Backend Developer Task - Find Duplicate Profiles

Given list of N number of profiles and M number of fields , compute all the possible duplicate profiles.

Below is the reference profile model

```
Class Profile(object):
  first_name = CharField(required=True)
  last_name = CharField(required=True)
  date_of_birth = DateField(required=False) # format will be Y
YYY-MM-DD
  class_year = PositiveInteger(required=False)
  email_field = EmailField(required=true)
```

How the algorithm should work

- Given two profiles, we will start with total score of 0
- if first_name + last_name + email match between two profiles is greater than 80% (you can try using a library like https://pypi.org/project/fuzzywuzzy/), increase the match score to 1
- if both the profiles have class year and they match, increase total score again to 1. If none of them have class year or even one of them does not have class year, skip the comparison and do not change the total score. If both of them have class year and they are different, reduce 1 from the total score.
- if both the profiles have birthdate and they match, increase total score to 1 again. If none of them have birth date or even one of them does not have birth date, skip the comparison and do not change the total score. If both of them have birth date and they are different, reduce 1 from the total score.
- After computing all the properties if the total score is greater than 1, both the profiles should be listed in the profile duplicate table with all the attributes matching results.

- Only consider fields which are passed as arguments to duplicate finder function.

Sample Input 1

```
profile 1 - { id: 1, email: 'knowkanhai@gmail.com', first_nam
e: ‘Kanhai’, last_name: ‘Shah’, class_year: 2012, date_of_birt
h: ’1990-10-11’, }
profile 2 - { id: 2, email: 'knowkanhai@gmail.com', first_nam
e: ‘Kanhai1’, last_name: ‘Shah’, class_year: 2012, date_of_bir
th: ’1990-10-11’}


find_duplicates(profiles=[profile1, profile2], fields=['emai
l',‘first_name’, ‘last_name’, ‘class_year’, ‘date_of_birth’])
```

Sample Output 1

Duplicate Profiles:

```
profile 1, profile 2, total match score : 3, matching_attribut
es: first_name, last_name, class_year, date_of_birth, non_matc
hing_attributes: None, ignored_attributes: None
```

Sample Input 2

```
profile 1 - { id: 1, email: 'knowkanhai@gmail.com', first_nam
e: ‘Kanhai’, last_name: ‘Shah’, class_year: None, date_of_birt
h: None}
profile 2 - { id: 2, email: 'knowkanhai@gmail.com', first_nam
e: ‘Kanhai1’, last_name: ‘Shah’, class_year: 2012, date_of_bir
th: ’1990-10-11’}


find_duplicates(profiles=[profile1, profile2], fields=['email'
, ‘first_name’, ‘last_name’, ‘class_year’, ‘date_of_birth’])
```

Sample Output 2

Duplicate Profiles:

```
profile 1, profile 2, total match score : 1, matching_attributes: first_name, last_name, non_matching_attributes: None, ignored_attributes: class_year, date_of_birth
```

Sample Input 3

```
profile 1 - { id: 1. email: 'knowkanhai@gmail.com', first_name: ‘Kanhai’, last_name: ‘Shah’, class_year: None, date_of_birth: None}
profile 2 - { id: 2, email: 'knowkanhai+donotcompare@gmail.com', first_name: ‘Kanhai1’, last_name: ‘Shah’, class_year: 2012, date_of_birth: ’1990-10-11’}


find_duplicates(profiles=[profile1, profile2], fields=[‘first_name’, ‘last_name’])
```

Sample Output 3

Duplicate Profiles:

```
profile 1, profile 2, total match score : 1, matching_attributes: first_name, last_name, non_matching_attributes: None, ignored_attributes: email, class_year, date_of_birth
```

What do we expect from you?

- Your solution should be written to optimise the time taken to flag all duplicates (Assume that this algorithm will be running on multi core machine).
- Make sure the solution you write scales for large number of profiles having large number of fields.

- Write clean, extensible code (duplicate finder algorithm should easily support more fields).
- Deploy your code to github and share it with us.
- Add proper documentation of how to run the code.