Final Project Report

ENSF 612 - Big Data

By: Yajur Vashisht, Satchytan Karalasingham, Momin Muhammad, Balkarn Gill

"Title page image created by ChatGPT's DALL-E, generated in November 2023. This AI-generated image, depicting a digital brain representing big data, was designed specifically for use as a title page in an educational context for a big data class."

Table of Contents

List of Figures

List of Tables

Introduction and Motivation

The ongoing climate crisis is the largest problem facing modern-day humanity. Due to the vast amounts of data available online, monitoring misinformation is imperative. Harmful misinformation can be identified, and good-faith activism can be empowered using media intelligence through the means of Machine Learning and Artificial Intelligence.

Data Collection

The data used in our ENSF 612 Final Project is taken from reddit via web scraping methods. There are two csv files within the Kaggle Dataset:

1. The-reddit-climate-change-dataset-comments.csv
    - Contains the comments from various users regarding climate change (a wide range from negative to positive sentiments and scores).
2. The-reddit-climate-change-dataset-posts.csv
    - Contains the actual posts from various users on Reddit regarding climate change. The data in the comments csv file contains the comments from these posts.

For our project we decided to focus on the comments file because most discussions on a topic happen in the comments section, the posts themselves spur on the conversation.

Data Inspection and Validation

In the the-reddit-climate-change-dataset-comments.csv file there were various columns, these included:

- Type
    - Type of data (in our case, comments)
- Id
    - Unique identifier of the comment itself
- Subreddit.id

    - Unique identifier of the subreddit the comment was posted in
- Subreddit.name
    - Name of the subreddit the comment was posted in
- Subreddit.nsfw
    - Whether the subreddit is NSFW(swearing, inappropriate imagery, etc.)
- Created_utc
    - Date and time the comment was posted
- Permalink
    - Permanent link to the comment
- Body
    - The actual text of the comment
- Sentiment
    - Using NLP what the overall sentiment of the comment was
- Score
    - The sum of the downvotes vs. upvotes of other reddit users

type	id	subreddit.id	subreddit.name	subreddit.nsfw	created_utc	permalink	body	sentiment	score
comment	imlddn9	2qh3l	news	false	1661990368	https://old.reddi...	Yeah but what the...	0.5719	2
comment	imldbeh	2qn7b	ohio	false	1661990340	https://old.reddi...	Any comparison of...	-0.9877	2
comment	imldado	2qhma	newzealand	false	1661990327	https://old.reddi...	I'm honestly wait...	-0.1143	1
comment	imld6cb	2qi09	sacramento	false	1661990278	https://old.reddi...	Not just Sacramen...	0.0	4
comment	imld0kj	2qh1i	askreddit	false	1661990206	https://old.reddi...	I think climate c...	0.6634	1

only showing top 5 rows

Figure 1. First 5 rows of the comments csv file

Data Filtering

The steps to ensure our data was ready for Machine Learning included:

1. Load Apache Spark and SparkNLP into Google Collab
2. Load dataset as spark dataframe
3. Check for duplicated rows
4. Drop null values in Score and Sentiment columns
    a. If null values are present in our data, machine learning cannot take place
5. The following columns were selected for our machine learning models:
    a. sentiment
    b. score
    c. created_utc
    d. body (text)
6. The other columns were subsequently dropped

Data Transformations

We transformed the dataframe to include the following columns:

- created_datetime
- date
- word_count
- time_only
- hour
- mean_value (mean embedding value after NLP)

Steps were:

1. From pyspark.sql.functions import from_unixtime, date_format, split, size, hour
2. Apply the from_unixtime function onto the 'created_utc' column to create created_datetime.
3. Apply the size and split functions onto the 'body' column to create word_count.
4. Apply the date_format function onto the 'created_datetime' column to create time_only.
5. Apply the hour function onto the 'time_only' column to create hour.

6. For creating the mean_value column, the following steps were completed:
    a. Used a pipeline with imported models from sparknlp (DocumentAssembler(), Tokenizer(), StopWordsCleaner().pretrained(), WordEmbeddingsModel.pretrained(), SentenceEmbeddings(), EmbeddingsFinisher()) to fit and transform 'text' (renamed 'body' column) into finished_embeddings.
    b. Explode finished_embeddings using spark explode function.
    c. Apply a user defined calculate_mean column to create the mean_value column.
7. Using df.show(5) to see if the function was applied correctly.

created_utc	text	sentiment	score	created_datetime	date	word_count	time_only	hour	finished_embeddings
1661990368	Yeah but what the...	0.5719	2	2022-08-31 23:59:28	2022-08-31	53	23:59:28	23	[[-0.09582342, 0....
1661990340	Any comparison of...	-0.9877	2	2022-08-31 23:59:00	2022-08-31	242	23:59:00	23	[[-0.20790112, 0....
1661990327	I'm honestly wait...	-0.1143	1	2022-08-31 23:58:47	2022-08-31	68	23:58:47	23	[[-0.11997872, 0....
1661990278	Not just Sacramen...	0.0	4	2022-08-31 23:57:58	2022-08-31	18	23:57:58	23	[[-0.12475941, 0....
1661990206	I think climate c...	0.6634	1	2022-08-31 23:56:46	2022-08-31	78	23:56:46	23	[[-0.20102279, 0....

Figure 2. Comments csv dataframe with created_datetime, date, word_count, time_only, hour and finished_embeddings columns.

score	exploded_values	mean_value
2	[-0.09582342, 0.1...	-0.015499057
2	[-0.20790112, 0.3...	-0.010763373
1	[-0.11997872, 0.2...	-0.03411759
4	[-0.12475941, 0.1...	-0.048009343
1	[-0.20102279, 0.1...	-0.015153618

Figure 3. Comments csv dataframe with exploded_values and mean_value columns.

<u>Exploratory Data Analysis</u>

For the exploratory data analysis, we used pyspark's mean function to calculate mean sentiment and mean score. We also used matplotlib and pandas to graph date created vs. average score (Figure 4) and date created vs. average sentiment (Figure

5). These graphs show that there might be a slight correlation between time and average score and no correlation between time and average sentiment.

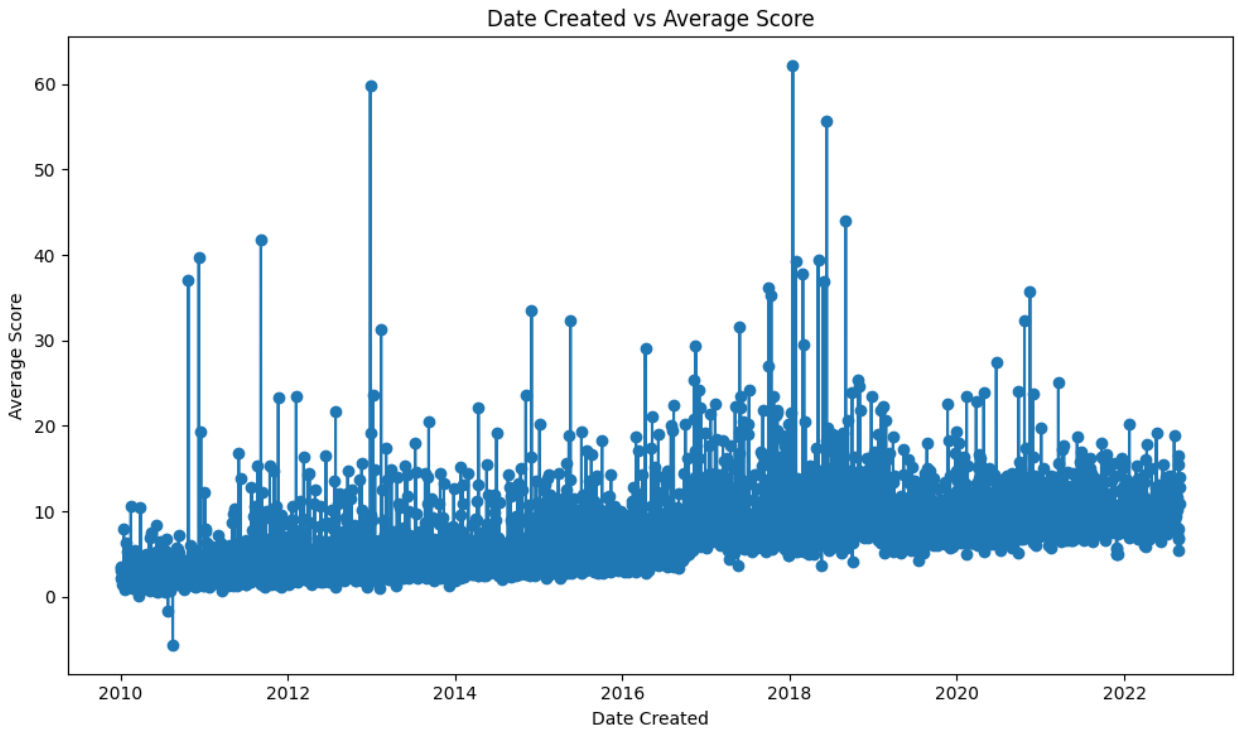

Figure 4. Graph of Date Created vs. Average Score

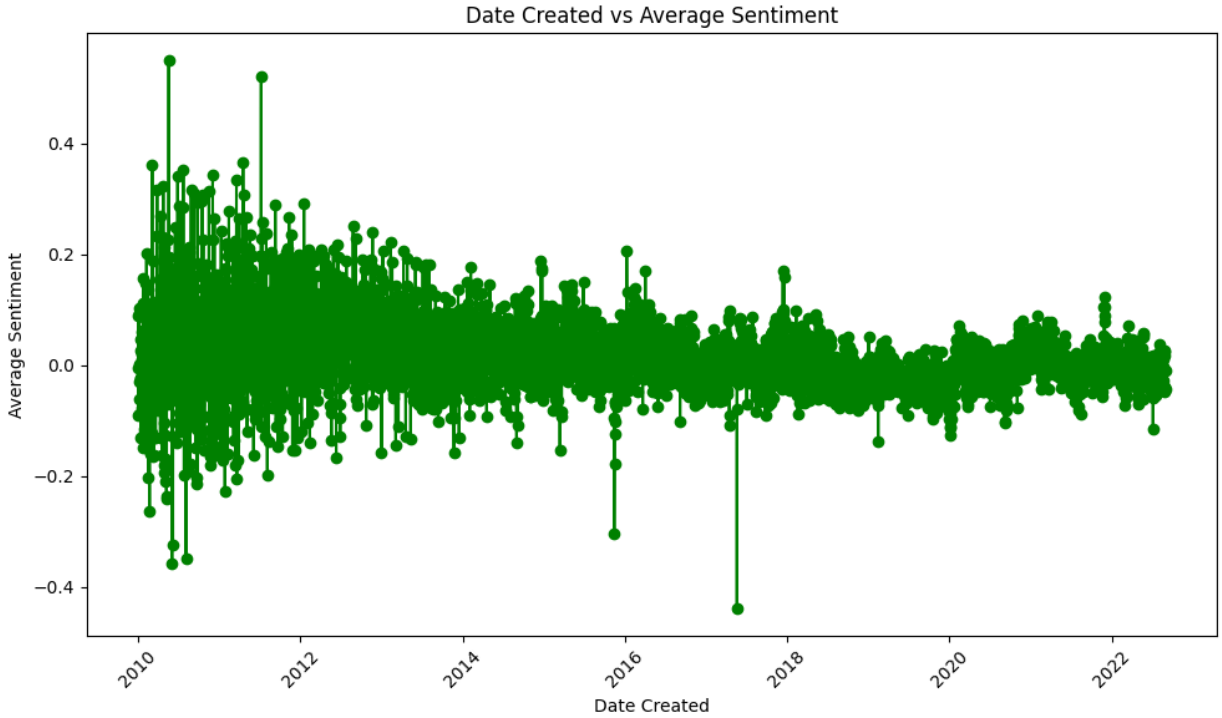

Figure 5. Graph of Date Created vs. Average Sentiment

Table 1. Mean Score and Average Sentiment of Reddit Comments

Mean Score	Average Sentiment
9.594	-0.00583

Model Building and Results

We built multiple models using a variety of features to try and predict score. We built linear regression, random forest regression, and gradient boosted regression models using the following feature sets:

1. sentiment
2. created_utc
3. Word_count

We built only linear regression and random forest models using the following feature sets:

1. hour
2. mean_value

The results of all models indicate there is no relationship with any of the features and score as shown by the r2 values close to zero or below.

Table 2. r2 Score and RMSE of Models

	Linear Regression Model		Random Forest Regression Model		Gradient Boosted Tree Regression Model	
Feature set	r2	RMSE	r2	RMSE	r2	RMSE
sentiment	-0.036	18.24	-0.458	21.64	-0.443	21.53
created_utc	-0.002	51.89	-4.011	51.85	-0.001	51.89
word_count	-0.021	18.11	-0.318	20.578	-0.325	20.63
hour	-0.035	18.23	-0.084	18.657	N/A	N/A
mean_value	-0.076	19.46	-0.212	20.654	N/A	N/A

Note: r2 and RMSE values are from fitting models on the first 1000 rows to reduce time it takes to run the full ipynb file .

Models were tested on 20000 rows to all rows and results tended to still be similar (r2 close to 0 and large RMSE).

In conclusion, it can clearly be seen that to create an accurate model, further features need to be extracted from the dataset as the current features are not correlated to score. This could be done by further NLP (use embedding values in different ways besides just averaging) or by using encoding techniques on dropped columns such as subreddit.name.