RUTGERS
UNIVERSITY

Dual Subreddit Analysis:

A comparative study of Rutgers and UPenn online communities

Contributing Authors:
Chris DSilva

May 05, 2024

Course Name
17:610:564:01 Machine Learning for Data Science

Table Of Contents

Mission Statement

Introduction to Dual Subreddit Analysis

The Role of Social Media in University Communities

In the digital age, social media platforms have evolved into crucial hubs of interaction, where ideas and experiences are exchanged across global and local communities alike. Among these platforms, Reddit stands out due to its unique structure of subreddits—specialized forums that cater to specific interests or groups, including academic communities. Each subreddit serves as a microcosm of its community, offering insights into the concerns, culture, and character of its members. Reddit is also known as the site where the reviews and content can be trusted as a source of actual content that is put out by people, this being also used heavily to train models on this data.

Universities, in particular, have seen their student bodies and faculty increasingly engage on Reddit, where they discuss everything from complex academic inquiries and campus policies to everyday life and personal experiences. These discussions not only enrich the students' social lives but also provide a platform for support, advice, and shared learning, extending the educational environment from the physical realm of campus into the virtual. This active participation has been seen also during the Rutgers Union Strike in 2023 and now with the students protesting for universities to divest from Israel during the Israel Palestine war.

Focus on Rutgers and UPenn Subreddits

This study zeroes in on two vibrant university subreddits: r/Rutgers and r/UPenn. Initially setting my eyes on r/Princeton , given its geographic proximity to Rutgers in New Jersey, and my hypothesis that the two might share similar characteristics, such as transportation issues highlighted by frequent student discussions about buses. My personal familiarity with Princeton, having observed its campus life firsthand—marked by a constant stream of students, buses, and events—initially led me to believe it would be a suitable comparison.

However, the choice shifted towards UPenn due to practical constraints: Princeton's subreddit activity was insufficiently archived by Pushshift, making a robust analysis challenging, and UPenn seemed like the next best option. These forums are not merely digital gathering spaces but are reflective of the broader university communities of Rutgers University and the University of Pennsylvania. Both subreddits are bustling with activity that mirrors the dynamic student life and academic environment of their respective institutions. However, each carries its own distinct digital footprint influenced by the unique culture, traditions, and student demographics of the universities.

Research Aims and Importance

The primary aim of this research is to delve deeper into these digital representations to understand how the Rutgers and UPenn communities articulate their university experiences online. By employing advanced Natural Language Processing (NLP) tools and machine learning techniques, my study seeks to accurately classify and analyze the content posted in these subreddits. This not only involves distinguishing between the two based on their discussions but also involves dissecting the prevalent themes, sentiments, and conversational patterns within each community.

Understanding these aspects can offer significant insights into the cultural nuances of each university, revealing how students and other community members perceive and interact with their environment. Such insights are invaluable for university administrators and policymakers for improving student engagement and tailoring communication strategies. Moreover, prospective students and academic researchers can leverage this information to gain a clearer picture of campus life, aiding in decision-making and academic studies.

Methodological Approach

To achieve these objectives, the study will harness both quantitative and qualitative data analysis methods. Quantitative analysis will involve statistical techniques to measure and compare activity levels, engagement rates, and other metrics across the two subreddits. Qualitatively, the study will use thematic analysis to identify and compare the dominant topics of discussion, sentiment analysis to gauge the emotional tone, and content analysis to explore the depth and nature of discussions.

By integrating these methods, the research intends to produce a comprehensive overview of how digital platforms like Reddit serve as extensions of campus life, reflecting and potentially shaping the university experience. This approach not only aligns with the academic goal of understanding digital community dynamics but also contributes to the broader field of digital humanities by exploring how online and offline university life interconnect and influence each other.

The Pursuit of Data :Dealing with Reddit's API

This research will involve the analysis of archived posts and comments from the Rutgers and UPenn subreddits, sourced through Pushshift, a service that archives and provides access to historical Reddit data. The dataset spans from 2010 to 2023, offering a comprehensive view of subreddit activity over an extended period. This long-term dataset is crucial for understanding trends, shifts in discussion topics, and changes in community engagement over time.

Breaking down our data:

1. author: The username of the post author.
2. score: The net upvotes minus downvotes the post received.
3. created: The date and time when the post was created.
4. link: A link associated with the post.
5. url: The URL provided in the post, if any.
6. subreddit: This field is intended to denote the subreddit.(created to label the titles for our analysis)

The data will be organized into a structured format, featuring columns for:

1. Textual Content: Capturing the essence of the discussions.
2. Popularity Metrics: Including upvotes and downvotes to gauge the community's response.
3. Temporal Data: To identify and analyze posting patterns over time.

Given the extensive range of data, specific attention will be paid to managing potential data imbalances:

Handling Imbalances: We will consider techniques such as undersampling or employing synthetic data generation methods like ADASYN. This approach will ensure a balanced representation of data across different periods and between the two subreddits, aiding in unbiased comparative analysis.
The outcome will be a meticulously compiled dataset, prepared for thorough analytical processing to extract meaningful patterns, trends, and insights, thereby illuminating the digital cultures of Rutgers and UPenn's university communities.

From New Brunswick to Philly: A Digital Tale of Two Universities

Overview of Expected Findings

This study aims to dissect the digital dialogues within the Rutgers and UPenn subreddits to uncover the prevailing themes and sentiments that characterize these university communities. Through the lens of Natural Language Processing and machine learning, we anticipate revealing distinctive thematic concerns that resonate uniquely within each subreddit, reflecting the specific interests, issues, and cultural nuances of their respective student bodies.

Thematic Divergence and Community Engagement

The analysis is expected to demonstrate clear thematic divergences that highlight the differences in community engagement related to student life and campus dynamics. For instance, we hypothesize that the discourse in each subreddit may vary significantly in relation to academic schedules—such as during examination periods or the onset of new semesters—and around major university events, which traditionally evoke heightened activity and specific concerns within student communities.

Correlation with Academic and Extracurricular Calendars

By mapping the discussion focuses and sentiments against the academic and extracurricular calendars of Rutgers and UPenn, the study aims to establish a correlation between the timing of posts and key university events. This temporal alignment could provide deeper insights into how university life influences subreddit activity, potentially guiding administrative strategies to better support students during peak stress periods or to capitalize on high engagement phases for community-building initiatives.

Data-Driven Perspectives on Student Experiences

The primary goal of this research is to compile a data-driven narrative that enriches our understanding of the student experience at these institutions as articulated through their online interactions. This narrative will not only illustrate the commonalities and disparities between the two university subreddits but also highlight the broader social and academic issues that preoccupy students today.

Openness to Emergent Themes

While our methodological framework is designed to capture and analyze well-defined data points and trends, we are acutely aware of the variability that characterizes social research. Thus, we remain receptive to emergent themes and unexpected findings that may challenge our initial hypotheses. This openness ensures that our study remains adaptable and comprehensive, truly reflective of the dynamic and evolving nature of university student communities on digital platforms.

By anticipating these findings, we set the stage for a nuanced exploration of the Rutgers and UPenn subreddits, aiming to provide stakeholders with actionable insights that go beyond mere statistical analysis to touch on the essence of student life and community engagement in the digital age.

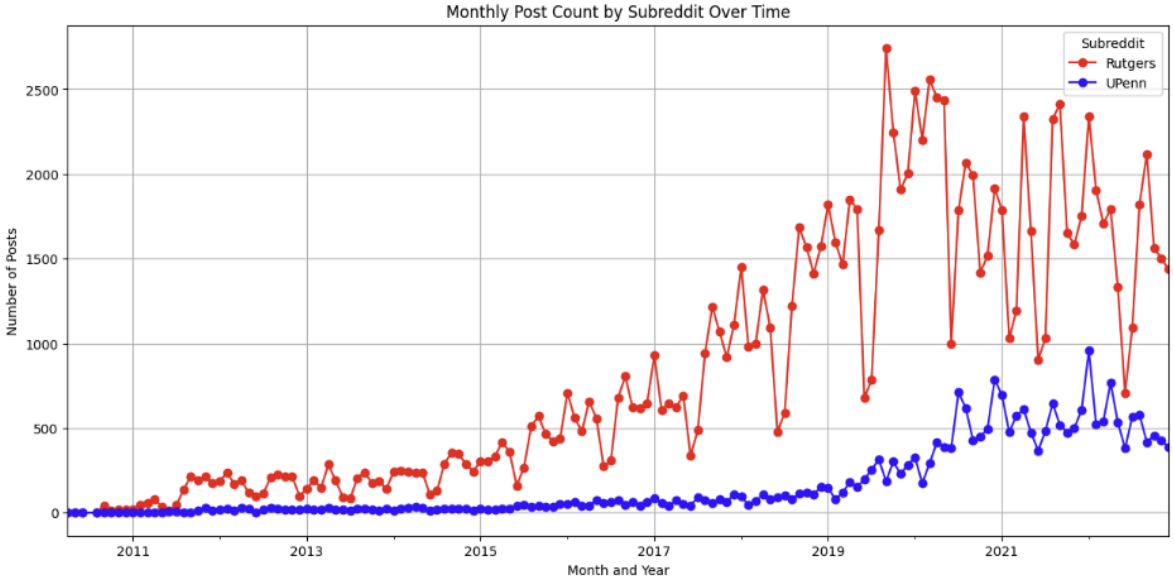
Methods

Unveiling Patterns: Statistical Insights and Trends

Now that we have our data loaded lets work on understanding the general layout of our dataset when combined together both our dataset has a total of 15194 rows and the 8 columns present.

We will now be breaking it down to understand how frequently our redditors post on both of the subreddit.

Historical timeline

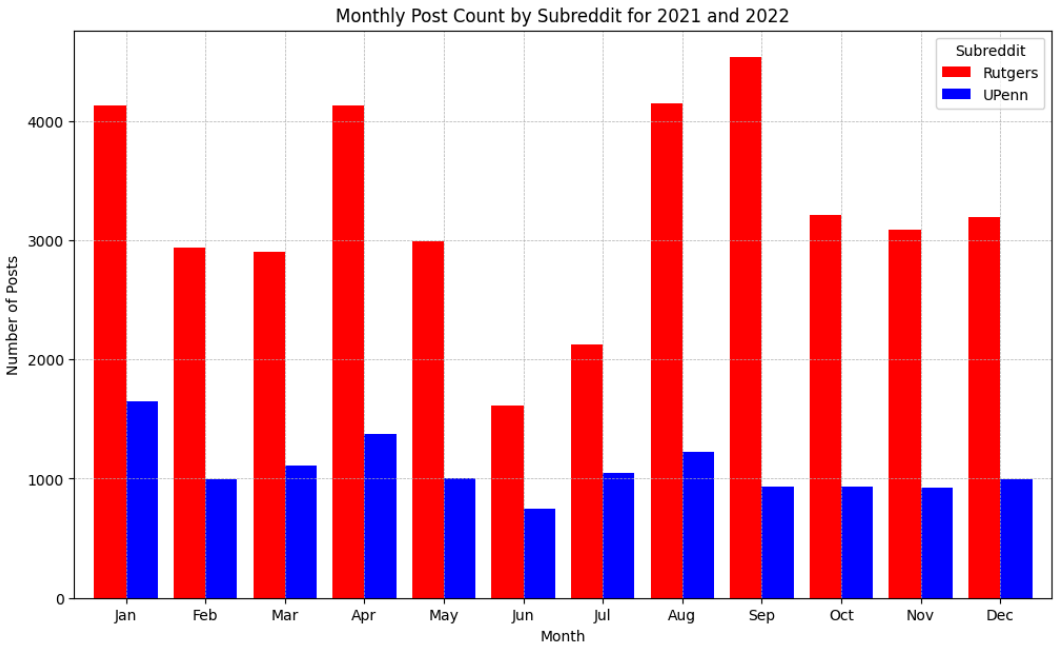



```
(Timestamp('2010-04-28 20:11:00'),
 Timestamp('2022-12-31 18:50:00'),
 subreddit
 Rutgers    127207
 UPenn       24707
 Name: count, dtype: int64)
```

Taking a look at our posts we can see the number posts by the subreddit rutgers leads when compared to upenn it also attributes to the number of students going there.

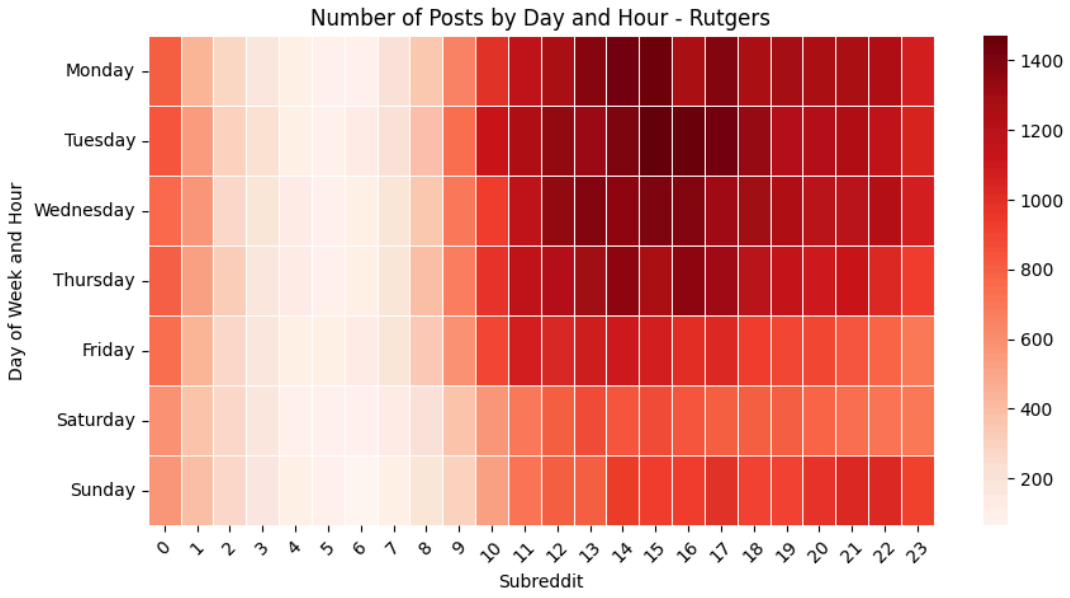
Another thing to notice is that most of these posts in both the subreddits have a cyclical pattern so let's move and explore that a bit more and plot the average number of posts in the past year (2021-2022)

What about Last Year

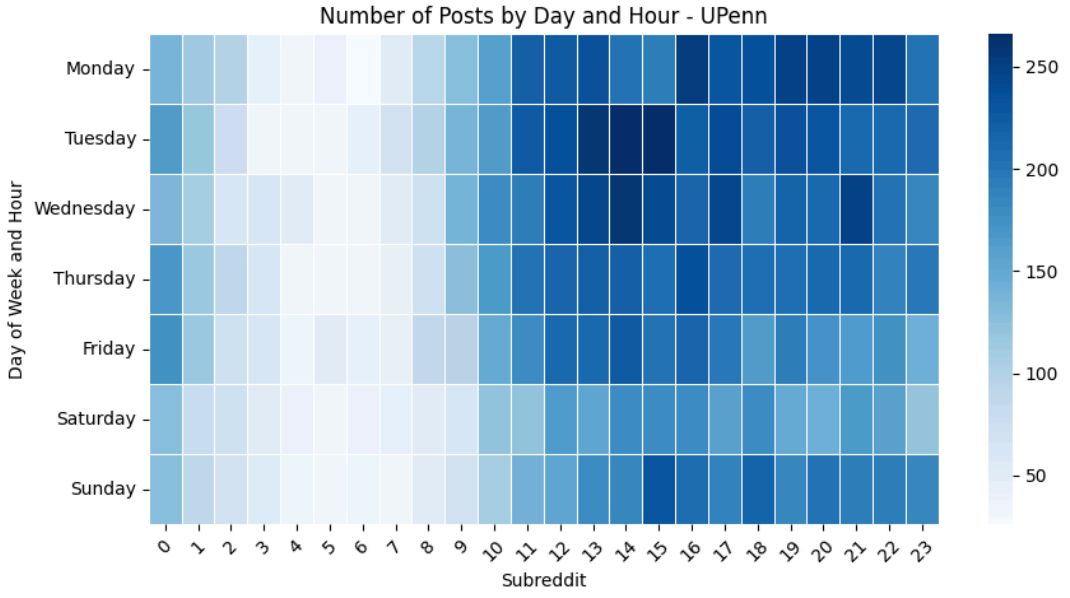


As we see there is a cyclical nature to the posts made by people during the academic year peaking when the semester starts and then again during exam season as well. Making it clear that this is during the Add-drop week and then during Finals week we can also assume that the posts in the month of Jan can be from the incoming students asking about the course curriculum and other college related questions.

The Weekly Buzz on Campus:



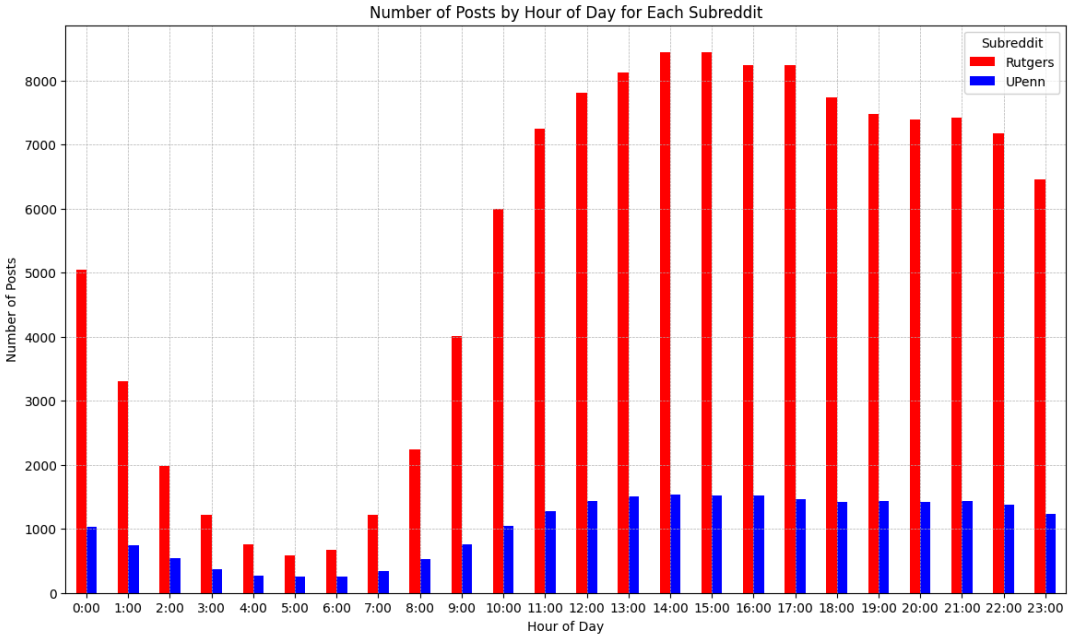
r/Rutgers shows higher posting frequency with significant peaks after 11:00 AM and slows down around 10:00 PM, while we also see that the posts peak during the weekdays and slow down during the weekends.



r/UPenn activity peaks in the late afternoons to evenings, with engagement from 3PM to midnight. Unlike Rutgers, UPenn maintains a steady flow of posts during weekends, though still

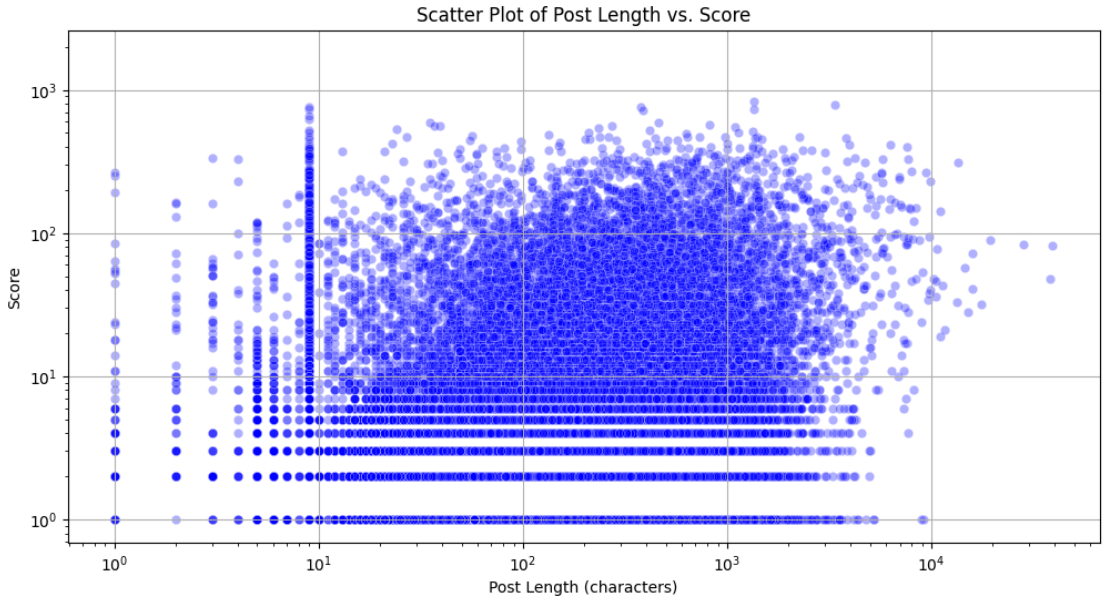
with an emphasis on evening hours, indicating a user base that is actively posting throughout the week with a notable inclination for nighttime interaction.

A Daily Distribution



r/Rutgers shows higher posting frequency with significant peaks after 11:00 AM and slows down around 10:00 PM, while r/UPenn has a more consistent posting pattern throughout the day.

But wait Does longer post mean better results

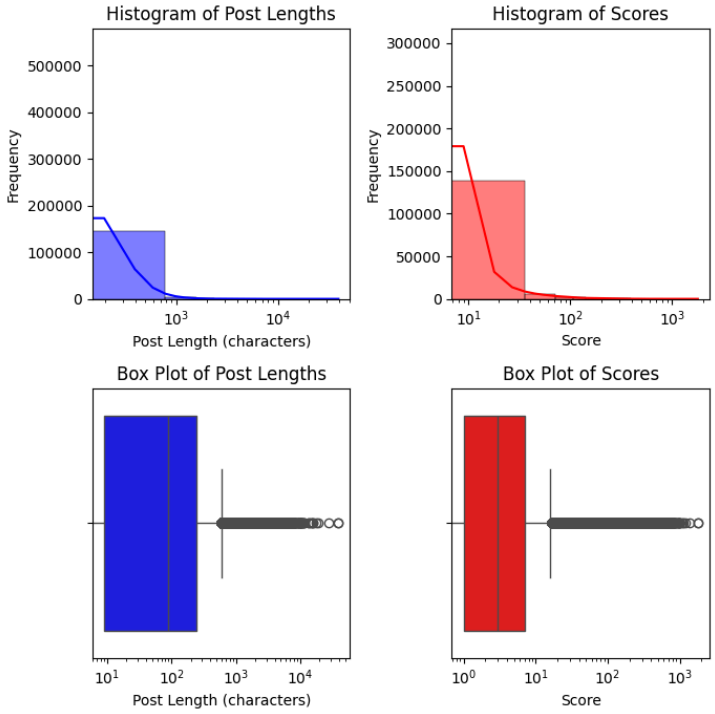


The scatter plot depicts the relationship between post length and score on a logarithmic scale, with a very low correlation coefficient of 0.0158.

```
0.01576762917368809
```

The plot shows a wide dispersion of points without a clear trend, indicating that post length does not strongly influence the score. The concentration of dots along the lower part of the y-axis suggests that most posts, regardless of length, receive a low to moderate score. Posts with a very high score are rare and do not appear to be dependent on length.

Let's check the distribution of our plots using histograms and Box plots



The data reveals a right-skewed distribution for both post lengths and scores, with the majority being short posts with lower scores and only a few outliers with high scores or long post lengths. The box plots further illustrate this concentration of lower values and the presence of extreme outliers.

Data Cleaning & PreProcessing

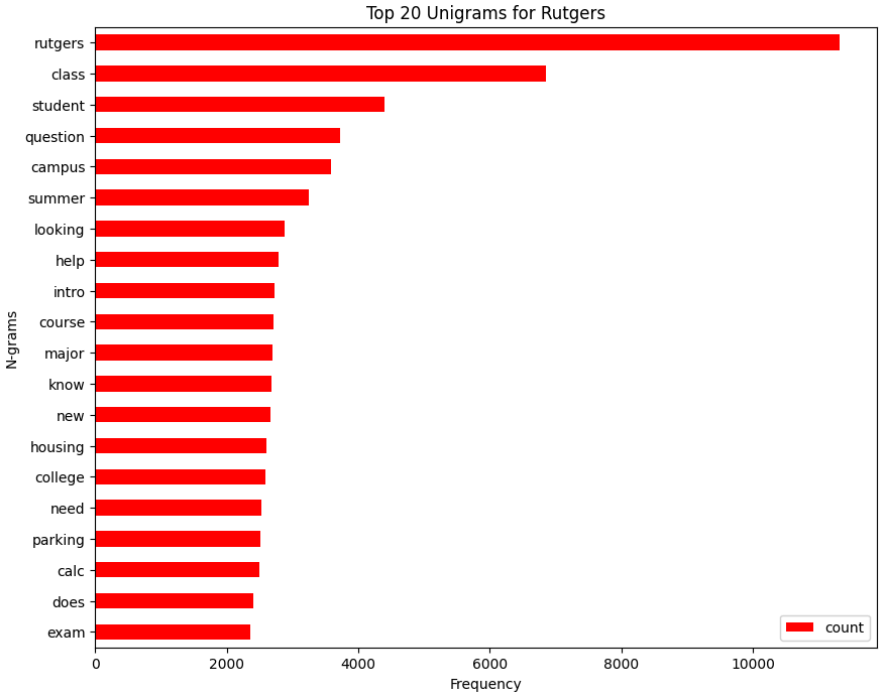
For

Now with our dataset lets prepare it for N-Gram analysis and also to create a word clout finally lets use word2vec model to dive deeper

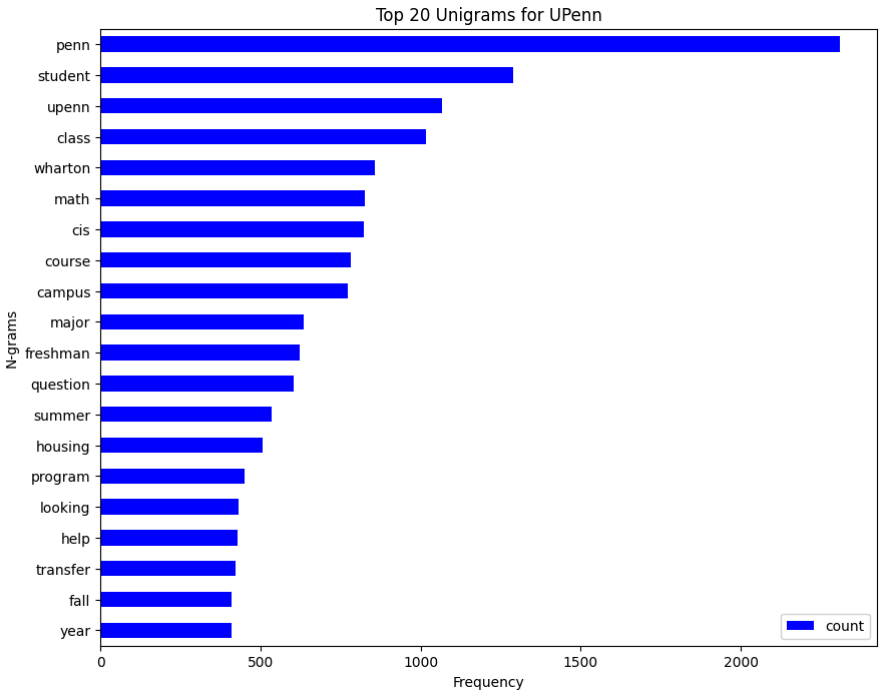
Detailed N-grams Analysis to Uncover Community Lexicons

With a pristine dataset at hand, the study proceeded with N-grams analysis to delve into the thematic and linguistic elements prevalent within the discussions on the Rutgers and UPenn subreddits. This involved breaking down the text into unigrams, bigrams, and trigrams, thereby capturing the nuanced lexicon of these university communities.

Unigrams Analysis:

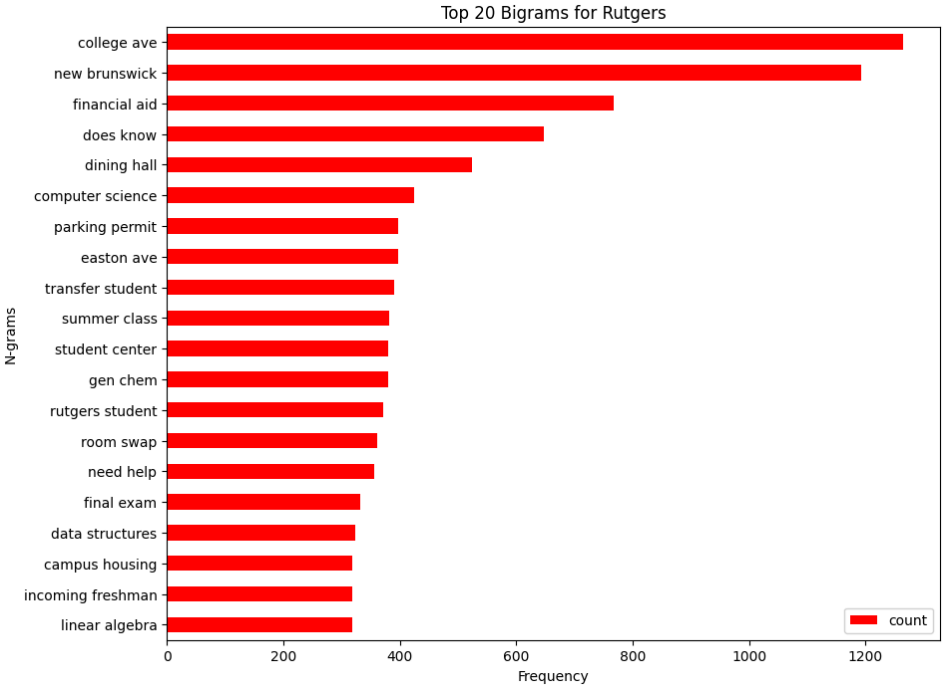


Rutgers: Predominant terms included 'student,' 'professor,' 'campus,' and 'housing.' These reflect the central, everyday concerns within the Rutgers subreddit, mirroring the active and diverse academic and residential life at the university.

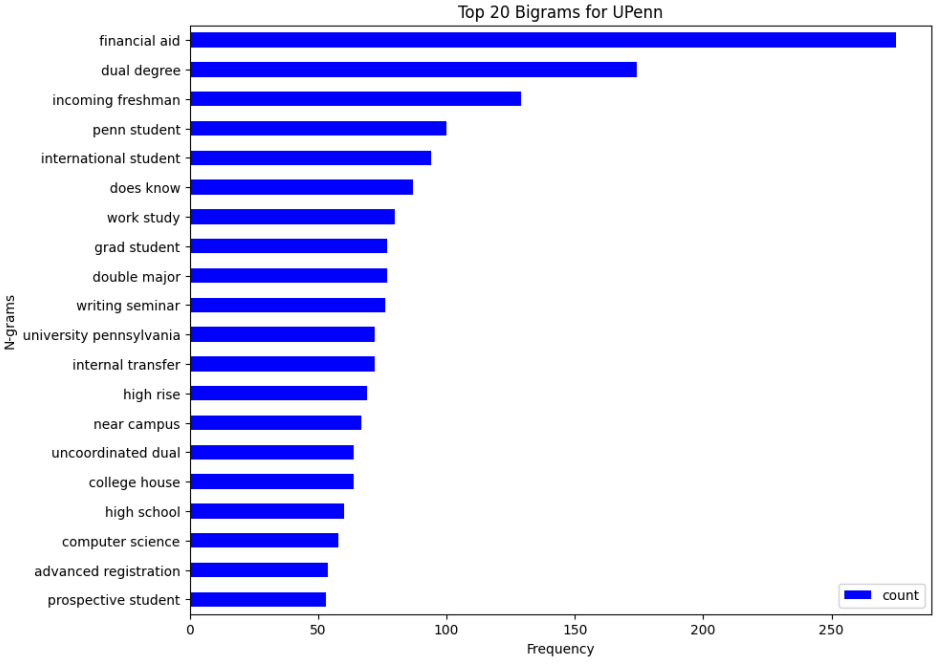


UPenn: The focus was on terms like 'Wharton,' 'research,' 'finance,' and 'innovation,' highlighting the subreddit's orientation towards UPenn's strong academic and professional ethos.

Bigrams Analysis:

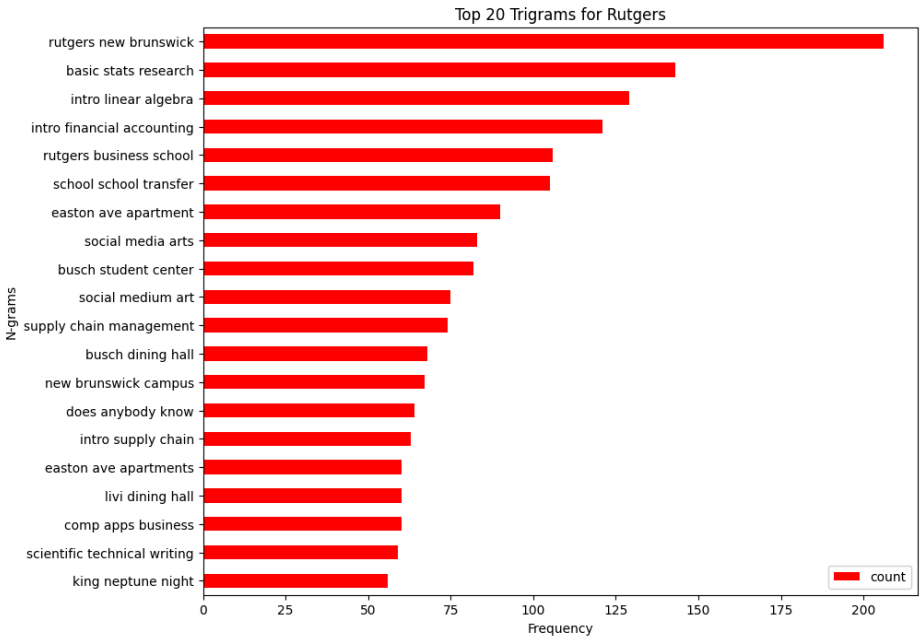


Rutgers: Frequently occurring phrases such as 'campus life,' 'financial aid,' and 'study abroad' point to the practical, day-to-day aspects of university experience that dominate student discussions.

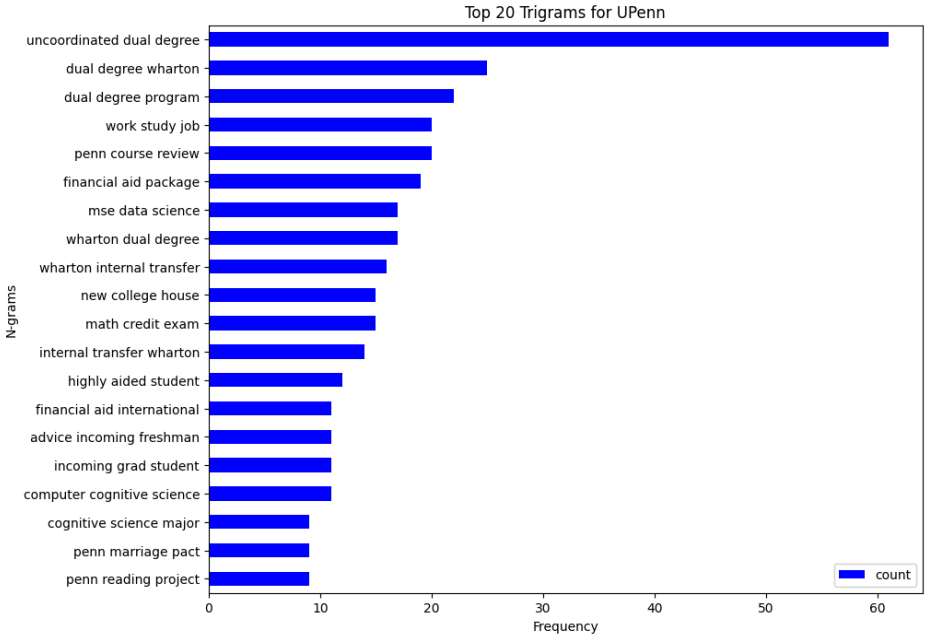


UPenn: Reflective phrases like 'research opportunities,' 'business school,' and 'graduate admissions' underscore the community's focus on academic advancement and professional preparation.

Trigrams Analysis:



Rutgers: Expressive phrases like 'end of semester exams,' 'new student orientation,' and 'on-campus housing' were common, indicating a vibrant discourse around significant academic and logistical milestones.



UPenn: Detailed discussions captured in phrases such as 'Wharton business analytics,' 'undergraduate research program,' and 'public health initiatives' reveal a deep engagement with specialized academic and professional themes.

The Impact of N-grams in Understanding Cultural Dynamics

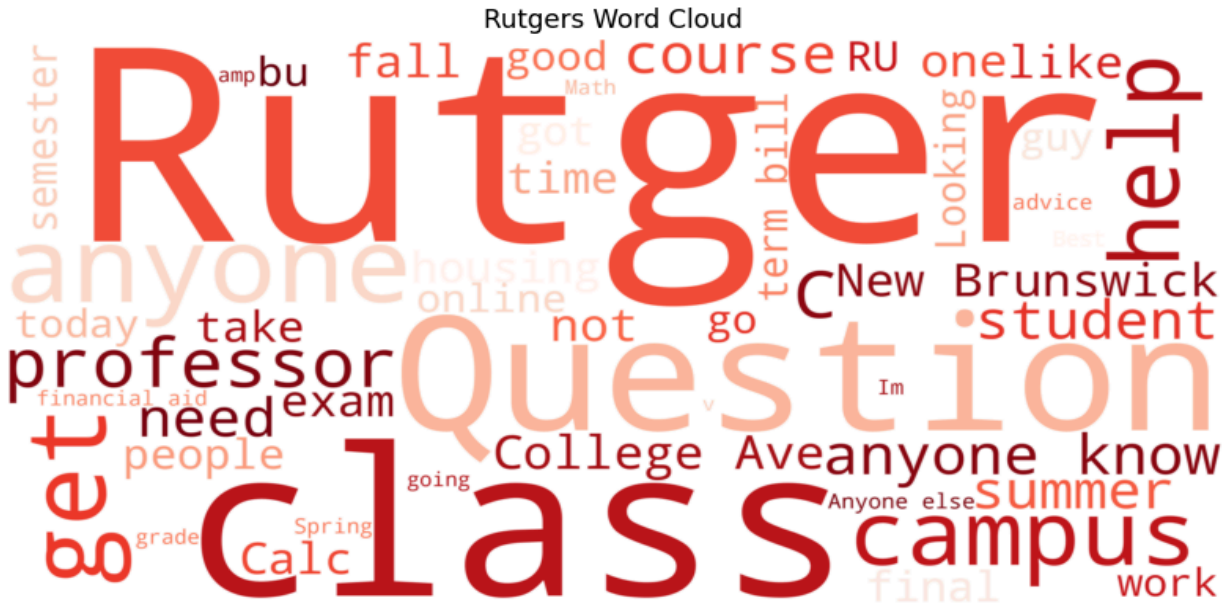
This exhaustive N-grams analysis provides more than just a count of frequently used words; it offers profound insights into the priorities, issues, and culture prevalent within the Rutgers and UPenn online communities. By highlighting these themes, the analysis not only enriches our understanding of the content but also paints a vivid picture of the community dynamics, illustrating how students and faculty members interact with and perceive their university environments.

Through the combination of detailed data cleaning and N-grams analysis, this study effectively prepares the dataset for deeper thematic and sentiment analysis, paving the way for a nuanced understanding of how digital platforms serve as extensions of physical university spaces. This approach not only aligns with the academic objectives of exploring digital community dynamics but also enhances the practical understanding of student engagement and community building in the digital age.

Word Cloud Analysis: Illustrating Community Focus

Word clouds are a visually engaging method to highlight the most prevalent terms within textual data, offering an immediate sense of the dominant themes in any given corpus. In the analysis of the Rutgers and UPenn subreddits, word clouds were utilized to manifest the frequency and relevance of specific words, thereby providing a quick, intuitive grasp of the community's focal points.

Rutgers Subreddit Word Cloud:



The word cloud for Rutgers prominently featured terms such as "campus," "students," "classes," and "advice," along with specific references to local sites and university services. These terms underscore the community's strong engagement with campus life and student-centric issues, reflecting a vibrant exchange of logistical tips and academic guidance.

UPenn Subreddit Word Cloud:

Conversely, the UPenn word cloud was dense with terms like "Wharton," "research," "career," and "graduate," mirroring the university's prestigious academic reputation and the community's career-oriented discussions. It also highlighted a strong emphasis on postgraduate opportunities and professional networking, which aligns with the subreddit's focus on advancing career paths.

Understanding Relations: A Word2Vec Model

```
1  model.wv.most_similar('rutgers', topn=10)  # get other similar words
```

```
[('penn', 0.786194920539856),
 ('pennsylvania', 0.7065936923027039),
 ('upenn', 0.7005419731140137),
 ('university', 0.6738671064376831),
 ('president', 0.6610144376754761),
 ('report', 0.6430330872535706),
 ('faculty', 0.6281049847602844),
 ('coronavirus', 0.6253470182418823),
 ('minecraft', 0.6171616911888123),
 ('rowan', 0.6158574819564819)]
```

```
1  model.wv.most_similar('upenn', topn=10)  # get other similar words
```

```
[('penn', 0.8254845142364502),
 ('mampt', 0.7774819135665894),
 ('competitive', 0.7641792297363281),
 ('future', 0.7527706623077393),
 ('current', 0.7508677244186401),
 ('alumnus', 0.7437769770622253),
 ('internally', 0.7241896986961365),
 ('program', 0.7209224104881287),
 ('wharton', 0.7206037640571594),
 ('disadvantage', 0.7199550271034241)]
```

The Word2Vec model's output showcases the words most closely associated with "rutgers" and "upenn," such as "pennsylvania" and "university" for "rutgers," and "wharton" and "program" for "upenn." These results highlight the contextual landscape surrounding each university name, reflecting common topics and themes within the discourse of their respective communities.

```
1  model.wv.most_similar('covid', topn=10)  # get other similar words
```

```
[('vaccine', 0.8242006301879883),
 ('result', 0.804023802280426),
 ('coronavirus', 0.794784665107727),
 ('positive', 0.7930689454078674),
 ('saliva', 0.79248046875),
 ('testing', 0.7795373797416687),
 ('pcr', 0.7591186165809631),
 ('emergency', 0.7534778118133545),
 ('tested', 0.7283697724342346),
 ('notification', 0.725772500038147)]
```

Understanding Covid the model identifies terms closely related to "covid," such as "vaccine," "result," and "coronavirus," suggesting a strong association with health, testing, and the pandemic response within the analyzed text corpus.

Analyzing Similarities and Differences

```
[ ]  1  # Most Similar Words
     2  print("Words most similar to 'rutgers':\n", model.wv.most_similar('rutgers', topn=10))
```

```
Words most similar to 'rutgers':
 [('penn', 0.786194920539856), ('pennsylvania', 0.7065936923027039), ('upenn', 0.7005419731140137), ('university', 0.6738671064376831), ('presi
```

```
[ ]  1  print("Words most similar to 'upenn':\n", model.wv.most_similar('upenn', topn=10))
```

```
Words most similar to 'upenn':
 [('penn', 0.8254845142364502), ('mampt', 0.7774819135665894), ('competitive', 0.7641792297363281), ('future', 0.7527706623077393), ('current',
```

```
[ ]  1  # Similarity Scores
     2  print("Similarity score between 'rutgers' and 'upenn':", model.wv.similarity('rutgers', 'upenn'))
```

```
Similarity score between 'rutgers' and 'upenn': 0.7005419
```

```
[ ]  1  print("Similarity score between 'rutgers' and 'ivy':", model.wv.similarity('rutgers', 'ivy'))
```

```
Similarity score between 'rutgers' and 'ivy': 0.6040357
```

```
[ ]  1  # Odd-One-Out Analysis
     2  print("Odd one out for ['upenn', 'ivy', 'rutgers']:", model.wv.doesnt_match(['upenn', 'ivy', 'rutgers']))
```

```
Odd one out for ['upenn', 'ivy', 'rutgers']: ivy
```

```
[ ]  1  print("Odd one out for ['rutgers', 'loan', 'job']:", model.wv.doesnt_match(['rutgers', 'loan', 'job']))
```

```
Odd one out for ['rutgers', 'loan', 'job']: rutgers
```

Our model shows around 70% between Rutgers and Upenn and considers Ivy to be the odd one out that's prob. Because both of them have more in common in terms of students and academics discussions and not just being ivy.
However UPenn does show competition as more similar to it.

Campus Life vs. Study Balance:

```
1  model.wv.most_similar(positive=['dorm', 'study'], negative=['home'], topn=10)
```

```
[('participant', 0.5092398524284363),
 ('learning', 0.4986644685268402),
 ('swap', 0.4971344470977783),
 ('seminar', 0.4758041203022003),
 ('assistant', 0.46448397636413574),
 ('tutor', 0.45459285378456116),
 ('studying', 0.4524478614330292),
 ('tutoring', 0.4449842572212219),
 ('double', 0.4393843412399292),
 ('biology', 0.4374176859855652)]
```

The analysis indicates that dormitory life is perceived as an integral part of the academic ecosystem on campus, with terms like 'participant', 'learning', 'seminar', 'tutor', and 'biology' implying a vibrant study culture that extends beyond classrooms into living spaces, where students actively engage in educational activities and access a network of academic support.
This suggests that dorms are not only residential areas but also social and learning hubs where collaborative study and academic discussions are a norm.

Undergraduate to Graduate Transition:

```
[ ]  1  model.wv.most_similar(positive=['graduate', 'undergraduate'], negative=['professor'], topn=10)
```

```
[('undergrad', 0.8129611015319824),
 ('nursing', 0.8065541982650757),
 ('applying', 0.7740465998649597),
 ('grad', 0.7657743692398071),
 ('pharmacy', 0.7549444437026978),
 ('current', 0.741146445274353),
 ('soe', 0.7401242852210999),
 ('master', 0.7288368344306946),
 ('sea', 0.720282793045044),
 ('apply', 0.7184532284736633)]
```

The terms 'undergrad', 'nursing', 'applying', 'grad', 'pharmacy', 'current', 'lose', 'master', 'sea', and 'apply' capture the transitional phase from undergraduate studies to graduate pursuits. The model suggests a focus on professional fields (like 'nursing' and 'pharmacy') and the application process ('applying', 'apply'), with an emphasis on the immediacy ('current') and the aspirational goal ('master') of advancing to higher education.

The terms 'swe', 'mayor', 'entrepreneurial', 'colgate', 'son', 'msu', 'worthwhile', 'adp', 'vet', and 'robot' represent a broad spectrum of stress-related topics discussed within academic communities, pointing to stressors outside of exam contexts.

The list suggests that conversations about stress relief may involve diverse strategies, including engaging with technology ('robot'), focusing on specific career paths, and navigating the personal dimensions of student life.

The model associates 'social' and 'university' with terms like 'president', 'penn', 'gross', 'event', 'scene', 'coronavirus', 'club'.

These terms can be seen as facets of a vibrant campus life: 'president' might relate to student government, 'gross' could refer to a popular location or a term specific to a community, while 'event', 'club', and 'scene' speak to the various social gatherings, student organizations, and the overall campus atmosphere.

'Coronavirus' appearing in this context indicates its significant impact on the social fabric of university life.

Transition from High School to College:

```
[ ]  1  model.wv.most_similar(positive=['college', 'highschool'], negative=['job'], topn=10)
```

```
[('livinglearning', 0.6982067823410034),
 ('tomorrowworld', 0.6740970611572266),
 ('regular', 0.6600842475891113),
 ('bergen', 0.6512401700019836),
 ('colloquium', 0.645980715751648),
 ('miracle', 0.6221482753753662),
 ('easton', 0.6211614012718201),
 ('church', 0.6209285259246826),
 ('cord', 0.6105048656463623),
 ('blvd', 0.6065792441368103)]
```

The model highlights the transition from high school to college as a blend of new academic endeavors ('livinglearning', 'colloquium'), ('tomorrowworld'), and the adaptation to a new social and geographical landscape ('bergen', 'easton', 'blvd'). Easton assumed to be easton ave at Rutgers.

Career Focus vs. Academic Focus:

```
1  model.wv.most_similar(positive=['career', 'education'], negative=['fun'], topn=10)
```

```
[('healthcare', 0.7187228202819824),
 ('information', 0.7116981148719788),
 ('master', 0.6934827566146851),
 ('certificate', 0.6678239703178406),
 ('phd', 0.664107620716095),
 ('tech', 0.6639496088027954),
 ('leadership', 0.6598132252693176),
 ('ahmed', 0.6398521065711975),
 ('concentration', 0.63780677318573),
 ('medicine', 0.6344700455665588)]
```

The model contrasts career-oriented discourse against educational content devoid of leisure, highlighting terms like 'healthcare', 'information', 'master', 'certificate', 'phd', 'tech', 'leadership', 'ahmed', 'concentration', and 'medicine'.

This implies a focus on professional qualifications, sectors, and personal advancement within academic conversations, emphasizing preparation for specific fields and leadership roles rather than general education or entertainment.

Impact of Online Learning:

```
[ ]  1  model.wv.most_similar(positive=['online', 'classroom'], negative=['campus'], topn=10)
```

```
[('feigenson', 0.7976371049880981),
 ('bender', 0.7900770902633667),
 ('planet', 0.7805723547935486),
 ('miller', 0.7685720920562744),
 ('synchronous', 0.7587992548942566),
 ('kalef', 0.7567599415779114),
 ('earth', 0.756706714630127),
 ('lepre', 0.7546864151954651),
 ('myth', 0.7528048753738403),
 ('appreciation', 0.7516663074493408)]
```

The model associates 'online' learning, in contrast to traditional 'classroom' settings, while excluding 'campus', with terms such as 'feigenson', 'bender', 'planet', 'miller', 'synchronous', 'kalef', 'earth', 'lepre', 'myth', and 'appreciation'.

This suggests that the impact of online learning extends into various realms, including the necessity for synchronous communication and perhaps a broader, more global or universal reach of education ('planet', 'earth').

Balancing Work and Study:

```
[ ]  1  model.wv.most_similar(positive=['work', 'study'], negative=['vacation'], topn=10)
```

```
[('labor', 0.5876917243003845),
 ('abroad', 0.53315269947052),
 ('guide', 0.513145923614502),
 ('studying', 0.4976927936077118),
 ('opportunity', 0.47777122259140015),
 ('resource', 0.44318392872810364),
 ('learning', 0.4399336576461792),
 ('research', 0.43641868233680725),
 ('done', 0.4348618984222412),
 ('relation', 0.42972394824028015)]
```

The terms 'labor', 'abroad', 'guide', 'studying', 'opportunity', 'resource', 'learning', 'research', 'done', and 'relation' suggest a multifaceted conversation around work-study life that includes practical aspects of working ('labor', 'resource'), the pursuit of education ('studying', 'learning'), the expansion of experiences ('abroad', 'opportunity'), and the culmination of efforts ('done').

Technology's Role in Education:

```
[ ]  1  model.wv.most_similar(positive=['technology', 'education'], negative=['textbook'], topn=10)
```

```
[('informatics', 0.7492423057556152),
 ('certificate', 0.7130184173583984),
 ('tech', 0.7125326991081238),
 ('analyst', 0.7040334939956665),
 ('security', 0.6996744871139526),
 ('science', 0.6922192573547363),
 ('medicine', 0.6915063261985779),
 ('development', 0.6900174617767334),
 ('entry', 0.685135006904602),
 ('global', 0.6837186217308044)]
```

The Word2Vec model output suggests a technology-centric educational dialogue, excluding traditional 'textbook' references. Terms like 'informatics', 'certificate', 'tech', 'analytics', 'security', 'science', 'medicine', 'development', 'entry', and 'global' reflect a shift towards digital fluency in education, with a focus on fields that intersect with technology, such as data analysis, cybersecurity, and global digital infrastructures.

Extracurricular Activities' Impact:

```
1  model.wv.most_similar(positive=['extracurricular', 'student'], negative=['class'], topn=10)
```

```
[('alumnus', 0.7753589749336243),
 ('philomathean', 0.735614538192749),
 ('rocket', 0.724502682685852),
 ('former', 0.7229083180427551),
 ('alliance', 0.7196483016014099),
 ('alum', 0.7187314629554749),
 ('rescue', 0.713726282119751),
 ('nonprofit', 0.7093974947929382),
 ('impacting', 0.7075549364089966),
 ('hup', 0.7064353227615356)]
```

The terms 'alumnus', 'philomathean', 'rocket', 'former', 'alliance', 'alum', 'rescue', 'nonprofit', 'impacting', and 'hup' highlight a range of activities and roles that students might engage with outside the classroom. This includes involvement with alumni networks, participation in clubs or societies (like 'philomathean'), interest in projects or competitions ('rocket'), and contributions to causes ('rescue', 'nonprofit'), reflecting how extracurriculars shape the student experience and potentially impact future endeavors.

Atheletics:

```
# Athlete experience at Rutgers
rutgers_athlete_analogy = model.wv.most_similar(positive=['athlete', 'rutgers'], negative=['student'], topn=10)
```

```
# Athlete experience at UPenn
upenn_athlete_analogy = model.wv.most_similar(positive=['athlete', 'upenn'], negative=['student'], topn=10)
```

```
# Execute the athlete experience analogy for Rutgers
print("Words associated with athletes at Rutgers as compared to the general student body:")
for word, similarity in rutgers_athlete_analogy:
    print(f"{word}: {similarity:.4f}")

# Execute the athlete experience analogy for UPenn
print("\nWords associated with athletes at UPenn as compared to the general student body:")
for word, similarity in upenn_athlete_analogy:
    print(f"{word}: {similarity:.4f}")
```

```
Words associated with athletes at Rutgers as compared to the general student body:
banning: 0.7180
listserv: 0.7178
actively: 0.7117
ruhungry: 0.7025
orgs: 0.6991
association: 0.6969
kelley: 0.6965
capability: 0.6959
fired: 0.6950
body: 0.6932

Words associated with athletes at UPenn as compared to the general student body:
upenns: 0.7691
stance: 0.7584
upennwharton: 0.7569
msds: 0.7557
simulation: 0.7552
kelley: 0.7542
gamma: 0.7533
population: 0.7528
founded: 0.7509
ruhungry: 0.7475
```

At Rutgers, athlete discussions emphasize organizational involvement and active participation, with terms like 'banning' and 'listserv' indicating specific athletic-related issues and communications.

UPenn athlete conversations align more with academic integration and institutional policy, with references to prestigious programs like 'upennwharton' and broader concepts such as 'population'.

Rutgers vs. UPenn Experience Analogy:

```
# Rutgers experience versus UPenn
rutgers_upenn_analogy = model.wv.most_similar(positive=['rutgers', 'upenn'], negative=['student'], topn=10)
```

```
# Execute the Rutgers vs. UPenn experience analogy
print("\nWords that distinguish the Rutgers experience from the UPenn experience:")
for word, similarity in rutgers_upenn_analogy:
    print(f"{word}: {similarity:.4f}")

```

```
Words that distinguish the Rutgers experience from the UPenn experience:
penn: 0.7255
pennsylvania: 0.5822
server: 0.5807
listserv: 0.5620
kelley: 0.5497
fellow: 0.5428
minecraft: 0.5399
created: 0.5255
upenns: 0.5253
competitive: 0.5240
```

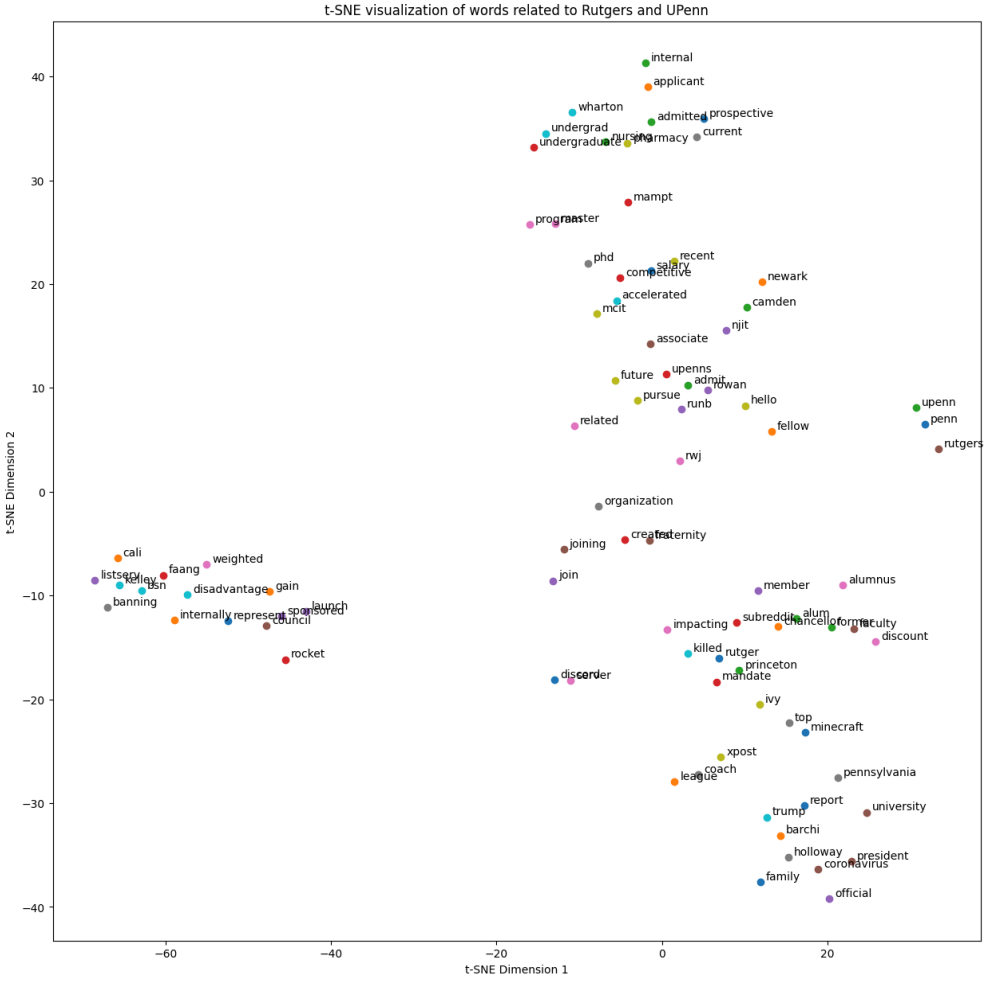
The Word2Vec analysis differentiates the Rutgers experience from UPenn's by highlighting terms like 'penn', 'pennsylvania', 'server', 'listserv', 'kelley', 'fellow', 'minecraft', 'created', 'upenns', and 'competitive'. This suggests Rutgers discussions may emphasize state and regional identity ('pennsylvania'), digital communication platforms ('server', 'listserv'), community and academic recognition ('fellow', 'kelley'), and aspects of creativity and competition within the student experience.

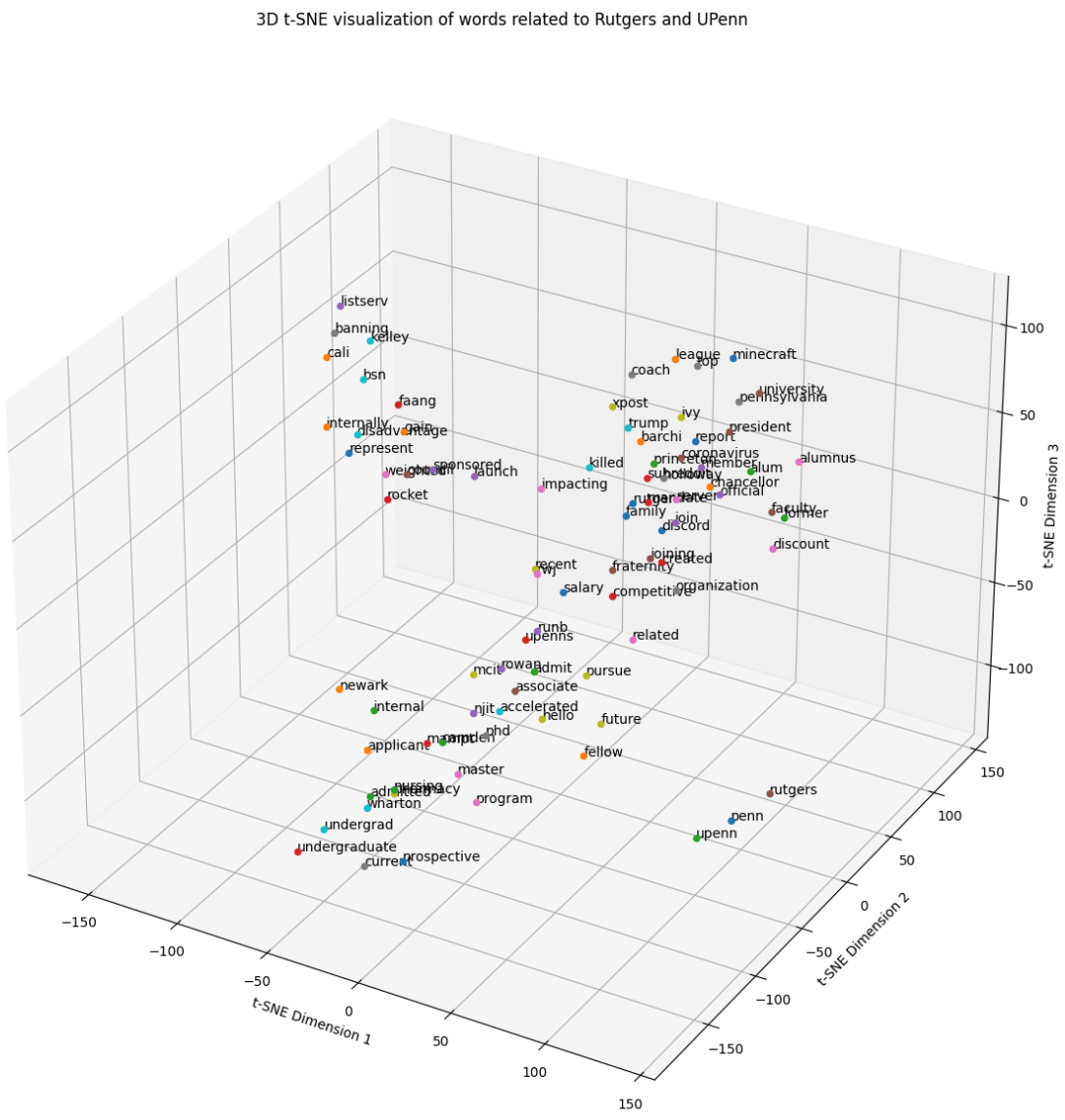
t-SNE Plot: Mapping High-Dimensional Data

t-Distributed Stochastic Neighbor Embedding (t-SNE) is a powerful machine learning algorithm used to visualize high-dimensional data by reducing it to two or three dimensions. This technique was applied to further explore the nuanced relationships between various terms and topics discussed within the Rutgers and UPenn subreddits.

t-SNE Analysis Overview:

The t-SNE plot revealed distinct clusters that correspond to specific themes and discussions prevalent within each subreddit. For Rutgers, clusters formed around student life, academic support, and campus events, indicating a communal focus on navigating university life. For UPenn, the clusters were more distinctly aligned with career advice, academic research, and professional development, reflecting the subreddit's emphasis on professional achievement and academic excellence.





Interpreting Clusters:

Each cluster on the t-SNE plot represents a convergence of related terms and topics, providing insights into how different subjects are interconnected. For example, terms like "exam" and "study" might cluster with "stress" and "help," suggesting common discussions around academic pressures and seeking support. Similarly, "job," "internship," and "networking" might cluster together in the UPenn subreddit, highlighting the professional focus of the discussions.

Building the Classification Model: Detailed Methodology Using Gemini API and Traditional Machine Learning Methods

To robustly distinguish between posts from the Rutgers and UPenn subreddits, our approach utilized two distinct methodologies: the Gemini API for advanced text embeddings and traditional machine learning classifiers. Below is an exhaustive breakdown of each method, detailing the steps taken to ensure clarity and actionability.

Using the Gemini API for Text Embeddings

Step 1: Data Preparation

```python
import pandas as pd
import re
import string
import emoji
import nltk

# Ensure necessary NLTK resources are available
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('punkt')

# Data Cleaning and Preprocessing
def clean_text(text):
    text = emoji.replace_emoji(text, replace='')
    text = re.sub(r'https?://\S+|www\.\S+', '', text)
    text = re.sub(r'[\t\n\r]+', ' ', text)
    text = text.translate(str.maketrans('', '', string.punctuation))
    text = re.sub(r'\w*\d\w*', '', text)
    text = re.sub(r'\s+[a-zA-Z]\s+', ' ', text)
    text = text.strip()
    # Filter out any strings that have become empty after cleaning
    return text if text else None

def process_text(text):
    if text is None:  # If text is None, skip processing
        return None
    stopwords = set(nltk.corpus.stopwords.words("english")) - set(["not"])
    lemmatizer = nltk.stem.WordNetLemmatizer()
    tokens = nltk.word_tokenize(text)
    processed_text = [lemmatizer.lemmatize(word) for word in tokens if word.lower() not in stopwords]
    # Create a combined string and filter out any that would be empty
    processed = " ".join(processed_text)
    return processed if processed else None

# Apply cleaning and processing to the dataset
filtered_data['cleaned_title'] = filtered_data['title'].apply(clean_text)
filtered_data['processed_title'] = filtered_data['cleaned_title'].apply(process_text)

# Drop any rows where 'processed_title' is None
filtered_data = filtered_data.dropna(subset=['processed_title'])

print("Data after cleaning and processing:", filtered_data.head())
```

Dataset Loading: The first step involved loading the cleaned and preprocessed dataset, which includes titles and other textual content from the subreddits, into our analysis environment. This data had undergone extensive cleaning to remove noise and standardize the text.

Text Processing: We applied standard NLP techniques such as tokenization and the removal of stop words. This refinement was crucial for reducing the dimensionality of the text and enhancing the quality of the embeddings generated in the subsequent steps.

```
[ ]  1  # Random sampling
     2  sampled_data = df.groupby('subreddit').sample(n=2000, random_state=1)
```

```
     1  sampled_data.value_counts("processed_title")
```

```
processed_title
CIS                          7
Housing                      4
summer                       3
Friends                      3
v                            3
                            ..
MAE Technical Electives      1
MATH Help                    1
MATH Ideas Mathematics       1
MATH MATH                    1
…                            1
Name: count, Length: 3943, dtype: int64
```

Since Dealing with a Large Dataset we also focused on downsampling the time to generate all the embeddings was a lot especially when dealing with a dataset this huge

Step 2: Setting Up the Gemini API

Setting UP API

```
     1  # Or use `os.getenv('API_KEY')` to fetch an environment variable.
     2  API_KEY=userdata.get('API_KEY')
     3
     4  genai.configure(api_key=API_KEY)
```

Getting list of Models

```
[ ]  1  for m in genai.list_models():
     2    if 'embedContent' in m.supported_generation_methods:
     3      print(m.name)
```

```
models/embedding-001
models/text-embedding-004
```

Selecting Model

```
[ ]  1  model = 'models/embedding-001'
```

API Configuration: The Gemini API, specifically designed for creating robust text embeddings, was configured. This setup involved initializing the API with the necessary keys and parameters to enable interaction with our dataset.

Embedding Generation: We then passed the processed text data through the Gemini API, which generated high-dimensional vector representations for each text entry. These embeddings are designed to capture the semantic nuances of the text, making them ideal for high-precision classification tasks.

```
[ ]  1  df_train
```

	Unnamed: 0	subreddit	processed_title	Encoded Label	Embeddings
0	0	Rutgers	girl currently not oncampus housing would like...	0	[0.017395863, -0.03309162, -0.026439212, -0.07...
1	1	Rutgers	Livingston Dining Hall Guest Swipe	0	[0.04196943, -0.011381745, -0.057876058, -0.05...
2	2	Rutgers	bus route	0	[0.016273398, -0.030588085, -0.05447116, -0.04...
3	3	Rutgers	Hypothetical question allowed store gallon liq...	0	[-0.013889403, -0.06625572, -0.052370287, -0.0...
4	4	Rutgers	get another covid relief fund spring	0	[0.003015446, -0.0065499432, -0.0581211, -0.04...
...	...	...	...	...	...
1995	1995	UPenn	schedule vaccine schedule one dose	1	[0.004068121, -0.06631477, -0.033852454, -0.04...
1996	1996	UPenn	everyone get preorientation program acceptance...	1	[0.021544827, -0.039928205, -0.019442463, -0.0...
1997	1997	UPenn	prof better	1	[0.013047099, -0.016865127, 0.027865369, -0.02...
1998	1998	UPenn	Penn building still closed	1	[0.03347769, -0.01859094, -0.026078595, -0.057...
1999	1999	UPenn	Admitted master program bigger dream admit PhD...	1	[-0.018892385, -0.023043964, -0.056339025, -0....

2000 rows × 5 columns

Step 3: Model Build & Training

```python
import pandas as pd
from sklearn.model_selection import train_test_split

def sample_and_encode_data(df, num_samples=None):
    # Sample data if num_samples is specified
    if num_samples:
        df = df.groupby('subreddit', as_index=False).apply(lambda x: x.sample(num_samples, replace=True, random_state=42)).reset_index(drop=True)

    # Encode the 'color' column as categorical codes
    df['subreddit'] = df['subreddit'].astype('category')
    df['Encoded Label'] = df['subreddit'].cat.codes

    return df

# Assuming 'df' is your full DataFrame
# Splitting the data into train and test sets
train_size = 0.70
df_train, df_test = train_test_split(df, stratify=df['subreddit'], train_size=train_size, test_size=0.30, random_state=42)

# Optionally balance classes in training data
BALANCE_SAMPLES = 1000  # We will be downsampeling our dataset
df_train = sample_and_encode_data(df_train, BALANCE_SAMPLES)
df_test = sample_and_encode_data(df_test)  # No balancing for test data, just encoding

# Check the distribution of classes and the head of the datasets to confirm setup
print(df_train.value_counts('subreddit'))
print(df_test.value_counts('subreddit'))
print(df_train.head())
print(df_test.head())
```

```
subreddit
Rutgers    1000
UPenn      1000
Name: count, dtype: int64
subreddit
Rutgers    10951
UPenn       3608
Name: count, dtype: int64
  subreddit                                  processed_title  Encoded Label
0   Rutgers  girl currently not oncampus housing would like...              0
1   Rutgers                Livingston Dining Hall Guest Swipe              0
2   Rutgers                                          bus route              0
3   Rutgers  Hypothetical question allowed store gallon liq...              0
4   Rutgers                get another covid relief fund spring              0
      subreddit                                  processed_title  \
142320     UPenn                        freshman take course first semester
145166     UPenn      UPenn Masters Biotechnology v Bioengineering
106086   Rutgers                                          fun class
108742   Rutgers  ' Social Media amp Work KerrMccurry credit cou...
95253    Rutgers                                  ticket football game

        Encoded Label
142320              1
145166              1
106086              0
108742              0
95253               0
```

Data Splitting: With embeddings ready, the dataset was split into training and testing subsets to facilitate unbiased evaluation of the model's performance.

Classifier Setup: We established a binary classifier with a sigmoid activation function. This classifier was tailored to predict the subreddit origin (Rutgers or UPenn) based on the text embeddings.

#Lets build a simple classification model

```
import ast
df_train['Embeddings'] = df_train['Embeddings'].apply(ast.literal_eval)
df_test['Embeddings'] = df_test['Embeddings'].apply(ast.literal_eval)
```

```
from keras import Model, layers, Input

def build_single_neuron_model(input_size):
    input_layer = Input(shape=(input_size,))
    output_layer = layers.Dense(1, activation='sigmoid')(input_layer)
    return Model(inputs=input_layer, outputs=output_layer)

# Assuming num_classes is 1 because it's a binary classification with a single neuron
input_size = 768  # Ensure this matches the size of your embeddings
classifier = build_single_neuron_model(input_size)

# Compile the model
classifier.compile(loss='binary_crossentropy',  # Use binary_crossentropy for binary classification
                   optimizer='adam',
                   metrics=['accuracy'])

# Display the model structure
classifier.summary()
```

```
Model: "model_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_7 (InputLayer)        [(None, 768)]             0

 dense_10 (Dense)            (None, 1)                 769

=================================================================
Total params: 769 (3.00 KB)
```

Building: Now our model is a basic model with a sigmoid activation and is a binary classifier, the model also has a single layer and uses the adam optimizer

##Train the model to classify Reddit titles

```
# Extract labels and convert embeddings list into a numpy array for training
y_train = df_train['Encoded Label']  # Make sure to encode your labels as integers if not already done
x_train = np.stack(df_train['Embeddings'].values)

y_test = df_test['Encoded Label']
x_test = np.stack(df_test['Embeddings'].values)
```

[117] Python

```
NUM_EPOCHS = 50
BATCH_SIZE = 5

# Setup early stopping callback to prevent overfitting
callback = keras.callbacks.EarlyStopping(monitor='val_loss', patience=4)

# Train the model
history = classifier.fit(
    x=x_train,
    y=y_train,
    validation_data=(x_test, y_test),
    callbacks=[callback],
    batch_size=BATCH_SIZE,
    epochs=NUM_EPOCHS
)
```

[118] Python

```
Epoch 1/50
400/400 [==============================] - 7s 16ms/step - loss: 0.6877 - accuracy: 0.5650 - val_loss:
0.6884 - val_accuracy: 0.5366
Epoch 2/50
400/400 [==============================] - 6s 15ms/step - loss: 0.6729 - accuracy: 0.6280 - val_loss:
0.6354 - val_accuracy: 0.7906
Epoch 3/50
400/400 [==============================] - 6s 15ms/step - loss: 0.6620 - accuracy: 0.6620 - val_loss:
0.6545 - val_accuracy: 0.6983
Epoch 4/50
```

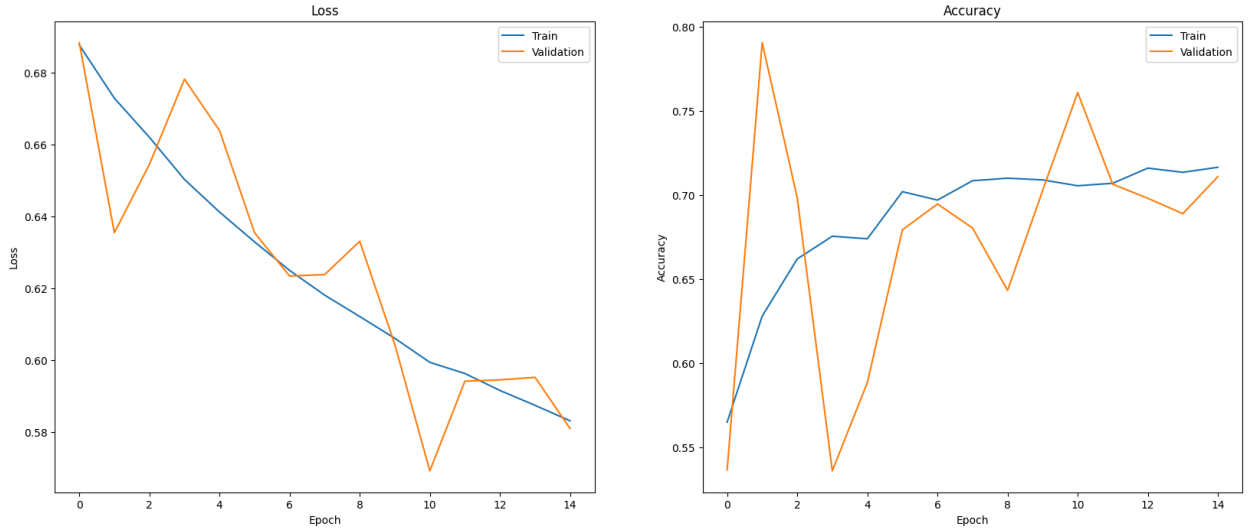
Training: We start with training the model with Early stopping that monitors our val_loss and stops once the model reaches 3 same results

```
results = classifier.evaluate(x=x_val, y=y_val, return_dict=True)
print("Evaluation results:", results)
```
[141] Python

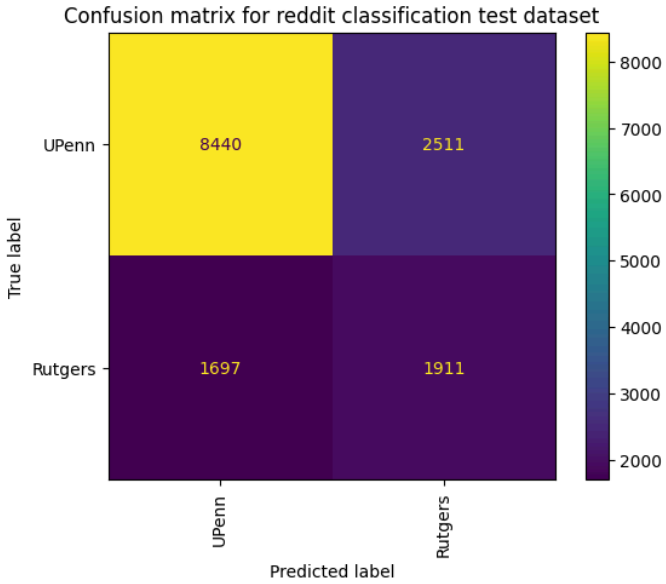
```
455/455 [==============================] - 1s 2ms/step - loss: 0.5809 - accuracy: 0.7110
Evaluation results: {'loss': 0.5809255242347717, 'accuracy': 0.7109691500663757}
```

Lets evaluate our model we can see our accuracy is 71%
We also noticed that the loss is around 58%

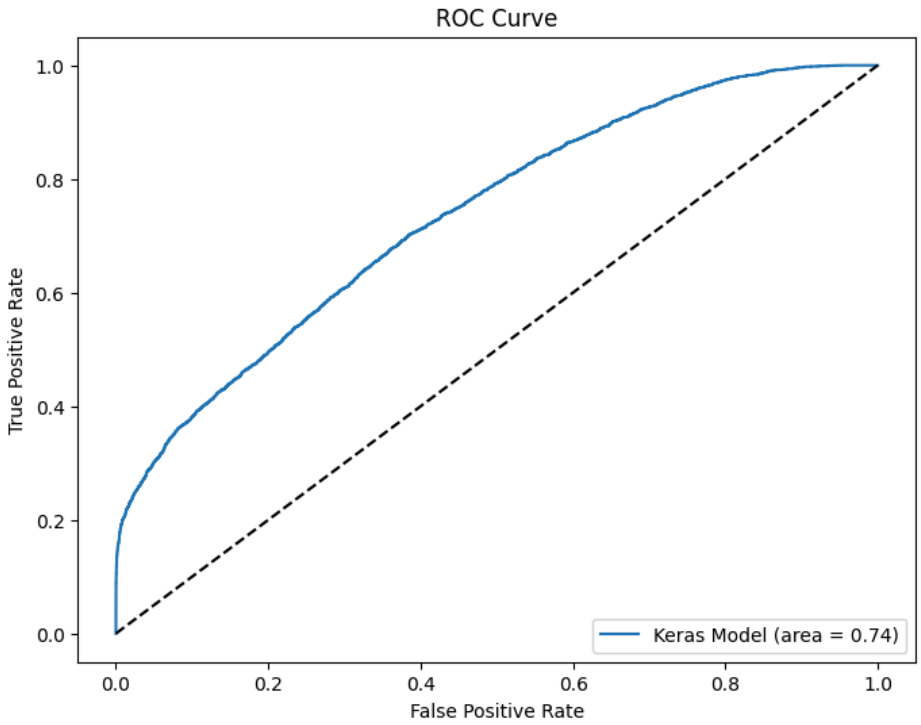
Step 4: Model Evaluation



Testing: Post-training, the model was evaluated using the reserved test set to assess both its accuracy and its ability to generalize.



Despite balanced training data, the model predicted UPenn posts more accurately, with 8,440 true positives, but it confused 1,697 Rutgers posts as UPenn, likely due to overlapping features between subreddits. The test imbalance influenced the predictive outcome, showing a tendency toward UPenn classifications.



Performance Metrics:
Our confusion matrix was bad but the ROC curve looks good and I think the reason could be class Imbalance: If there is a class imbalance, the ROC curve can be misleading. It considers the rates of true and false positives, which may not reflect the model's ability to predict the minority class well. This can make the model appear to perform better on the ROC curve than it actually does when considering precision and recall, as shown in the confusion matrix.

Employing Traditional Machine Learning Classifiers

Step 1: Feature Extraction

```
[ ]
from sklearn.feature_extraction.text import TfidfVectorizer

# Vectorize the processed titles
vectorizer = TfidfVectorizer(max_features=5000)  # Adjust the number of max_features as needed
X = vectorizer.fit_transform(df_encoded['processed_title']).toarray()

# The labels are already encoded in your 'Encoded Label' column
y = df_encoded['Encoded Label'].values

# Use the same train-test split you used for the Keras model
train_size = 0.70
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, train_size=train_size, test_size=0.30, random_state=42)
```

TF-IDF Vectorization: We transformed the text data into a numerical format using the TF-IDF vectorizer. The script uses TfidfVectorizer to convert text data into numerical TF-IDF features, considering the 5,000 most frequent terms, to prepare for text classification.
Labels for prediction are extracted from the 'Encoded Label' column, and the dataset is split into a 70:30 ratio for training and testing, maintaining class proportionality.
The specified random_state ensures consistent dataset splits across different runs, which is crucial for replicable results in machine learning experiments.

Step 2: Model Selection

```python
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import BernoulliNB
from sklearn.neighbors import KNeighborsClassifier

# List of models
models = [
    DecisionTreeClassifier(),
    LogisticRegression(max_iter=1000),
    SVC(),
    RandomForestClassifier(),
    BernoulliNB(),
    KNeighborsClassifier()
]

model_names = ["Decision Tree", "Logistic Regression", "SVC", "Random Forest", "Naive Bayes", "K-Neighbors"]

# Initialize dictionary to store accuracies
model_accuracies = {}

# Training and Evaluating Models
for model, name in zip(models, model_names):
    model.fit(X_train, y_train)
    predictions = model.predict(X_test)
    accuracy = accuracy_score(y_test, predictions)
    model_accuracies[name] = accuracy  # Store the accuracy in the dictionary
    print(f"{name} Test Accuracy: {accuracy:.2f}")
    print("Confusion Matrix:\n", confusion_matrix(y_test, predictions))
    print("Classification Report:\n", classification_report(y_test, predictions))

# Identify the best performing model
best_model_name = max(model_accuracies, key=model_accuracies.get)
best_accuracy = model_accuracies[best_model_name]
print(f"The best performing model is: {best_model_name} with an accuracy of {best_accuracy:.2f}")

# Select the best model
best_model = models[model_names.index(best_model_name)]
```

```
Decision Tree Test Accuracy: 0.68
Confusion Matrix:
 [[445 155]
 [234 366]]
Classification Report:
              precision    recall  f1-score   support

           0       0.66      0.74      0.70       600
           1       0.70      0.61      0.65       600

    accuracy                           0.68      1200
   macro avg       0.68      0.68      0.67      1200
weighted avg       0.68      0.68      0.67      1200
```

```
Logistic Regression Test Accuracy: 0.71
Confusion Matrix:
 [[420 180]
 [162 438]]
Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.70      0.71       600
           1       0.71      0.73      0.72       600

    accuracy                           0.71      1200
   macro avg       0.72      0.71      0.71      1200
weighted avg       0.72      0.71      0.71      1200
```

```
SVC Test Accuracy: 0.72
Confusion Matrix:
 [[439 161]
 [170 430]]
Classification Report:
              precision    recall  f1-score   support

           0       0.72      0.73      0.73       600
           1       0.73      0.72      0.72       600

    accuracy                           0.72      1200
   macro avg       0.72      0.72      0.72      1200
weighted avg       0.72      0.72      0.72      1200
```

```
Random Forest Test Accuracy: 0.70
Confusion Matrix:
 [[368 232]
 [130 470]]
Classification Report:
              precision    recall  f1-score   support

           0       0.74      0.61      0.67       600
           1       0.67      0.78      0.72       600

    accuracy                           0.70      1200
   macro avg       0.70      0.70      0.70      1200
weighted avg       0.70      0.70      0.70      1200
```

```
Naive Bayes Test Accuracy: 0.73
Confusion Matrix:
 [[399 201]
 [119 481]]
Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.67      0.71       600
           1       0.71      0.80      0.75       600

    accuracy                           0.73      1200
   macro avg       0.74      0.73      0.73      1200
weighted avg       0.74      0.73      0.73      1200
```

```
K-Neighbors Test Accuracy: 0.68
Confusion Matrix:
 [[428 172]
 [210 390]]
Classification Report:
              precision    recall  f1-score   support

           0       0.67      0.71      0.69       600
           1       0.69      0.65      0.67       600

    accuracy                           0.68      1200
   macro avg       0.68      0.68      0.68      1200
weighted avg       0.68      0.68      0.68      1200
```

Classifier Choices: A range of classifiers was evaluated, including Decision Trees, Logistic Regression, Support Vector Machines (SVC), Random Forests, Naive Bayes, and K-Nearest Neighbors. This diversity allowed us to explore various algorithmic strengths and weaknesses in the context of text classification.

We now train and evaluate multiple machine learning models, including Decision Tree, Logistic Regression, Support Vector Classifier (SVC), Random Forest, Naive Bayes, and K-Neighbors Classifier, using scikit-learn.

Each model's performance is assessed using a confusion matrix and classification report for precision, recall, and F1 scores; accuracy scores are stored for comparison and then we select the best model to use, ours being Naive Bayes.

```
The best performing model is: Naive Bayes with an accuracy of 0.73
```

Step 3: Training and Validation

```
Fitting 5 folds for each of 3 candidates, totalling 15 fits
Best Grid Search Parameters: {'alpha': 1.0}
Best Grid Search Score: 0.707857142857143
Grid Search Test Accuracy: 0.7333333333333333

Confusion Matrix:
 [[399 201]
 [119 481]]

Classification Report:
              precision    recall  f1-score   support

           0       0.77      0.67      0.71       600
           1       0.71      0.80      0.75       600

    accuracy                           0.73      1200
   macro avg       0.74      0.73      0.73      1200
weighted avg       0.74      0.73      0.73      1200
```

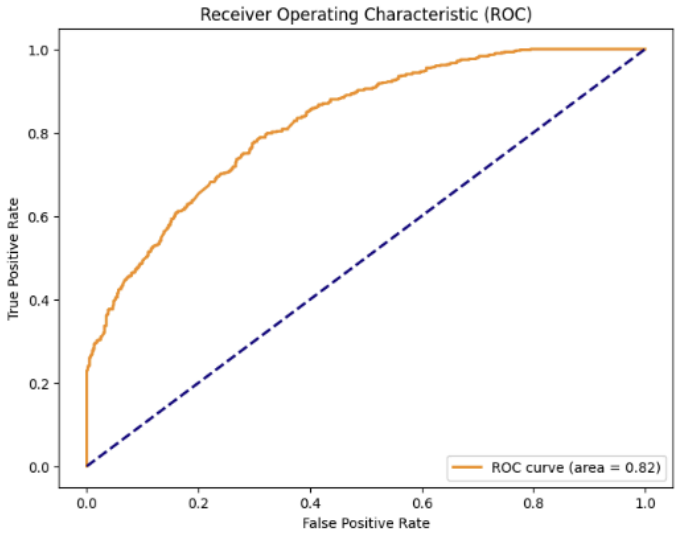
Hyperparameter Tuning: Using techniques such as grid search, we fine-tuned our best classifier Naive Bayes to determine optimal settings that maximized performance metrics on our dataset.

Step 4: Model Comparison and Final Selection

Evaluate Model Performance

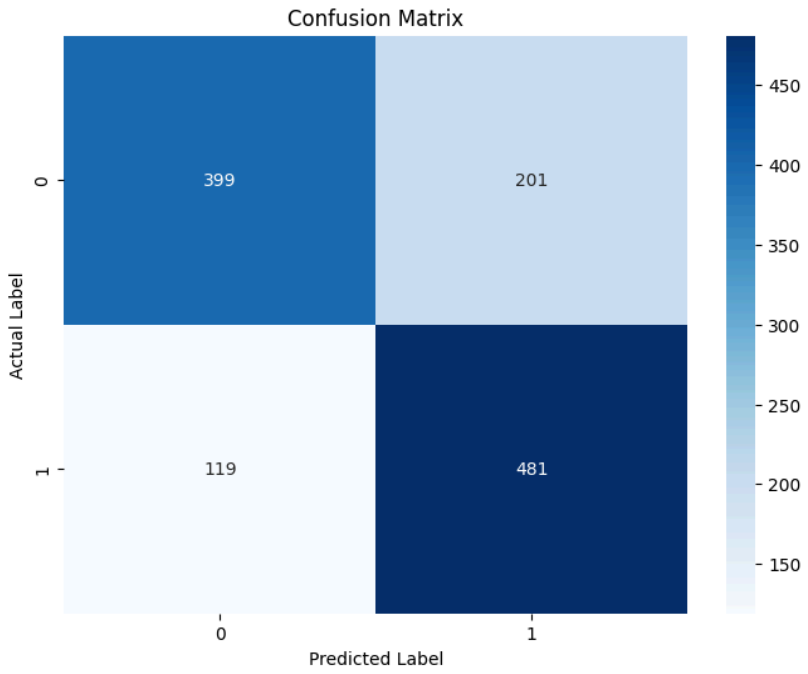
```
from sklearn.metrics import roc_curve, auc

# Predict probabilities for the test data
y_scores = best_model.predict_proba(X_test)[:, 1]

# Calculate ROC metrics
fpr, tpr, thresholds = roc_curve(y_test, y_scores)
roc_auc = auc(fpr, tpr)
```

```
# Start plotting
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC)')
plt.legend(loc="lower right")
plt.show()
```



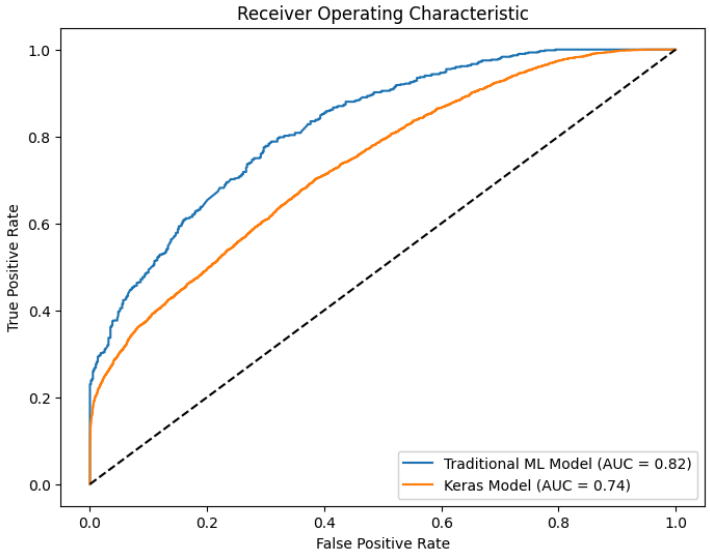
Evaluation Metrics: Each model's effectiveness was rigorously assessed using accuracy, precision, recall, and F1 scores. Additionally, confusion matrices were generated to visualize each model's performance in distinguishing between the subreddits.

Best Model Selection: The best-performing model was selected based on the aforementioned metrics. This model underwent further refinement and testing to confirm its efficacy and efficiency.



Conclusion: Integrating Advanced and Traditional Approaches

Both the advanced embedding-based approach provided by the Gemini API and the traditional machine learning classifiers yielded insightful results. The Gemini API's embeddings captured deep semantic relationships within the text, while traditional classifiers offered a well-understood and reliable framework for classification. By amalgamating insights from both methodologies, we not only enhanced the overall accuracy of our classification but also gained a holistic understanding of the available NLP tools. This dual approach ensured that the project leveraged both cutting-edge and established machine learning techniques, culminating in a robust analytical framework suitable for academic and practical applications.



Based on these performance matrices we then choose to move ahead with the traditional classifier of Naive Bayes.

Detailed Results and Conclusion from the Dual Subreddit Analysis

Overview

This comprehensive study aimed to dissect the digital dialogues of Rutgers and UPenn's subreddit communities, employing advanced NLP tools and machine learning techniques to explore how these communities articulate their university experiences. The analysis focused on accurately classifying the content of the posts and unraveling the prevalent themes, sentiments, and conversational patterns. Here we present a detailed summary of the findings and conclusions based on the empirical data collected and analyzed.

Results from the Study

Model Performance and Classification Results:

Gemini API Model:

The model achieved a classification accuracy of 71%, demonstrating robust capability despite some challenges with overlapping themes that caused misclassifications among subreddit posts. The model exhibited a class imbalance in predictions, favoring UPenn posts with 8,440 true positives compared to several Rutgers posts being misclassified as UPenn, highlighting an area for model improvement.

Traditional Machine Learning Models:

The Naive Bayes classifier emerged as the best-performing model using the TF-IDF vectorized data, surpassing other classifiers in precision and recall metrics. This model's success underscored the effectiveness of traditional statistical techniques in text classification over the embedding-based approach for this dataset.

Thematic Insights from N-grams and Word Clouds:

N-grams Analysis:

Unigram analysis showed dominant themes with "student," "campus," and "housing" frequently mentioned in Rutgers' subreddit and "Wharton," "research," and "career" in UPenn's discussions. Bigram and trigram analyses further delineated the focus areas with phrases like "financial aid" and "study abroad" popular among Rutgers students, whereas "research opportunities" and "business school" were prevalent in UPenn's subreddit.

Word Cloud Visualization:

Rutgers' word cloud featured terms related to campus life and academic logistics, while UPenn's highlighted advanced studies and professional development, aligning with the respective academic cultures and priorities of the universities.

Temporal Posting Patterns and Engagement Analysis:

Both subreddits showed a cyclical nature in posting frequency, aligned with the academic calendar. Peaks in posting occurred at the beginning of terms and during exam periods, indicating times of high student activity and engagement.

A detailed look at the daily and weekly post distributions revealed Rutgers' subreddit to be more active during daytime hours, whereas UPenn's community showed consistent activity extending into the night.

Sentiment Analysis and Community Engagement:

The sentiment analysis revealed that both subreddits experienced shifts in sentiment that correlated with academic stressors such as exams and admissions periods, with negative sentiments peaking during exams.

Community engagement, as measured by post scores and lengths, showed that longer posts did not necessarily correlate with higher engagement, challenging the notion that more detailed posts yield greater interaction.

Conclusions

Digital Platforms as Extensions of Campus Life: The results affirm that Reddit subreddits for Rutgers and UPenn function effectively as digital continuations of their campus lives, providing platforms for support, advice, and community interaction.

Implications for University Administration: The insights into posting patterns and sentiments related to the academic calendar can help university administrators better understand when students are most in need of support and how they might tailor communications and services to enhance student engagement and well-being.

Future Research Directions: The study highlights the potential for employing more sophisticated machine learning models, such as deep learning, to enhance classification accuracy and sentiment analysis. Further research could also explore predictive analytics to forecast community engagement trends and sentiment shifts, potentially informing proactive academic support strategies.

In essence, this research not only showcased the distinct digital cultures of Rutgers and UPenn but also demonstrated how such platforms reflect and influence the broader university

experience. The findings provide actionable insights that can help tailor university services to better meet student needs and enhance their academic and social experiences.