SUBMISSION REPORT (Round 1)

TCS Talent Ocean

Ketan Patil
k2p1020@gmail.com
7744895120

MY DETAILS

Name: Ketan Shantaram Patil
CT Number: CT20203174022
Gmail: k2p1020@gmail.com
Contact Number: 7744895120
College Name: Vidyalankar Institute of Technology, Mumbai
Current Education: Bachelor of engineering Information Technology (T.E)

Approach to solve problem:

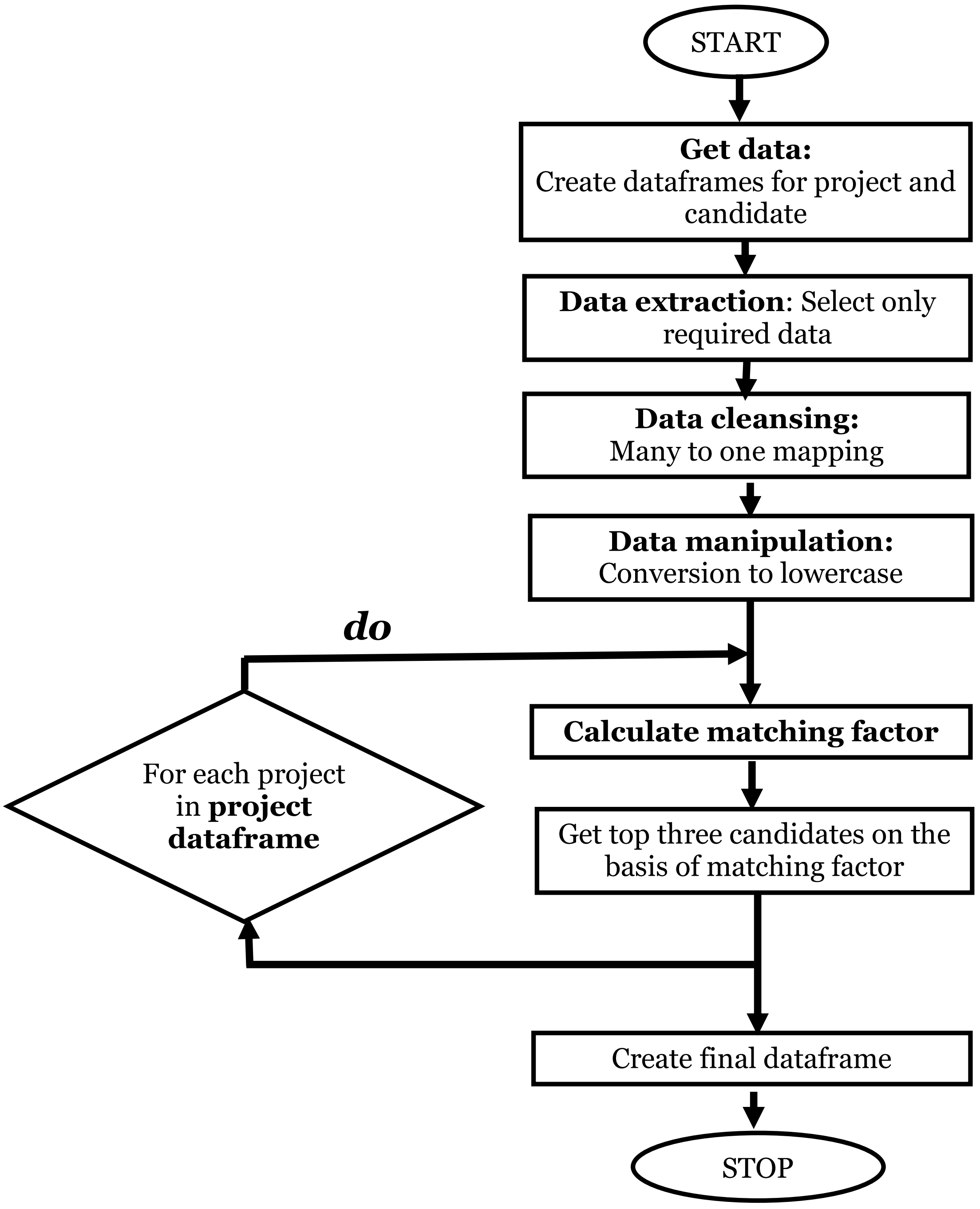
My approach to solve given problem includes following steps:

1. Get data
2. Data extraction
3. Data cleansing
4. Data manipulation
5. Calculation of matching factor
6. Get suitable candidates on the basis of matching factor

Algorithm:

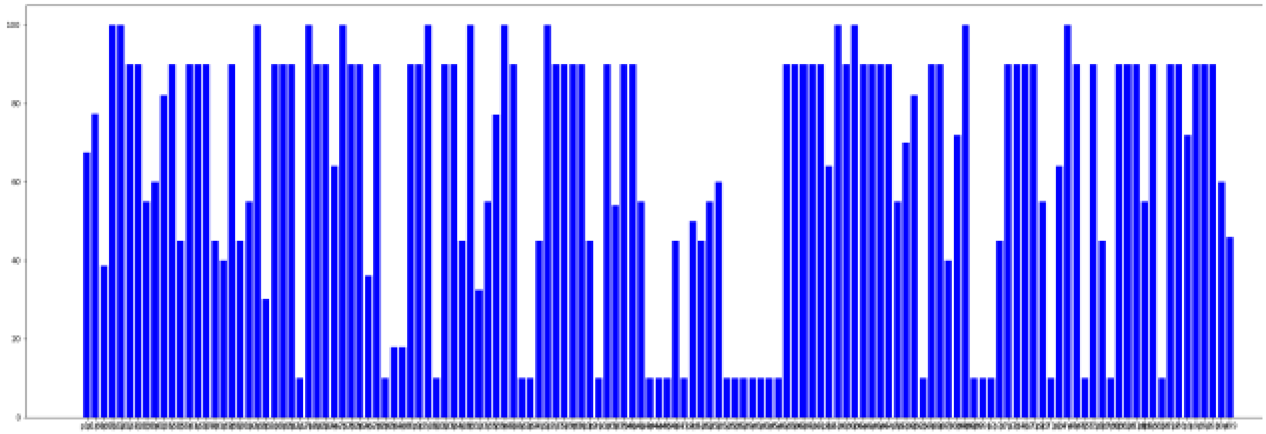
1. Start
2. Create two dataframes for candidate and project details.
3. Extract required columns (**project id**, **location**, **required skills**) from main dataframe of **project.**
4. **Data extraction:** Extract required columns (**candidate id**, **location**, **skills**) from main dataframe of **candidate.**
5. **Data manipulation:** Convert both dataframes data into lower case for simplicity of comparison.
6. **Data cleansing:** for both dataframes replace multiples by one.
   Example**: java programming**, **java language**, **core java**, **programming in java**, **java se** can be replaced by simple one-word **java**.
7. Calculate **matching factor** of every candidate for every project
   Matching factor = 90% of (matching skills) + 10% of (location match)
8. Get top 5 candidates having high matching factor for the project.
9. Final dataframe **(project id, location, required skills, top 5 suitable candidates, matching factor for top 5 candidates.)**

Flowchart:


START
Get data:
Create dataframes for project and candidate
Data extraction: Select only required data
Data cleansing:
Many to one mapping
Data manipulation:
Conversion to lowercase
do
Calculate matching factor
For each project in **project dataframe**
Get top three candidates on the basis of matching factor
Create final dataframe
STOP

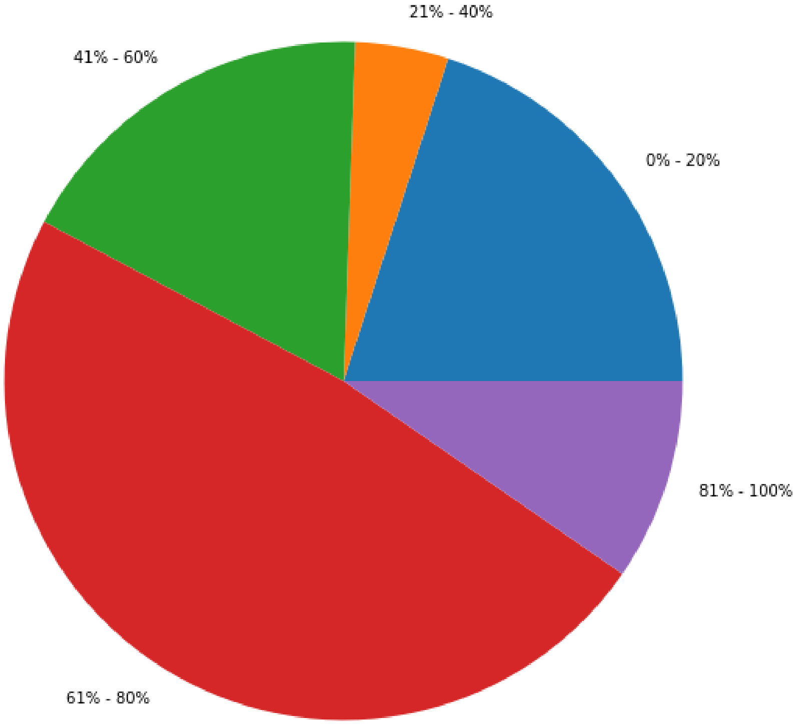
Data visualization:

a. Matching factor vs projects



Above graph shows project against matching factor

b. Matching factors range

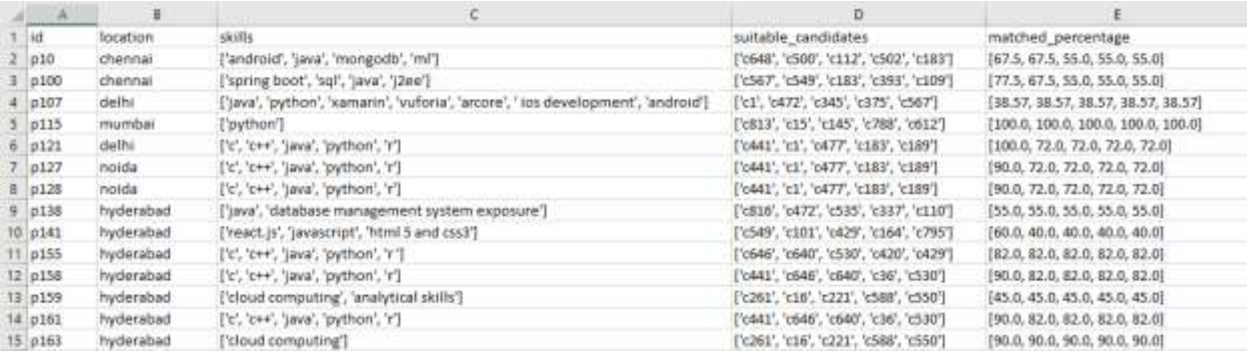


In above graph, orange green shows number of projects which has matching factor between 41 – 60% and so on.

Solution Format:

Attached *solution.csv* file contains following attributes

Attribute	Description
id	project ID.
location	Location of the project.
skill	Required skills for the project.
suitable_candidates	Top 5 suitable candidate for project.
matching_percentage	matching factor of suitable candidates respectively.



	A	B	C	D	E
1	id	location	skills	suitable_candidates	matched_percentage
2	p10	chennai	['android', 'java', 'mongodb', 'ml']	['c648', 'c500', 'c112', 'c502', 'c183']	[67.5, 67.5, 55.0, 55.0, 55.0]
3	p100	chennai	['spring boot', 'sql', 'java', 'j2ee']	['c567', 'c549', 'c183', 'c393', 'c109']	[77.5, 67.5, 55.0, 55.0, 55.0]
4	p107	delhi	['java', 'python', 'xamarin', 'vuforia', 'arcore', ' ios development', 'android']	['c1', 'c472', 'c345', 'c375', 'c567']	[38.57, 38.57, 38.57, 38.57, 38.57]
5	p115	mumbai	['python']	['c813', 'c15', 'c145', 'c788', 'c612']	[100.0, 100.0, 100.0, 100.0, 100.0]
6	p121	delhi	['c', 'c++', 'java', 'python', 'r']	['c441', 'c1', 'c477', 'c183', 'c189']	[100.0, 72.0, 72.0, 72.0, 72.0]
7	p127	noida	['c', 'c++', 'java', 'python', 'r']	['c441', 'c1', 'c477', 'c183', 'c189']	[90.0, 72.0, 72.0, 72.0, 72.0]
8	p128	noida	['c', 'c++', 'java', 'python', 'r']	['c441', 'c1', 'c477', 'c183', 'c189']	[90.0, 72.0, 72.0, 72.0, 72.0]
9	p138	hyderabad	['java', 'database management system exposure']	['c816', 'c472', 'c535', 'c337', 'c110']	[55.0, 55.0, 55.0, 55.0, 55.0]
10	p141	hyderabad	['react.js', 'javascript', 'html 5 and css3']	['c549', 'c101', 'c429', 'c164', 'c795']	[60.0, 40.0, 40.0, 40.0, 40.0]
11	p155	hyderabad	['c', 'c++', 'java', 'python', 'r ']	['c646', 'c640', 'c530', 'c420', 'c429']	[82.0, 82.0, 82.0, 82.0, 82.0]
12	p158	hyderabad	['c', 'c++', 'java', 'python', 'r']	['c441', 'c646', 'c640', 'c36', 'c530']	[90.0, 82.0, 82.0, 82.0, 82.0]
13	p159	hyderabad	['cloud computing', 'analytical skills']	['c261', 'c16', 'c221', 'c588', 'c550']	[45.0, 45.0, 45.0, 45.0, 45.0]
14	p161	hyderabad	['c', 'c++', 'java', 'python', 'r']	['c441', 'c646', 'c640', 'c36', 'c530']	[90.0, 82.0, 82.0, 82.0, 82.0]
15	p163	hyderabad	['cloud computing']	['c261', 'c16', 'c221', 'c588', 'c550']	[90.0, 90.0, 90.0, 90.0, 90.0]

Requirements:

Programming language: Python 3
Python library: Pandas, matplotlib, numpy
Tool used: Google Colab

Time complexity:

Time complexity to reach solution using above mentioned approach is:
T(n) = O(Log(n²))