DATA INSIGHTS

EXPLORATORY DATA ANALYSIS (EDA)

Team: Deep Learners

Team Details:

1. Rahul Jha (076BCT)
2. Sumit Yadav (076BCT)

Topics:

- Text Preprocessing
- Word Cloud
- Analysis of Categorical Distribution
- N-gram Exploration
- Multiple Models Performance

Text Preprocessing

- Text preprocessing is a method to clean the text data and make it ready to feed data to the model. Text data contains noise in various forms like emotions, punctuation, text in a different case.
- In python, some of the popular libraries used for text preprocessing are:
    1. Natural Language Toolkit (NLTK)
    2. Gensim
    3. spaCy (used here)
    4. TextBlob

Text Preprocessing Techniques

- Expand Contractions
- Lower Case
- Remove Punctuations
- Remove words and digits containing digits
- Remove Stopwords
- Rephrase Text
- Stemming and Lemmatization
- Remove White spaces

Example

For a demo of the preprocessing we performed on our dataset, the following example is presented:

Original Abstract:

" *This paper describes the derivation of the level 5 versions of Ramanujan's system of ordinary differential equations satisfied by the Eisenstein series,$E_2(q),E_4(q)$, and $E_6(q).$ "*

This is converted into:

"paper describ deriv level version ramanujan s system ordinari differenti equat satisfi eisenstein seri e q e q e q "

Details of Preprocessing

- We created a single function called clean_abstract which does the following operations:
- In our preprocessing, we first cleansed the data by removing extra spaces and numbers as well as punctuations. We just let the alphabets stay in our dataset.
- We then lowercased all the data and removed the stopwords.
- Through PorterStemmer, we stemmed the data.

WORDCLOUD

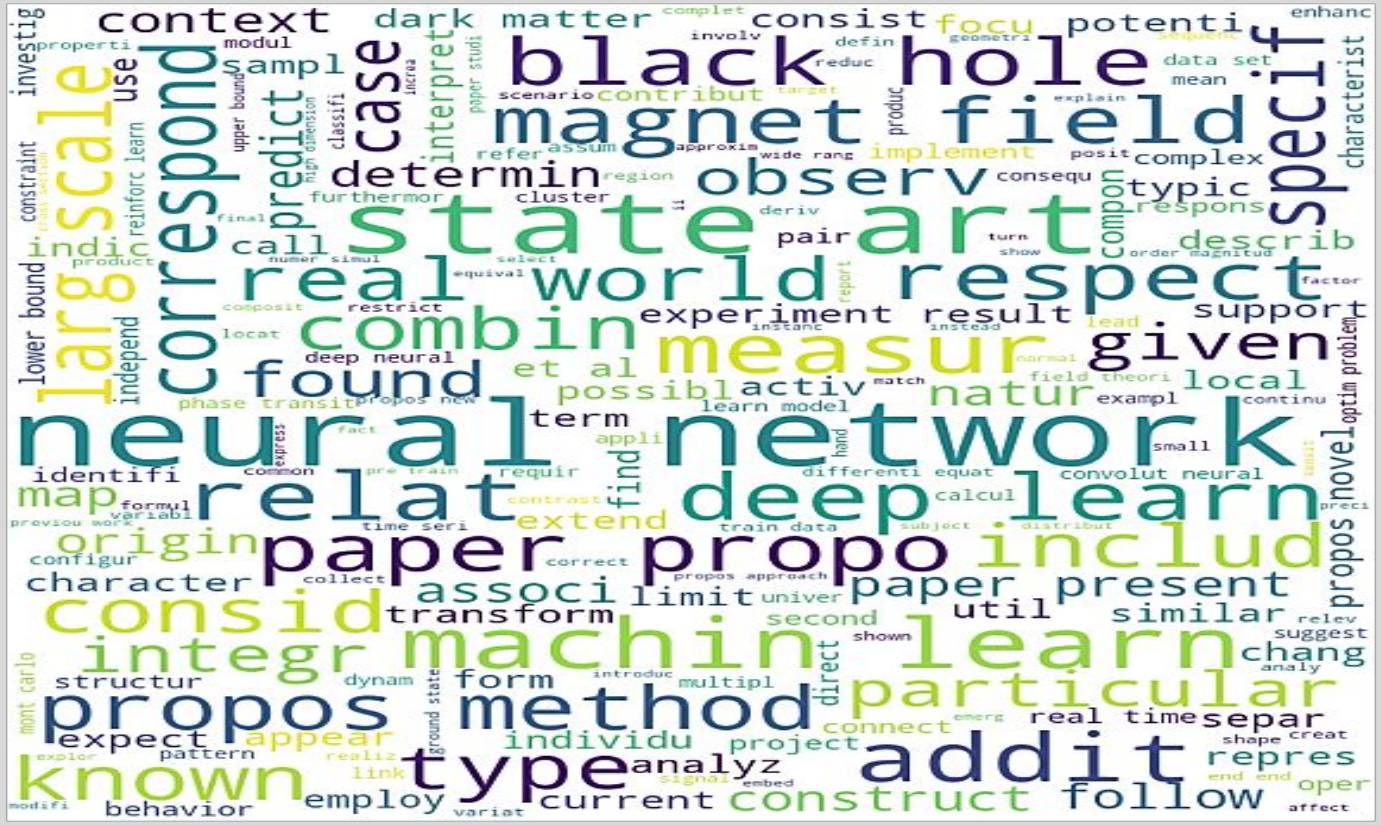
Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.

For generating word cloud in Python, modules needed are – matplotlib, pandas and wordcloud.

Wordcloud for any covid-related text data:

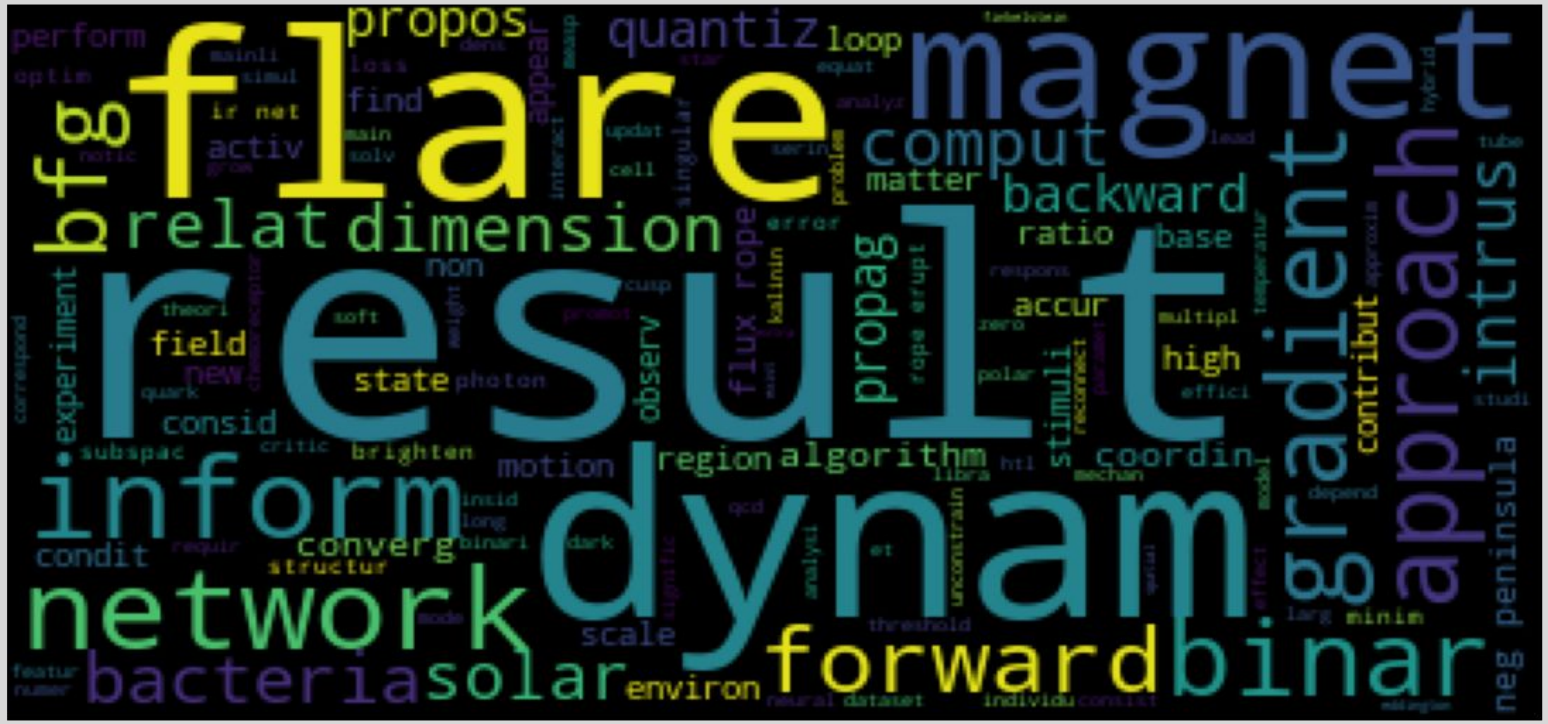
Wordcloud 1

The first wordcloud we generated was for the overall data *(abstract column)* of our dataset.



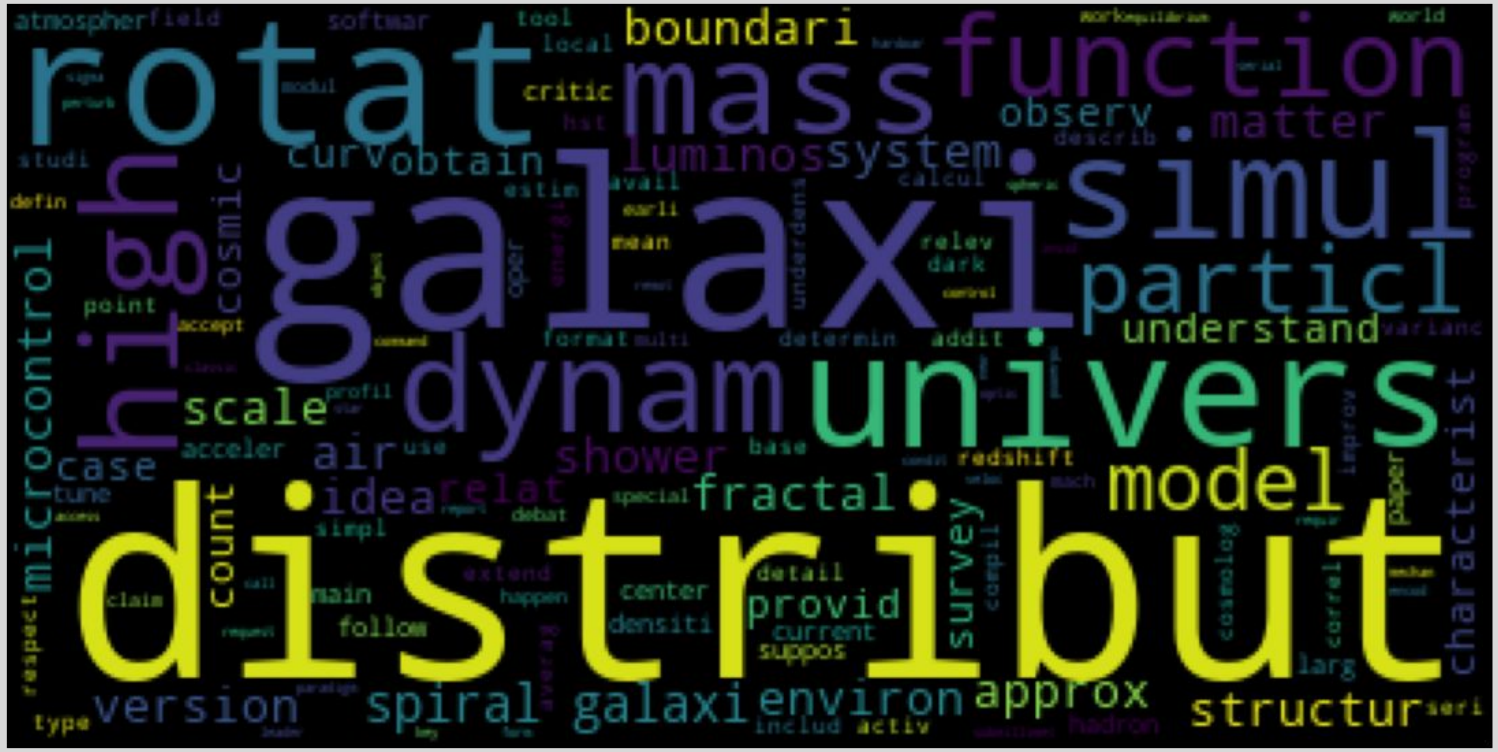
Wordcloud 2

The second wordcloud we generated was for the overall data *(abstract column)* of our dataset after reshuffling the data.



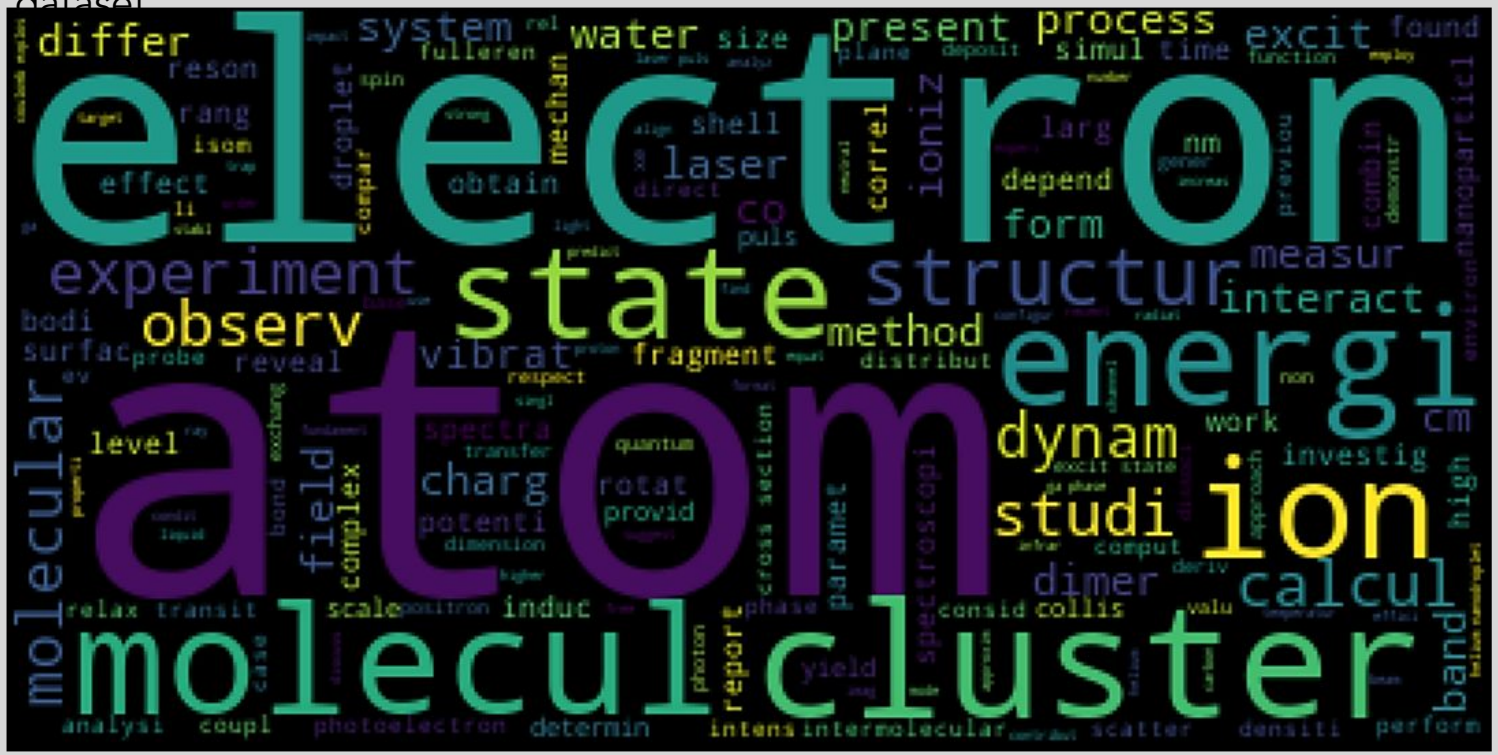
Wordcloud 3

The third wordcloud we generated was for the data *(abstract column)* of category "Astro-ph" of our dataset.



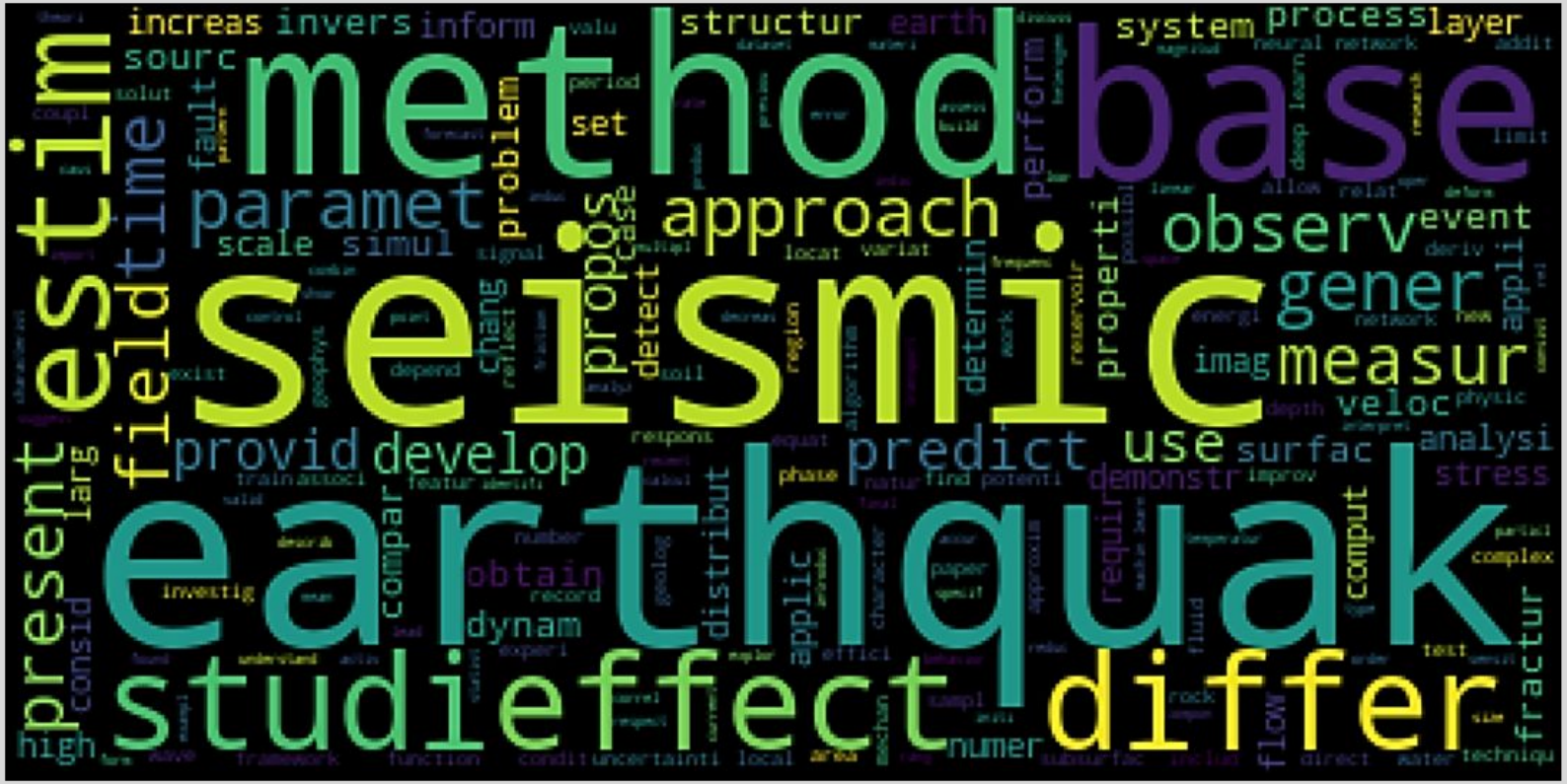
Wordcloud 4

The fourth wordcloud we generated was for the data *(abstract column)* of category "physics-atm-clus" of our dataset



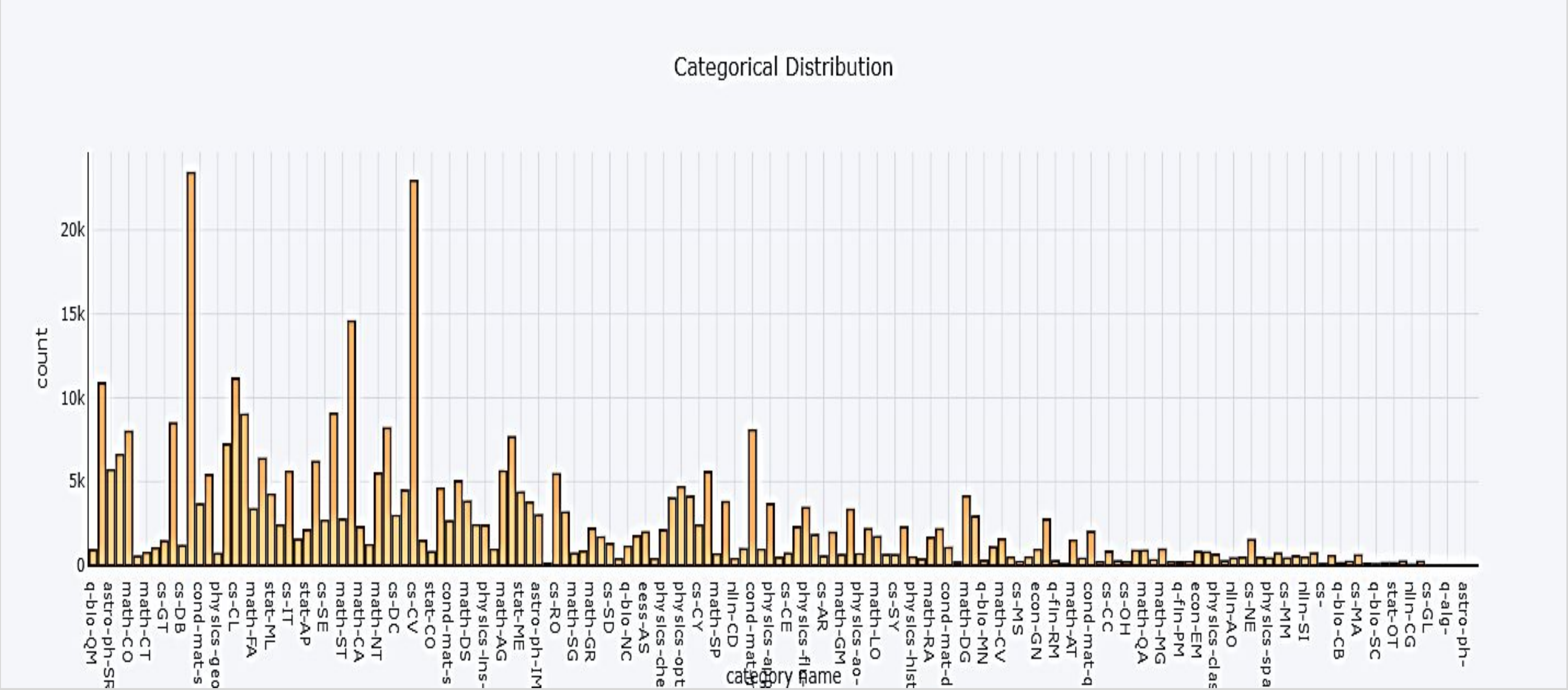
Wordcloud 5

The fifth wordcloud we generated was for the data *(abstract column)* of category "physics-geo" of our dataset.



Categorical Distribution

Now, for our EDA section, we created a plot which shows categorical distribution of data, i.e. number of data in each category of our dataset.



Categorical Distribution
count
20k
15k
10k
5k
0
category name
q-bio-QM
math-CO
math-CT
cs-GT
cs-DB
cs-CL
math-FA
stat-ML
cs-IT
stat-AP
cs-SE
math-ST
math-CA
math-NT
cs-DC
cs-CV
stat-CO
math-DS
math-AG
stat-ME
cs-RO
math-SG
math-GR
cs-SD
q-bio-NC
eess-AS
cs-CY
math-SP
nlin-CD
cs-CE
cs-AR
math-GM
math-LO
cs-SY
math-RA
math-DG
q-bio-MN
math-CV
cs-MS
econ-GN
q-fin-RM
math-AT
cs-CC
cs-OH
math-QA
math-MG
q-fin-PM
econ-EM
nlin-AO
cs-NE
cs-MM
nlin-SI
q-bio-CB
cs-MA
q-bio-SC
stat-OT
nlin-CG
cs-GL

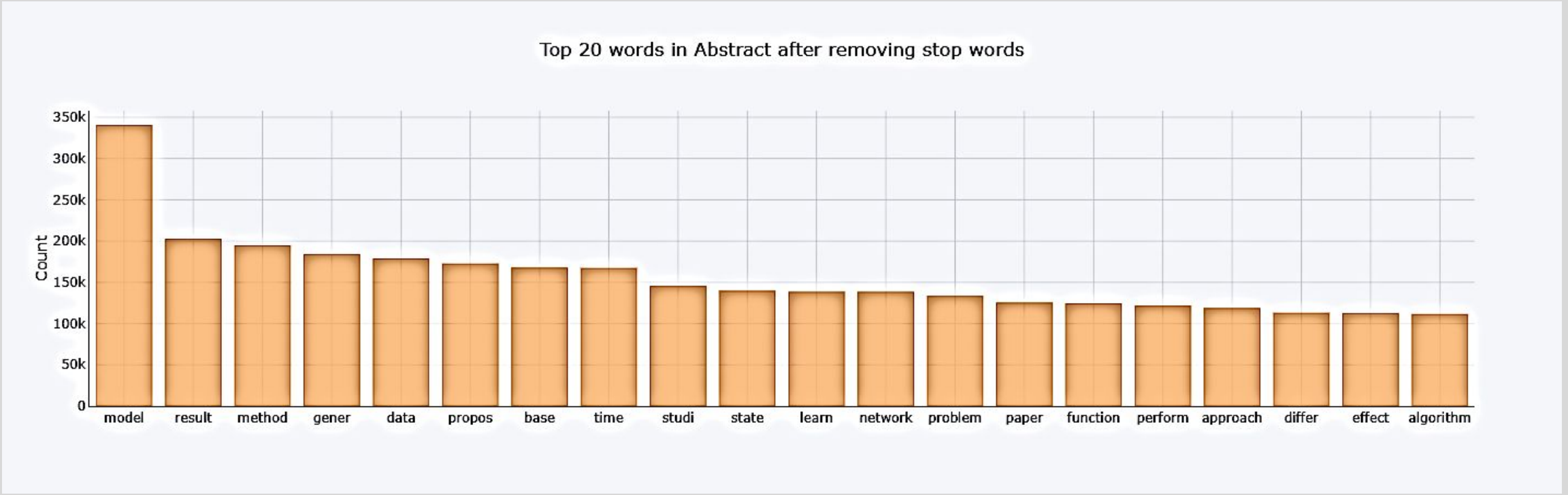
N-gram exploration

In natural language processing, an n-gram is an arrangement of n words. For example “Python” is a unigram (n = 1), “Data Science” is a bigram (n = 2), “Natural language preparing” is a trigram (n = 3) etc.

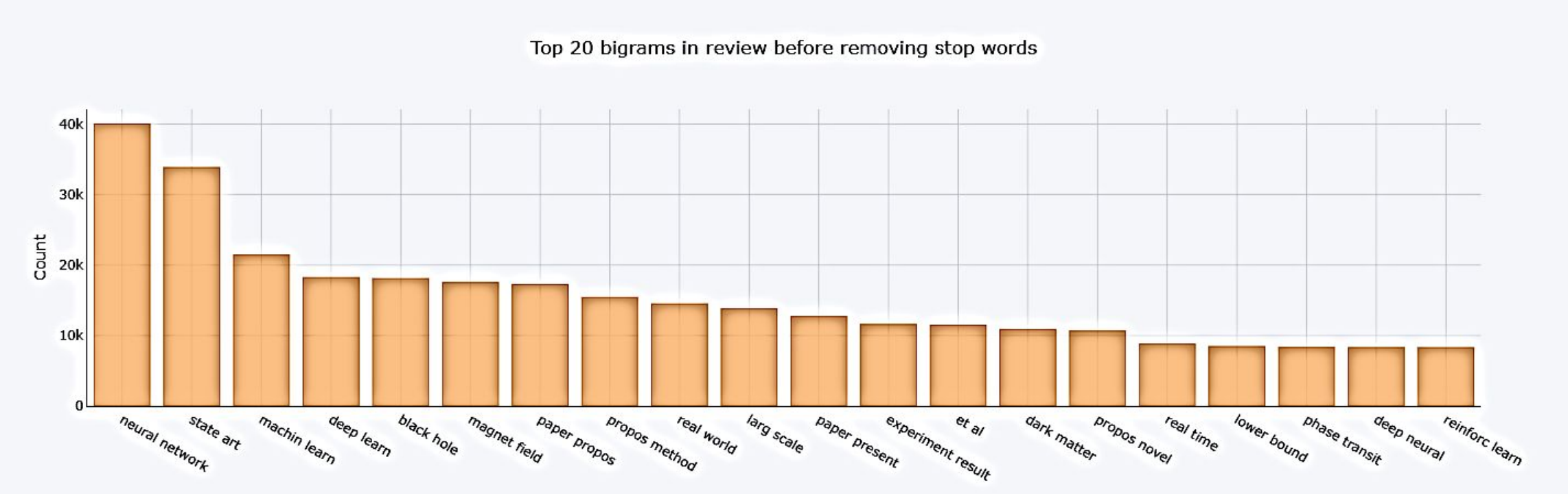
Here our focus will be on implementing the unigrams (single words), bigrams (double words) and trigrams (triple words) models in python for our dataset.

Note: We will consider only top 20 N-grams here.

Unigram Exploration



Bigram Exploration



Trigram Exploration