A Comparative Analysis of Advertising in the 2020 Presidential Elections & Phoenix Mayoral Elections using Natural Language Processing

Jun Wang, Jessica King, Likhith Sri Vatsa Uppada, Yogananda Reddy Murikinati

Arizona State University

Natural Language Processing

https://doi.org/10.32388/X0E9AF

Abstract

This language model examines advertisements used for political campaigns, specifically the 2023 Phoenix mayoral election. Political microtargeting looks at the tone of content throughout Instagram and Facebook ads during the 8 months leading up to the election. Through various processing techniques, both written and spoken language within ads was collected and used Facebook ad API, associated with a target demographic or group within the platform. This allows campaigns to send targeted messages to voters based on factors like location, media preference, associated political beliefs, age, and other critical variables.

This data was then compared to a peer-reviewed study by the University of Amsterdam, that studied Facebook and Instagram ads 8 months before the 2020 US Presidential election. Media companies often offer paid advertisements to a targeted audience, and during highly polarized world events such as an election, uncivil negative ads can be pushed to groups to which the shown ideology appeals. (Votta, F.)

The question posed by Votta in the comparative study was:

"To what extent is political microtargeting used by political advertisers to deliver toxic campaign messages?"

To study whether this optimization was occurring, the advertisements were run through a multilevel ordinal regression(Votta et al., 2023). They took ads from official political campaigns and compared them to third-party groups. They then analyzed the size of the desired audience and compared the size of the audience to the verbal tone of the listed advertisement.

Our results indicate that while traditional media remains an important part of electoral voting, show out, and postage signs and campaign events serve a purpose in reaching a wide,

unrestricted audience, social media advertising presents a valuable resource in targeting specific groups of individuals who will be most susceptible to guided advertisement. Last, we analyze the implications of our findings in the 2020 Phoenix mayoral election, how city elections differ from a presidential election, and how micro-targeted advertisements might affect voter behavior.

Introduction

In the project, we find the question "Compared to the 2020 US presidential campaign, how extensively is microtargeting used for political advertisement in the city of Phoenix 2020 mayoral election?". Using the NLP model, we try to find the solution to answer this question.

Through our model, there is a number to show the accuracy from 0% to 100%. In this range, it shows how extensively microtargeting is used for political advertisement in the city of Phoenix's 2020 mayoral election. Different percentages can represent different extents. For example, 0 to 30, 30 to 60, and 60 to 100 percent can be set as voters' weak, medium, and extensive levels. "Spreading uncivil negative campaign messages is a "high-risk, high reward" campaign strategy since certain voters are more likely to be swayed by negative messaging whereas other voters are more inclined to feel sympathy with the attack (Votta et al., 2023)". With advertising being dependent on monetary resources in most cases, it is important to understand how much money is being spent on advertising and the nature of this advertising.

Related work

In the project, the paper “Going Micro to Go Negative? Targeting Toxicity using Facebook and Instagram Ads” has been used for us. This paper introduces the basic idea of political communication, microtargeting, negative campaigning, toxicity, and ad library. And how they do work during the period of the campaign. Moreover, the paper gives the link to the Facebook Ad library API. Through this API, we can find the dataset to analyze and train our model. In the dataset, it exists several variables such as ad creation time, ad delivery start time, ad delivery stop time, byline, ad creative bodies, ad creative link titles, ad creative link captions, ad creative link descriptions, impressions, spend, demographic distribution, delivery by region, publisher platforms, and estimated audience size. Through analysis of the ad creative bodies, ad creative link titles, and creative link descriptions, we can find out how toxic messages are from the description of the Ad.

Using NLP model for analysis

NLP techniques are the best tools for answering the question. In the NLP techniques, the message needs to be processed such as grammar check, word stemming, remove stop words. Also, the most important thing is that the NLP model can use some mature algorithms such as decision trees, SVM, ANN, and RNN. It will help us build the best model to predict how extensive the effect of political microtargeting.

Identify the textual data.

Our dataset comes from the Facebook library API. Facebook uses its data from advertisements to support our question. Facebook library API provides reliable data sources from year to year, and every election campaign. Although the dataset still has some empty values or outlines, we still can remove them with data clean technology. Based on these, we think the dataset can support and answer our question very well.

Process and Outcome

This project is a comparative analysis of the usage of microtargeting for political advertisement in the 2020 US presidential campaign and the 2020 City of Phoenix Mayoral Elections. We intended to recreate the model used by the original used in the study that studied political microtargeting in the 2020 US presidential campaign (Votta et al., 2023). After the model was created, we used the advertising data collected from the 2020 City of Phoenix Mayoral Elections collected from the Facebook Ad library API.

Data Acquisition

Data Source:

The Data Source for this project is the Facebook Ad library API, the same source researchers have primarily used to collect their dataset. The researchers have also used financial sponsorships different contestants receive and the different advertisers' information for data analysis. For many reasons, like city elections focusing on a small group of people or low advertising budget, the Mayoral elections in the city of Phoenix have very minimal information available related to how much money was spent on advertising. However, this narrowed down the different sources of political advertising. The data we have collected consists of all the textual information about the advertising campaigns, the timeline the sponsors have used for a certain type of advertising, impressions count, estimated audience, and demographic information. This can be used to understand which type of messaging reflects what type of audience.

Relation to NLP:

In politics, smear campaigns play a critical role. This is the basis of toxic advertising. If money can be spent on promoting a candidate, it can also be used to tarnish their reputation. Sometimes advertising can be used to manipulate the facts to fit a narrative that creates hatred or fear among certain communities. So, it is important to understand the kind of advertising being done before an election to analyze the sentiments of voters.

The nature of advertising will change based on the demographics of the region, if we can define an objective metric to measure the toxicity, we can use this model to understand the nature of advertising being done. This requires a deeper understanding of the ways spoken language is used to veil toxic messages. In other terms, sentiment analysis of the political campaign advertisements is being performed to compare the toxicity of advertisements.

For Natural Language Models to be well-trained, one must capture the multiple layers of linguistic understanding. To develop this understanding of language, iterative training of models is required. In this case, the model must capture toxicity. To understand veiled toxic messaging, the model requires training in an iterative process and with a huge dataset. A reason for our case to not require an iterative approach is that the model has a small dataset. This will lead to the model being improperly trained and could be overfitted.

Data Cleaning

Based on our scenario, the NLP dataset comes from the Facebook Ad library API. This API allows us to use custom keywords to search for ads stored in the Ad Library. So, the data on all social issues, elections, or political ads can be searched, regardless of whether the ad is live or stopped (Fabio Votta, Arman Noroozian, Tom Dobber, Natali Helberger, Claes de Vreese, 2023).

For data sensitivity, the three main things need to be considered such as data privacy, business secrets, and government data privacy.

Data privacy

If the data includes personally identifiable information, social security numbers, bank account numbers, health records, etc. When dealing with this kind of data, the relevant laws, regulations, and privacy policies need to be considered to ensure personal data security and confidentiality (Anna Rogers, Tim Baldwin, Kobi Leins, 2021).

Business secrets

Companies or other organizations may have commercially confidential data, such as business plans, customer lists, research, and development results, etc. This data is critical to the competitiveness of businesses and measures need to be taken to prevent unauthorized access, disclosure, or misuse (Anna Rogers, Tim Baldwin, Kobi Leins, 2021).

Government data

Government agencies may process sensitive data including national security, law enforcement, and intelligence. The security and confidentiality of this data are critical to the security and stability of the nation (Anna Rogers, Tim Baldwin, Kobi Leins, 2021).

Data Security

Data classification and labeling

Classify and label sensitive data and clarify its sensitivity level and access rights so that it can be identified and managed in subsequent processing and storage.

Access Control

Restrict access to sensitive data to only authorized personnel. This can be achieved through means such as authentication, authorization, and rights management to ensure that only authorized users can access and process sensitive data (Gitanjali Gupta & Kamlesh Lakhwani, 2022).

Data encryption

Encrypts sensitive data to protect the security of data during transmission and storage. Methods such as symmetric encryption or asymmetric encryption can be used to ensure that data cannot be encrypted (Gitanjali Gupta & Kamlesh Lakhwani, 2022).

Data backup and recovery

Regularly back up sensitive data and establish a reliable data recovery mechanism to prevent data loss or damage and ensure data reliability and integrity (Gitanjali Gupta & Kamlesh Lakhwani, 2022).

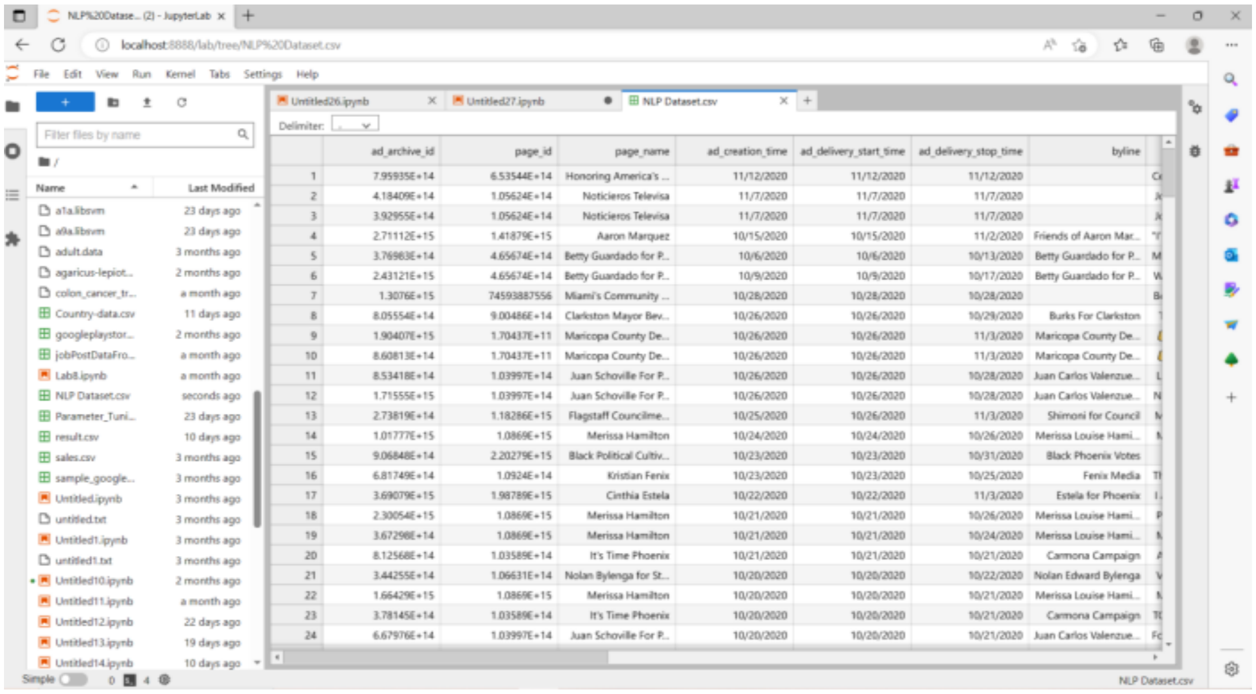
Process and Methodology

To clean the dataset, several things need to be considered. First, remove the irrelevant columns (Data reduction) from our dataset such as ID, currency type, and languages. Second, remove the rows that include null values. Third, some texts need to be transferred to numbers. Data Preparation is the process of transforming data from its raw format to the input format required by the data mining libraries. Data quality issues that might exist within the data such as noise or missing values (Asmaa Elbadrawy, 2023).

If the null values are negligible or do not significantly impact the overall dataset, they can be simply removed from the dataset (Asmaa Elbadrawy, 2023). Irrelevant columns are those that do not contain meaningful information for the intended analysis or purpose of the dataset. Removing irrelevant columns can help reduce the size and complexity of the dataset, simplify data analysis, and improve data processing efficiency (Asmaa Elbadrawy, 2023). Noise and outlier can be detected by using some techniques such as machine learning algorithms that can identify data points that fall outside of expected ranges, have extreme values, or exhibit abnormal patterns (Asmaa Elbadrawy, 2023).

For the data cleaning tools, the anaconda navigator was used for data cleaning tasks. In the Jupyter Notebook, the dataset can be cleaned with Python code and command.

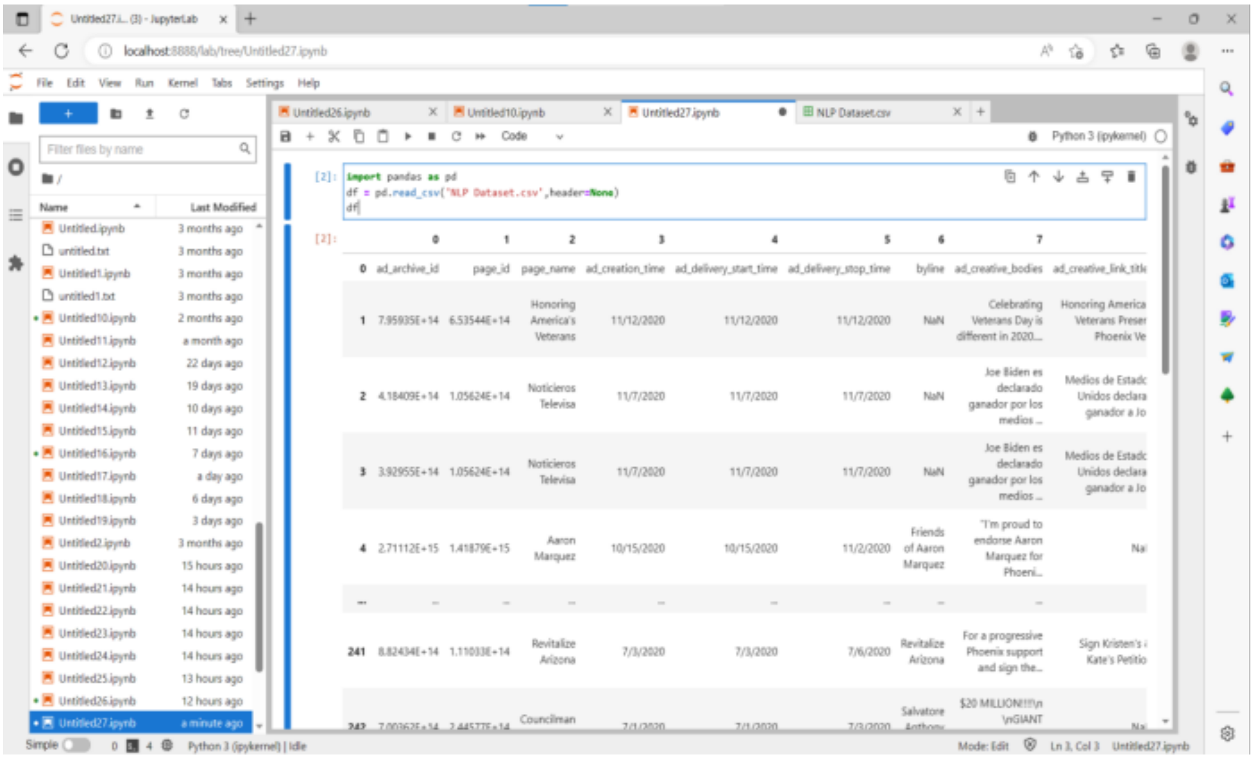
Upload the dataset:


NLP%20Datase... (2) - JupyterLab
localhost:8888/lab/tree/NLP%20Dataset.csv
File Edit View Run Kernel Tabs Settings Help
Filter files by name
Untitled26.ipynb
Untitled27.ipynb
NLP Dataset.csv
Delimiter:
Simple
NLP Dataset.csv

Name	Last Modified
a1a.libsvm	23 days ago
a9a.libsvm	23 days ago
adult.data	3 months ago
agaricus-lepiot...	2 months ago
colon_cancer_tr...	a month ago
Country-data.csv	11 days ago
googleplaystor...	2 months ago
jobPostDataFro...	a month ago
Lab8.ipynb	a month ago
NLP Dataset.csv	seconds ago
Parameter_Tuni...	23 days ago
result.csv	10 days ago
sales.csv	3 months ago
sample_google...	3 months ago
Untitled.ipynb	3 months ago
untitled.txt	3 months ago
Untitled1.ipynb	3 months ago
untitled1.txt	3 months ago
Untitled10.ipynb	2 months ago
Untitled11.ipynb	a month ago
Untitled12.ipynb	22 days ago
Untitled13.ipynb	19 days ago
Untitled14.ipynb	10 days ago

	ad_archive_id	page_id	page_name	ad_creation_time	ad_delivery_start_time	ad_delivery_stop_time	byline
1	7.95935E+14	6.53544E+14	Honoring America's ...	11/12/2020	11/12/2020	11/12/2020	
2	4.18409E+14	1.05624E+14	Noticieros Televisa	11/7/2020	11/7/2020	11/7/2020	
3	3.92955E+14	1.05624E+14	Noticieros Televisa	11/7/2020	11/7/2020	11/7/2020	
4	2.71112E+15	1.41879E+15	Aaron Marquez	10/15/2020	10/15/2020	11/2/2020	Friends of Aaron Mar...
5	3.76983E+14	4.65674E+14	Betty Guardado for P...	10/6/2020	10/6/2020	10/13/2020	Betty Guardado for P...
6	2.43121E+15	4.65674E+14	Betty Guardado for P...	10/9/2020	10/9/2020	10/17/2020	Betty Guardado for P...
7	1.3076E+15	74593887556	Miami's Community ...	10/28/2020	10/28/2020	10/28/2020	
8	8.05554E+14	9.00486E+14	Clarkston Mayor Bev...	10/26/2020	10/26/2020	10/29/2020	Burks For Clarkston
9	1.90407E+15	1.70437E+11	Maricopa County De...	10/26/2020	10/26/2020	11/3/2020	Maricopa County De...
10	8.60813E+14	1.70437E+11	Maricopa County De...	10/26/2020	10/26/2020	11/3/2020	Maricopa County De...
11	8.53418E+14	1.03997E+14	Juan Schoville For P...	10/26/2020	10/26/2020	10/28/2020	Juan Carlos Valenzue...
12	1.71555E+15	1.03997E+14	Juan Schoville For P...	10/26/2020	10/26/2020	10/28/2020	Juan Carlos Valenzue...
13	2.73819E+14	1.18286E+15	Flagstaff Councilme...	10/25/2020	10/26/2020	11/3/2020	Shimoni for Council
14	1.01777E+15	1.0869E+15	Merissa Hamilton	10/24/2020	10/24/2020	10/26/2020	Merissa Louise Hami...
15	9.06848E+14	2.20279E+15	Black Political Cultiv...	10/23/2020	10/23/2020	10/31/2020	Black Phoenix Votes
16	6.81749E+14	1.0924E+14	Kristian Fenix	10/23/2020	10/23/2020	10/25/2020	Fenix Media
17	3.69079E+15	1.98789E+15	Cinthia Estela	10/22/2020	10/22/2020	11/3/2020	Estela for Phoenix
18	2.30054E+15	1.0869E+15	Merissa Hamilton	10/21/2020	10/21/2020	10/26/2020	Merissa Louise Hami...
19	3.67298E+14	1.0869E+15	Merissa Hamilton	10/21/2020	10/21/2020	10/24/2020	Merissa Louise Hami...
20	8.12568E+14	1.03589E+14	It's Time Phoenix	10/21/2020	10/21/2020	10/21/2020	Carmona Campaign
21	3.44255E+14	1.06631E+14	Nolan Bylenga for St...	10/20/2020	10/20/2020	10/22/2020	Nolan Edward Bylenga
22	1.66429E+15	1.0869E+15	Merissa Hamilton	10/20/2020	10/20/2020	10/21/2020	Merissa Louise Hami...
23	3.78145E+14	1.03589E+14	It's Time Phoenix	10/20/2020	10/20/2020	10/21/2020	Carmona Campaign
24	6.67976E+14	1.03997E+14	Juan Schoville For P...	10/20/2020	10/20/2020	10/21/2020	Juan Carlos Valenzue...

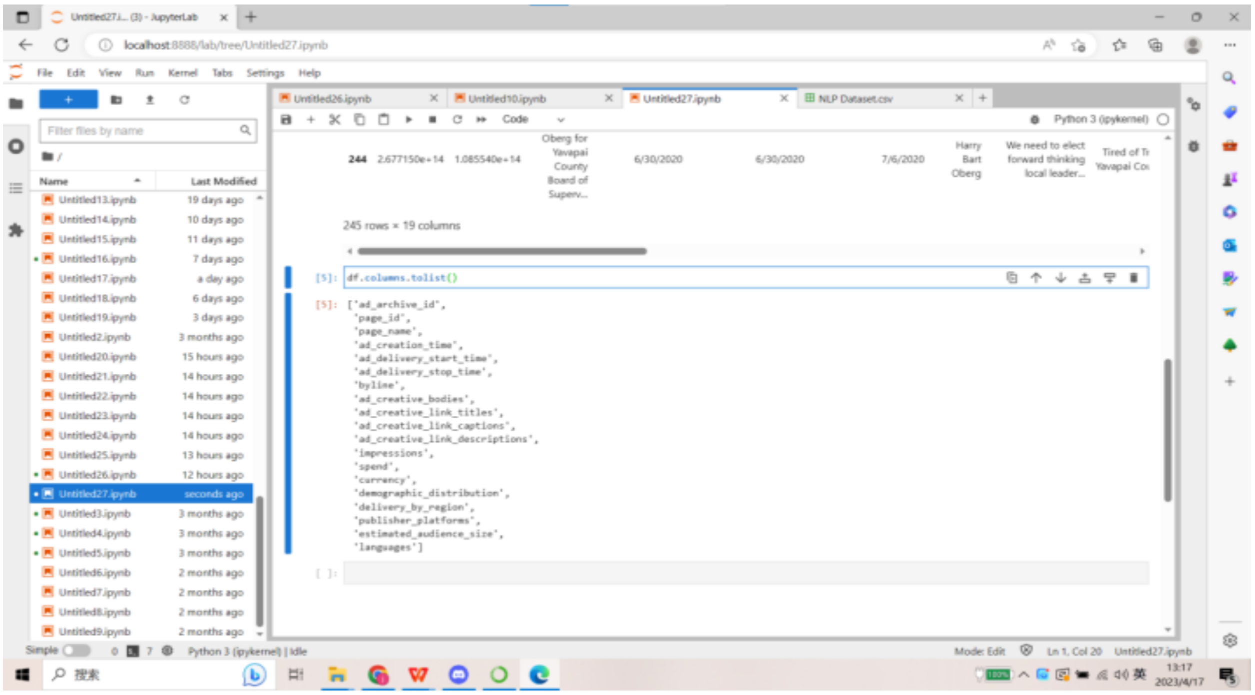
Display the dataset using Python:



```
[2]: import pandas as pd
     df = pd.read_csv('NLP Dataset.csv',header=None)
     df
```

```
[2]:          0            1            2                 3                       4                      5           6                     7
     0   ad_archive_id  page_id     page_name   ad_creation_time  ad_delivery_start_time  ad_delivery_stop_time  byline  ad_creative_bodies   ad_creative_link_title
     1   7.95935E+14  6.53544E+14  Honoring        11/12/2020            11/12/2020             11/12/2020     NaN    Celebrating          Honoring America
                                   America's                                                                          Veterans Day is      Veterans Preser
                                   Veterans                                                                           different in 2020....  Phoenix Ve
     2   4.18409E+14  1.05624E+14  Noticieros      11/7/2020             11/7/2020              11/7/2020     NaN    Joe Biden es         Medios de Estadc
                                   Televisa                                                                           declarado            Unidos declara
                                                                                                                      ganador por los      ganador a Jo
                                                                                                                      medios ...
     3   3.92955E+14  1.05624E+14  Noticieros      11/7/2020             11/7/2020              11/7/2020     NaN    Joe Biden es         Medios de Estadc
                                   Televisa                                                                           declarado            Unidos declara
                                                                                                                      ganador por los      ganador a Jo
                                                                                                                      medios ...
     4   2.71112E+15  1.41879E+15  Aaron           10/15/2020            10/15/2020             11/2/2020    Friends  "I'm proud to               Na
                                   Marquez                                                                   of Aaron endorse Aaron
                                                                                                             Marquez  Marquez for
                                                                                                                      Phoeni...
    ...      ...          ...          ...            ...                   ...                    ...        ...       ...
   241   8.82434E+14  1.11033E+14  Revitalize      7/3/2020              7/3/2020               7/6/2020    Revitalize For a progressive  Sign Kristen's
                                   Arizona                                                                   Arizona  Phoenix support      Kate's Petitio
                                                                                                                      and sign the...
   242   7.00362E+14  2.44577E+14  Councilman      7/1/2020              7/1/2020               7/3/2020    Salvatore $20 MILLION!!!\n        Na
                                                                                                             Anthony  \nGIANT
```

Display all columns:

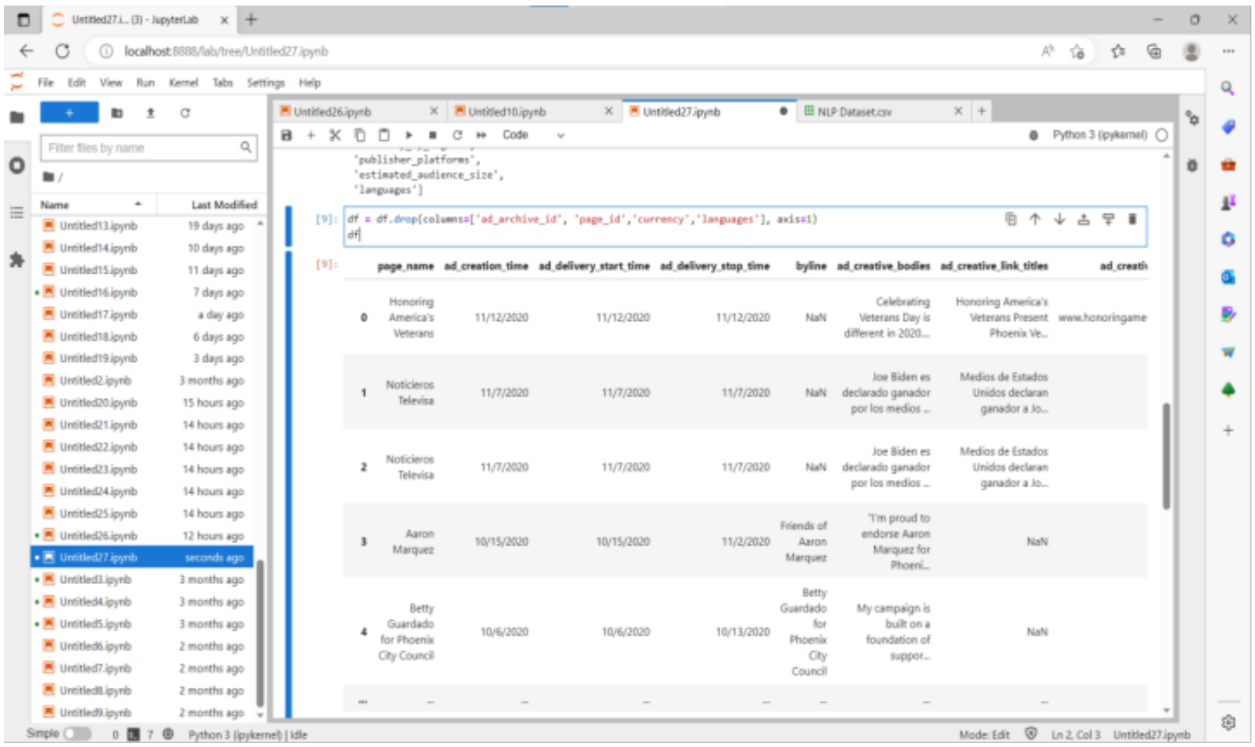


```
   244  2.677150e+14  1.085540e+14  Oberg for      6/30/2020        6/30/2020        7/6/2020   Harry    We need to elect    Tired of Tr
                                    Yavapai                                                    Bart     forward thinking    Yavapai Co
                                    County                                                     Oberg    local leader...
                                    Board of
                                    Superv...

   245 rows × 19 columns
```

```
[5]: df.columns.tolist()
```

```
[5]: ['ad_archive_id',
      'page_id',
      'page_name',
      'ad_creation_time',
      'ad_delivery_start_time',
      'ad_delivery_stop_time',
      'byline',
      'ad_creative_bodies',
      'ad_creative_link_titles',
      'ad_creative_link_captions',
      'ad_creative_link_descriptions',
      'impressions',
      'spend',
      'currency',
      'demographic_distribution',
      'delivery_by_region',
      'publisher_platforms',
      'estimated_audience_size',
      'languages']
```

Delete not useful columns such as 'ad_archive_id', 'page_id','currency','languages'



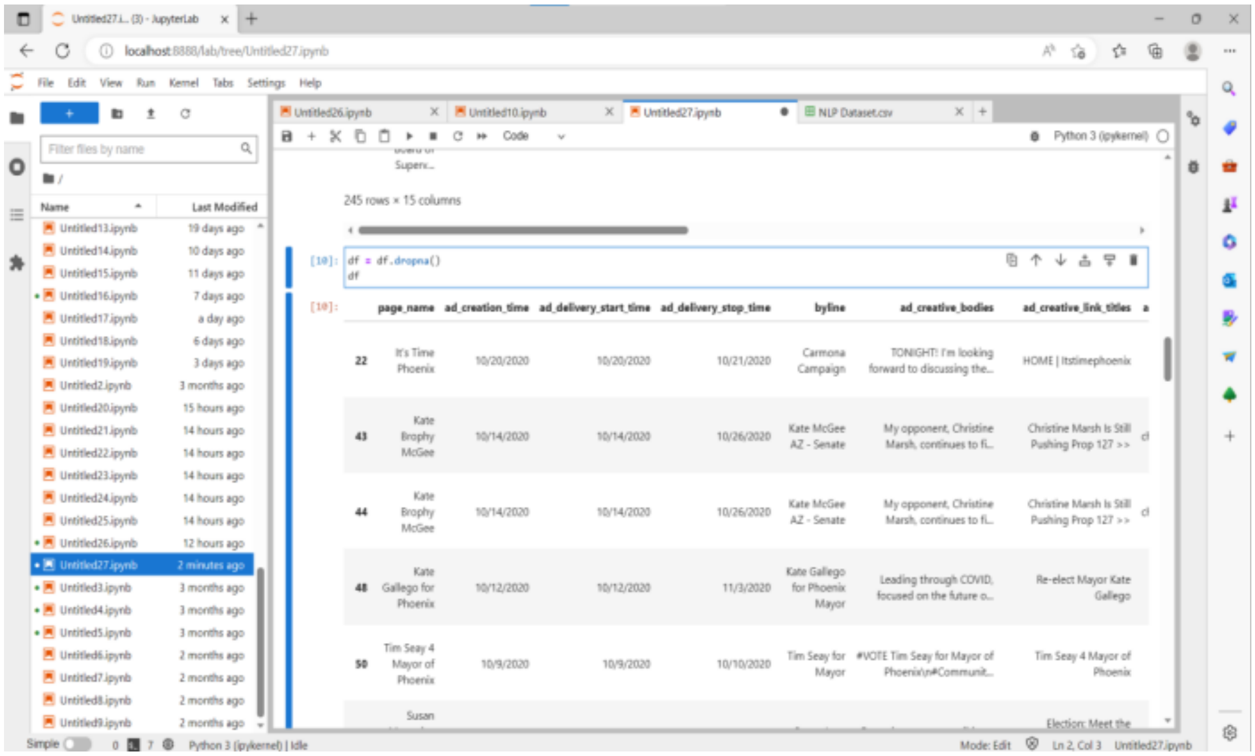
```
 'publisher_platforms',
 'estimated_audience_size',
 'languages']
```

```
[9]: df = df.drop(columns=['ad_archive_id', 'page_id','currency','languages'], axis=1)
     df
```

[9]:

	page_name	ad_creation_time	ad_delivery_start_time	ad_delivery_stop_time	byline	ad_creative_bodies	ad_creative_link_titles	ad_creati
0	Honoring America's Veterans	11/12/2020	11/12/2020	11/12/2020	NaN	Celebrating Veterans Day is different in 2020...	Honoring America's Veterans Present Phoenix Ve...	www.honoringame
1	Noticieros Televisa	11/7/2020	11/7/2020	11/7/2020	NaN	Joe Biden es declarado ganador por los medios ...	Medios de Estados Unidos declaran ganador a Jo...	
2	Noticieros Televisa	11/7/2020	11/7/2020	11/7/2020	NaN	Joe Biden es declarado ganador por los medios ...	Medios de Estados Unidos declaran ganador a Jo...	
3	Aaron Marquez	10/15/2020	10/15/2020	11/2/2020	Friends of Aaron Marquez	"I'm proud to endorse Aaron Marquez for Phoeni...	NaN	
4	Betty Guardado for Phoenix City Council	10/6/2020	10/6/2020	10/13/2020	Betty Guardado for Phoenix City Council	My campaign is built on a foundation of suppor...	NaN	
...	...	...	...	...	...	...	...	

Remove null value:



```
Superv...

245 rows × 15 columns
```

```
[10]: df = df.dropna()
      df
```

[10]:

	page_name	ad_creation_time	ad_delivery_start_time	ad_delivery_stop_time	byline	ad_creative_bodies	ad_creative_link_titles	a
22	It's Time Phoenix	10/20/2020	10/20/2020	10/21/2020	Carmona Campaign	TONIGHT: I'm looking forward to discussing the...	HOME | Itstimephoenix	
43	Kate Brophy McGee	10/14/2020	10/14/2020	10/26/2020	Kate McGee AZ - Senate	My opponent, Christine Marsh, continues to fi...	Christine Marsh Is Still Pushing Prop 127 >>	cl
44	Kate Brophy McGee	10/14/2020	10/14/2020	10/26/2020	Kate McGee AZ - Senate	My opponent, Christine Marsh, continues to fi...	Christine Marsh Is Still Pushing Prop 127 >>	cl
48	Kate Gallego for Phoenix	10/12/2020	10/12/2020	11/3/2020	Kate Gallego for Phoenix Mayor	Leading through COVID, focused on the future o...	Re-elect Mayor Kate Gallego	
50	Tim Seay 4 Mayor of Phoenix	10/9/2020	10/9/2020	10/10/2020	Tim Seay for Mayor	#VOTE Tim Seay for Mayor of Phoenix\n#Communit...	Tim Seay 4 Mayor of Phoenix	
	Susan						Election: Meet the	

The Stoplist:

For improving the efficiency of the NLP process, a stoplist needs to be used to remove some words from our dataset. These words are not helpful, at the same time, they take a lot of CPU and memory resources. As a result, we will create a stoplist for our dataset. For example, "a"、"an"、"the"、"in"、"is"、"of"、"and" will be put into the stoplist. When we do this, the accuracy of searching will be improved efficiently. As a result, it also can reduce the noise and extract more helpful text. When we need to find the stoplist, appendix 2 can be used for our data cleaning. It is a very general stoplist. Based on our scenario, we still can add more stop words depending on the growth of our scenario (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Unique Jargon

Unique Jargon is a challenge for NLP, due to NLP dealing with general language and grammar is much better. Some unique Jargon cannot be marked as general terms. Hence, we need to find some solutions to deal with it. To solve these problems, Term Extraction, Term Dictionary, Domain-specific Word embedding, Custom Rules, and Manual Annotation can be used for unique jargon in NLP (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Acronyms

Acronyms pose a significant trouble for NLP. To solve these problems, we can use Acronym Expansion, Contextual Disambiguation, Word embedding, and Manual Annotation. When these strategies have been used. It will help the machine to better understand the meaning of words(Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Any other "noise" that might prove problematic for your process.

The stop words such as conjunctions, prepositions, and pronouns can be seen as noise. Also, spelling errors or grammar errors can also be seen as noise.

Spelling errors

The text may contain spelling errors, which may cause the model to misidentify and interpret words. Processing methods can include spelling correction, such as using automatic spell checkers, error correction models, or rule-based spelling correction methods, to fix spelling errors in the text (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Grammatical errors

The text may contain grammatical errors, such as wrong grammatical structures, wrong word forms, etc. Processing methods can include the use of syntax analysis tools or rule-based syntax error correction methods to identify and fix grammatical errors in the text (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

In NLP tasks, noise can negatively affect a model, as it can introduce noise that makes it difficult for the model to understand and process text correctly. For example, in a text classification task, noise may include missing words, wrong punctuation, or other text formatting errors, which can cause the model to make incorrect predictions (Hobson Lane, Cole Howard, Hannes Hapke, 2019). To tackle these issues some solutions such as Data Cleaning, Text preprocessing, and Error Correction can be employed.

Data cleaning

By cleaning the text data, unnecessary characters, punctuation marks, HTML tags, special characters, etc. are removed, hence, reducing the noise in the text data (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Text preprocessing

Preprocessing text data, including lexical analysis, stem extraction, stop word removal, spelling correction, etc., to normalize text and reduce the impact of noise (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Error Correction

Use automated error correction techniques, such as spell checking, grammatical error correction, etc., to identify and repair errors in text, thereby reducing the impact of noise on NLP models (Hobson Lane, Cole Howard, Hannes Hapke, 2019).

Data Modeling

In the project, we use two models such as SVM and RNN for training our dataset.

Support Vector Machine (SVM)

Support Vector Machine (SVM) is a very popular machine learning algorithm. It has been used for classification and regression. The main goal of SVM is to find a hyperplane that divides the data into two or more categories and maximizes the margin (Vikramaditya Jakkula, 2006).

Our dataset mainly deals with classification problems, SVM maps the input data into a high-dimensional space, making the data easier to classify. In high-dimensional space, SVM finds a hyperplane that maximizes the minimum distance between two different classes. This hyperplane can be expressed as a linear equation; some of the parameters are automatically learned from the data (Vikramaditya Jakkula, 2006).

For nonlinear problems, SVM can use kernel functions to deal with this situation. Kernel makes the data separable by mapping the input data into a higher dimensional. SVM can use a variety of kernel functions, such as linear, polynomial, and Gaussian kernels (Vikramaditya Jakkula, 2006).

Recurrent Neural Network (RNN)

Recurrent Neural Network (RNN) is a very useful model in Natural Language Processing (NLP). RNN is a type of neural network capable of processing sequence data, so it is very useful when processing text (Robert DiPietro, Gregory D. Hager, 2020).

In NLP, RNNs are commonly used for tasks such as text classification, sentiment analysis, language modeling, and machine translation. By using a recurrent structure, RNN can take the output of the previous time step as the input of the current time step, Hence, capturing the time dependence of the text sequence. The focus of RNN is memory, which can retain previous information internally and use this information in subsequent steps (Robert DiPietro, Gregory D. Hager, 2020).

When processing text, RNNs typically use word embeddings to represent words. Word embedding is a technology that maps words to a low-dimensional vector space, which can encode the semantic information of words into a vector form so that RNN can better understand the meaning of text (Robert DiPietro, Gregory D. Hager, 2020).

Another important RNN variant is the Long Short-Term Memory (LSTM). LSTM solves the vanishing gradient problem of traditional RNNs when processing long sequences by using a gating mechanism to control the flow of information. LSTM is very useful in NLP because it can handle long text sequences and has excellent performance in tasks such as language modeling and machine translation (Robert DiPietro, Gregory D. Hager, 2020).

Advantage of SVM:

Support for high-dimensional feature spaces.

NLP often needs to deal with high-dimensional feature spaces, and SVM can map features into high-dimensional spaces through kernel functions for classification, so it has a good ability to process high-dimensional features (Vikramaditya Jakkula, 2006).

Solve the problem of small samples.

It is common to have a small amount of data in NLP, and SVM can effectively avoid overfitting by controlling the regular parameters, and it has a better classification effect on small sample data (Vikramaditya Jakkula, 2006).

Good theoretical support

The theoretical basis of SVM is relatively rigorous, and the generalization ability of the model can be guaranteed through mathematical proof (Vikramaditya Jakkula, 2006).

Disadvantage of SVM:

Sensitive to noise

SVM is sensitive to noise, so some preprocessing of the data is required in NLP, such as removing stop words and punctuation marks (Vikramaditya Jakkula, 2006).

Difficult to handle large-scale data.

When the amount of data is very large, the training time of SVM will be relatively long and require a large amount of storage space, so it is not suitable for processing large-scale NLP data (Vikramaditya Jakkula, 2006).

Advantage of RNN:

Processing sequence data

RNN can process variable-length sequence data, such as text data, so it can be well applied to various tasks in NLP, such as text classification, sentiment analysis, machine translation, etc.(Robert DiPietro, Gregory D. Hager, 2020).

Capable of capturing context information: RNN can generate subsequent text content based on previous text content, so it can capture text context information well and improve the effect of text processing (Robert DiPietro, Gregory D. Hager, 2020).

Word embeddings can be used in conjunction to represent words and convert words into vector representations to better handle natural language (Robert DiPietro, Gregory D. Hager, 2020).

Disadvantage of RNN:

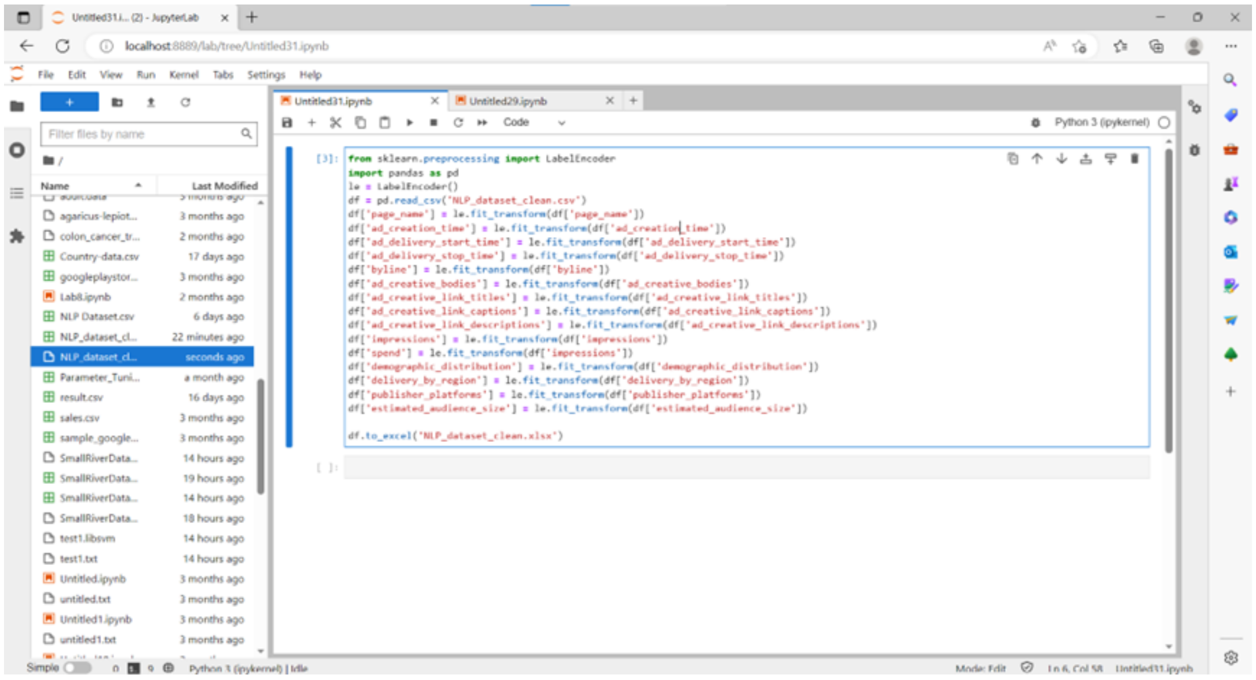
Difficulty in training

Training RNN requires backpropagation on the entire sequence, which makes the training time relatively long, and it is also prone to gradient disappearance or gradient explosion problems (Akshay Tondak, 2022).

Overfitting

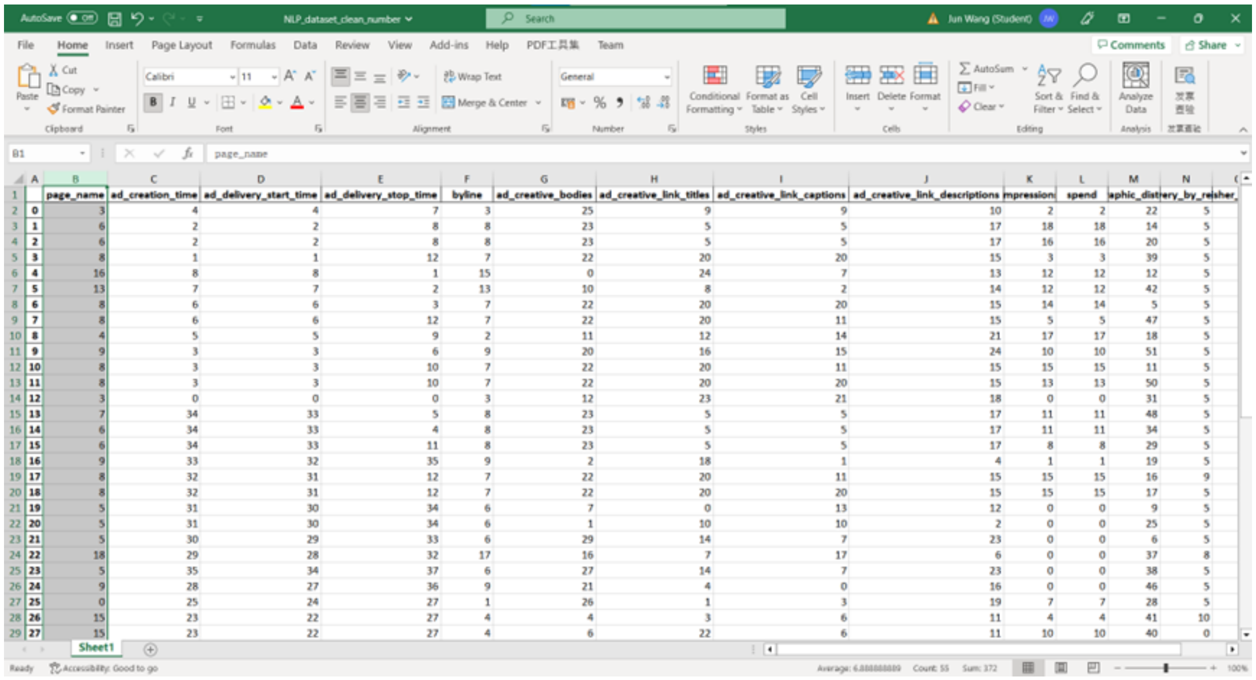
There are many parameters in RNN, so it is prone to overfitting problems (Akshay Tondak, 2022).

Transform our dataset from string to number:



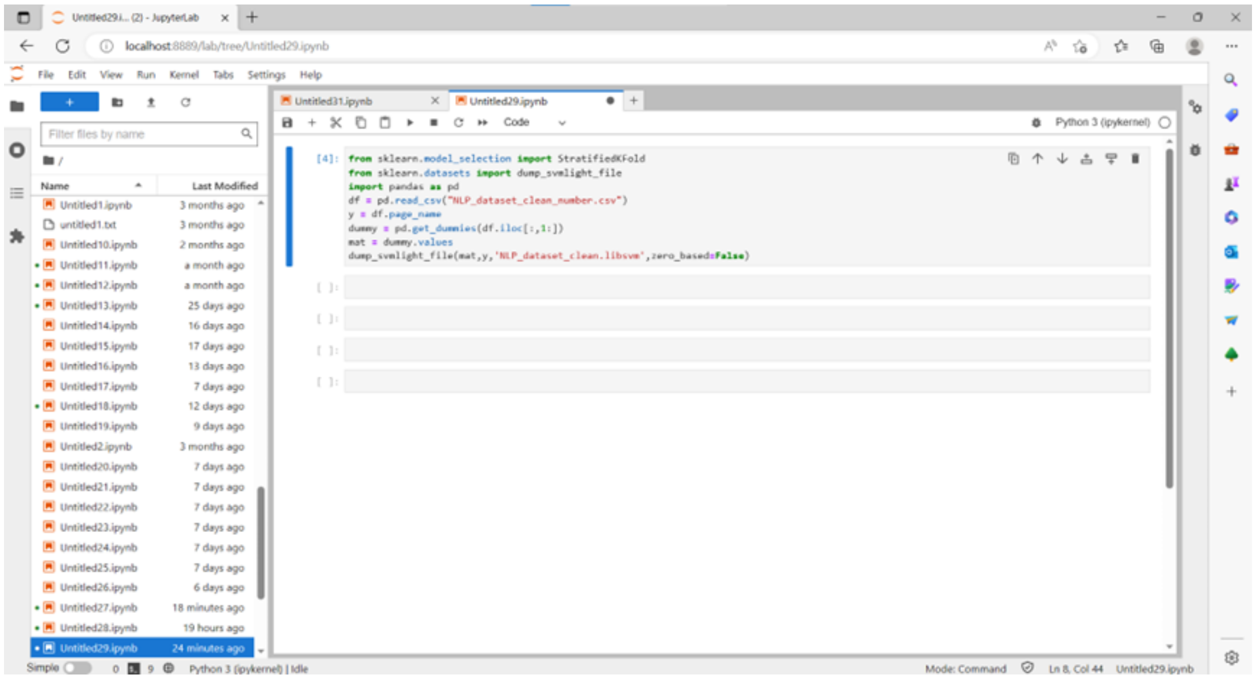
```
[3]: from sklearn.preprocessing import LabelEncoder
     import pandas as pd
     le = LabelEncoder()
     df = pd.read_csv('NLP_dataset_clean.csv')
     df['page_name'] = le.fit_transform(df['page_name'])
     df['ad_creation_time'] = le.fit_transform(df['ad_creation_time'])
     df['ad_delivery_start_time'] = le.fit_transform(df['ad_delivery_start_time'])
     df['ad_delivery_stop_time'] = le.fit_transform(df['ad_delivery_stop_time'])
     df['byline'] = le.fit_transform(df['byline'])
     df['ad_creative_bodies'] = le.fit_transform(df['ad_creative_bodies'])
     df['ad_creative_link_titles'] = le.fit_transform(df['ad_creative_link_titles'])
     df['ad_creative_link_captions'] = le.fit_transform(df['ad_creative_link_captions'])
     df['ad_creative_link_descriptions'] = le.fit_transform(df['ad_creative_link_descriptions'])
     df['impressions'] = le.fit_transform(df['impressions'])
     df['spend'] = le.fit_transform(df['impressions'])
     df['demographic_distribution'] = le.fit_transform(df['demographic_distribution'])
     df['delivery_by_region'] = le.fit_transform(df['delivery_by_region'])
     df['publisher_platforms'] = le.fit_transform(df['publisher_platforms'])
     df['estimated_audience_size'] = le.fit_transform(df['estimated_audience_size'])

     df.to_excel('NLP_dataset_clean.xlsx')

[ ]:
```

```
from sklearn.preprocessing import LabelEncoder
import pandas as pd
le = LabelEncoder()
df = pd.read_csv('NLP_dataset_clean.csv')
df['page_name'] = le.fit_transform(df['page_name'])
df['ad_creation_time'] = le.fit_transform(df['ad_creation_time'])
df['ad_delivery_start_time'] = le.fit_transform(df['ad_delivery_start_time'])
df['ad_delivery_stop_time'] = le.fit_transform(df['ad_delivery_stop_time'])
df['byline'] = le.fit_transform(df['byline'])
df['ad_creative_bodies'] = le.fit_transform(df['ad_creative_bodies'])
df['ad_creative_link_titles'] = le.fit_transform(df['ad_creative_link_titles'])
df['ad_creative_link_captions'] = le.fit_transform(df['ad_creative_link_captions'])
df['ad_creative_link_descriptions'] = le.fit_transform(df['ad_creative_link_descriptions'])
df['impressions'] = le.fit_transform(df['impressions'])
df['spend'] = le.fit_transform(df['impressions'])
df['demographic_distribution'] = le.fit_transform(df['demographic_distribution'])
df['delivery_by_region'] = le.fit_transform(df['delivery_by_region'])
df['publisher_platforms'] = le.fit_transform(df['publisher_platforms'])
df['estimated_audience_size'] = le.fit_transform(df['estimated_audience_size'])
df.to_excel('NLP_dataset_clean.xlsx')
```



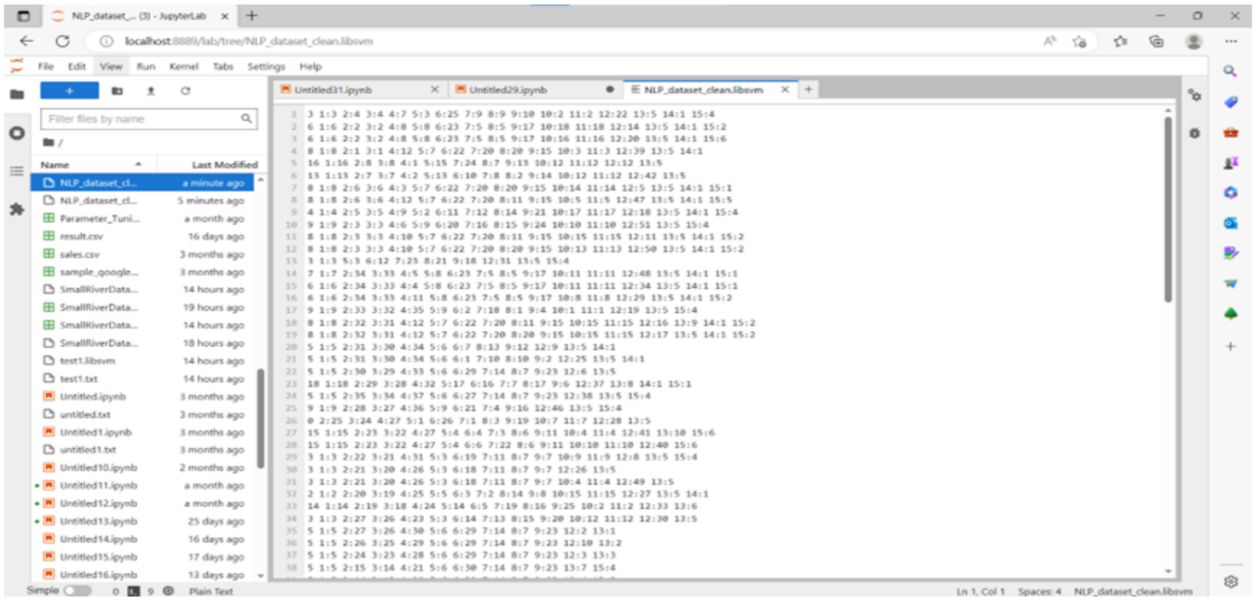
	page_name	ad_creation_time	ad_delivery_start_time	ad_delivery_stop_time	byline	ad_creative_bodies	ad_creative_link_titles	ad_creative_link_captions	ad_creative_link_descriptions	mpression	spend	aphic_dist	ery_by_re	isher_
0	3	4	4	7	3	25	9	9	10	2	2	22	5	
1	6	2	2	8	8	23	5	5	17	18	18	14	5	
2	6	2	2	8	8	23	5	5	17	16	16	20	5	
3	8	1	1	12	7	22	20	20	15	3	3	39	5	
4	16	8	8	1	15	0	24	7	13	12	12	12	5	
5	13	7	7	2	13	10	8	2	14	12	12	42	5	
6	8	6	6	3	7	22	20	20	15	14	14	5	5	
7	8	6	6	12	7	22	20	11	15	5	5	47	5	
8	4	5	5	9	2	11	12	14	21	17	17	18	5	
9	9	3	3	6	9	20	16	15	24	10	10	51	5	
10	8	3	3	10	7	22	20	11	15	15	15	11	5	
11	8	3	3	10	7	22	20	20	15	13	13	50	5	
12	3	0	0	0	3	12	23	21	18	0	0	31	5	
13	7	34	33	5	8	23	5	5	17	11	11	48	5	
14	6	34	33	4	8	23	5	5	17	11	11	34	5	
15	6	34	33	11	8	23	5	5	17	8	8	29	5	
16	9	33	32	35	9	2	18	1	4	1	1	19	5	
17	8	32	31	12	7	22	20	11	15	15	15	16	9	
18	8	32	31	12	7	22	20	20	15	15	15	17	5	
19	5	31	30	34	6	7	0	13	12	0	0	9	5	
20	5	31	30	34	6	1	10	10	2	0	0	25	5	
21	5	30	29	33	6	29	14	7	23	0	0	6	5	
22	18	29	28	32	17	16	7	17	6	0	0	37	8	
23	5	35	34	37	6	27	14	7	23	0	0	38	5	
24	9	28	27	36	9	21	4	0	16	0	0	46	5	
25	0	25	24	27	1	26	1	3	19	7	7	28	5	
26	15	23	22	27	4	4	3	6	11	4	4	41	10	
27	15	23	22	27	4	6	22	6	11	10	10	40	0	

Change to libsvm format:



```
[4]: from sklearn.model_selection import StratifiedKFold
     from sklearn.datasets import dump_svmlight_file
     import pandas as pd
     df = pd.read_csv("NLP_dataset_clean_number.csv")
     y = df.page_name
     dummy = pd.get_dummies(df.iloc[:,1:])
     mat = dummy.values
     dump_svmlight_file(mat,y,'NLP_dataset_clean.libsvm',zero_based=False)
```

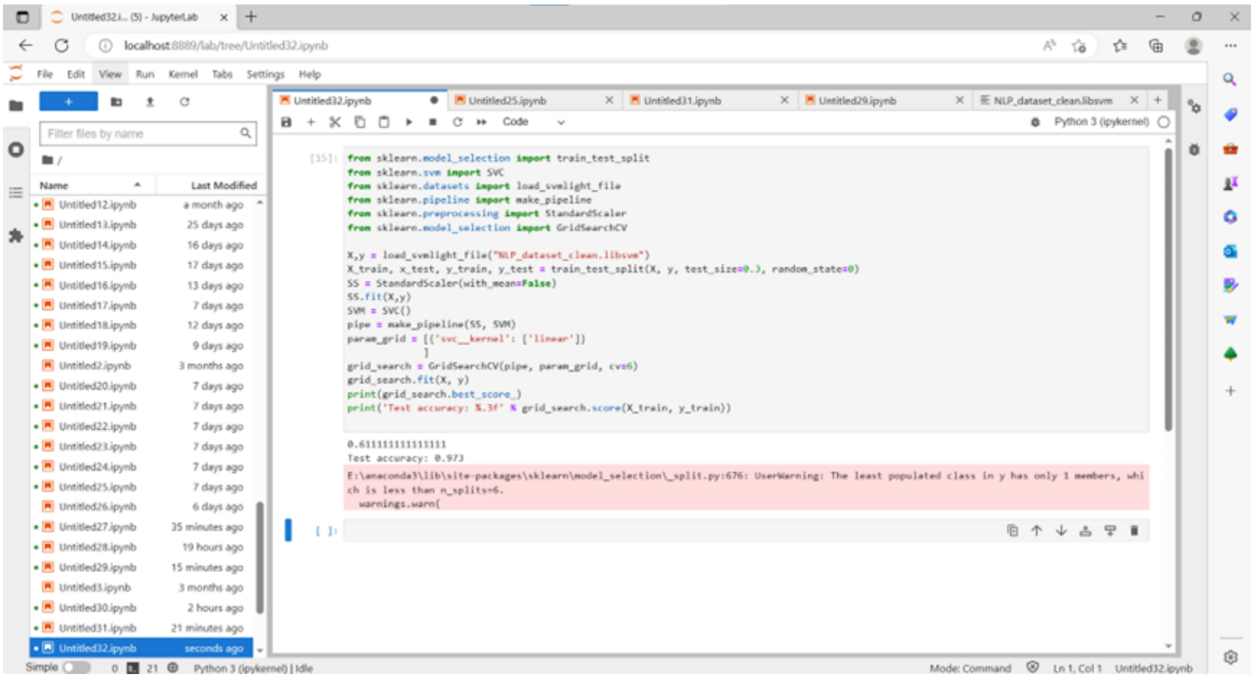
```
from sklearn.model_selection import StratifiedKFold

from sklearn.datasets import dump_svmlight_file

import pandas as pd

df = pd.read_csv("NLP_dataset_clean_number.csv")

y = df.page_name

dummy = pd.get_dummies(df.iloc[:,1:])

mat = dummy.values

dump_svmlight_file(mat,y,'NLP_dataset_clean.libsvm',zero_based=False)
```


NLP_dataset_... (3) - JupyterLab
localhost:8889/lab/tree/NLP_dataset_clean.libsvm
File Edit View Run Kernel Tabs Settings Help
Filter files by name
Untitled31.ipynb
Untitled29.ipynb
NLP_dataset_clean.libsvm
Simple 0 9 Plain Text
Ln 1, Col 1 Spaces: 4 NLP_dataset_clean.libsvm

Name	Last Modified
NLP_dataset_cl...	a minute ago
NLP_dataset_cl...	5 minutes ago
Parameter_Tuni...	a month ago
result.csv	16 days ago
sales.csv	3 months ago
sample_google...	3 months ago
SmallRiverData...	14 hours ago
SmallRiverData...	19 hours ago
SmallRiverData...	14 hours ago
SmallRiverData...	18 hours ago
test1.libsvm	14 hours ago
test1.txt	14 hours ago
Untitled.ipynb	3 months ago
untitled.txt	3 months ago
Untitled1.ipynb	3 months ago
untitled1.txt	3 months ago
Untitled10.ipynb	2 months ago
Untitled11.ipynb	a month ago
Untitled12.ipynb	a month ago
Untitled13.ipynb	25 days ago
Untitled14.ipynb	16 days ago
Untitled15.ipynb	17 days ago
Untitled16.ipynb	13 days ago

```
3 1:3 2:4 3:4 4:7 5:3 6:25 7:9 8:9 9:10 10:2 11:2 12:22 13:5 14:1 15:4
6 1:6 2:2 3:2 4:8 5:8 6:23 7:5 8:5 9:17 10:18 11:18 12:14 13:5 14:1 15:2
6 1:6 2:2 3:2 4:8 5:8 6:23 7:5 8:5 9:17 10:16 11:16 12:20 13:5 14:1 15:6
8 1:8 2:1 3:1 4:12 5:7 6:22 7:20 8:20 9:15 10:3 11:3 12:39 13:5 14:1
16 1:16 2:8 3:8 4:1 5:15 7:24 8:7 9:13 10:12 11:12 12:12 13:5
13 1:13 2:7 3:7 4:2 5:13 6:10 7:8 8:2 9:14 10:12 11:12 12:42 13:5
8 1:8 2:6 3:6 4:3 5:7 6:22 7:20 8:20 9:15 10:14 11:14 12:5 13:5 14:1 15:1
8 1:8 2:6 3:6 4:12 5:7 6:22 7:20 8:11 9:15 10:5 11:5 12:47 13:5 14:1 15:5
4 1:4 2:5 3:5 4:9 5:2 6:11 7:12 8:14 9:21 10:17 11:17 12:18 13:5 14:1 15:4
9 1:9 2:3 3:3 4:6 5:9 6:20 7:16 8:15 9:24 10:10 11:10 12:51 13:5 15:4
8 1:8 2:3 3:3 4:10 5:7 6:22 7:20 8:11 9:15 10:15 11:15 12:11 13:5 14:1 15:2
8 1:8 2:3 3:3 4:10 5:7 6:22 7:20 8:20 9:15 10:13 11:13 12:50 13:5 14:1 15:2
3 1:3 5:3 6:12 7:23 8:21 9:18 12:31 13:5 15:4
7 1:7 2:34 3:33 4:5 5:8 6:23 7:5 8:5 9:17 10:11 11:11 12:48 13:5 14:1 15:1
6 1:6 2:34 3:33 4:4 5:8 6:23 7:5 8:5 9:17 10:11 11:11 12:34 13:5 14:1 15:1
6 1:6 2:34 3:33 4:11 5:8 6:23 7:5 8:5 9:17 10:8 11:8 12:29 13:5 14:1 15:2
9 1:9 2:33 3:32 4:35 5:9 6:2 7:18 8:1 9:4 10:1 11:1 12:19 13:5 15:4
8 1:8 2:32 3:31 4:12 5:7 6:22 7:20 8:11 9:15 10:15 11:15 12:16 13:9 14:1 15:2
8 1:8 2:32 3:31 4:12 5:7 6:22 7:20 8:20 9:15 10:15 11:15 12:17 13:5 14:1 15:2
5 1:5 2:31 3:30 4:34 5:6 6:7 8:13 9:12 12:9 13:5 14:1
5 1:5 2:31 3:30 4:34 5:6 6:1 7:10 8:10 9:2 12:25 13:5 14:1
5 1:5 2:30 3:29 4:33 5:6 6:29 7:14 8:7 9:23 12:6 13:5
18 1:18 2:29 3:28 4:32 5:17 6:16 7:7 8:17 9:6 12:37 13:8 14:1 15:1
5 1:5 2:35 3:34 4:37 5:6 6:27 7:14 8:7 9:23 12:38 13:5 15:4
9 1:9 2:28 3:27 4:36 5:9 6:21 7:4 9:16 12:46 13:5 15:4
0 2:25 3:24 4:27 5:1 6:26 7:1 8:3 9:19 10:7 11:7 12:28 13:5
15 1:15 2:23 3:22 4:27 5:4 6:4 7:3 8:6 9:11 10:4 11:4 12:41 13:10 15:6
15 1:15 2:23 3:22 4:27 5:4 6:6 7:22 8:6 9:11 10:10 11:10 12:40 15:6
3 1:3 2:22 3:21 4:31 5:3 6:19 7:11 8:7 9:7 10:9 11:9 12:8 13:5 15:4
3 1:3 2:21 3:20 4:26 5:3 6:18 7:11 8:7 9:7 12:26 13:5
3 1:3 2:21 3:20 4:26 5:3 6:18 7:11 8:7 9:7 10:4 11:4 12:49 13:5
2 1:2 2:20 3:19 4:25 5:5 6:3 7:2 8:14 9:8 10:15 11:15 12:27 13:5 14:1
14 1:14 2:19 3:18 4:24 5:14 6:5 7:19 8:16 9:25 10:2 11:2 12:33 13:6
3 1:3 2:27 3:26 4:23 5:3 6:14 7:13 8:15 9:20 10:12 11:12 12:30 13:5
5 1:5 2:27 3:26 4:30 5:6 6:29 7:14 8:7 9:23 12:2 13:1
5 1:5 2:26 3:25 4:29 5:6 6:29 7:14 8:7 9:23 12:10 13:2
5 1:5 2:24 3:23 4:28 5:6 6:29 7:14 8:7 9:23 12:3 13:3
5 1:5 2:15 3:14 4:21 5:6 6:30 7:14 8:7 9:23 13:7 15:4
```

For SVM, the accuracy is 0.6 and test accuracy is 0.97.



Name	Last Modified
Untitled12.ipynb	a month ago
Untitled13.ipynb	25 days ago
Untitled14.ipynb	16 days ago
Untitled15.ipynb	17 days ago
Untitled16.ipynb	13 days ago
Untitled17.ipynb	7 days ago
Untitled18.ipynb	12 days ago
Untitled19.ipynb	9 days ago
Untitled2.ipynb	3 months ago
Untitled20.ipynb	7 days ago
Untitled21.ipynb	7 days ago
Untitled22.ipynb	7 days ago
Untitled23.ipynb	7 days ago
Untitled24.ipynb	7 days ago
Untitled25.ipynb	7 days ago
Untitled26.ipynb	6 days ago
Untitled27.ipynb	35 minutes ago
Untitled28.ipynb	19 hours ago
Untitled29.ipynb	15 minutes ago
Untitled3.ipynb	3 months ago
Untitled30.ipynb	2 hours ago
Untitled31.ipynb	21 minutes ago
Untitled32.ipynb	seconds ago

```
[15]: from sklearn.model_selection import train_test_split
      from sklearn.svm import SVC
      from sklearn.datasets import load_svmlight_file
      from sklearn.pipeline import make_pipeline
      from sklearn.preprocessing import StandardScaler
      from sklearn.model_selection import GridSearchCV

      X,y = load_svmlight_file("NLP_dataset_clean.libsvm")
      X_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
      SS = StandardScaler(with_mean=False)
      SS.fit(X,y)
      SVM = SVC()
      pipe = make_pipeline(SS, SVM)
      param_grid = [{'svc__kernel': ['linear']}
                   ]
      grid_search = GridSearchCV(pipe, param_grid, cv=6)
      grid_search.fit(X, y)
      print(grid_search.best_score_)
      print('Test accuracy: %.3f' % grid_search.score(X_train, y_train))

      0.611111111111111
      Test accuracy: 0.973
      E:\anaconda3\lib\site-packages\sklearn\model_selection\_split.py:676: UserWarning: The least populated class in y has only 1 members, whi
      ch is less than n_splits=6.
        warnings.warn(

[ ]:
```

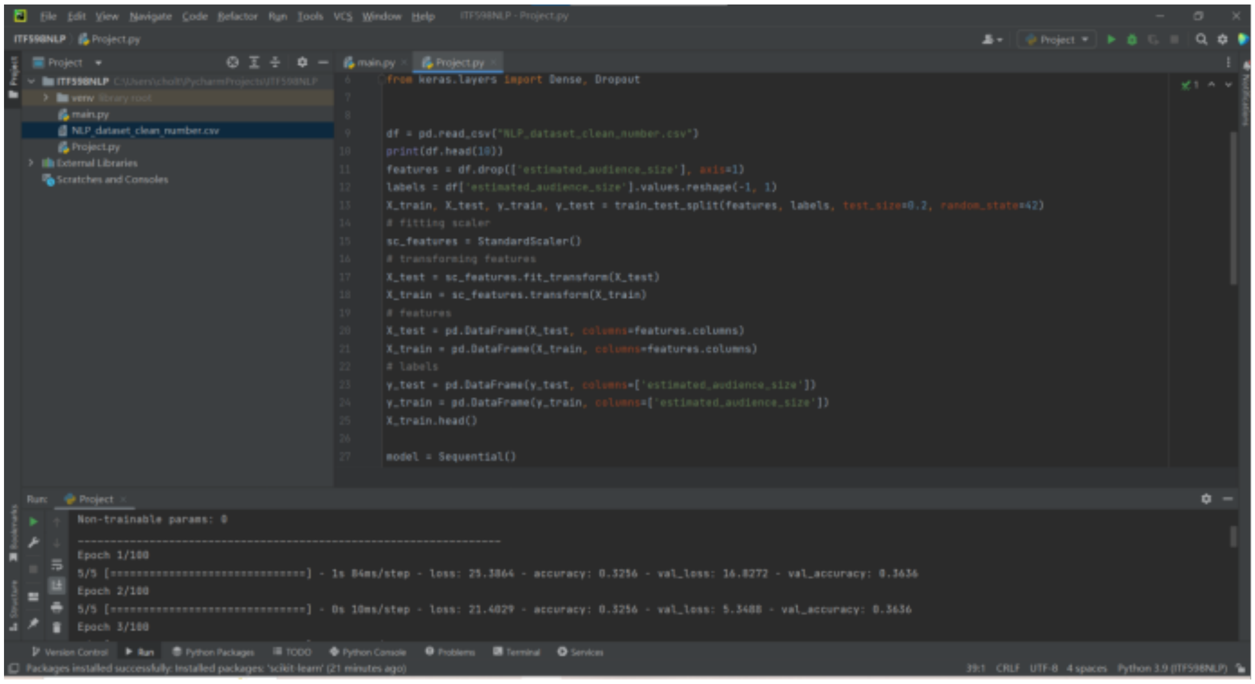
Code:

```
from sklearn.model_selection import train_test_split

from sklearn.svm import SVC

from sklearn.datasets import load_svmlight_file

from sklearn.pipeline import make_pipeline

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import GridSearchCV



X,y = load_svmlight_file("NLP_dataset_clean.libsvm")

X_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)

SS = StandardScaler(with_mean=False)

SS.fit(X,y)

SVM = SVC()
```

```
pipe = make_pipeline(SS, SVM)
param_grid = [{'svc__kernel': ['linear']}
          ]
grid_search = GridSearchCV(pipe, param_grid, cv=6)
grid_search.fit(X, y)
print(grid_search.best_score_)
print('Test accuracy: %.3f' % grid_search.score(X_train, y_train))
```

For RNN, the result of accuracy is 0.093 and valid is 0.364.

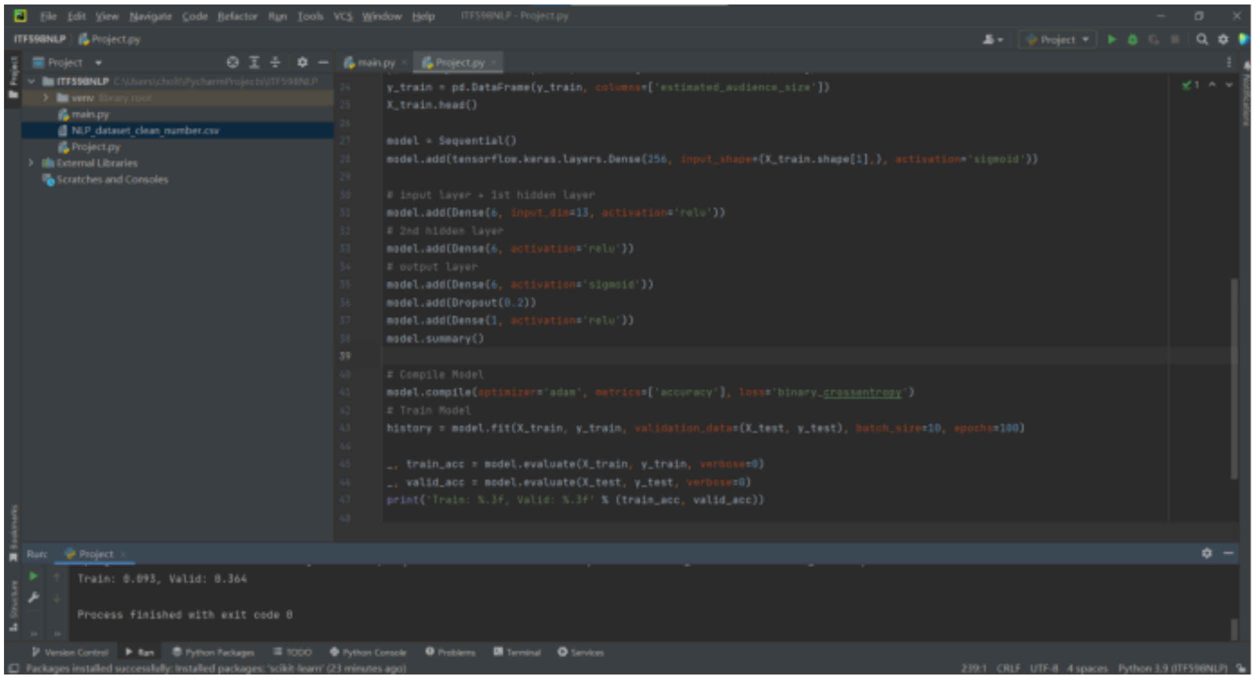
We use the dataset already to transform it into numbers. The result is not ideal. This parameter still needs tuning, and the dataset needs to be enlarged for accuracy. In the code, we use TensorFlow and Keras as our libraries. All of them provide a powerful function to us. Also, in our RNN, we use four layers as the input layer, 1st layer, 2nd layer, and the output layer to calculate. Different active functions are used in the model such as sigmoid and relu.



```python
from keras.layers import Dense, Dropout


df = pd.read_csv("NLP_dataset_clean_number.csv")
print(df.head(10))
features = df.drop(['estimated_audience_size'], axis=1)
labels = df['estimated_audience_size'].values.reshape(-1, 1)
X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.2, random_state=42)
# fitting scaler
sc_features = StandardScaler()
# transforming features
X_test = sc_features.fit_transform(X_test)
X_train = sc_features.transform(X_train)
# features
X_test = pd.DataFrame(X_test, columns=features.columns)
X_train = pd.DataFrame(X_train, columns=features.columns)
# labels
y_test = pd.DataFrame(y_test, columns=['estimated_audience_size'])
y_train = pd.DataFrame(y_train, columns=['estimated_audience_size'])
X_train.head()

model = Sequential()
```

```
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
5/5 [==============================] - 1s 84ms/step - loss: 25.3864 - accuracy: 0.3256 - val_loss: 16.8272 - val_accuracy: 0.3636
Epoch 2/100
5/5 [==============================] - 0s 10ms/step - loss: 21.4029 - accuracy: 0.3256 - val_loss: 5.3488 - val_accuracy: 0.3636
Epoch 3/100
```

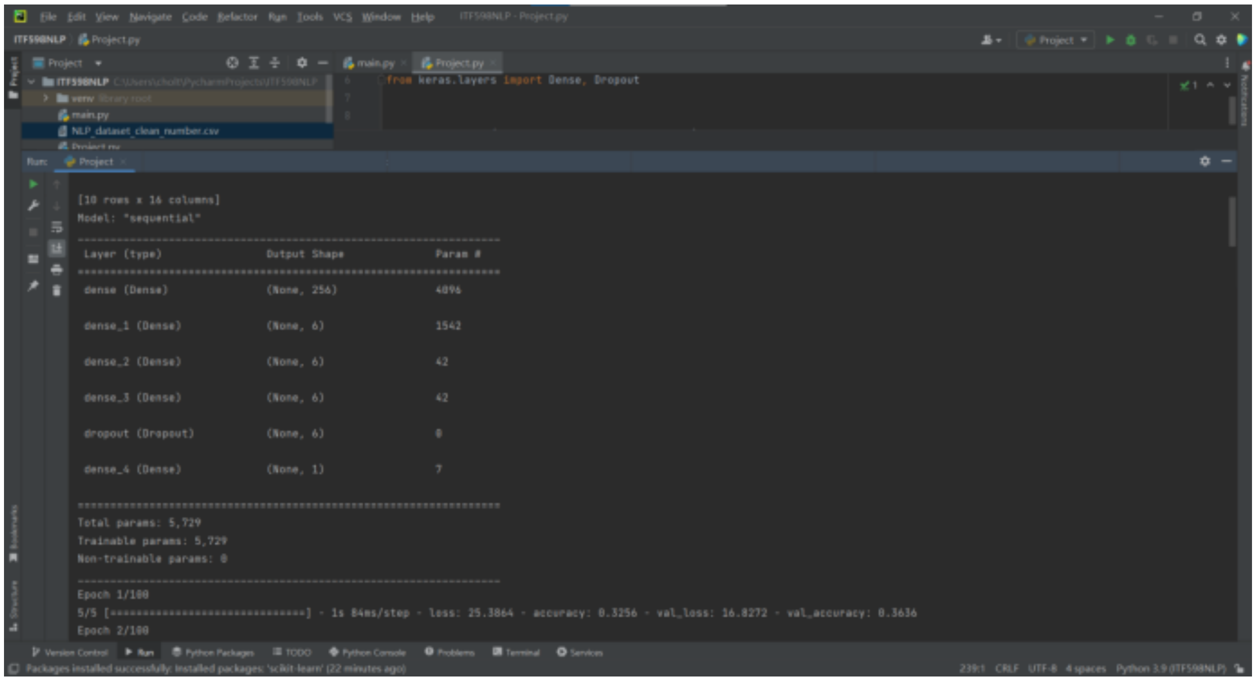
Packages installed successfully: Installed packages: 'scikit-learn' (21 minutes ago)



```python
y_train = pd.DataFrame(y_train, columns=['estimated_audience_size'])
X_train.head()

model = Sequential()
model.add(tensorflow.keras.layers.Dense(256, input_shape=(X_train.shape[1],), activation='sigmoid'))

# input layer + 1st hidden layer
model.add(Dense(6, input_dim=13, activation='relu'))
# 2nd hidden layer
model.add(Dense(6, activation='relu'))
# output layer
model.add(Dense(6, activation='sigmoid'))
model.add(Dropout(0.2))
model.add(Dense(1, activation='relu'))
model.summary()

# Compile Model
model.compile(optimizer='adam', metrics=['accuracy'], loss='binary_crossentropy')
# Train Model
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), batch_size=10, epochs=100)

_, train_acc = model.evaluate(X_train, y_train, verbose=0)
_, valid_acc = model.evaluate(X_test, y_test, verbose=0)
print('Train: %.3f, Valid: %.3f' % (train_acc, valid_acc))
```

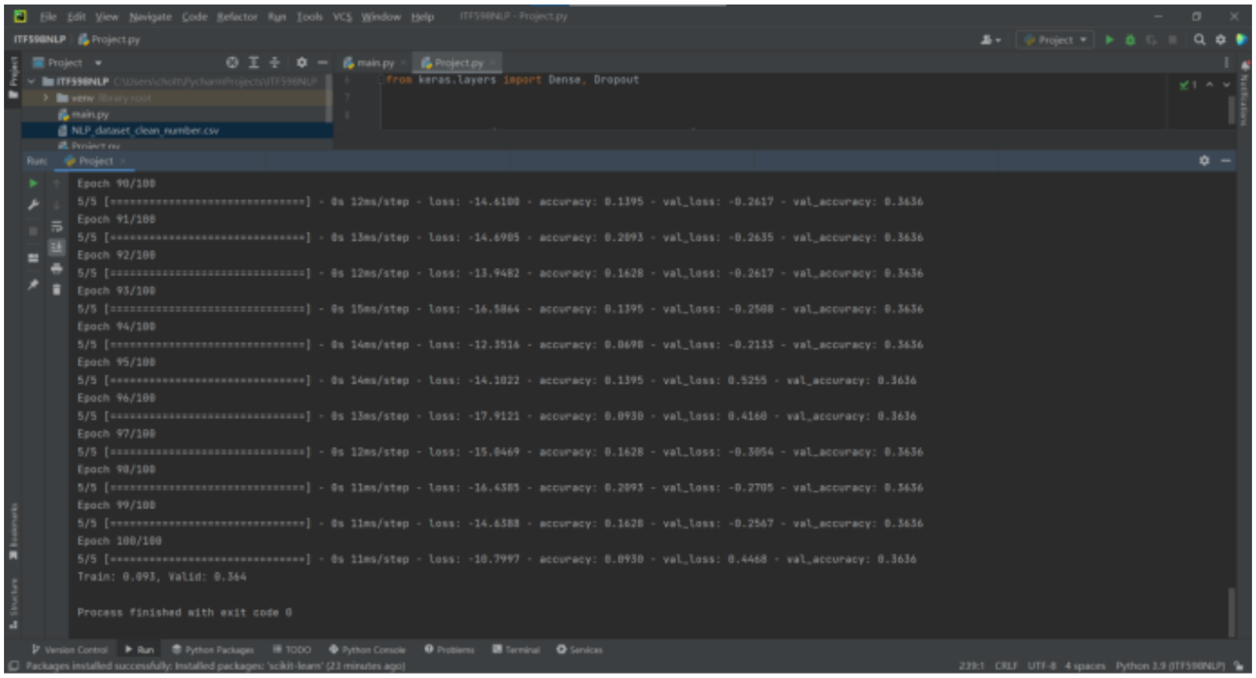
```
Train: 0.093, Valid: 0.364

Process finished with exit code 0
```

Packages installed successfully: Installed packages: 'scikit-learn' (23 minutes ago)



```
from keras.layers import Dense, Dropout
```

```
[10 rows x 16 columns]
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 256)               4096

 dense_1 (Dense)             (None, 6)                 1542

 dense_2 (Dense)             (None, 6)                 42

 dense_3 (Dense)             (None, 6)                 42

 dropout (Dropout)           (None, 6)                 0

 dense_4 (Dense)             (None, 1)                 7

=================================================================
Total params: 5,729
Trainable params: 5,729
Non-trainable params: 0
_________________________________________________________________
Epoch 1/100
5/5 [==============================] - 1s 84ms/step - loss: 25.3864 - accuracy: 0.3256 - val_loss: 16.8272 - val_accuracy: 0.3636
Epoch 2/100
```



```
from keras.layers import Dense, Dropout
```

```
Epoch 90/100
5/5 [==============================] - 0s 12ms/step - loss: -14.6100 - accuracy: 0.1395 - val_loss: -0.2617 - val_accuracy: 0.3636
Epoch 91/100
5/5 [==============================] - 0s 13ms/step - loss: -14.6985 - accuracy: 0.2093 - val_loss: -0.2635 - val_accuracy: 0.3636
Epoch 92/100
5/5 [==============================] - 0s 12ms/step - loss: -13.9482 - accuracy: 0.1628 - val_loss: -0.2617 - val_accuracy: 0.3636
Epoch 93/100
5/5 [==============================] - 0s 15ms/step - loss: -16.5864 - accuracy: 0.1395 - val_loss: -0.2508 - val_accuracy: 0.3636
Epoch 94/100
5/5 [==============================] - 0s 14ms/step - loss: -12.3516 - accuracy: 0.0698 - val_loss: -0.2133 - val_accuracy: 0.3636
Epoch 95/100
5/5 [==============================] - 0s 14ms/step - loss: -14.1022 - accuracy: 0.1395 - val_loss: 0.5255 - val_accuracy: 0.3636
Epoch 96/100
5/5 [==============================] - 0s 13ms/step - loss: -17.9121 - accuracy: 0.0930 - val_loss: 0.4160 - val_accuracy: 0.3636
Epoch 97/100
5/5 [==============================] - 0s 12ms/step - loss: -15.0469 - accuracy: 0.1628 - val_loss: -0.3054 - val_accuracy: 0.3636
Epoch 98/100
5/5 [==============================] - 0s 11ms/step - loss: -16.4385 - accuracy: 0.2093 - val_loss: -0.2705 - val_accuracy: 0.3636
Epoch 99/100
5/5 [==============================] - 0s 11ms/step - loss: -14.6388 - accuracy: 0.1628 - val_loss: -0.2567 - val_accuracy: 0.3636
Epoch 100/100
5/5 [==============================] - 0s 11ms/step - loss: -10.7997 - accuracy: 0.0930 - val_loss: 0.4468 - val_accuracy: 0.3636
Train: 0.093, Valid: 0.364

Process finished with exit code 0
```

Code:

```
import pandas as pd

from sklearn.preprocessing import StandardScaler

from sklearn.model_selection import train_test_split

import tensorflow as tensorflow

from keras.models import Sequential

from keras.layers import Dense, Dropout


df = pd.read_csv("NLP_dataset_clean_number.csv")

features = df.drop(['estimated_audience_size'], axis=1)

labels = df['estimated_audience_size'].values.reshape(-1, 1)

X_train, X_test, y_train, y_test = train_test_split(features, labels, test_size=0.2,
random_state=42)

# fitting scaler

sc_features = StandardScaler()

# transforming features

X_test = sc_features.fit_transform(X_test)

X_train = sc_features.transform(X_train)

# features

X_test = pd.DataFrame(X_test, columns=features.columns)

X_train = pd.DataFrame(X_train, columns=features.columns)

# labels

y_test = pd.DataFrame(y_test, columns=['estimated_audience_size'])
```

```
y_train = pd.DataFrame(y_train, columns=['estimated_audience_size'])
X_train.head()


model = Sequential()
model.add(tensorflow.keras.layers.Dense(256, input_shape=(X_train.shape[1],),
activation='sigmoid'))


# input layer + 1st hidden layer
model.add(Dense(6, input_dim=13, activation='relu'))
# 2nd hidden layer
model.add(Dense(6, activation='relu'))
# output layer
model.add(Dense(6, activation='sigmoid'))
model.add(Dropout(0.2))
model.add(Dense(1, activation='relu'))
model.summary()


# Compile Model
model.compile(optimizer='adam', metrics=['accuracy'], loss='binary_crossentropy')
# Train Model
history = model.fit(X_train, y_train, validation_data=(X_test, y_test), batch_size=10,
epochs=100)
```

```
_, train_acc = model.evaluate(X_train, y_train, verbose=0)
_, valid_acc = model.evaluate(X_test, y_test, verbose=0)
print('Train: %.3f, Valid: %.3f' % (train_acc, valid_acc))
```

For training the SVM model, Anaconda has been used for SVM. In the notebook, we can use python3 and Sklearn to fit and train our model. For the RNN, we have used the powerful Pycharm as IDE, TensorFlow, and Keras as our library. They can easily train the neural network model with activation functions.

Data Analysis

Due to the product in the analysis phase being abstract, users cannot see what the product looks like. In the analysis phase of the product, we can provide a graphical representation of the statistics and the results achieved from the modeling. This representation of the data, using toxicity vs audience reach and ad toxicity comparison can help the audience understand the role of toxic advertising in political campaigns. We have a very small data set which prevents us from getting accurate results or even training the model to have good accuracy.

Our model is successful when the whole process, data acquisition to generate the model and results, is completed and when a deeper understanding of the toxicity of political campaigns is achieved. This can be achieved when the model returns a prediction or classifies a given set of features related to an advertising campaign as toxic on the given scale.

"We find SVM is better than RNN". RNN should be better than SVM when it comes to the entire model. In the beginning, we thought RNN was a better algorithm than SVM. However, we found that SVM is much better than RNN when we finished our model. The reasons for this conclusion are the size of data points and a powerful GPU is not involved. In theory, The RNNs is a good neural network model. To achieve that a bigger data set and equipment will be necessary.

Conclusion

In conclusion, our study examined the extent to which microtargeting was used in political advertisements during the 2023 Phoenix mayoral election, compared to the 2020 US presidential campaign. Our results suggest that social media advertising presents a valuable resource in targeting specific groups of individuals who are most susceptible to guided advertisements. This finding highlights the importance of understanding the nature of political advertising and the strategies used to target voters. As political campaigns become more dependent on monetary resources, understanding how much money is being spent on advertising and the nature of this advertising becomes increasingly important. Ultimately, our findings provide insight into how microtargeting may affect voter behavior in city elections and underscore the need for continued research in this area.

Based on the analysis of political advertisements used in the 2023 Phoenix mayoral election, it can be concluded that microtargeting is extensively used for political advertisement in city elections. Social media advertising presents a valuable resource in targeting specific groups of individuals who are most susceptible to guided advertisement.

While traditional media remains an important part of electoral voting, social media advertising allows political campaigns to send targeted messages to voters based on factors like location, media preference, associated political beliefs, age, and other critical variables.

For the project, the model of our training can be re-used for any project at any project, process, and solution. There are two ways to enhance products they want to use in the future.

First, the model can be published as an API. The users can upload their data points in their area to judge how toxic Ads affect voters. When they upload the data points based on their region, they can get the accuracy of the model, which is an index to indicate the level of toxic Ads. This model can also be used to understand the sentiments of an area.

Second, if considering data privacy, the users can copy the code into their project to generate their own model to fit their business predictions. In this way, it prevents the data from being exposed to untrustful transfer through the network. This way this model can enable the creation of inclusive and friendly advertising.

The techniques of NLP are very powerful when we deal with this situation. For measuring the toxic Ads, word embracing and removing the stop words can improve the efficiency when we train the model. The Ads do not have grammar errors, we can escape this process in our code. But if other users use the dataset that doesn't come from the Facebook Ads API, they still need to think about checking the grammar errors. Also, we just use SVM and RNNs to train the dataset as an example. In the future, the decision tree, ANN, or other algorithms can still try to train the model. It might be better than the current model.

When working with data from a remote rural area where the use of language varies from the use of language based on which the model is trained, the model might not be as successful. This opens the door to understanding language in a new light. This can potentially require the model to be retrained if there is minimal linguistic similarity.

Reference

Akshay Tondak. (2022). Recurrent Neural Networks (RNN) Tutorial: RNN Training, Advantages & Disadvantages (Complete Guidance). https://k21academy.com/datascience-blog/machine-learning/recurrent-neural-networks/

Anna Rogers, Tim Baldwin, Kobi Leins. (2021). Just What do You Think You're Doing, Dave?' A Checklist for Responsible Data Use in NLP. https://arxiv.org/abs/2109.06598

Asmaa Elbadrawy. (2023). Handling Data Quality Issues. https://asu.instructure.com/courses/145901/pages/handling-data-quality-issues?module_item_id=10172859

Fabio Votta, Arman Noroozian, Tom Dobber, Natali Helberger, Claes de Vreese. (2023). Going Micro to Go Negative? Targeting Toxicity using Facebook and Instagram Ads. https://www.aup-online.com/content/journals/10.5117/CCR2023.1.001.VOTT#CIT0059

Gitanjali Gupta & Kamlesh Lakhwani. (2022). An enhanced approach to improve the encryption of big data using intelligent classification techniques. https://link.springer.com/article/10.1007/s11042-022-12401-5

Hobson Lane, Cole Howard, Hannes Hapke. (2019). Natural Language Processing in Action.

Robert DiPietro, Gregory D. Hager. (2020). Recurrent Neural Network—An overview. https://www.sciencedirect.com/topics/engineering/recurrent-neural-network

Vikramaditya Jakkula. (2006). Tutorial on support vector machine (SVM). https://course.ccs.neu.edu/cs5100f11/resources/jakkula.pdf