THE EFFECTS OF DIFFERENT COURSE FACTORS ON STUDENTS’ SENTIMENT

Samer AL Masri

UNIVERSITY OF ALBERTA Edmonton AB T6G 2E8

Introduction:

As we step in the technological age, the number of internet users is increasing across the world, which gives companies more data and hence more power over people. This data can be used for multiple reasons; governments use it to keep the countries safe, corporate companies sell it for ads, however some organizations use it as anonymous statistical data to make their service better. For example, when Ubuntu operating system (OS) collects bug reports, although these reports may contain some information about the user, their main usage is to understand and fix anomalies in the OS.

Other types of organizations that use data for improving are MOOCs; in fact, there is a study field specifically for that: learning analytics. Learning analytics (LA) is the use of big data in order to optimize the methods and ways to teach and deliver information to students. Although this field is very specific with respect to Computer Science, LA is still generic and contains various sub-fields. For example, one can improve students' performance by tracking their grades, performance, whether they completed the MOOC or not, demographic information. Even more personalized data can be collected, we can track students' eye movement, skin temperature, active area of brain…

Tracking one characteristic alone may not be very efficient, or it may not give very accurate results, hence many scientists decide to track multiple characteristics together and create hypothesis of how these characteristics are related. This is called multimodal LA. In other words, multimodal LA is collecting information from multiple sources about students and using it to come up with theories about how to optimize the learning efficiency and experience of students. One source of information that can be used when applying multimodal LA is student's feedback. When we look at different forms of collecting feedback, we see courses applying rating systems, asking students to explicitly give their comments about the course, or checking students' sentiment throughout the course.

The most accurate way is obviously to collect students' comments however this is a very laborious task in MOOCs since the number of students is huge.

Collecting feedback is generally a common way to collect feedback in MOOCs, however students may be biased when rating the course; the bias can come from previous reviews of the course, the student's performance in the course, or student's involvement in the course (active students would feel like they are making an impact on the course trajectory and hence would rate the course higher).

A student's feeling about the course is very important since it's one of the most accurate sources of feedback we can collect. Since people cannot easily control how they feel, this source of feedback is less impacted by external factors like previous reviews of the course. However, it is important to note that a student's sentiment is still not 100% accurate since it can be impacted by other areas of the student's life. For example, a student who is having personal problems will display less positive sentiment in the course even if the course is really good. Or a student who is having a series of successes in their life will show more positive sentiment even if the course was depressing.

My project for this course is to study how the students' sentiment is affected by different factors of MOOCs.

Goals and Hypothesis:

The project aim to provide the following contributions:

- Analyze the effect of different course factors on students' sentiment.
- Provide some manually tone analyzed text and compare it to IBM AlchemyLanguage Tone Analyzer's results.
- Publishing the code used to analyze the data on GitHub.

The following are the hypothesis that I was planning to verify/disprove throughout my project:

- Students display better sentiment when the number of students in the MOOC is smaller: a smaller number of students means it's easier for students to interact with each other. This hypothesis comes for the general theory that a smaller class means the teacher can give a more customized learning experience for the students. However, this doesn't necessarily hold true in MOOCs since a small class is still more than a hundred students. However, I still expect students' sentiment to increase since a smaller number of students means it's easier to track the forum and hence students will feel they know each other more.
- Students who get higher grades are happier with the course: it is intuitive that people who are successful at something will feel good about it.
- Students who feel better about a course don't necessarily give the course a high rating: this hypothesis comes from the discussion about the accuracy of students' feedback in the introduction (students may be biased to rate the course in a specific way or some external factor is affecting their sentiment)
- A course with lots of peer work increases students' sentiment: peer work means students interacting with each other, and hence students should feel more engaged and making an impact in the course and on other students. Hence, I expect the latter to increase their sentiment.
- Shorter course will have happier students: completing a course gives students a sense of achievement. Hence progressing through a course also give students this feeling since they are getting closer to completing what they started. Shorter courses progress more quickly and hence students should display a better sentiment when progressing rapidly through a short course compared to the relatively slow progress of the longer course.
- A post that displays higher sentiment will have higher rating: when someone is delivering information in a respectful and merry-full way, I expect viewers to like it and give it a thumb up more than posts that give information while disrespecting the target person for example.

Technical description:

Implementation mechanisms:

Although I had the same main goal in mind, however this project, like any other project, went through a series of different directions before finding the best way to achieve this goal. At first I wanted to analyze the sentiment behind posts, and use several clustering algorithms to divide the posts based on their topic, which would give how students felt about different components of the course. However, after facing difficulties with the clustering tools I decided to try another implementation. My second plan was to get the sentiment of each student via the posts he/she posted, and relate these sentiments to the students' performance and demographic data. However, I discovered later that the anonymized student IDs in the database provided by Coursera are done in such a way that it's impossible to link a student's post to their grade or demographic data. Finally, I decided to get the average sentiment of each course and relate it to the majority's demographic characteristics and performance. More about the difficulties will be discussed in later sections of this report.

<u>The Coursera Database:</u>

Coursera collects various information related to courses and stores them in a database. This section explains what parts of the database I used and how I used them. The database provided multiple categories of information:

Course Grades: The set of tables in Figure 1 show information stored about students' performance. Each user has the overall grade he/she scored in the course. In addition, each user has a grade for each item they completed in the course. Course items in the courses I used in this project include: exams, supplements, lectures, quiz, and peer work.

Demographic data: The set of tables shown in Figure 2 visualize how the demographic data are organized in the Coursera database. All studied demographic questions are multiple choice question except one (year born), which has a numeric answer. Hence, the main table links each student to which questions he/she answered, with the chosen choice. Demographic data studied are: country where a student is living, age, gender, education level, and whether the student is of Hispanic origin or not.

Forum posts: As shown in figure 3, the tables link students to the posts they posted in addition to linking students to the posts they voted for. I used the post content to do sentimental analysis and find the average sentiment of the course. In addition, I used the answer votes table to calculate the score of each post and linked the results to the sentiments.

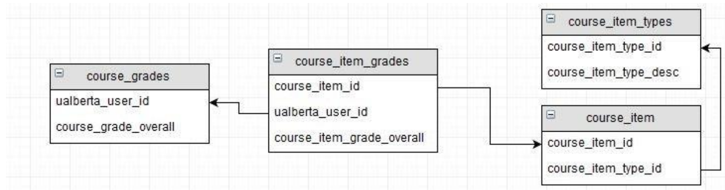

Figure 1: Coursera database tables related to students' grades

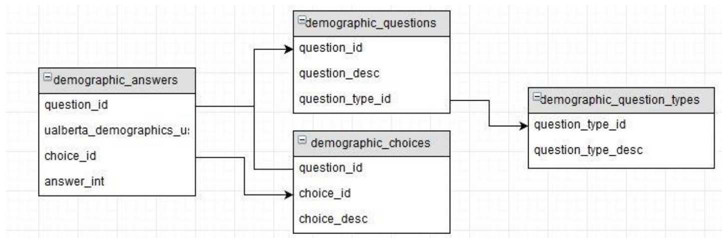

Figure 2: Coursera database tables related to demographic data

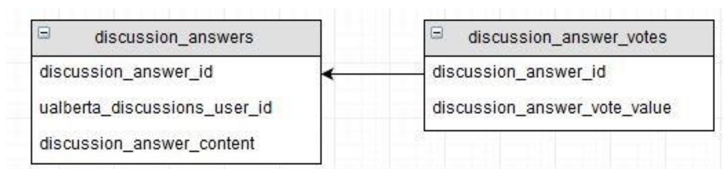

Figure 3: Coursera database tables related to forum posts

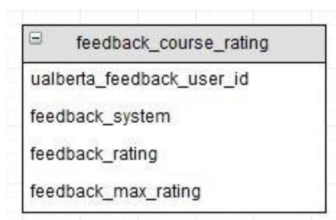

Figure 4: Coursera database table about students' feedback

Feedback: I also used the table shown in Figure 4 to inspect students' feedback and relate them to their sentiment. Note that Coursera is using 2 systems for rating, one called NPS and another called STAR. Although NPS is the newer rating system and is believed by Coursera to be more accurate, however most of the ratings recorded were in STAR system and hence I ignored the NPS system when forming my results for consistency.

Note that each figure has a different name for student IDs. This is because each category of studentIDs is anonymized differently. In other words, the same students are behind the demographic information and the posts in the database, however discussion_answer_ids are not the same as demographic_student_ids. This implementation hindered me from relating each student's sentiment to other information as mentioned in the beginning of this section.

IBM Watson AlchemyLanguage:

Watson is an IBM supercomputer created to process data with the help of artificial intelligence (AI) [1]. One service specified by Watson is sentimental analysis of text, the service is called AlchemyLanguage. Note that this service has been deprecated as of April 7, 2017 as shown on the service's website [2].

The way I used the API is as follows: using Java code, I send the text that I want to analyze and get back a JSON file containing different sentiment categories. Hence, I programmatically sent each forum post to the API, got back the JSON and then created a map to link every post ID to the JSON file name containing its sentiment.

The received JSON file contains the following [4]:

Sentence Tone: tone of each sentence in the text. Since the posts I'm sending are not very long (approximately 2 sentences on average), this component on the JSON file is ignored.

Document Tone: the tone of the entire document (which is in my case, the post). The following sentiment categories are provided:

- **Emotional Tone:** A value between zero and one that shows how much likely is it that the sentiment behind the post to include the component. The components are: Joy-Fear-Sadness-Disgust-Anger. Hence inside the emotional sentiment, we have 5 values, each of them is between zero and one. For example, getting a zero for joy means the sentiment behind the post isn't joyful at all. Whereas getting 1 for Disgust means that the sentiment behind the post certainly (100% chance) contains disgust.
- **Language tone:** This part analyzes the language in an attempt to understand characteristics about the author of that post. The value given to its subcomponents is also between zero and one.
  If the value is less than 0.5 it means the studied characteristic is not apparent, instead the opposite characteristic shows from the text. As the value decreases away from 0.5, it means the opposite characteristic is showing more clearly in the author.
  Whereas a value above 0.5 means the characteristic is apparent, and the higher the value gets the clearer this characteristic shows.
  Under the Language tone category, there are 3 subcomponents (and hence 3 sentimental values): Analytical-Confidence-Tentative.
  For example, when a post scores 1 on analytical it means the author shows clear analytical skills by writing this post whereas scoring 0 on confidence shows that the author showed 0 confidence in this post, which means the author showed was very doubtful when writing the post.

Emotion	Language Tone	Social Tone (big 5 personality traits)
Joy	Analytical	Openness
Fear	Confidence	Conscientiousness ◦ acting in an organized way
Sadness	Tentative	Extraversion ◦ being an extrovert
Disgust		Agreeableness ◦ cooperativeness
Anger		Emotional Range or Neuroticism ◦ emotional sensitivity

Figure 5: IBM Watson ToneAnalyzer Output

- **Social tone:** in this part, Watson ToneAnalyzer attempts to understand the author's personality by judging the author based on the big 5 personality traits (openness-conscientiousness-extraversion-agreeableness-emotional range or sensitivity). Similar to language tone, a value higher than 0.5 shows a probability that the corresponding trait is found whereas a value under 0.5 shows a probability that the corresponding trait is absent, which means the opposite trait is found.

Figure 5 shows a visual summary of all the sentiments analyze by IBM Watson ToneAnalyzer.

Computing the average sentiment:

Having all these detailed sentiments is really nice however it raises an important question: what is the best way to compute the average sentiment of a student? Since there is no one correct way to compute the sentiment, I decided to compute it in 3 ways:

1. **Total average:** I get the average of everything by adding positive sentiments and subtracting negative ones.
   First, I denote S(x) to be the sentiment of x.
   Second, I denote F(x) to be the fixed sentiment of x. For values in language tone and social tone categories, I get the difference between the recorded score and 0.5. Since values above 0.5 shows the corresponding tone whereas values above 0.5 shows the opposite of that characteristic. Hence, in symbols:
   S(x) = value in JSON file
   F(x) = S(x) – 0.5
   More specifically, I computed as follows:
   Average Sentiment = S(Joy) – S(Fear) – S(Sadness) – S(Disgust) – S(Anger) + F(Analytical) + F(Confidence) – F(Tentative) + F(Openness) +F(Conscientiousness) + F(Extraversion) + F(Agreeableness) – F(Emotional Range)
2. **Emotions average:** The same computation as above but only took into consideration the emotional sentiment:
   Average Sentiment = S(Joy) – S(Fear) – S(Sadness) – S(Disgust) – S(Anger)
3. **Language Tone:** The same as above but only took into consideration the language tone:
   Average Sentiment = F(Analytical) + F(Confidence) – F(Tentative)
4. **Social Tone:** The same as above but only took into consideration the big 5 personality traits:
   Average Sentiment = F(Openness)

+F(Conscientiousness) + F(Extraversion) + F(Agreeableness) – F(Emotional Range)

Java Program:

The source code consists of 3 packages:

The first package is org.json, I got that from a Github repository [3] belonging to the creators of json.org website.

The second package, called ToneParsing: the main job of this package is to convert the JSON file to a hierarchy of objects representing all the details in the JSON file. Hence, it defines structs that hold information as an analogue for the JSON objects (it's easier to work with Java structs, or classes without methods, instead of navigating through the JSON file every time I need one attribute). In addition, it contains a library class that contains static methods that create a *DocumentTone* object from the JSON files. The returned *DocumentTone* object contains the 3 tone categories, and in each *ToneCategory* object, the relevant information is found.
It is important to note that the average sentiment is calculated in 2 phases. First I calculate the average sentiment in each *ToneCategory*, which is a method inside the class *ToneCategory*. Then I calculate the average of the 3 categories in the third package.

The third package, default package, is where most of the work is done. The package consists of some structs to organize information (with minimal number of functions for type-specific purposes, like a customized constructor or a *toString()* function to print a better representation of the object when called by *System.out.println()*). In addition, the package has 2 main classes:

- *ForumPostsSentimentalAnalysis*: This class collects the posts from the database csv file, send each of them to the IBM ToneAnalyzer, and saves the JSON file with the filename = ID of post. After that it creates a map file, that links each post ID to its average sentiment.
- *Database*: This class contains all the methods that deal with taking information from the database and presenting them as results. More specifically, the main function starts by calling *fillMaps()*, which basically stores all the information we may use when creating results in a series of maps and arrays of objects. After that, I alter the main function by calling different functions to get different results. Finally, I enter these results into Microsoft PowerPoint to get a visualization. I will end this section by giving a brief description of methods in the *Database* class:

specifyToneCalculationMethod(): The *ForumPostsSentimentalAnalysis* class main function will produce 4 maps; each one mapping the post IDs with a specific calculation of the average. For example, *FileToneMap.all* contains the total average. *FileToneMap.emotions* contains only the sentiment emotion average… Hence, the *specifyToneCalculationMethod()* specifies which *FileToneMap* to use.

genderStatistics(): gets the percentage of males and females in the cours.

printAverageSentiment(): Prints the average sentiment of the course.

getAverageGrade(): Gets the average grade of the course.

getTopCountries(): prints the top N countries that students are from (N is provided as a parameter).

getHispanicePercentage(): gets the percentage of students in the course who are of Hispanic origin.

printTopEducationLevel(): prints, in decreasing order, the majority of students' education level.

getAverageAge(): get the average age of students in the course.

postScoreSentimentRelation(): categorizes the posts by the amount of likes each of them got (which is computed from the *discussion_answer_votes* table). Then, the average sentiment of each category is computed.

feedbackSentimentRelation(): prints the average rating and average sentiment of the course for comparison.

As shown here, most of the functions are self-explanatory. In addition, the code contains more comments to make it more readable.

Discussion and Interpretation of Results:

I started by using the data to prove/disprove the provided hypothesis. However, while going through the data I realized additional patterns that I will also discuss here.

The Studied Courses:

First, I give some general information about the courses and the student population.

The 6 courses analyzed were a series of a Coursera specialization (Software Product Management Specialization) where successfully completing the 6 courses would give students a certificate. The 6 courses are named as follows and I will refer to them in the order I list them below:

1. Introduction to Software Product Management
2. Software Processes and Agile Practices
3. Client Needs and Software Requirements
4. Agile Planning for Software Products
5. Reviews & Metrics for Software Improvements
6. Software Product Management Capstone

It is noteworthy that courses 1 and 6 are 3 and 7 weeks respectively whereas the other 4 courses are 5 weeks each. Also note that Course 3, 4, and 6 contained peer work were students had to work with each other for an assignment whereas the other 3 courses didn't contain that kind of work.

The Student Population:

	Introduction to Software Product Management	Software Processes and Agile Practices	Client Needs and Software Requirements	Agile Planning for Software Products	Reviews and Metrics for Software Improvements	Software Product Management Capstone
Country #1	United States	United States	United States	United States	United States	United States
Country #2	India	India	India	India	India	United Kingdom
Country #3	Ukraine	Ukraine	Ukraine	Ukraine	Ukraine	Germany
Country #4	Russian Federation	Russian Federation	Russian Federation	Russian Federation	Russian Federation	Russian Federation
Country #5	Brazil	Brazil	Brazil	Germany	Germany	Netherlands
Country #6	United Kingdom	Spain	United Kingdom	Spain	Brazil	Bulgaria

Figure 6: Top 6 countries where students are from in each course

As shown in Figure 6, most courses had the majority of students from US and India. Which shows that most of the students are proficient in English language; US students have English as their first language and India ranks as the 4th country in Asia for English proficiency according to Education First [5].

Other demographic data are shown in Figure 7. It appears that age is almost strictly increasing throughout courses. Assuming that most of the students in a course, are a subset of the students in the previous course (for example all students in course 2, already attended course 1; even if this is not entirely

Courses	Hispanic Origin %	Education Level	Gender	Age
Course 1	13.33%	MSc – BS	75.09% males	35.0178
Course 2	16.06%	MSc – BS	75.25% males	35.24
Course 3	15.64%	MSc – BS	72.99% males	35.405
Course 4	14.51%	MSc – BS	73.11% males	35.16
Course 5	14.48%	MSc – BS	72.78% males	35.74
Course 6	20.37%	MSc – BS	75.92% males	36.88

Figure 7: Various demographic data of students according to courses

true, it is the majority of cases since the courses are bundled in a Coursera specialization), the increase of age shows that slightly more young people are dropping than older people. This shows some correlation between age and being determined and consistent (older people tend to be more consistent with what they do).

We also notice that the capstone course had the highest percentage of males and Hispanic people.

It is also worthy to note that the dominating level of education in all the courses was the Master's degree, followed by Bachelor's degree. Hence, most students who took the series of courses already had their Master's degree (but not PhD). A primary interpretation would be that students are getting their Master's degree, going to work in the field, and aiming to get the Coursera certificate in order to get promoted in their job.

Manual Sentiment Analysis:

I manually judged 30 posts from each course by giving each post a rating between -1 and 1 representing their sentiment. Below are some notes about how I judged the posts, and other interesting realizations:

- Many posts are just asking other students to grade their work (peer work), I put 0.5 for these posts since authors usually use please and thank you.
- I put 0 for posts with questions only since there is no emotion in such posts.
- I put 0.5 for posts explaining a concept since they didn't show emotion, but it is an initiative for students to explain stuff for their peers which is associated with positive emotion.
- I put negative score when there's a disagreement in opinions, especially when it's not in a diplomatic way (for example if a student just said I disagree instead of acknowledging the other person's opinion).
- I put -1 when a word is in CAPS (shouting voice) or a student showed aggressiveness against others.
- Using text emoticons (like :D) or exclamation marks that shows positive sentiment affected the posts score in my judgement (IBM machine may not have done that).
- I sometimes prioritized emotions over way of speech and big5 personality features when deciding on the average sentiment (not linear as in the average results I computed from the JSON files, which may be more accurate).
- English language wasn't a problem for the students discussing in the

forums; most posts had good English.
  - Some students were taking advantage of the forums to meet new people professionally and add them on LinkedIn, which supports the idea that many people are taking these courses to for professional career related reasons. They also went as far as creating a LinkedIn group for themselves (students who are taking the course).
  - I was aiming to give the sentiment of the person's feeling about the course, not absolute feeling when writing the post. For example, if a student is asking about trouble in the course, even if they said please and thanks the score will still be negative since the student wasn't feeling good when asking the question.
  - Some students even decided to talk in their own language in the forum, I put a bad score for these posts since they make other students feel not welcome to join the chat.
- In course "Review and Metrics for Software Improvement" (Course 5), there was an issue that students lately discovered that they must pay for the course, which resulted the sharp drop in sentiment average in that course shown in Figure 8.

Difference Between Manual and IBM Sentimental Analysis:

When comparing the manual sentimental in Figure 9, we realize that the results are not synchronized. First, the IBM Watson results have negative averages, whereas my results have positive averages. In addition, the slopes in my results are different than the ones in the Watson results (courses where sentiment increases in my results is sometimes decreasing the Watson's results). The difference in results could be associated with multiple interpretations:

1. I analyzed a small number of posts per course (since I wanted to analyze the same number of posts for all courses).
2. The way I computed the average sentiment is controversial; although the 3 ways I calculated the sentiments are intuitive, I believe more accurate results may be achieved with more educated guesses (by specialists in the psychology and sentimental analysis field).
3. The IBM Watson machine may not be very accurate; as shown by the preceding notes, when judging the results, I took multiple things into consideration that only makes sense for a human intelligence.

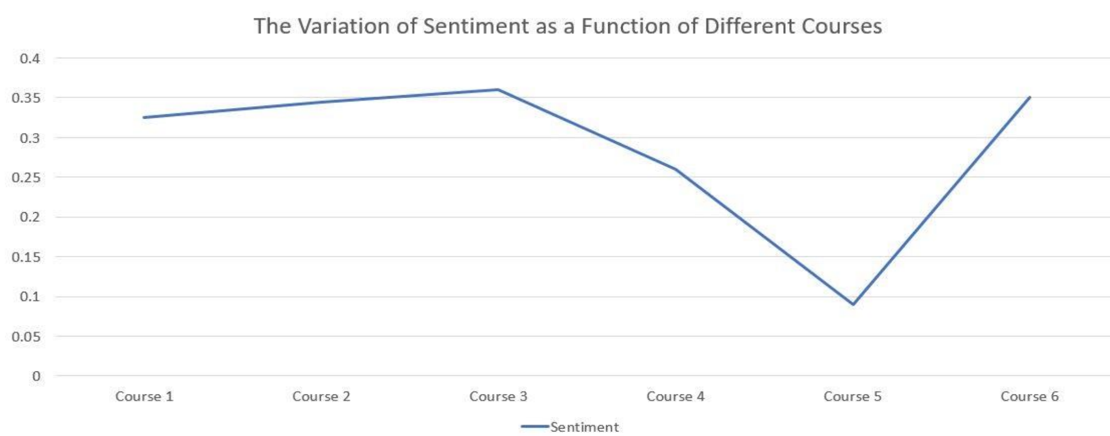

Figure 8: The results of manual sentimental analysis of posts across different courses

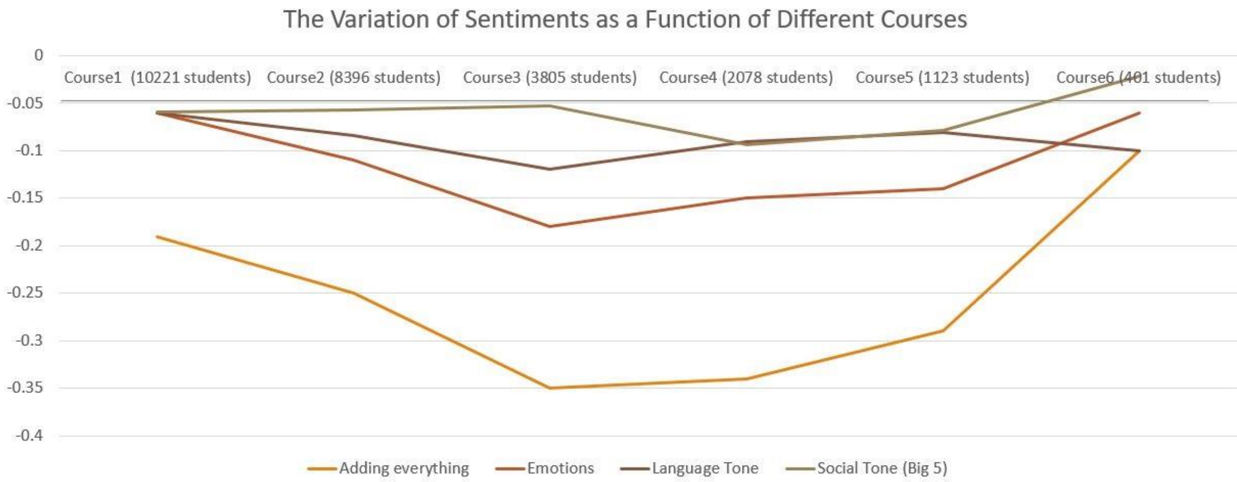

Figure 9: The IBM ToneAnalyzer sentimental analysis results across different courses

Interpretation of IBM Watson ToneAnalyzer Results:

Since the courses' student population is strictly decreasing thoroughout the courses, Figure 9 also shows the variation of sentiment as a function of different student population size. We can see that the social tone is increasing (almost always) as the number of students decrease. This can be correlated to the fact that when the number of students decrease, communication becomes easier (as mentioned before) which makes students more open-minded to other students' (which is represented by the social tone).

Hence, it is apparent from the above results that first hypothesis (in the Goals and Hypothesis section) is proved by some of sentiment categories but debunked by others.

Another important note is that lanugage tone is the lowest in courses 3, 4, and 6. Since these are the only courses that have peer work, we can deduce that peer work negatively effects language tone. As mentioned by professor Kenny Wong, who instructed the course series on Coursera, that there may have been a problem with phrasing the peer work instructions which caused confusion, and hence dropped the language tone in these courses.

Hence, this result also partly prove the hypothesis about peer work increasing students' sentiment.

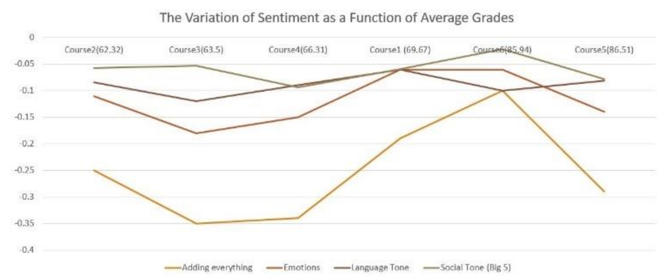

Figure 10: The variation of sentiment as a function of average grades

Figure 10 shows that there is no correlation between students' grades and their sentiment, which is backed up by the manual sentiment results in Figure 8. Although this is anti-intuitive, there are many other factors affecting the students' sentiments and grades aren't a deal breaker apparently. In other words, Figure 10 proves that students can get low grades and still enjoy the course, or get high grades and still not like the course.

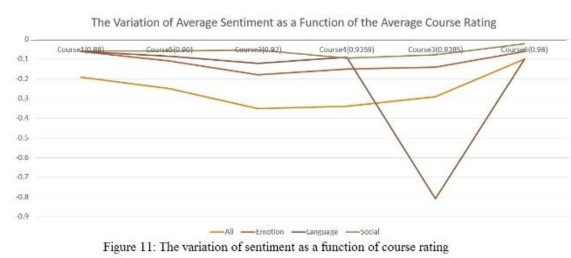

Figure 11: The variation of sentiment as a function of course rating

In Figure 11, my third hypothesis is proved; students' ratings are not directly affected by their sentiment. This may be due to the students' opinion bias towards the course or the students' bias in sentiment, as pointed out earlier.

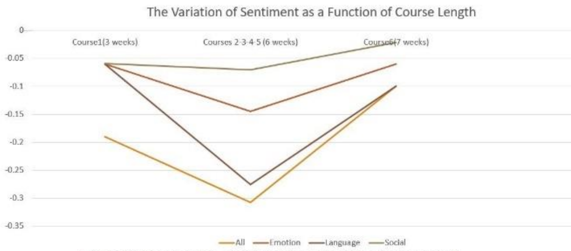

Figure 12: The variation of sentiment as a funcion of course length

If we ignore course 6, Figure 12 proves the hypothesis that shorter courses' students display better sentiment. I am ignoring course 6 since it is the capstone course, and it's expected for students to display high sentiment in a course where the biggest deliverable is the capstone project. This is directly related to the sense of achievement humans get from doing a relatively big project in a short period of time.

Last but not least, I categorized posts by the score they got from students (how many thumbs ups or likes) and computed the average sentiment for each category in each post. The results are displayed in Figure 13.

This figure shows that the post rating is not directly related to the sentiment behind the post. After reading some manual posts, I realized that many posts say please and thanks for example; these posts although they have high sentiment score, but they are not expected to get higher number of likes. The opposite can also be true: some posts may be complaining about some specific part of the course (like when in course 4 students realized they have to pay later during the course), but everyone will put a thumbs up on such posts since they reflect what others feel (even if the sentiment is negative).

Figure 13: The variation of sentiment as a function of different post ratings across different courses

Challenges and Lessons Learned:

As I was going through this project, I passed through several challenges and learned multiple values lessons.

First, I wasn't able to practically cluster the forms. When trying to use a program called Carrot2 to cluster the posts based on their content, Carrot2 algorithms used a word as a key to cluster the posts. For example, I would give the program a word "assignment" and the program would cluster posts related to assignments in a category, and other posts are also clustered into different categories. Hence, the only way I had in mind was to repeat the clustering with different manually defined keywords but this way would result in some posts in multiple categories and other posts in no categories. Another challenge with Carrot2 is inputting the posts for the program. Since, the tool is designed to search for the keywords from a web search engine like bing, I had to convert the posts to a specific XML convention so that Carrot2 would use these posts as an "offline search engine." Fortunately, I found a tool on GitHub [6] that converted text files to XML files that can be inputted to Carrot2. However, I still had the first problem and hence I decided not to cluster the posts.

The second main challenge that impacted my project was the fact that userIDs are different across different information, for example demographic information has student IDs different than student IDs for forum posts information. Linking this information together would give information about every student, what were their grades, demographic information, course rating, forum answers, and sentiment which would give a way more accurate correlations how sentiment affect as is affected by different course factors. I disagree with Coursera about anonymizing the information that way. Although I am a strong proponent of privacy and hence I understand why Coursera would anonymize the student IDs in the first place, however using a different ID for each categorie of information is hindering data scientists and it is barely giving students any extra privacy protection. Linking these information together would help produce more literature in the Learning Analytics field which would help Coursera play a leading role in renovating the methods of education around the world.

Other challenges include some discrepencies in Coursera's provided databases. For example I found some discussion post IDs in the discussion_answer_vote table which I didn't find in the main discussion_answers table. Hence, some posts I could find how many likes they got from other students but I couldn't find this post's content to evaluate the sentiment behind it. Another issue with coursera's material was the provided guide. The guide is provided with the database for each course explaining some background informationa about it, however the guide missed explaining the different types of peer work. In the course item types, there was: graded peer, phased peer, and closed peer. The difference between the 3 was not clear.

By implementing a Java program to process the information I learned a lot about dealing with databases. At first I used HashMaps to store the information that I found in the database. However after that I found out that creating structures with multiple fields and storing an array of structures is more abstract and organized. I also had some difficuties with reading the CSV files with Java (the Scanner object was stop reading the file before its done, which I solved by using a BufferedReader instead of a Scanner). Hence, I would advise new experimentors to use Python which has built in libraries that read and write to CSV files.

Related Work:

Natural Language Processing (NLP): Using NLP, paper [7] suggests an algorithm that divides a paper into clusters of words, and each cluster can be positive or negative.

Twitter opinion mining: In [8], the authors process tweets and other microblogging posts on Twitter looking forward to find the general opinion of the users (which are the public) and the sentiments behind it.

Pre-processing sentimental analysis: An underestimated factor in sentimental analysis is pre-processing the results. Pre-processing affects the accuracy of sentimental analysis greatly. For instance, processing the forum posts data in MOOCs before removing the introductions at the beginning of the course will make the results look more positive (due to the artificial positivity showed when meeting someone new). Pre-processing the data is explored in [9].

Another work that leverages pre-processing the results is done by Pang et al. [10]. They aim to process a series of posts that are found in a discussion forum where people are reviewing a movie or a product, and create a summary of all posts where authors displayed positive sentiment towards that product and posts that displayed a negative one. This is done by first labeling the posts as subjective or objective, and then running a machine-learning algorithm on subjective posts. By removing objective posts, the artificial intelligence (AI) algorithm will become more accurate. The proposed methodology is backed up by experiments that showed up to 4% of improvement in accuracy.

References:

[1] https://www.ibm.com/watson/

[2] https://www.ibm.com/watson/developercloud/alchemy-language.html

[3] https://github.com/stleary/JSON-java

[4] https://www.ibm.com/watson/developercloud/doc/tone-analyzer/understand-tone.html

[5] http://www.ef.com/ca/epi/regions/asia/india/

[6] https://github.com/carrot2/folder2index

[7] Tetsuya Nasukawa, Jeonghee Yi, "Sentiment analysis: capturing favorability using natural language processing," K-CAP, 2nd conference, October 2003, p. 70-77

[8] Alexander Pak, Patrick Paroubek, "Twitter as a Corpus for Sentiment Analysis and Opinion Mining," Proceedings of the International Conference on Language Resources and Evaluation, May 2010, p. 17-23

[9] Emma Haddi, Xiaohui Liu, Yong Shi, "The Role of Text Pre-processing in Sentiment Analysis," First International Conference on Information Technology and Quantitative Management, Volume 17, 2013, p. 26-32

[10] Bo Pang, Lillian Lee, "A Sentimental Education: Sentiment Analysis Using Subjectivity Summarization Based on Minimum Cuts," Proceeding ACL '04 Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Article No. 271, July 2004