National Science Foundation (NSF) Awards Granted History Analysis

Prepared and presented by:

YARA SEIF

MOHAMMAD AL-SHAMI

FAIRUS TANZIM

ASIF KAMAL TURZO

RADUANUL CHOWDHURY

Abstract

The National Science Foundation (NSF) are the funding source for approximately 25 percent of all federally supported basic research conducted by America's colleges and universities. Our team is conducting an analysis on the NSF Award data shown over the years and visualize the data to conclude awards analysis granted, to do that we collect/export our XML formatted data directly from the provided link (https://www.nsf.gov/awardsearch/download.jsp) We parsed the data using Python (pandas data frame) and extracted the file as CSV to start our analysis. Then we used multiple software to demonstrate our findings such as (Tableau, PowerBI, and Jupiter Notebook.)

Post our visualization we were able to analyze the spread of the awards are in the East coast, West coast, and South of the United States, also showed the highest dollar amount of NSF grant was given to Directorate for Mathematical & Physical Science, a total of $30 Billion. We also analyzed the abstract and title of the NSF proposal using the Natural Language Processing (NLP) technique. We identified that Machine Learning/Data analysis is the topic that has received the most grant.

Background

NSF funds research and education in most fields of science and engineering. They do this through grants and cooperative agreements to more than 2,000 colleges, universities, K-12 school systems, businesses, informal science organizations and other research organizations throughout the U.S. The Foundation considers proposals submitted by organizations on behalf of individuals or groups for support in most fields of research. Interdisciplinary proposals also are eligible for consideration. Awardees are chosen from those who send us proposals asking for a specific amount of support for a specific project.

The goal of our class study is to analyze awards grants given by the NSF, National Science Foundation, to the different programs throughout the years. In our project we are examining more than 10 years of data, and we can conclude that the NSF awards different fellowship programs based on their importance relevant to the year we look at. Also, our dataset provides us with the national spread of the awards by state, also by money amount. Therefore, we can conclude that the more educationally developed the state or city is the more money amount we see, due to more accredited institutions for example, based on our heat map visual, we seem to find that NSF had given more award money in Massachusetts than it has to Idaho.

Methodology

In the study we have conducted on the NSF Awards data, we first collected the data in its XML schema file formats and parsed data in all years in the study and consolidated it into one CSV file format to be analyzed. We have used Python Jupyter and treating the dataset as a data frame using pandas, we used the code below:

```python
import requests
import os
import csv
from zipfile import ZipFile
import xml.etree.ElementTree as ET


def file_download():
    path = "datasets"
    isExist = os.path.exists(path)
    if not isExist:
       os.makedirs(path)
       print("The new directory is created!")

    for i in range(2000,2022):
        URL = "https://www.nsf.gov/awardsearch/download?DownloadFileName={0}&All=true".format(i)
        # Download the data behind the URL
        response = requests.get(URL)
        # Open the response into a new file
        file_name = "datasets/{0}.zip".format(i)

        open(file_name, "wb").write(response.content)
        print("File download complete for year {0}".format(i))


def extract_zip():
    with os.scandir('datasets/') as entries:
        for entry in entries:
            if(".zip" in entry.name):
                with ZipFile('datasets/'+entry.name, 'r') as zipObj:
                    strn = entry.name.split(".")
                    zipObj.extractall('datasets/'+strn[0])


def parseXML(xmlfile,headers = []):

    tree = ET.parse(xmlfile)
    root = tree.getroot()
    file_name="dataset.csv"
    exists = os.path.exists(file_name)
    row_header=[]
    row_value=[]
    if exists:
        row_header_2=[None] * len(headers)

    for i in range(len(root[0])):
        if root[0][i].text is not None:
            if not root[0][i].text.strip():
                for j in range(len(root[0][i])):
                    if root[0][i][j].text is not None:
                        if not root[0][i][j].text.strip():
                            for k in range(len(root[0][i][j])):
                                #print(str(root[0][i][j][k].tag)+ " : "+str(root[0][i][j][k].text))
                                if not exists:
                                    row_value.append(str(root[0][i][j][k].text))
                                    row_header.append(str(root[0][i][j].tag)+"_"+str(root[0][i][j][k].tag))
                                else:
                                    s_match=str(root[0][i][j].tag)+"_"+str(root[0][i][j][k].tag)
                                    ind=0
                                    try:
                                        ind=headers.index(s_match)
                                    except ValueError:
                                        ind=-1
                                    if ind!=-1:
                                        row_header_2[ind]=str(root[0][i][j][k].text)
                        else:
                            #print(str(root[0][i][j].tag)+ " : "+str(root[0][i][j].text))
                            if not exists:
                                row_value.append(str(root[0][i][j].text))
                                row_header.append(str(root[0][i].tag)+"_"+str(root[0][i][j].tag))
                            else:
                                s_match=str(root[0][i].tag)+"_"+str(root[0][i][j].tag)
                                ind=0
```

```python
                                if ind!=-1:
                                try:
                                    ind=headers.index(s_match)
                                except ValueError:
                                    ind=-1
                                if ind!=-1:
                                    row_header_2[ind]=str(root[0][i][j].text)
                    else:
                        #print(str(root[0][i].tag)+ " : "+str(root[0][i].text))
                        if not exists:
                            row_value.append(str(root[0][i].text))
                            row_header.append(str(root[0][i].tag))
                        else:
                            s_match=str(root[0][i].tag)
                            ind=0
                            try:
                                ind=headers.index(s_match)
                            except ValueError:
                                ind=-1
                            if ind!=-1:
                                row_header_2[ind]=str(root[0][i].text)

    if not exists:
        with open(file_name, 'w', newline='') as f:
            write = csv.writer(f)
            write.writerow(row_header)

    with open(file_name, 'a', newline='') as f:
        write = csv.writer(f)
        if not exists:
            write.writerow(row_value)
        else:
            write.writerow(row_header_2)
        #print("writing complete for file: "+xmlfile)

    if len(headers)==0:
        return row_header
    else:
        return None


def create_dataset():
    rootdir = 'datasets/'
    file_name="dataset.csv"
    flag=0
    headers=[]
    exists = os.path.exists(file_name)
    if exists:
        os.remove(file_name)
    list=sorted(os.listdir(rootdir),reverse=True)
    for file in list:
        d = os.path.join(rootdir, file)
        if os.path.isdir(d):
            #print(d)
            file_list = os.listdir(d)
            for file in file_list:
                try:
                    if flag==0:
                        headers=parseXML(d+"/"+file)
                        flag=1
                    else:
                        parseXML(d+"/"+file,headers)
                except:
                    continue

#file_download()
#extract_zip()
create_dataset()
```

We parsed the data into useful fields (shown below) to be able to begin our text analysis, and visualization using Tableau and PowerBI:

	A	B	C	D	E	F	G	H	I	J	K
1	AwardTitle	AGENCY	AwardEffectiveDate	AwardExpirationDate	AwardTotalIntnAmount	AwardAmount	AwardInstrument_Value	Organization_Code	Directorate_Abbreviation	Directorate_LongName	Division_Abbreviation
2	Collaborative	NSF	10/1/2022	9/30/2025	214373	69413	Continuing Grant	6040200	GEO	Directorate For Geoscience	OCE
3	The Role of Int	NSF	10/1/2022	12/31/2022	318962	34403	Standard Grant	7020000	ENG	Directorate For Engineering	CBET
4	CNS Core: Smal	NSF	10/1/2022	6/30/2025	484041	484041	Standard Grant	5050000	CSE	Direct For Computer & Info	CNS
5	GEM: Modeling	NSF	10/1/2022	6/30/2024	449411	386928	Standard Grant	6020200	GEO	Directorate For Geoscience	AGS
6	RAPID: Critical	NSF	11/1/2022	4/30/2023	49970	49970	Standard Grant	6030000	GEO	Directorate For Geoscience	EAR
7	Collaborative	NSF	11/1/2022	7/31/2025	399803	130038	Continuing Grant	6040300	GEO	Directorate For Geoscience	OCE
8	RUI: SpecEES: C	NSF	10/1/2022	12/31/2023	250000	179126	Standard Grant	7010000	ENG	Directorate For Engineering	ECCS
9	Collaborative	NSF	10/1/2022	12/31/2025	250000	250000	Standard Grant	7010000	ENG	Directorate For Engineering	ECCS
10	CAREER: Deep I	NSF	10/1/2022	4/30/2024	550000	351355	Continuing Grant	5020000	CSE	Direct For Computer & Info	IIS
11	Conference: Hc	NSF	11/1/2022	10/31/2023	5000	5000	Standard Grant	3010000	MPS	Direct For Mathematical &	PHY
12	Iterative Algori	NSF	10/1/2022	6/30/2023	300000	259825	Continuing Grant	3040000	MPS	Direct For Mathematical &	DMS
13	FET: Small: Opt	NSF	10/15/2022	9/30/2024	399995	399995	Standard Grant	5010000	CSE	Direct For Computer & Info	CCF
14	Collaborative	NSF	10/15/2022	11/30/2024	368331	333153	Standard Grant	8010000	BIO	Direct For Biological Scienc	DEB
15	Research Initi	NSF	10/1/2022	8/31/2024	199998	199998	Standard Grant	7050000	ENG	Directorate For Engineering	EEC
16	Collaborative	NSF	11/1/2022	10/31/2023	24559	24559	Standard Grant	6030000	GEO	Directorate For Geoscience	EAR
17	Collaborative	NSF	11/1/2022	10/31/2023	21750	21750	Standard Grant	6030000	GEO	Directorate For Geoscience	EAR
18	RR: The Validit	NSF	10/1/2022	8/31/2023	704337	149554	Standard Grant	4050000	SBE	Direct For Social, Behav & E	SES
19	Collaborative	NSF	11/1/2022	9/30/2025	253472	253472	Standard Grant	5020000	CSE	Direct For Computer & Info	IIS
20	Collaborative	NSF	10/1/2022	9/30/2026	275000	275000	Standard Grant	7010000	ENG	Directorate For Engineering	ECCS
21	RUI: Simulatio	NSF	11/1/2022	8/31/2023	282833	120814	Standard Grant	3090000	MPS	Direct For Mathematical &	CHE
22	CNS Core: Smal	NSF	10/1/2022	10/31/2023	500000	440014	Standard Grant	5050000	CSE	Direct For Computer & Info	CNS
23	Collaborative	NSF	10/1/2022	9/30/2024	160241	137683	Standard Grant	5010000	CSE	Direct For Computer & Info	CCF
24	CAREER: Bound	NSF	10/1/2022	7/31/2025	528649	483306	Standard Grant	7030000	ENG	Directorate For Engineering	CMMI
25	CRII: SaTC: Phy	NSF	10/1/2022	6/30/2023	174428	152994	Standard Grant	5050000	CSE	Direct For Computer & Info	CNS
26	CAREER: Distrit	NSF	10/15/2022	5/31/2026	489750	189676	Continuing Grant	5010000	CSE	Direct For Computer & Info	CCF
27	Collaborative	NSF	10/1/2022	10/31/2023	250000	231301	Standard Grant	7030000	ENG	Directorate For Engineering	CMMI
28	CAREER: Cognit	NSF	11/15/2022	10/31/2023	731165	145597	Continuing Grant	11090000	EDU	Directorate for STEM Educa	DRL
29	Collaborative	NSF	11/1/2022	7/31/2025	48579	48579	Standard Grant	6040200	GEO	Directorate For Geoscience	OCE
30	CAREER: Optic	NSF	11/1/2022	12/31/2025	703000	537452	Continuing Grant	3090000	MPS	Direct For Mathematical &	CHE
31	Collaborative	NSF	10/1/2022	10/31/2023	79996	58089	Standard Grant	7010000	ENG	Directorate For Engineering	ECCS
32	CAREER: Tubul	NSF	11/1/2022	12/31/2026	679989	598844	Continuing Grant	3090000	MPS	Direct For Mathematical &	CHE
33	Collaborative	NSF	10/15/2022	8/31/2024	288583	132303	Standard Grant	6090300	GEO	Directorate For Geoscience	OPP
34	CDS&E: Co	NSF	10/1/2022	7/31/2023	270802	203483	Standard Grant	5090000	CSE	Direct For Computer & Info	OAC
35	Collaborative	NSF	11/15/2022	10/31/2023	12007	12007	Standard Grant	6030000	GEO	Directorate For Geoscience	EAR
36	Collaborative	NSF	11/15/2022	10/31/2023	7390	7390	Standard Grant	6030000	GEO	Directorate For Geoscience	EAR
37	NSF-BSF: AF: Sr	NSF	10/1/2022	9/30/2024	453382	430152	Standard Grant	5010000	CSE	Direct For Computer & Info	CCF
38	NSF Student Tr	NSF	11/1/2022	10/31/2023	5000	5000	Standard Grant	5090000	CSE	Direct For Computer & Info	OAC
39	CAREER: Elucid	NSF	11/15/2022	3/31/2023	499593	97162	Continuing Grant	3090000	MPS	Direct For Mathematical &	CHE
40	Collaborative	NSF	10/15/2022	6/30/2023	944037	652363	Continuing Grant	6010000	GEO	Directorate For Geoscience	RISE

Natural Language Processing (NLP) Task Visualization:

NLP can be applied to the title of the projects that have received NSF grants to understand interesting characteristics. As well as we have access to the NSF-funded project's abstract narration. So, we can apply topic modeling techniques to find what are different topics that have received NSF grants. For achieving this, we collected titles of the projects that have received NSF grants from the years 2000 to 2023. To understand which keywords are frequent in NSF grant titles, we draw a WordCloud of the titles. Before drawing the WordCloud, we removed the stop words and unwanted character sequences from the titles. The WordCloud is presented below:

From WordCloud, we can visualize the most frequent words appearing in the NSF grant's title. We further identified some unwanted stop words such as 'sbir' from WordCloud and removed those stop words.

Then for identifying the underlying topics from the collection of titles, we used an unsupervised topic modeling technique. The algorithm we have used is called Latent Dirichlet Allocation (LDA) [1]. LDA is an unsupervised machine learning technique used for identifying topics from a text collection. LDA can also be used for classifying documents and grouping the documents based on topic similarity. We used the python package 'gensim' for implementing the LDA algorithm.

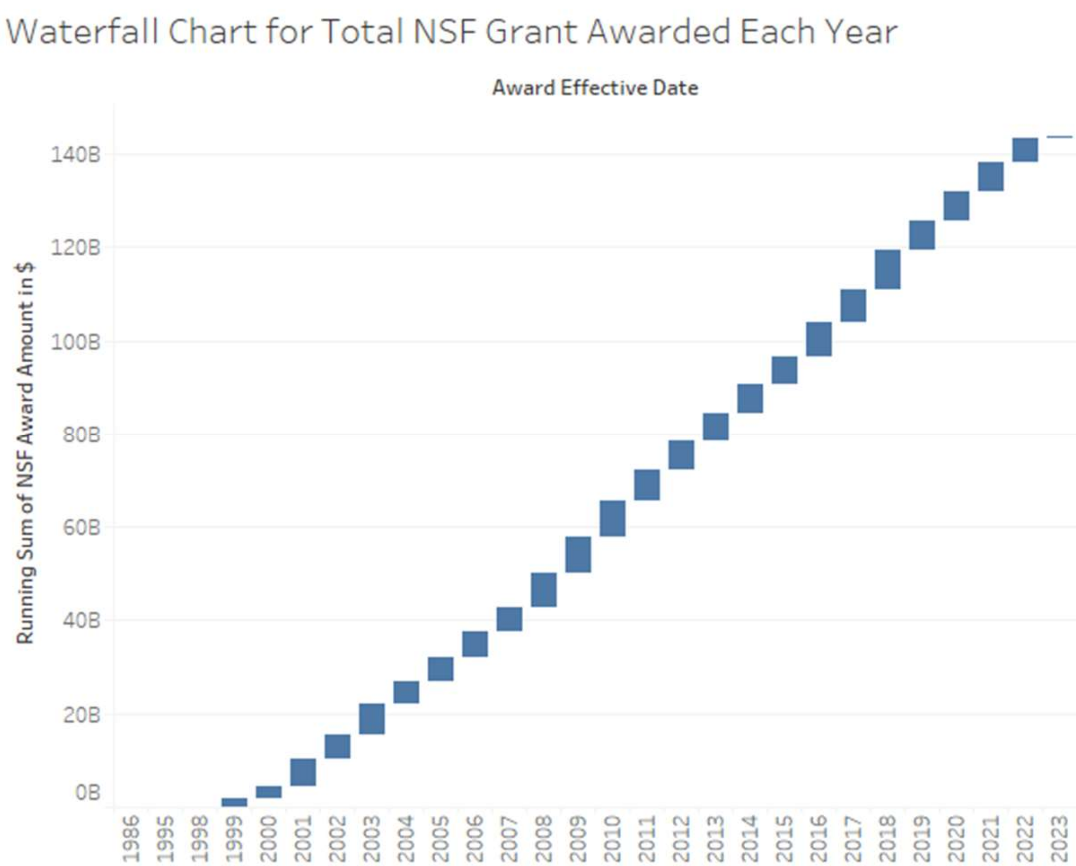
By analyzing the titles of NSF-funded projects, we can identify what are the topics that receive the most grants as well as visualizing which keywords appear frequently in NSF grant titles. Moreover, we have analyzed the abstracts of the funded NSF grants to understand the dominant topics and understand which document contains which topics.

Result

Tableau Visualization

We have made several visualizations in the tableau desktop to analysis the data and made some recommendations.

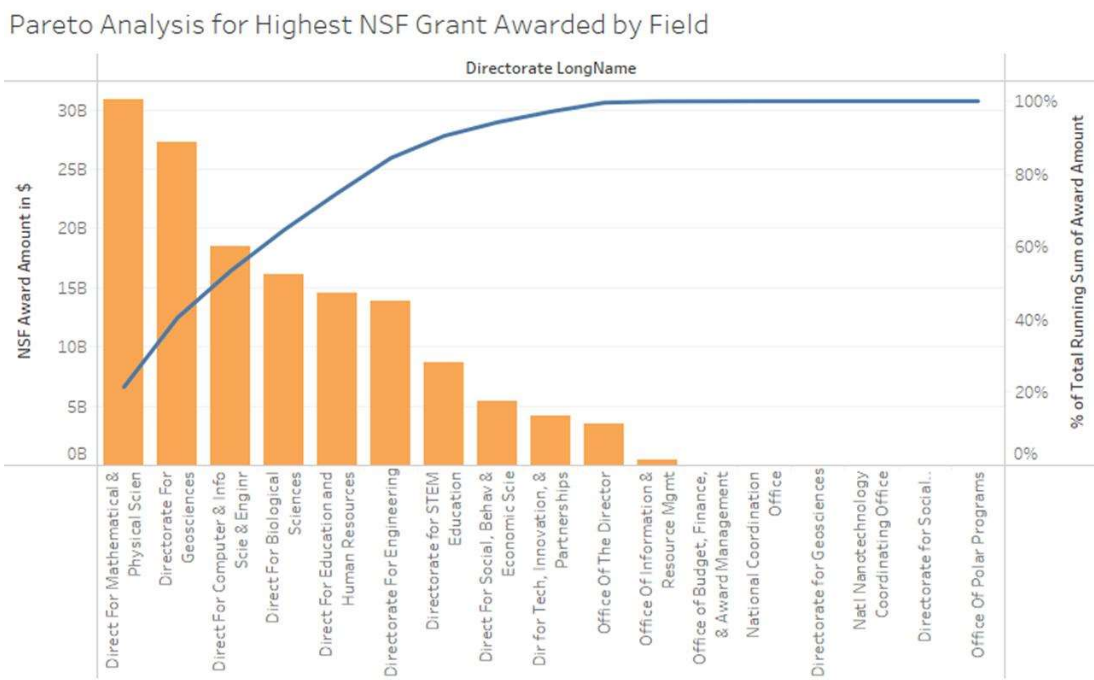
At first, this Waterfall Chart is showing the running total amount of NSF grants awarded from the year 1986 to 2023. 3. Total of $140B has been awarded over 35 years and highest amount of NSF grant was awarded in 2008, 2009, and 2010 and again in 2017, 2018, and 2019, which was more than $7B.



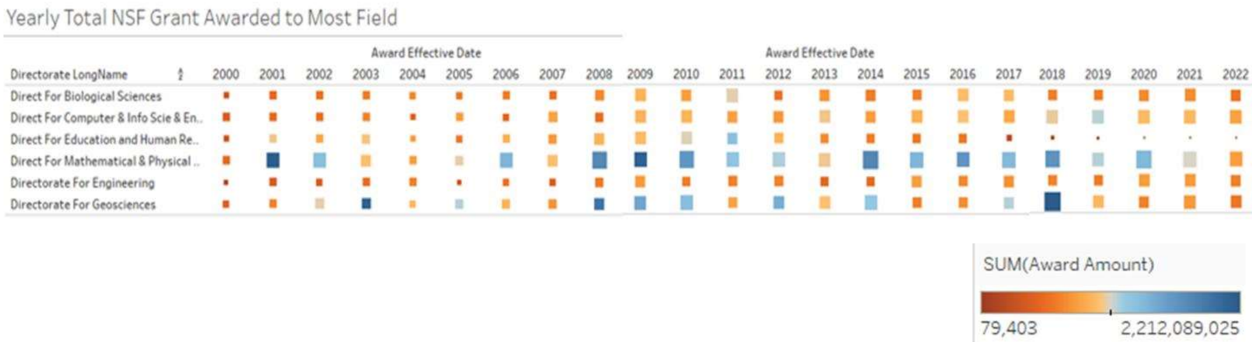
Then, we tried to find out which Directorate Field of studies received a significant amount of NSF grant. Pareto analysis shows that

- Directorate for Mathematical & Physical Science
- Directorate for Geoscience
- Directorate for Computer & Information Science & Engineering
- Directorate for Biological science
- Directorate for Education and human resource
- Directorate for Engineering

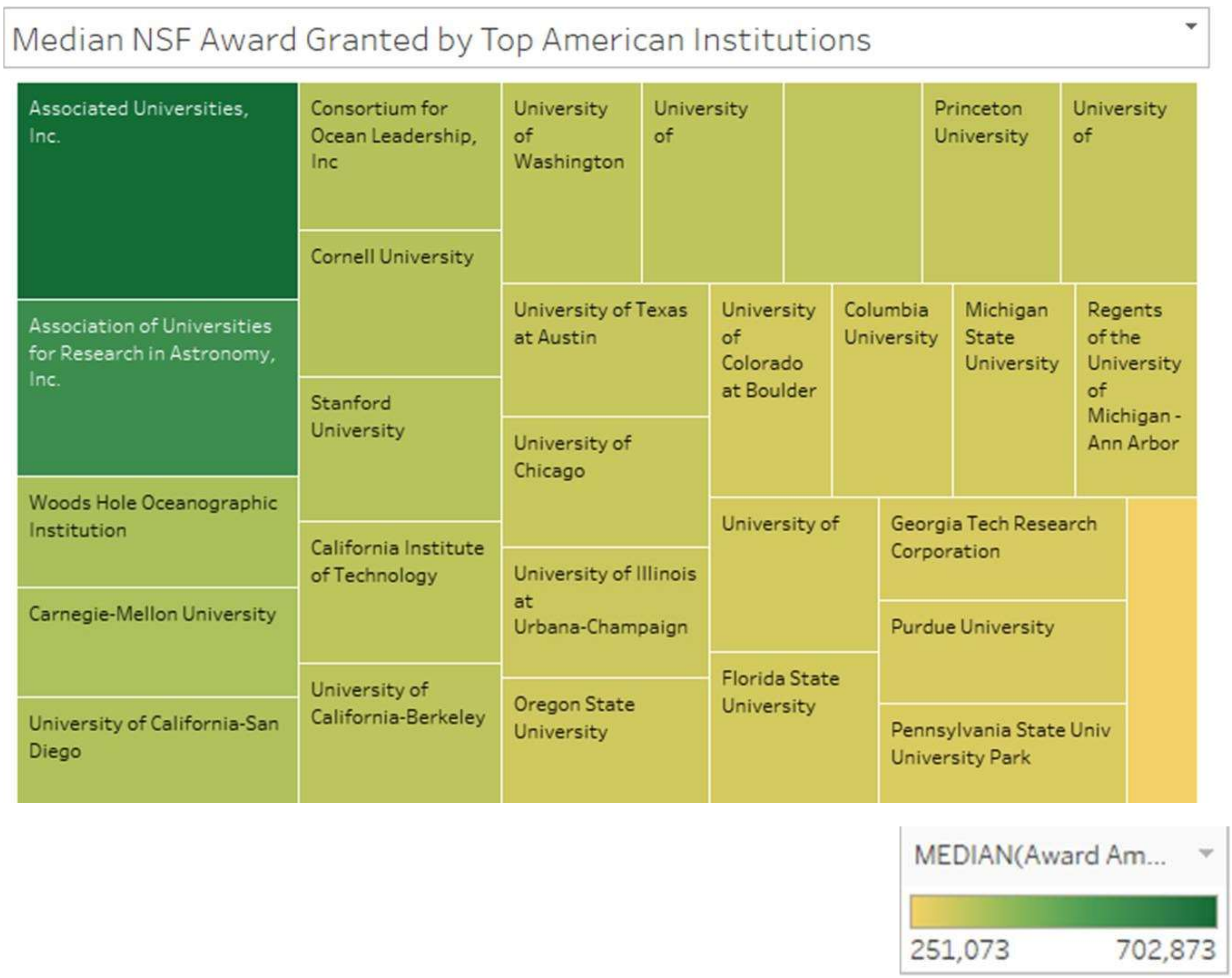
These fields received 80% of the total NSF grant over these 35 years.



After that, we tried to see which Directorate field of study received the highest amount of total NSF Grant each year over last 22 years. The highest amount of NSF grants has been awarded to Directorate of Mathematical & Physical Science Directorate for last 22 years constantly. Whereas, Directorate of Geoscience has received over $2B during 2018 and Grant for Directorate for Computer & Information Science & Engineering has been increased since 2018. On the other hand, Directorate for Education and Human Resource has been losing NSF grant since 2016.



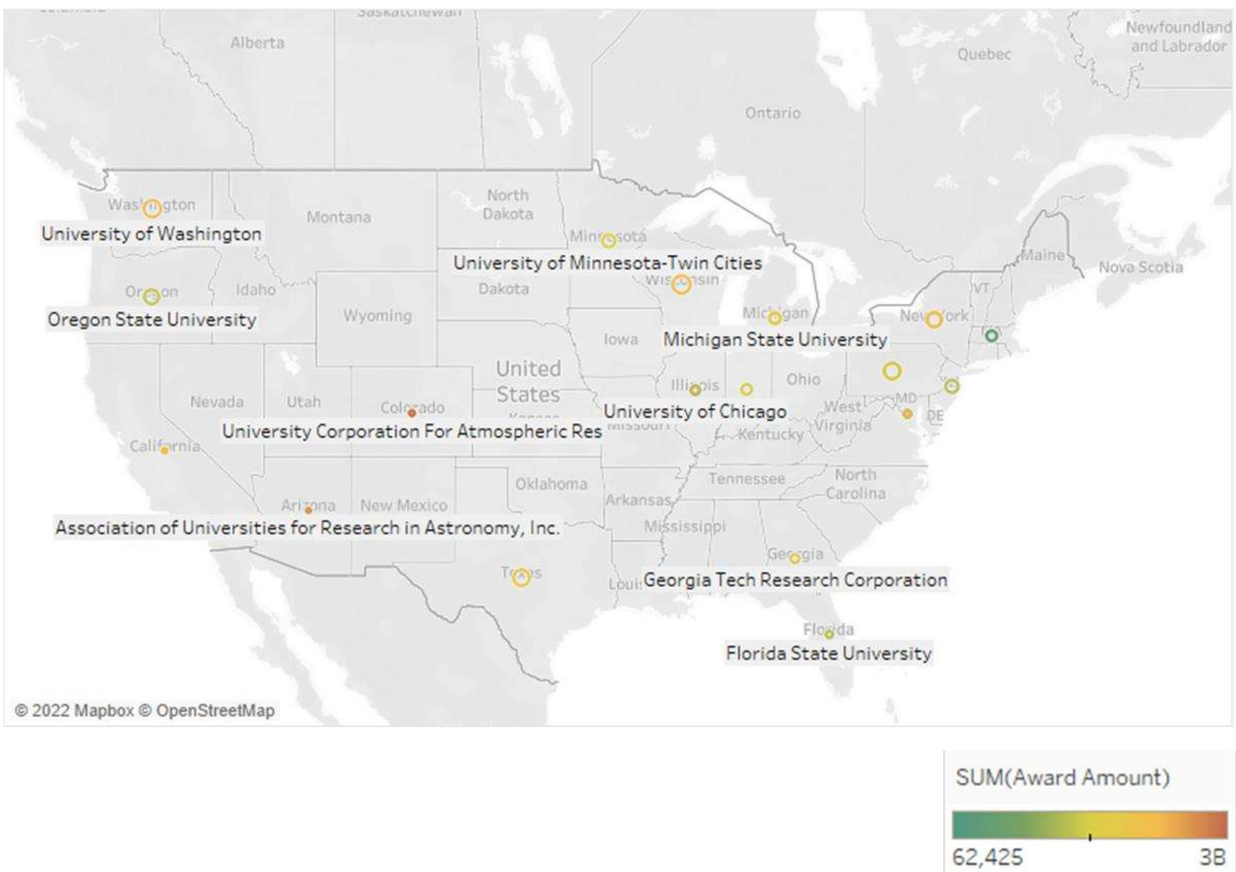
We also tried to analysis the institutions which received the highest median NSF grant each year over these 6 Directorate fields of studies. We listed 29 Institutions which received the highest Median NSF grant each year. These institutions are working on a focused 6 Directorate field of studies.



Finally, we tried to locate those 29 institutions in United States Map. We have found that all the institutions that are receiving highest median NSF grant are in East coast, West coast and South of United States. There are few institutions in Colorado and Arizona that received a good amount of NSF grant too. But the institutions from the Midwest of USA like Montana, Idaho, Wyoming, North and South Dakota and other states are not receiving competitive NSF grant each year.

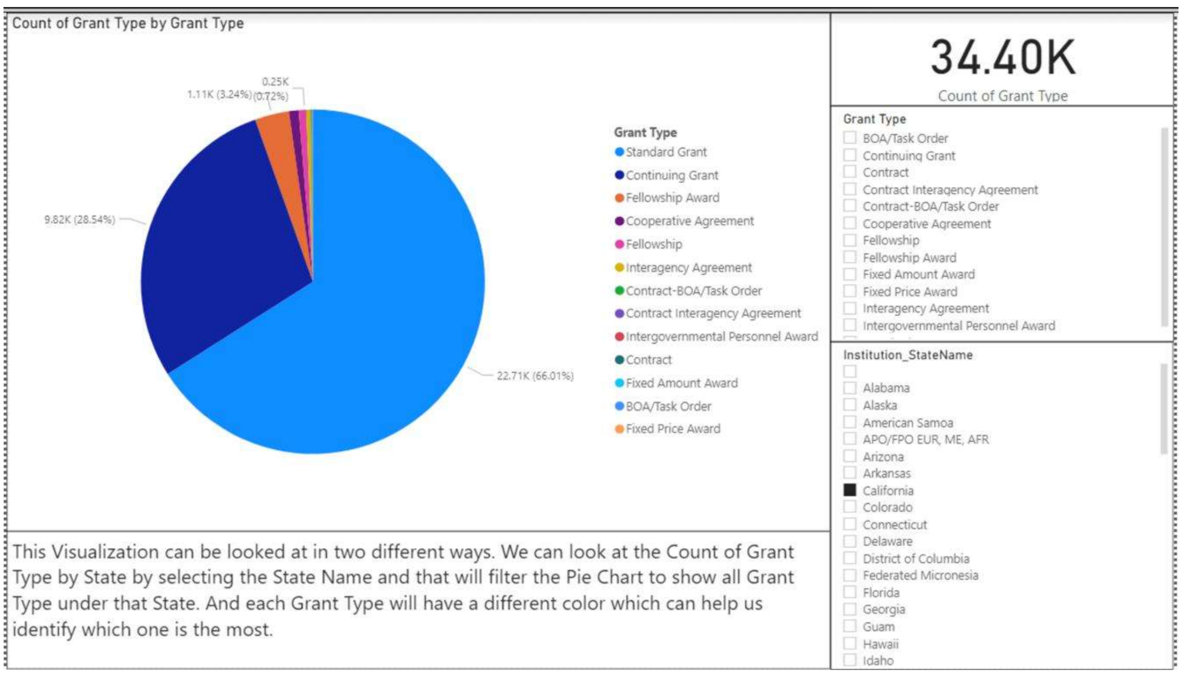
Proper steps need to be taken for the institutes from the Midwest of the USA to make them competitive with the institutions from the East and West coast of the USA. That is how overall educational improvement will happen.

Most NSF Grant Awarded Institution's State.

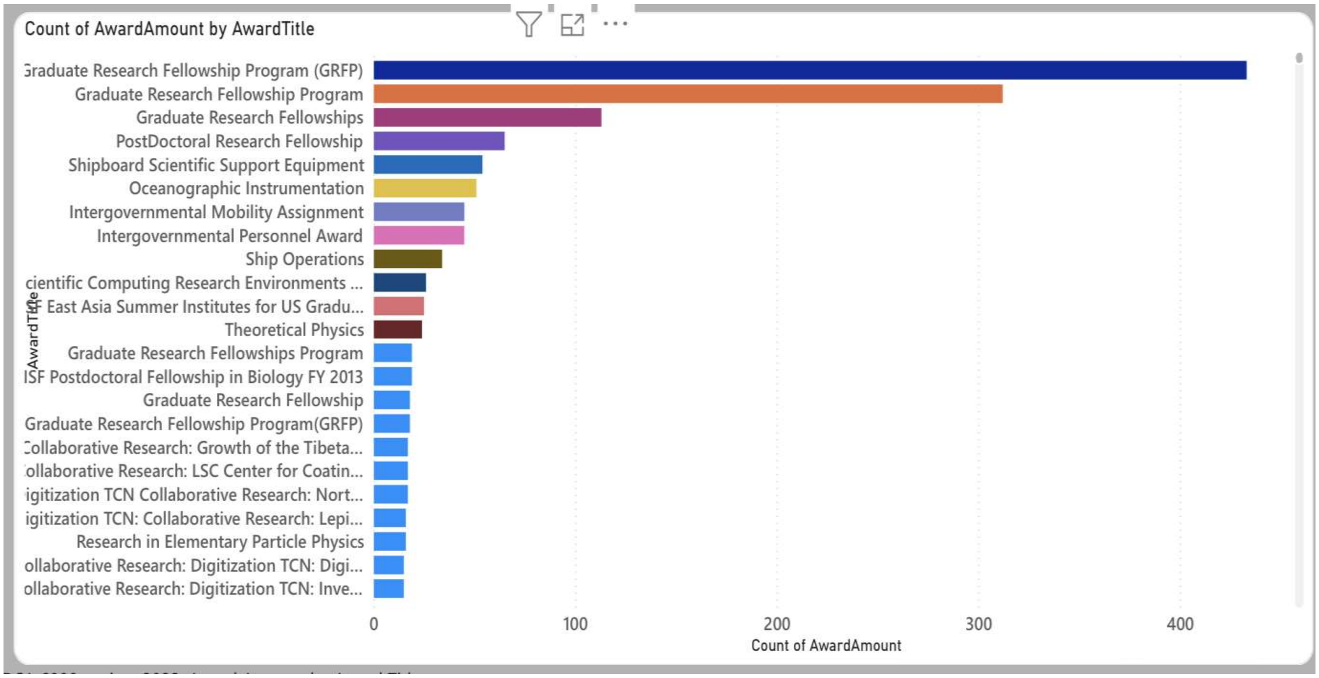


Power BI Visualizations

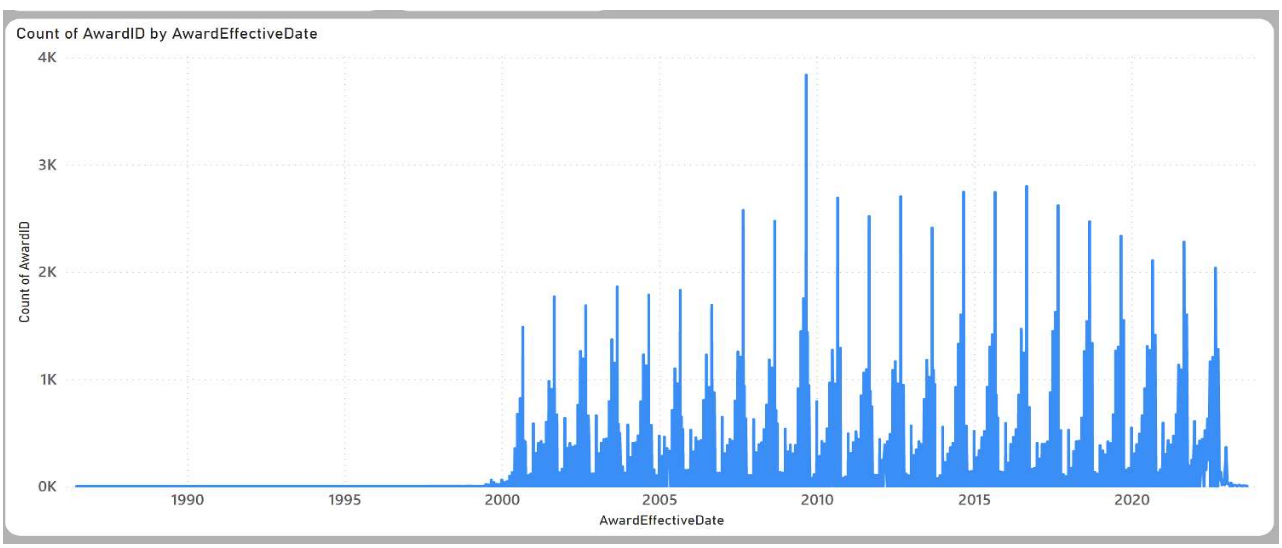
In the Visualization below, we are looking at the Count of Grant-by-Grant Type and State. When selecting the state, it will filter the Pie chart and display all Grant Types for that State. Each Grant Type has its own color which helps us quickly identify it. We can also look at it the other way around by filtering by Grant Type and that will show us the Count of that Grant in each State.



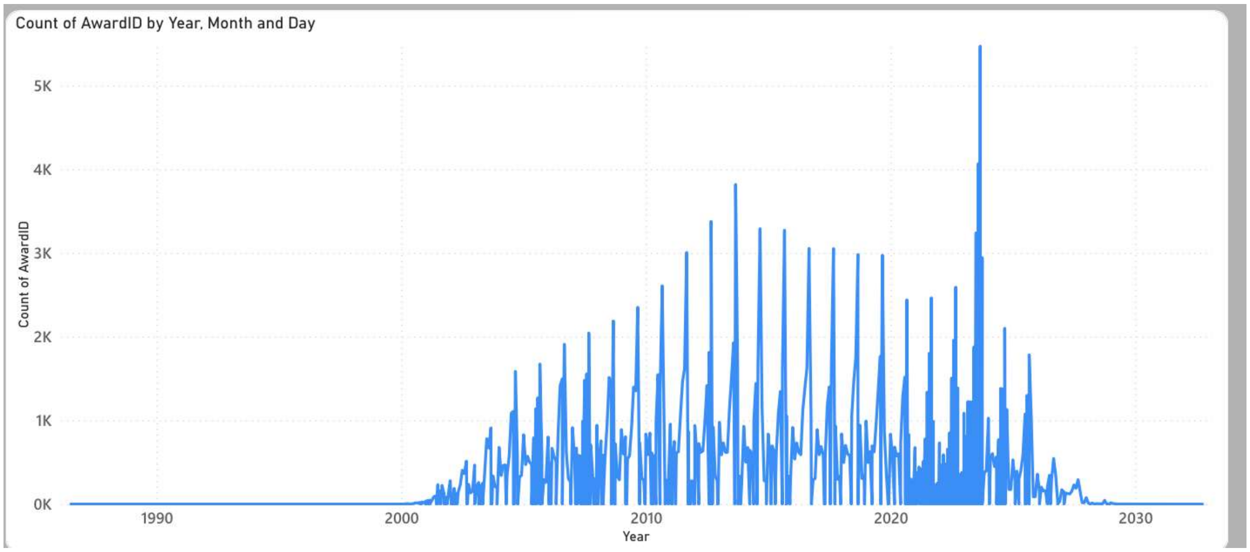
This dashboard displays the visual of Award amount by name, or title as it is displayed. The top Award granted in our National Science Foundation data is "Graduate Research Fellowship Program (GRFP)" with an award amount of 433. the Slicer provided to select the award title to help the data experts to analyze Award by single name.



This dashboard displays the bar chart of the Award effective date compared to the Award ID; we are showing when the Award was granted to those Ids in our data. Based on the visual below it's saying that the most awards granted were mostly clustered between 2009 and 2011 and it looks to continue to be in the high numbers of awards granted up to the year 2020.

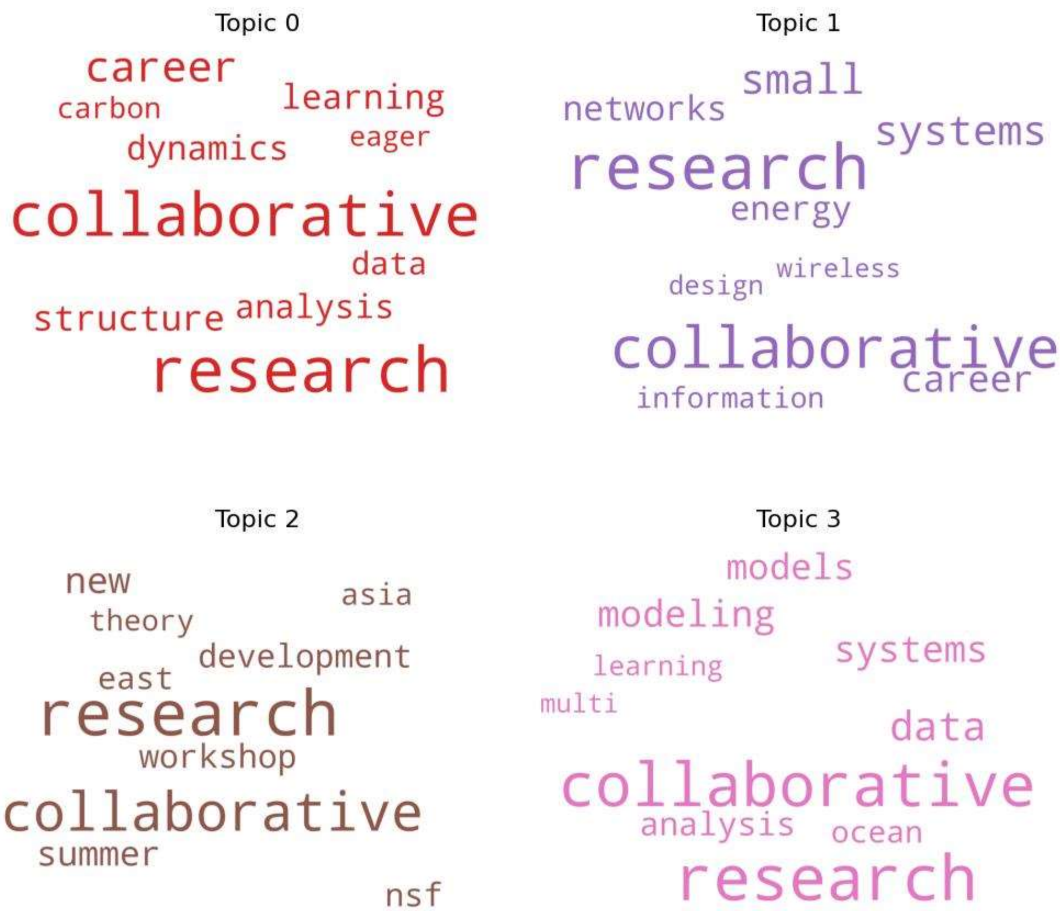


This dashboard displays the bar chart of the Award expiration date compared to the Award ID; we are showing when the expiration date was for the Award granted to those Ids in our data. Based on the visual below its saying that the most awards for expiration date for awards granted were mostly clustered to expire September 2023 and 2011 and to decline after to continue to fade in 2030.



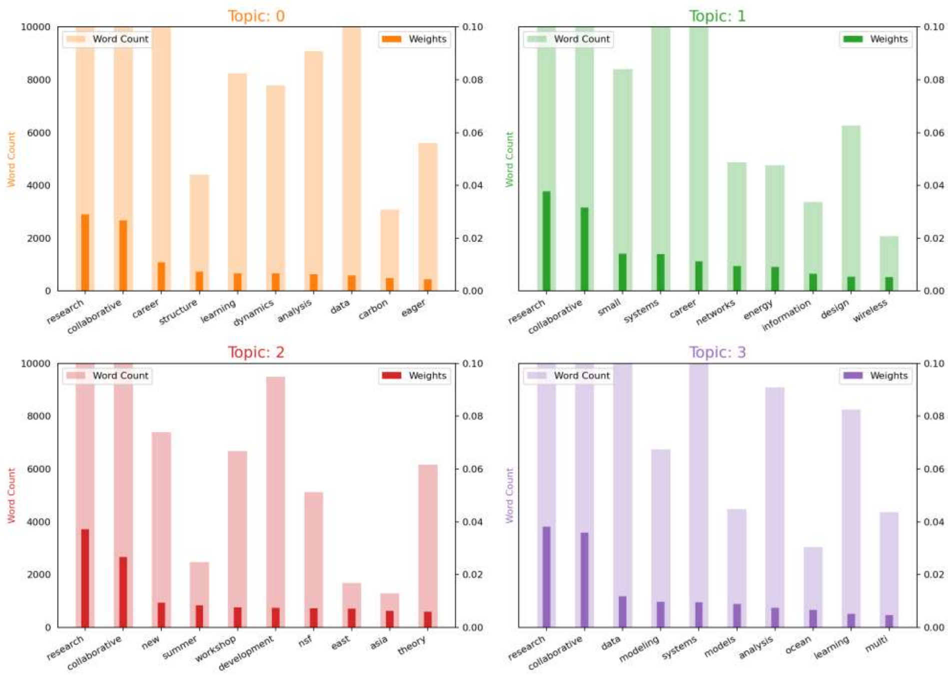
Visualize findings for NLP technique applied to NSF grant titles:

First, analyzing the titles of NSF grants, we have identified the topmost ten topics that received the most NSF grants. Below we present the topmost four topics that received the most NSF grant.



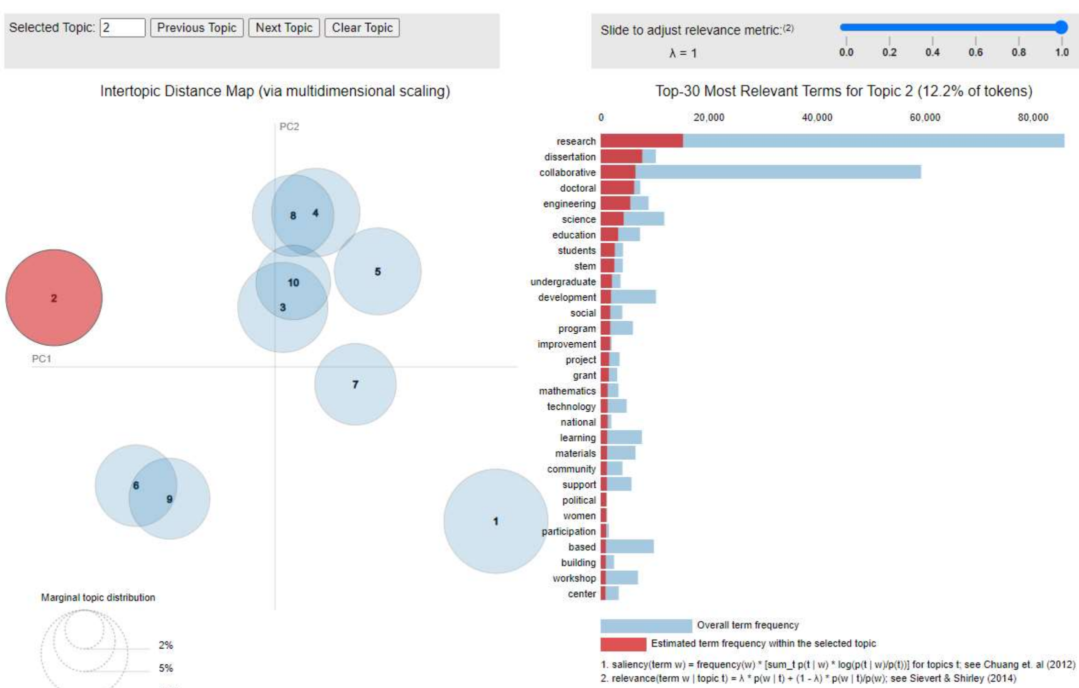
By visualizing the frequent words from each topic, we can easily identify what the topic is about. For example, in topic 0, some representative keywords are ‘data’, ‘analysis’, and ‘learning’. So, we can guess that the topic that received the most grants is machine learning/data science topic.

Next, we analyzed the frequency of representative words from each topic to understand which keywords appear frequently in the NSF title. Below we present a bar chart to show the frequency of different words for each topic.



From the bar chart above, we can easily visualize the dominant keywords in NSF titles and understand their frequency.

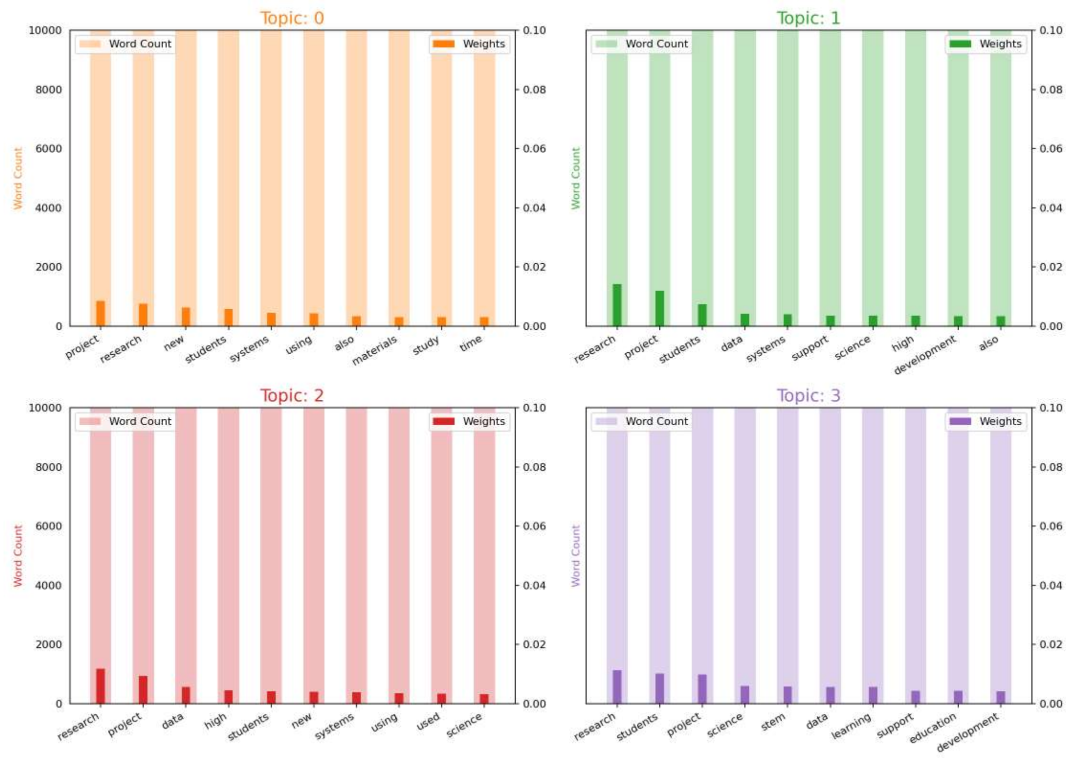
Our last visualization for the NSF title is an interactive visualization of LDA results. From this graph, we can understand what the keywords of the ten most important NSF-funded topics are. We can also visualize the most important keywords of each topic and also, the distance measure among the topics.



Here, we are showing the top 30 most relevant keywords for Topic 2. In the input box of 'Selected Topic', we can change the number to visualize keywords for different topics. We can also visualize the cluster distance among topics on the left side. For example, Topic 2 and Topic 1 have a large cluster distance, which indicates the two topics are completely different.

Visualize findings for NLP technique applied to NSF grant abstract:

Similarly, we applied LDA on the funded NSF grant abstract to identify the different keyword frequencies that appeared in the abstract of the NSF grant, cluster them into different topics, and to summarize each document's abstract. Below is the word frequency map for the top four topics.

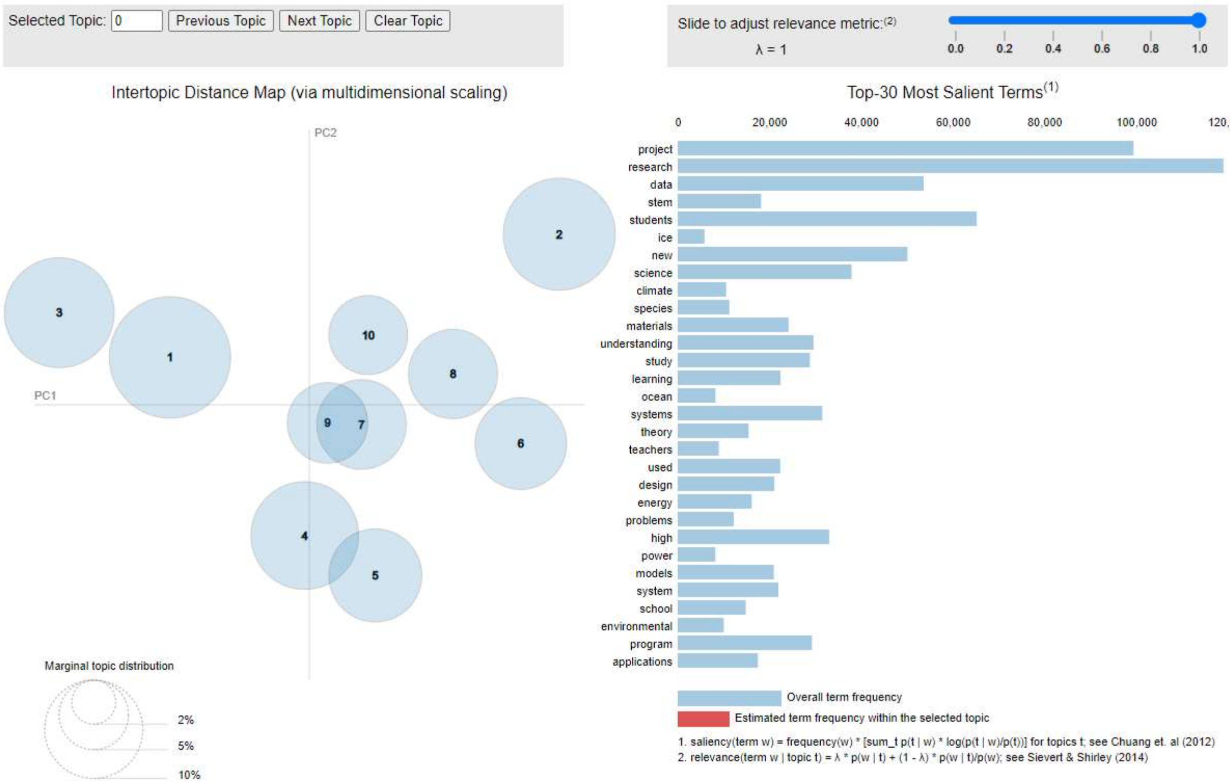


Next, we want to identify which topic is dominant within each abstract and text representation to summarize the document's abstract.

	Document_No	Dominant_Topic	Topic_Perc_Contrib	Keywords	Text
0	0	0.0	0.9929	project, research, new, students, systems, usi...	[recent, surge, induced, seismicity, midwester...
1	1	6.0	0.9842	research, project, students, science, universi...	[objective, research, investigate, integration...
2	2	7.0	0.7434	research, project, new, understanding, data, m...	[simple, sub, ice, investigation, marine, plan...
3	3	9.0	0.9384	research, project, data, new, students, also, ...	[project, involves, research, empirical, metho...
4	4	8.0	0.4838	project, research, new, students, science, und...	[proposed, research, focuses, materials, growt...
5	5	6.0	0.6224	research, project, students, science, universi...	[gordon, award, supports, professor, mark, gor...
6	6	9.0	0.6650	research, project, data, new, students, also, ...	[broad, agreement, indigent, defense, counsel,...
7	7	5.0	0.3755	research, project, students, new, also, progra...	[assessing, assessments, historical, philosoph...
8	8	2.0	0.6124	research, project, data, high, students, new, ...	[yang, yong, conventional, cell, culture, meth...
9	9	7.0	0.7101	research, project, new, understanding, data, m...	[goal, research, understand, large, stand, rep...

In the table above, we can see which topic is dominant within each document, what are the keywords of the dominant topic, and last, the 'Text' column presents the representative text that summarizes the document.

Lastly, we present the LDA map summary for complete visualization of our LDA analysis result below.



Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0
0.2
0.4
0.6
0.8
1.0
Intertopic Distance Map (via multidimensional scaling)
Top-30 Most Salient Terms(1)
PC2
PC1
1
2
3
4
5
6
7
8
9
10
Marginal topic distribution
2%
5%
10%
0
20,000
40,000
60,000
80,000
100,000
120,0
project
research
data
stem
students
ice
new
science
climate
species
materials
understanding
study
learning
ocean
systems
theory
teachers
used
design
energy
problems
high
power
models
system
school
environmental
program
applications
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Conclusion

Analyzing the NSF grant application from the year 2000 to 2022, we found which region in USA has received the most grant and uncovered that the highest dollar amount of NSF grant was given to Directorate for Mathematical & Physical Science which is a total of $30 Billion. We also analyzed the abstract and title of the NSF proposal using the Natural Language Processing (NLP) technique.

We identified that Machine Learning/Data analysis is the topic that has received the most grant. We also identified descriptive keywords that appear most frequently in the NSF proposals and titles. We also presented a way to summarize each proposal's abstract and identified the underlying dominant topic of each proposal. The dataset collection techniques, dataset collection code, and NLP visualization code are attached to the document.

References

[1] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." Journal of machine Learning research 3, no. Jan (2003): 993-1022.

[2] NSF Award Search. NSF AWARD SEARCH: Simple search. (n.d.). Retrieved December 10, 2022, from https://www.nsf.gov/awardsearch

Python code for data parsing: file:///Users/yara/Downloads/NSF_data_collection.py

Appendices

NLP visualization Code:

```
import pandas as pd
import os
import csv

import gensim
from gensim.utils import simple_preprocess
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from wordcloud import WordCloud
import pyLDAvis
import pyLDAvis.gensim_models as gensimvis
import gensim.corpora as corpora
from pprint import pprint
from matplotlib import pyplot as plt
from wordcloud import WordCloud, STOPWORDS
import matplotlib.colors as mcolors
from collections import Counter


#load dataset
input_fd = open('dataset_nlp.csv', encoding="utf8", errors='ignore')
dataset=pd.read_csv(input_fd)

###################### data preprocessing

# Remove punctuation
dataset['AwardTitle_clean'] = dataset['AwardTitle'].str.replace('[^\w\s]',' ')

# Convert the titles to lowercase
dataset['AwardTitle_clean'] = dataset['AwardTitle_clean'].str.lower()

print(dataset['AwardTitle_clean'].head())

# Join the different processed titles together.
long_string=dataset['AwardTitle_clean'].str.cat(sep=' ')

# Create a WordCloud object
wordcloud = WordCloud(background_color="white", max_words=5000, contour_width=3, contour_color='steelblue')
# Generate a word cloud
wordcloud.generate(long_string)
# Visualize the word cloud
image=wordcloud.to_image()
image.save("word_cloud.png")

stop_words = stopwords.words('english')
stop_words.extend(['from', 'subject', 're', 'edu', 'use', 'th', 'us', 'ii', 'fy', 'sbir'])

def sent_to_words(sentences):
    for sentence in sentences:
        # deacc=True removes punctuations
        yield(gensim.utils.simple_preprocess(str(sentence), deacc=True))

def remove_stopwords(texts):
    return [[word for word in simple_preprocess(str(doc))
             if word not in stop_words] for doc in texts]
```

```python

data = dataset.AwardTitle_clean.values.tolist()
data_words = list(sent_to_words(data))

# remove stop words
data_words = remove_stopwords(data_words)
print(data_words[:2][0][:20])

# Create Dictionary
id2word = corpora.Dictionary(data_words)
# Create Corpus
texts = data_words
# Term Document Frequency
corpus = [id2word.doc2bow(text) for text in texts]
# View
print(corpus[:1][0][:30])

# number of topics
num_topics = 10
# Build LDA model
lda_model = gensim.models.LdaMulticore(corpus=corpus,
                                       id2word=id2word,
                                       num_topics=num_topics)
# Print the Keyword in the 10 topics
pprint(lda_model.print_topics())
doc_lda = lda_model[corpus]


cols = [color for name, color in mcolors.TABLEAU_COLORS.items()]  # more colors: 'mcolors.XKCD_COLORS'

cloud = WordCloud(stopwords=stop_words,
                  background_color='white',
                  width=2500,
                  height=1800,
                  max_words=10,
                  colormap='tab10',
                  color_func=lambda *args, **kwargs: cols[i+3],
                  prefer_horizontal=1.0)

topics = lda_model.show_topics(formatted=False)

fig, axes = plt.subplots(2, 2, figsize=(10,10), sharex=True, sharey=True)

for i, ax in enumerate(axes.flatten()):
    fig.add_subplot(ax)
    topic_words = dict(topics[i][1])
    cloud.generate_from_frequencies(topic_words, max_font_size=300)
    plt.gca().imshow(cloud)
    plt.gca().set_title('Topic ' + str(i), fontdict=dict(size=16))
    plt.gca().axis('off')


plt.subplots_adjust(wspace=0, hspace=0)
plt.axis('off')
plt.margins(x=0, y=0)
plt.tight_layout()
```

```python
fig1 = plt.gcf()
plt.show()
fig1.savefig('topic_wordcloud.png')

topics = lda_model.show_topics(formatted=False)
data_flat = [w for w_list in data_words for w in w_list]
counter = Counter(data_flat)

out = []
for i, topic in topics:
    for word, weight in topic:
        out.append([word, i , weight, counter[word]])

df = pd.DataFrame(out, columns=['word', 'topic_id', 'importance', 'word_count'])

# Plot Word Count and Weights of Topic Keywords
fig, axes = plt.subplots(2, 2, figsize=(14,10), sharey=True, dpi=160)
cols = [color for name, color in mcolors.TABLEAU_COLORS.items()]

for i, ax in enumerate(axes.flatten()):
    ax.bar(x='word', height="word_count", data=df.loc[df.topic_id==i, :], color=cols[i+1], width=0.5, alpha=0.3, label='Word Count'
    ax_twin = ax.twinx()
    ax_twin.bar(x='word', height="importance", data=df.loc[df.topic_id==i, :], color=cols[i+1], width=0.2, label='Weights')
    ax.set_ylabel('Word Count', color=cols[i+1])
    ax_twin.set_ylim(0, 0.10); ax.set_ylim(0, 10000)
    ax.set_title('Topic: ' + str(i), color=cols[i+1], fontsize=16)
    ax.tick_params(axis='y', left=False)
    ax.set_xticklabels(df.loc[df.topic_id==i, 'word'], rotation=30, horizontalalignment= 'right')
    ax.legend(loc='upper left'); ax_twin.legend(loc='upper right')

fig.tight_layout(w_pad=2)
fig.suptitle('Word Count and Weights of Topic Keywords', fontsize=22, y=1.05)
fig2 = plt.gcf()
plt.show()
fig2.savefig('topic_word_frequency.png')

pyLDAvis.enable_notebook()
vis = gensimvis.prepare(lda_model, corpus, dictionary=lda_model.id2word)
vis
pyLDAvis.save_html(vis,"title_visualiation.html")

dataset_abstract = dataset.sample(n = 50000)

dataset_abstract['AbstractNarration'] = dataset_abstract['AbstractNarration'].str.replace('[^\w\s]',' ')

# Convert the titles to lowercase
dataset_abstract['AbstractNarration'] = dataset_abstract['AbstractNarration'].str.lower()

print(dataset_abstract['AbstractNarration'].head())

stop_words.extend(['it', 'gt', 'br', 'lt'])

def sent_to_words(sentences):
    for sentence in sentences:
        # deacc=True removes punctuations
        yield(gensim.utils.simple_preprocess(str(sentence), deacc=True))
```

```python

def remove_stopwords(texts):
    return [[word for word in simple_preprocess(str(doc))
             if word not in stop_words] for doc in texts]

data_abs = dataset_abstract.AbstractNarration.values.tolist()
data_words_abs = list(sent_to_words(data_abs))

# remove stop words
data_words_abs = remove_stopwords(data_words_abs)
print(data_words_abs[:2][0][:20])

# Create Dictionary
id2word_abs = corpora.Dictionary(data_words_abs)
# Create Corpus
texts_abs = data_words_abs
# Term Document Frequency
corpus_abs = [id2word_abs.doc2bow(text) for text in texts_abs]
# View
print(corpus_abs[:1][0][:30])

num_topics = 10
# Build LDA model
lda_model_abs = gensim.models.LdaMulticore(corpus=corpus_abs,
                                           id2word=id2word_abs,
                                           num_topics=num_topics)
# Print the Keyword in the 10 topics
pprint(lda_model_abs.print_topics())
doc_lda_abs = lda_model_abs[corpus_abs]

pyLDAvis.enable_notebook()
vis_abs = gensimvis.prepare(lda_model_abs, corpus_abs, dictionary=lda_model_abs.id2word)
vis_abs

def format_topics_sentences(ldamodel=None, corpus=corpus, texts=data):
    # Init output
    sent_topics_df = pd.DataFrame()

    # Get main topic in each document
    for i, row_list in enumerate(ldamodel[corpus]):
        row = row_list[0] if ldamodel.per_word_topics else row_list
        # print(row)
        row = sorted(row, key=lambda x: (x[1]), reverse=True)
        # Get the Dominant topic, Perc Contribution and Keywords for each document
        for j, (topic_num, prop_topic) in enumerate(row):
            if j == 0:  # => dominant topic
                wp = ldamodel.show_topic(topic_num)
                topic_keywords = ", ".join([word for word, prop in wp])
                sent_topics_df = sent_topics_df.append(pd.Series([int(topic_num), round(prop_topic,4), topic_keywords]), ignore_index=True)
            else:
                break
    sent_topics_df.columns = ['Dominant_Topic', 'Perc_Contribution', 'Topic_Keywords']

    # Add original text to the end of the output
    contents = pd.Series(texts)
    sent_topics_df = pd.concat([sent_topics_df, contents], axis=1)
```

```python

    # Add original text to the end of the output
    contents = pd.Series(texts)
    sent_topics_df = pd.concat([sent_topics_df, contents], axis=1)
    return(sent_topics_df)


df_topic_sents_keywords = format_topics_sentences(ldamodel=lda_model_abs, corpus=corpus_abs, texts=data_words_abs)

# Format
df_dominant_topic = df_topic_sents_keywords.reset_index()
df_dominant_topic.columns = ['Document_No', 'Dominant_Topic', 'Topic_Perc_Contrib', 'Keywords', 'Text']
df_dominant_topic.head(10)

topics = lda_model_abs.show_topics(formatted=False)
data_flat = [w for w_list in data_words_abs for w in w_list]
counter = Counter(data_flat)

out = []
for i, topic in topics:
    for word, weight in topic:
        out.append([word, i , weight, counter[word]])

df = pd.DataFrame(out, columns=['word', 'topic_id', 'importance', 'word_count'])

# Plot Word Count and Weights of Topic Keywords
fig, axes = plt.subplots(2, 2, figsize=(14,10), sharey=True, dpi=160)
cols = [color for name, color in mcolors.TABLEAU_COLORS.items()]

for i, ax in enumerate(axes.flatten()):
    ax.bar(x='word', height="word_count", data=df.loc[df.topic_id==i, :], color=cols[i+1], width=0.5, alpha=0.3, label='Word Count')
    ax_twin = ax.twinx()
    ax_twin.bar(x='word', height="importance", data=df.loc[df.topic_id==i, :], color=cols[i+1], width=0.2, label='Weights')
    ax.set_ylabel('Word Count', color=cols[i+1])
    ax_twin.set_ylim(0, 0.10); ax.set_ylim(0, 10000)
    ax.set_title('Topic: ' + str(i), color=cols[i+1], fontsize=16)
    ax.tick_params(axis='y', left=False)
    ax.set_xticklabels(df.loc[df.topic_id==i, 'word'], rotation=30, horizontalalignment= 'right')
    ax.legend(loc='upper left'); ax_twin.legend(loc='upper right')

fig.tight_layout(w_pad=2)
fig.suptitle('Word Count and Weights of Topic Keywords', fontsize=22, y=1.05)
fig3 = plt.gcf()
plt.show()
fig3.savefig('abstract_word_frequency.png')


```