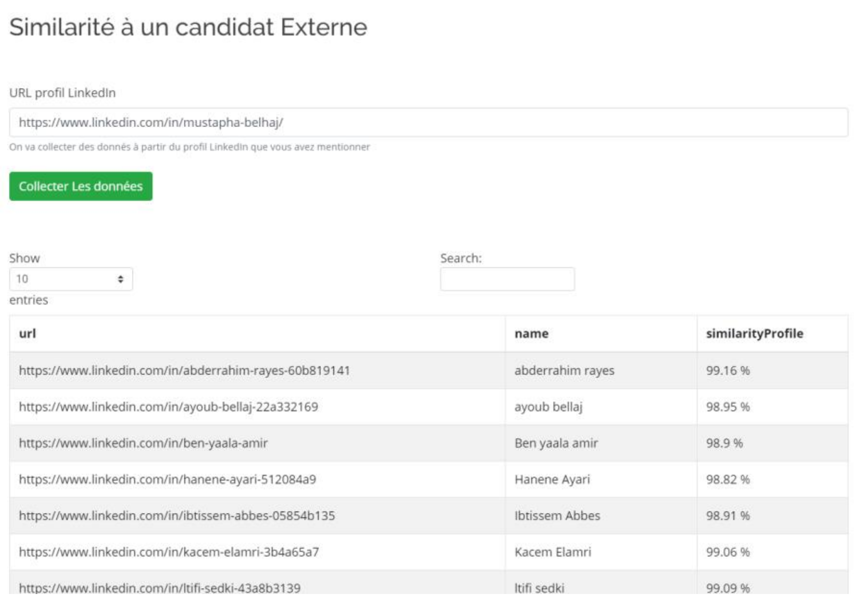


Similarité à un candidat Interne

LinkedIn URL	Name and LastName	Pourcentage de similarité	action
https://www.linkedin.com/in/maatougwassim	wassim MAATOUG	100.0 %	voir 10 similaires
https://www.linkedin.com/in/houssem-mtimet-83342671	Houssem Mtimet	99.97 %	voir 10 similaires
https://www.linkedin.com/in/marwa-belgasmi-206757154	Marwa Belgasmi	99.96 %	voir 10 similaires
https://www.linkedin.com/in/hasni-neji	Hasni NEJI	99.96 %	voir 10 similaires
https://www.linkedin.com/in/boujemaa-khaldi-9a0a3371	Boujemaa Khaldi	99.96 %	voir 10 similaires
https://www.linkedin.com/in/thameur-abdelghani	Thameur ABDELGHANI	99.95 %	voir 10 similaires
https://www.linkedin.com/in/mohamed-yassine-dammak-4054a313	Mohamed Yassine DAMMAK	99.95 %	voir 10 similaires

Figure 51- Internal Similarity

- External similarity: In this page the recruiter has the possibility to paste the candidate's LinkedIn URL in order to see the 10 similar candidates in our database.



Similarité à un candidat Externe

URL profil LinkedIn

https://www.linkedin.com/in/mustapha-belhaj/

On va collecter des donnés à partir du profil LinkedIn que vous avez mentionner

url	name	similarityProfile
https://www.linkedin.com/in/abderrahim-rayes-60b819141	abderrahim rayes	99.16 %
https://www.linkedin.com/in/ayoub-bellaj-22a332169	ayoub bellaj	98.95 %
https://www.linkedin.com/in/ben-yaala-amir	Ben yaala amir	98.9 %
https://www.linkedin.com/in/hanene-ayari-512084a9	Hanene Ayari	98.82 %
https://www.linkedin.com/in/ibtissem-abbes-05854b135	Ibtissem Abbes	98.91 %
https://www.linkedin.com/in/kacem-elamri-3b4a65a7	Kacem Elamri	99.06 %
https://www.linkedin.com/in/ltifi-sedki-43a8b3139	ltifi sedki	99.09 %

Figure 52 - External similarity

b) Back office

- Dashboard: This page presents an interface which organizes and presents information in an interactive way that is easy to read such as : mobility percentage, Language, experiences duration, top skills, high degree.

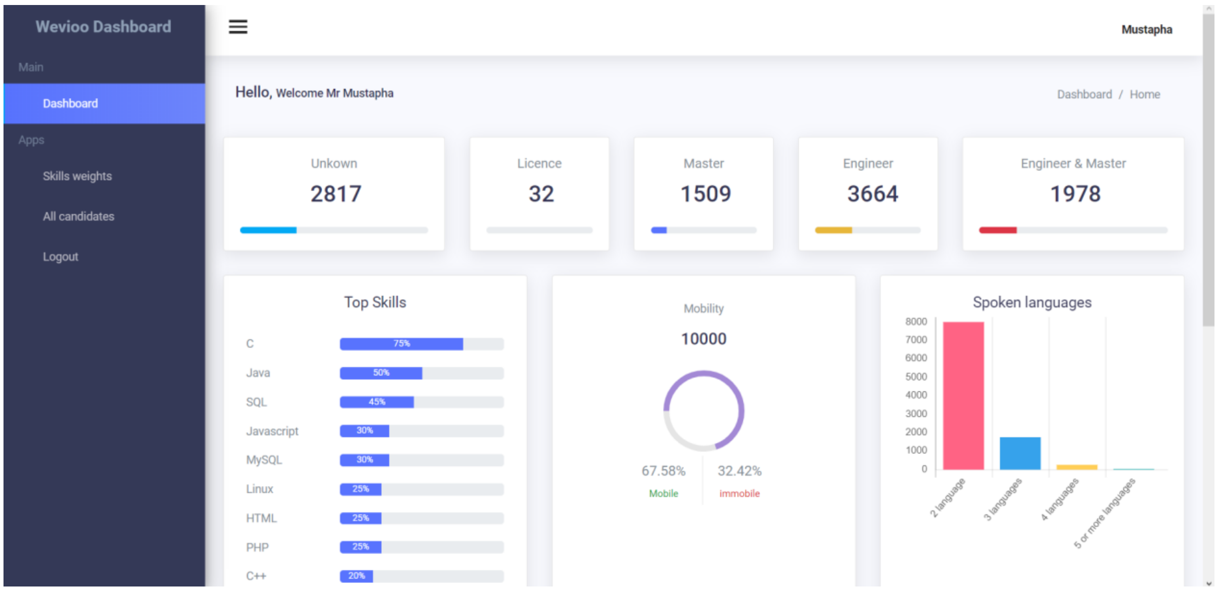
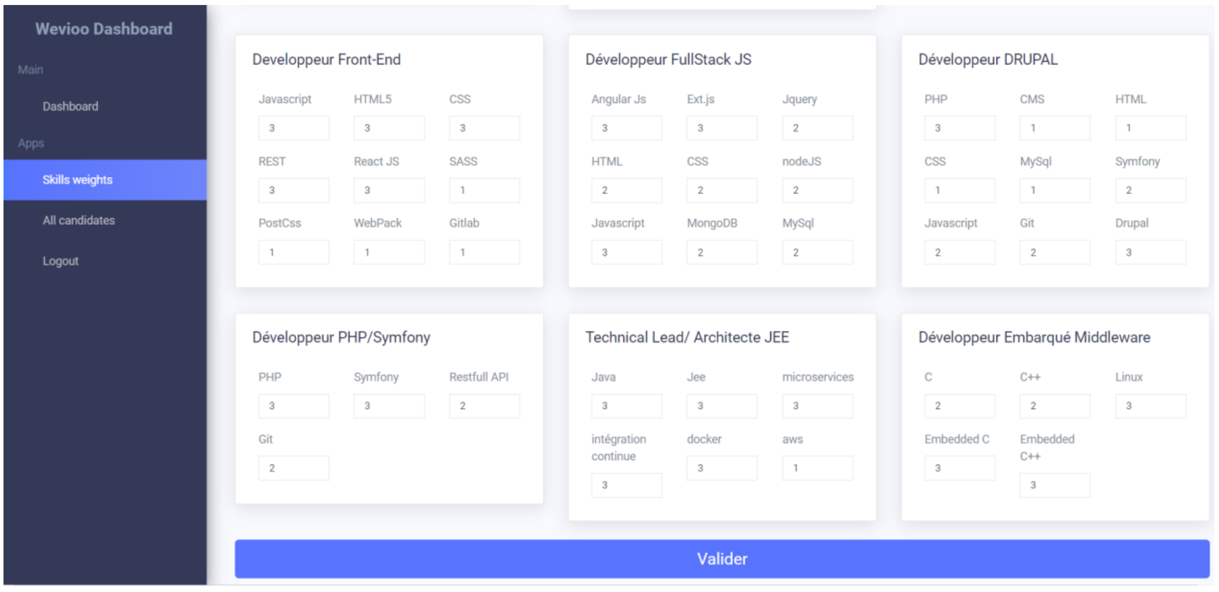


Figure 53 - Dashboard

- Skills weights: In case the recruiter changed his mind about the weights given for the skills, It is possible to change them through this interface which contains all the skills desired for each profile with their initial weights.



Developpeur Front-End

Javascript	HTML5	CSS
3	3	3
REST	React JS	SASS
3	3	1
PostCss	WebPack	Gitlab
1	1	1

Développeur FullStack JS

Angular Js	Ext.js	Jquery
3	3	2
HTML	CSS	nodeJS
2	2	2
Javascript	MongoDB	MySql
3	2	2

Développeur DRUPAL

PHP	CMS	HTML
3	1	1
CSS	MySql	Symfony
1	1	2
Javascript	Git	Drupal
2	2	3

Développeur PHP/Symfony

PHP	Symfony	Restfull API
3	3	2
Git		
2		

Technical Lead/ Architecte JEE

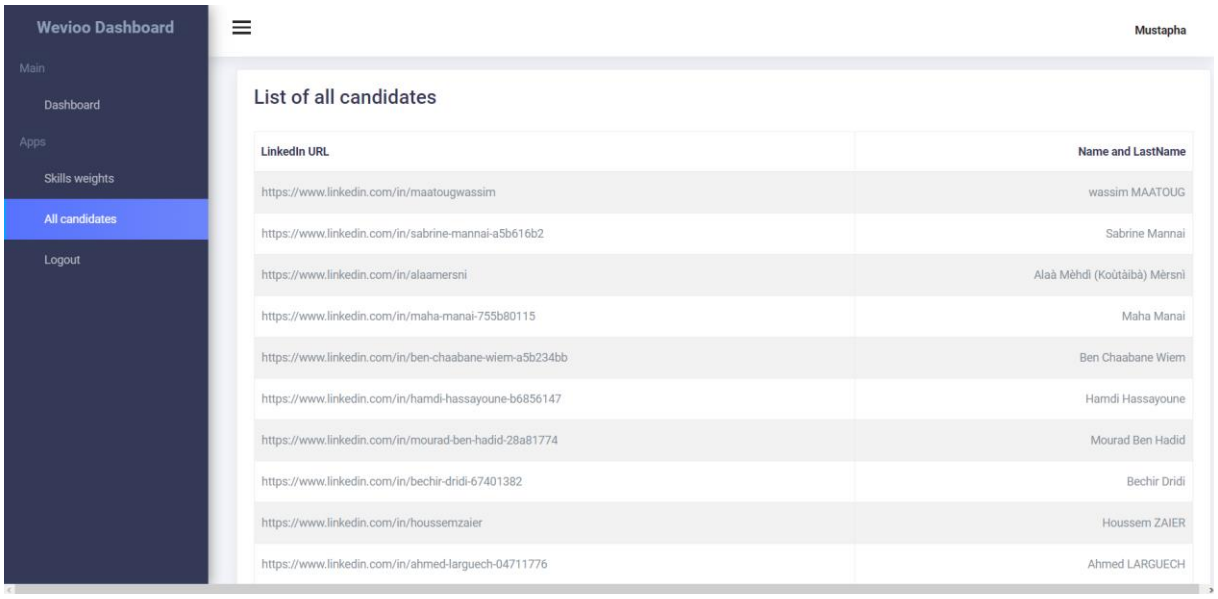
Java	Jee	microservices
3	3	3
intégration continue	docker	aws
3	3	1

Développeur Embarqué Middleware

C	C++	Linux
2	2	3
Embedded C	Embedded C++	
3	3	

Figure 54 - Skills weight

- Display the database of candidates: Whenever the recruiter wants to check the list of the candidates available in the database he can access this page.



List of all candidates

LinkedIn URL	Name and LastName
https://www.linkedin.com/in/maatougwassim	wassim MAATOUG
https://www.linkedin.com/in/sabrine-mannai-a5b616b2	Sabrine Mannai
https://www.linkedin.com/in/alaamersni	Alaà Mèhdì (Koùtàibà) Mèrsnì
https://www.linkedin.com/in/maha-manai-755b80115	Maha Manai
https://www.linkedin.com/in/ben-chaabane-wiem-a5b234bb	Ben Chaabane Wiem
https://www.linkedin.com/in/hamdi-hassayoune-b6856147	Hamdi Hassayoune
https://www.linkedin.com/in/mourad-ben-hadid-28a81774	Mourad Ben Hadid
https://www.linkedin.com/in/bechir-dridi-67401382	Bechir Dridi
https://www.linkedin.com/in/houssemzaier	Houssem ZAIER
https://www.linkedin.com/in/ahmed-larguech-04711776	Ahmed LARGUECH

Figure 55 - All candidates

3. Conclusion

In this chapter we have presented the different parts of the deployment phase in which we have described the content of each interface in our delivered website application.

VI. Conclusion

Customer satisfaction is the key to success of any business, whatever

its sector of activity. Their loyalty is even more so, because it is not enough that the

customer is satisfied, but on top of that, he will come back again to ask for

another service, As much as the number of services increases the turnover of the company will do so. The points mentioned above are not achievable until the company resolve many problems, on top of them, the process of recruitment.

The Internet caused a substantial impact on the recruitment process through the creation of e-recruiting platforms that become a primary recruitment channel in most companies. While Wevioo was suffering of recruitment's time and cost. E-recruitment platforms accomplished clear advantages for recruiters by allowing them to reduce the time of recruitment and its process cost.

This is what WEVIOO is looking for when launching this project, because they know well that all digital service companies feel that they have mastered the assignment of consultants to the corresponding missions. Unfortunately, this is not the case as long as they are limited to matching the skills requested and don't dig into skills that are similar.

During this project, our mission was to design and develop a system of recommendation which will be a practical tool to improve and facilitate the assignment consultants to WEVIOO recruiters. So, to achieve this goal we have started with an understanding of the mission which allowed us to determine the need of our customer “WEVIOO” for business and data science objectives that are the subject of our project. then, we have entered the data understanding phase, through which we were able to clarify the data given by WEVIOO, and its issues. Based on this understanding, we moved on to a preparation of data. This phase allowed us to prepare our data for the next steps, in other words to extract new features, use the NLP and to apply the data cleaning in order to make it compatible with the modeling algorithms. Following this preparation, our data is ready to use. We have created 4 models which are based on similarity, during of experience, skills and scores.

After gathering the results of the models we passed through the decision making step which allowed us to return a final recommendation based on the intersection between our 4 models. Finally, we finished with the deployment phase in which we developed a website application.