# APPROACH

APPROACH: For rating the similarity between two resumes on scale of 0-1, we compare the keywords from a pre-defined list between the two resumes.

Q: How keywords list is prepared?

Ans: The list of keywords is prepared manually by examining the 5 example resume text files given in the problem statement and picking out the most promising words.

|  |  |  |
| --- | --- | --- |
| Keywords | Resume1 | Resume2 |
| Keyword-1 | 1 | 0 |
| Keyword-2 | 1 | 1 |
| Keyword-3 | 0 | 1 |
| Keyword-4 | 0 | 0 |
| Keyword-5 | 0 | 1 |
| Keyword-6 | 1 | 0 |
| Keyword-7 | 1 | 1 |
| ……………. | -- | - |
| ………………. | -- |  |

Then after keywords list is prepared a matrix is made similar to the below figure:

Fig. Comparing resumes Resume1 and Resume2

Here in the above table ‘1’ represents corresponding keyword is present in Resume and ‘0’ represents that the keyword is not present in the corresponding resume.

In the sim.py file we firstly make a l\_main\_list, These vectors of [1,1] or [0,1] or [1,0] or [0,0] values of each keyword are appended to main list in form of sublists. So, the main list contains sublists as: l\_mainlist=[[1,1],[1,0],[0,1],[0,0],……..] with each sublist denoting to a row corresponding to a keyword in the above table.

But some of the keywords repeat themselves very often they cannot be treated as other keywords which occur only few times. In order to address and avoid this problem, we make a vector containing the values min( Resume1.count(keyword) , Resume2.count(keyword) ) for every keyword in our keywords list .(Let us call this list as count\_vector)

min(a,b) gives the minimum of values a and b.

Resume1.count(keyword)—gives the number of times keyword is repeating itself in Resume1.

Resume2.count(keyword)—gives the number of times keyword is repeating itself in Resume2.

Regarding why we take minimum of these two values is discussed afterwards.

If all keywords are occurring only once in each resume the similarity between two resumes can be calculated as:

(No. of features in both resumes) divided by

((No. of features in both resumes) + (No. of features present either in Resume1 or Resume2))

Mathematically No. of features present in both resumes is nothing but the AND-GATE between the vector values of Resume1 and Resume2=No. of [1,1]’s in the above table

Similarly, No. of features present in either in Resume1 or Resume2 = No. of [1,0]’s + No. of [0,1]’s

The keywords not present in both the resumes are ignored because similarity comes from the keywords present in both resumes but not from the keywords not present in either of them. So, the No. of [0,0]’s is just ignored.

The above made calculation is good approach when keywords are not repeating but when keywords are repeating we take the count-vector into consideration and add value of corresponding keyword in the count-vector instead of one for each [1,1] case.(This is similar to weighted mean)

Q: Why we take minimum when calculating the count-vector?

Ans: Let’ say that we have three resumes belonging to programmer1, programmer2 and manager. Let’s say the keyword we are considering is ‘python’. Let the no. of times this keyword ‘python’ occurs in programmer1’s Resume be 25.The Resume belonging to a manager has less chance of repetition of keyword ‘python’ but concretely it may occur once or twice. Without loss in generality the programmer2 has more chance of repetition of this keyword and let us take this value as 10.

So, when we are comparing the Resumes of programmer1 and manager and if we take maximum of these two values we get a value of 25 into the count-vector. Similarly if we take the maximum value between programmer1 and programmer2 we still get value of 25 into count-vector.

But as we have same values in count-vector between the above two cases of comparing programmer1,manager(case1) and programmer1,programmer2(case2) we tend to predict the similarity as same in both cases, so this above method of taking maximum is not valid in reality as similarity between the case1 must be lower than case2 generally.

But taking the minimum gives the values min(1,25)=1 in case1 and min(10,25)=10 in case2.Concretely,this method gives more similarity between case2 than case1 which more probabilistic and close to reality. So,minimum of two values is best way of selecting the effective number of times the keyword is repeated without the loss of generality.