Project Report:-

Project Statement:

A large dataset of NSF funded projects (NSF is the US national funding agency which funds many scientific investigations in the US -- read more online) is available from 1959. The data consists of information of the funded projects along with several details (metadata) of the funding, e.g., the principal investigator, which university/organization got this funding, what is the amount of funding, and several similar things. The goal of the mini-project is twofold:

1. Clean the dataset (e.g., download and automatically extract the metadata to form a csv file comprising of the details like the principal investigator, which university/organization got this funding, what is the amount of funding, and all such things) and save that.
2. The second part is to gain insight from the data, e.g.,
    1. finding the topics within which the proposals fall, e.g., deep learning, healthcare, astronomy, theoretical physics, etc. this information must be available in the xml file itself or there must be a code for which you'll need to search NSF site a little more), and also their corresponding funding amount. The insight we gain is what are the topics that receive the most funding.
    2. each proposal has an abstract (brief description of the problem and their proposed solution. We want to extract what are the topics and their proposed solutions by running NLP tools, e.g., topic modeling, from this abstract. The insight is which type of solution technique is mostly funded in each area of research.
    3. do these exercise for each decade starting from 1960, and show how the above insights change over time.
    4. final goal: NSF also has a funding policy document. Do the insights we got so far match with their stated policies?

1. First problem - Collecting and Cleaning Data-Set

The given Data-set in the NSF website was present in a zip file format for a particular year.

After extracting the zip data file ,It came out to be a number of XML files that had the information about the awards corresponding researches that were written in XML format.

So for each year and each research there was an XML present in that particular zip of a specific year.

So the task now arises to clean the data and convert it into CSV format.

I found out three ways to Convert an XML to CSV which are as follows:

- Using Python Element Tree.
- Using Excel Work-Book.
- Running SQL Queries (creating a database and table for each XML of each year).

STEPS :-

- The Python Element Tree was a great way to automate the conversion of XML into CSV ,but it is not possible for a low configuration PC to process that amount of data using a python script.

- The second way was to use the XSD Schema file and create column headers using the XSD file as an XML map sourcing developer method and then import data of each XML file ,which can also be automated to process the complete data-set at once , but only works for a workstation or a high specification computer.

- The third way was to apply SQL queries to each XML file and create a certain column a primary key and convert it to CSV ,then bind each CSV of the respective XML to form a particular year which cannot be automated .

My Approach :-

My approach was to download each zip-file of each year and sort them according to the decade they fall in ,then extract them in a folder of the corresponding decade.After that extract the XML files in the same folder of that decade.

Then using Microsoft Excel to create a workbook and parse the XSD Schema file as a mapping source file Developer Tool to create columns for each specific data type present in XML data files.(The XSD Schema file for XML format of the Data-set is present in the NSF Site from which the data is extracted.)

Then manually import every XML file and parse them to create a CSV and delete the non-required columns .
Repeating the process for each decade and creating the CSV until the complete data-set is converted into CSV.

The reason I prefer or selected this approach was the lack of specs that my Laptop had and the large amount of data that needed to be processed .

2. The second part is to gain insight from the data

- finding the topics within which the proposals fall, e.g., deep learning, healthcare, astronomy, theoretical physics, etc. this information must be available in the xml file itself or there must be a code for which you'll need to search NSF site a little more), and also their corresponding funding amount. The insight we gain is what are the topics that receive the most funding.

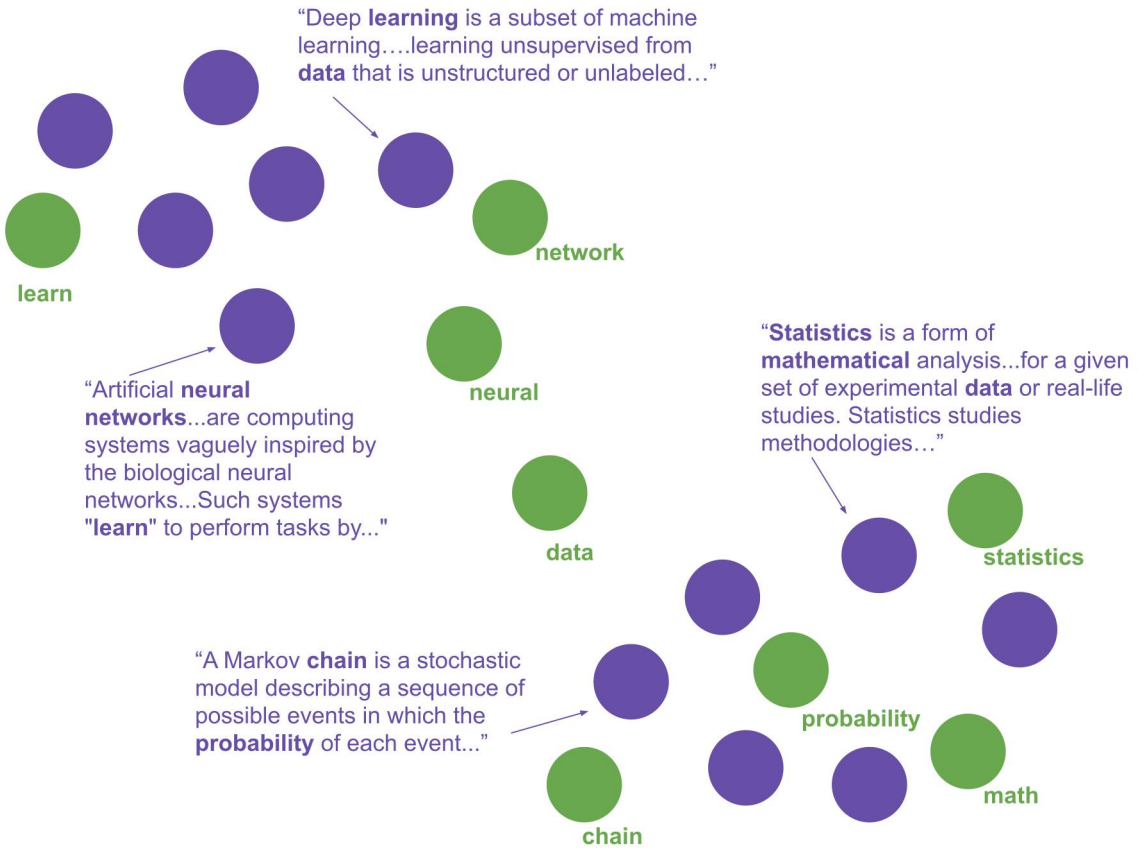
In the CSV file for a particular decade we apply topic extraction in the columns of the dataset and find most occurred topic using count of each topic with its tagged id and then creating a table that contains the total funding of that topic and the count as well then sorting them according to the increasing funding amount .

Repeating this process for each decade and storing the manipulated data with useful attributes to find trends in the funding amount with diversity around funding.

This can be done using top2vec or LDA library build in Python for topic modeling using word tokenization and word prediction as well . The working of top2vec is as follows:

- **Create jointly embedded document and word vectors using Doc2Vec or Universal Sentence Encoder or BERT Sentence Transformer.**

  Documents will be placed close to other similar documents and close to the most distinguishing words.



- **Create lower dimensional embedding of document vectors using UMAP.**

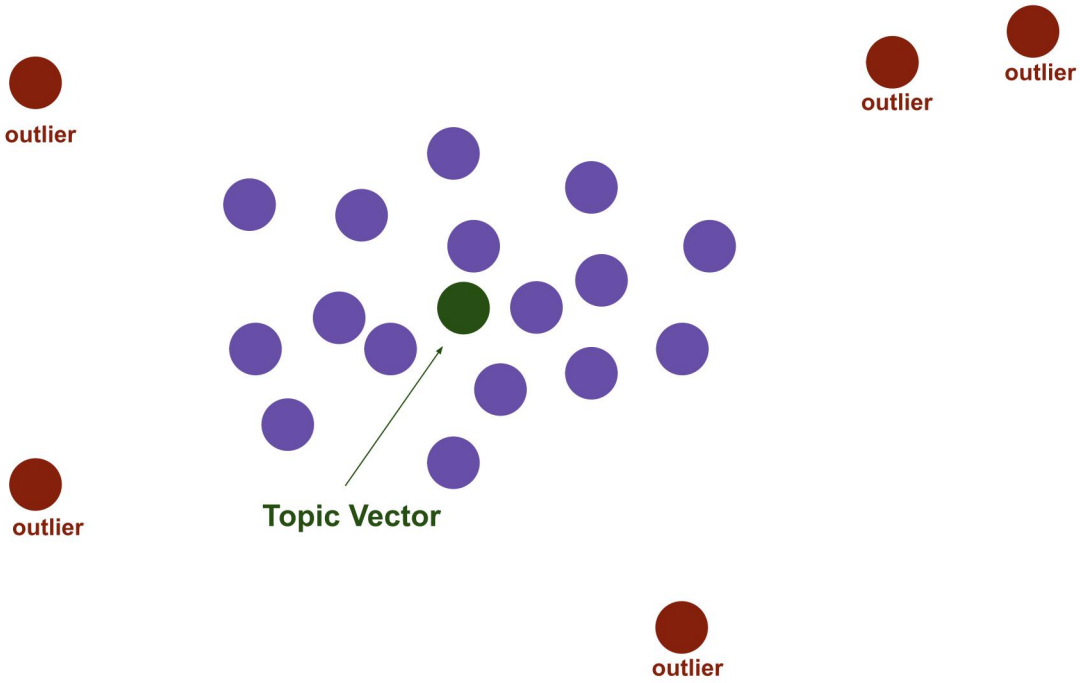
  Document vectors in high dimensional space are very sparse, dimension reduction helps for finding dense areas. Each point is a document vector.

- **Find dense areas of documents using HDBSCAN.**

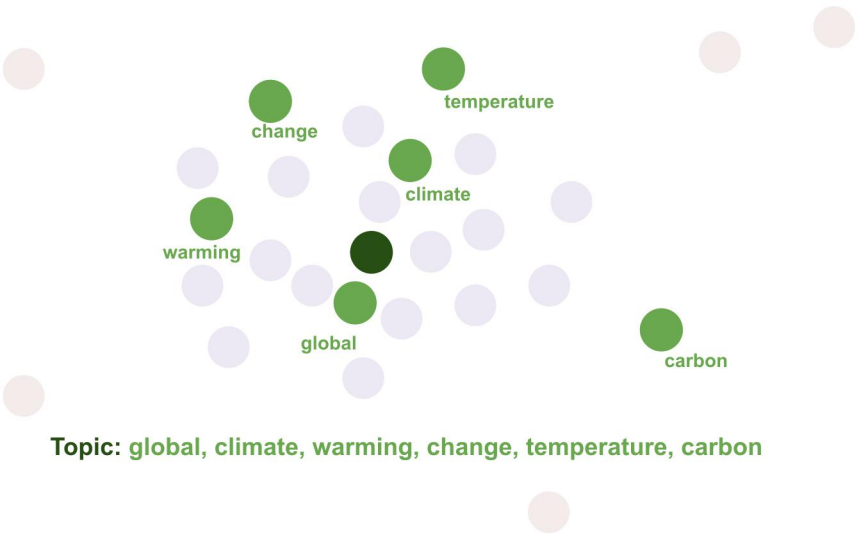
  The colored areas are the dense areas of documents. Red points are outliers that do not belong to a specific cluster.

- **For each dense area calculate the centroid of document vectors in original dimension, this is the topic vector.**

The red points are outlier documents and do not get used for calculating the topic vector. The purple points are the document vectors that belong to a dense area, from which the topic vector is calculated.



- **Find n-closest word vectors to the resulting topic vector.**The closest word vectors in order of proximity become the topic words.



- **Topic Modelling using LDA:**
  Latent Dirichlet Allocation (LDA) is one of the ways to implement Topic Modelling.

It is a generative probabilistic model in which each document is assumed to be consisting of a different proportion of topics**.**

- ## Working of LDA :

The following steps are carried out in LDA to assign topics to each of the documents:

1) For each document, randomly initialize each word to a topic amongst the K topics where K is the number of pre-defined topics.

2) For each document d:

For each word w in the document, compute:

- **P(topic t| document d)**: Proportion of words in document d that are assigned to topic t
- **P(word w| topic t)**: Proportion of assignments to topic t across all documents from words that come from w

3) Reassign topic T' to word w with probability p(t'|d)*p(w|t') considering all other words and their topic assignments

The last step is repeated multiple times till we reach a steady state where the topic assignments do not change further. The proportion of topics for each document is then determined from these topic assignments.

- ## Illustrative Example of LDA:

Let us say that we have the following 4 documents as the corpus and we wish to carry out topic modelling on these documents.

Document 1: We watch a lot of videos on YouTube.

Document 2: YouTube videos are very informative.

Document 3: Reading a technical blog makes me understand things easily.

Document 4: I prefer blogs to YouTube videos.

LDA modelling helps us in discovering topics in the above corpus and assigning topic mixtures for each of the documents. As an example, the model might output something as given below:

Topic 1: 40% videos, 60% YouTube

Topic 2: 95% blogs, 5% YouTube

Document 1 and 2 would then belong 100% to Topic 1. Document 3 would belong 100% to Topic 2. Document 4 would belong 80% to Topic 2 and 20% to Topic 1. This assignment of topics to documents is carried out by LDA modelling using the steps that we discussed in the previous section. Let us now apply LDA to some text data and analyze the actual outputs in Python.

3. Analysis of topics.

The extracted topic are displayed with their corresponding counts with their probability that depicts the favourable topic for each award that has been funded for its research.

In the second part of analysis the awards are arranged according to the topics in which they fit. This makes the analysis much easier and efficient

After the data has been configured the visualization is done using pyhton matplotlib and numpy library in which the data is represented in the form of a bar graph and a pie chart for ease in recognition of the extracted topics and their corresponding funding amount which helps us in finding the most funded topic in a decade and recognize the trend in funding amounts for various topics of different fields.

The analysis helps us in the following fields of analysis :

- Finding the topics within which the proposals fall and also their corresponding funding amount.
- The insight we gain is what are the topics that receive the most funding.
- The insight is which type of solution technique is mostly funded in each area of research.
- And how the above insights change over time.

Sources used to create the model are as follows:

- https://drive.google.com/drive/folders/1KnSlPw0IoW4cockhCoN3eQ87YwTkMBo?usp=sharing
- https://github.com/ddangelov/Top2Vec
- https://github.com/rsreetech/LDATopicModelling
- https://www.youtube.com/watch?v=r6dyk68gymk
- https://www.youtube.com/watch?v=nNvPvvuPnGs
- https://www.youtube.com/watch?v=bEaxKSQ4Av8
- https://www.geeksforgeeks.org/matplotlib-tutorial/

Shortcomings of the implementation:

- The implementation is not able to name the topics that are extracted but is able to identify them through the topic index.
- Due to lack of specifications of my laptop I was not able to process the whole data to convert more XML files and analyze the overall trend for each decade.

Future Scope of the Project:

- Using data mapping techniques the model will be able to name the topics that have are extracted and give classify the themes of each Awards.
- The model will be more automated to detect insights directly after just feeding data to it.
- More accurate results can be deduced using different classifiers and increase the overall pace of the project.

Alternative Approach:

Another approach I used to create the model is LDA Topic modeling which is described above the first decade model is created using Top2vec and named as firstDecadeModel.ipynb and the Second decade model is created using LDA topic modeling which is named as secondDecadeModel.ipynb in the Github repository.

I opted for this approach so as to test the accuracy of the topic extraction and diversity of results that the LDA topic modelling provides .

The outputs that LDA modeling and top2vec provide are confine but LDA is more accurate for a larger data-set .

Project Links :

Github Repository of the Project:

https://github.com/iceticshacker7/miniprojectiitb

The cleaned data of each decade and year is uploaded in drive and the public link to the drive is:-

https://drive.google.com/drive/folders/1KnSlPw0IoW4cockhCoN3eQ87YwTkMBo?usp=sharing

First decade CSV Link :

https://drive.google.com/file/d/18ZGDtyQsinvo1mswAB7Xp9bzO7NmL6J/view?usp=sharing

Second decade CSV Link :

https://drive.google.com/file/d/18ZGDtyQsinvo1mswAB7Xp9bzO7NmL6J/view?usp=sharing