## INTRODUCTION

All code mentioned in this report could be found in my github repositiry, and  it licensed under MIT, so feel free to use it.

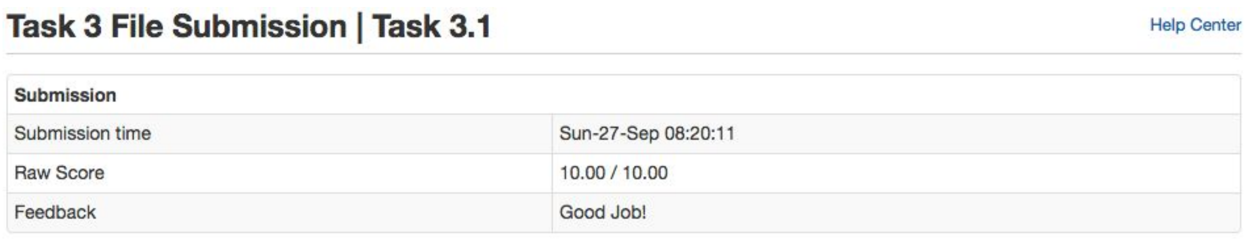
For both tasks I used Mediterranean cuisine, because i live in Spain, and familiar with topic.

## TASK 3.1

This task was pretty straight forward. Simply scrolling through the file and switching flags on false positive, and false negatives.

There were 24 true positive out of 120+ positive candidates, and 5 false negative out of 250+ negative candidates.

Which give us, around 30 dishes, or 8% of match, if count in percents.



**Task 3 File Submission | Task 3.1**

| Submission | |
|---|---|
| Submission time | Sun-27-Sep 08:20:11 |
| Raw Score | 10.00 / 10.00 |
| Feedback | Good Job! |

## TASK 3.2

As i was unimpressed with results from SegPhrase, which were given to us to work with in the task 3.1. I decided to write my own algorithm, just to try how it will work.

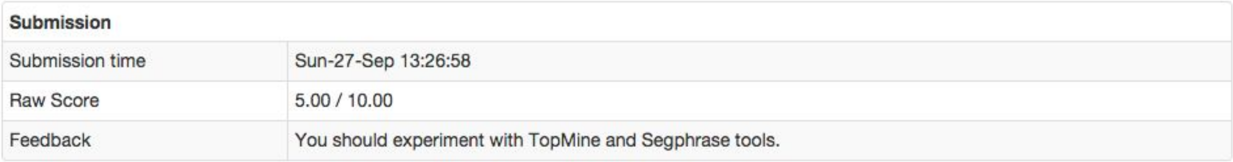
So, I took reviews of Mediterranean cuisine, generated using scripts i wrote for previous weeks tasks.

My algorithm is based on mining word association and word pattern mining. After browsing reviews, I found that people tend to use common text patterns when they mention dishes in their reviews.

The most common patterns are:

- I ordered *“dish name”*.
- *“dish name”* was *“adjective”*
- we ordered *“dish name”*, *“other dish”* and *“one more dish”*

My initial approach was simply try to mine dishes from the first pattern. Result wasn’t bad, I got 5 out 10 points.

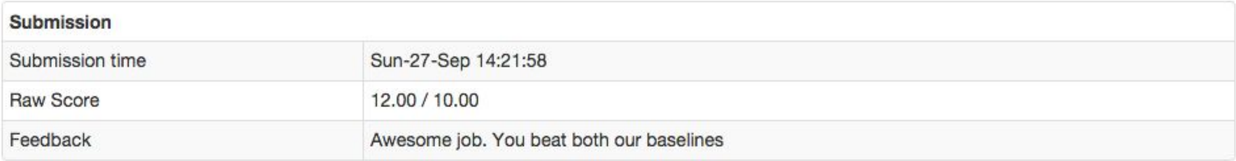


| **Submission** | |
|---|---|
| Submission time | Sun-27-Sep 13:26:58 |
| Raw Score | 5.00 / 10.00 |
| Feedback | You should experiment with TopMine and Segphrase tools. |

The problem was mainly in the fact that i was getting around 50% of false positive results from my initial run of the script (which you can find in the github repo mention earlier). The problem was that i used “greedy” approach when mining actual dish name from the pattern. So i tweaked my algorithm, added few stop words, to limit dish names. For example, there were many patterns like:

- *dish name*, which
- *dish name* that
- *dish name* with

these patterns were occurring with previously described patterns. So adding checks for cleaning data using these stopwords, i was able to make my algorithm less “greedy”. This worked like a charm. I got 12 out 10:



| **Submission** | |
|---|---|
| Submission time | Sun-27-Sep 14:21:58 |
| Raw Score | 12.00 / 10.00 |
| Feedback | Awesome job. You beat both our baselines |

## CONCLUSION

My algorithm may be not the most efficient one, and it haven’t produced few thousands of candidate dishes. But from the ~500 candidates around 80% were true positives. Which is pretty impressive for a few hundred lines code.

I think this shows that it’s important to first look for the text patterns, before starting to do text mining, as we can see from results, mining  from known patterns can become trivial task, and could give very solid results.

As an improvement of my approach i can see, using results of my code, as input data labels for the SegPhrase tool.