## A) Exploring the biases in search results.

Explore the nature and extent of biases in search results. The assignment will involve conducting searches on different search engines and comparing the results to identify any biases or differences in the search results and submitting a report detailing the findings.

The report should include:

1. The methodology used to conduct the searches and collect the data.
2. Description of the search results obtained.
   a. Frequency Analysis on the biases or differences observed in the search results (more guidelines below). (You can specifically analyse on any one of the topics: privacy breach, hate speech, Fake news, related biases in search results)
3. Brief description of the possible causes for these biases in search results.
4. Brief recommendations for reducing or eliminating these biases in search results.
5. Any limitations or challenges faced during the assignment.

## B) Behavioral analysis using Drift Diffusion Models

Use the Online shoppers purchasing intention dataset in this link: https://archive.ics.uci.edu/dataset/468/online+shoppers+purchasing+intention+dataset for obtaining the below information- Model the population intention for purchase (revenue: True) using DDM to identify the mean and sd for the Response times that ended up in a purchase, report your results and interpretations. Explore the drift rates for a few specific product types and identify the products ended with quickest purchase.

Try to include visualizations and statistical tests to support your findings. All the data and code used for the data collection and analysis should be included in the report along with the steps to reproduce the results. The code should be well-documented and easy to understand. Make sure to include all relevant citations and references in the report.

Any analysis or reports based on personal data will be dealt with sensitively and confidentially. This guarantee does not apply to any data that is given to you or is available in a public domain.

Kindly contact the instructor or the teaching assistants for any concerns or clarifications regarding the scope or requirements of the assignment. TAs: Lazar Tony

lazartony23@iitk.ac.in, Amal Jude Ashwin amaljude22@iitk.ac.in

Within project evaluation: 40% code, 30% report and 30% interpretations (interpretations will also be assessed through in-person meeting reviews if needed)

Deadline for the Assignment 1: Jan 26, 2025 You are also allowed to work in teams of 2-3, Clearly mention the team members name and contributions (if any) and upload your assignments as executable notebooks/codes in HelloIITK portal within the deadline. For the review process, the teaching team will be running your notebooks and calling for an inperson interview for any clarifications.

## Specific Guidelines for part A

You can use the following as a guideline, but feel free to expand upon it as needed:

Differences that can be explored:

1. Difference in search results based on the search engine used (Google, Bing, DuckDuckGo, etc.).
2. Difference in search results based on the topic of the question (Bias based on gender, race, etc. eg. "Black girl" vs "White girl").
3. Difference in search results based on the location of the user (US, UK, India, etc.).
4. Other possible causes for differences or biases in search results (eg. phrasing of the query, time of the search, user history, etc.).

Analyses that can be performed:

1. Frequent words in the search results (ex, presenting WordClouds).
2. Frequency and ordering of results, links and domains.
3. Diversity of perspectives.
4. Other possible metrics/dimensions are for Bonus (eg. Topics, sentiments etc.).

## Starter code

Provided below is a sample code to get you started with the assignment. This is just a basic template you will need to modify and expand upon it to conduct a thorough analysis. You are also free to use any other techniques or tools that you see fit for the assignment.

In [1]:
```
import time
from selenium import webdriver
from selenium.webdriver.chrome.options import Options
from bs4 import BeautifulSoup
import re
```

```
from urllib.parse import quote

def get_search_results(search_engine_name, search_engine_url, search_phrase)
    search_url = search_engine_url + quote(search_phrase)
    chrome_options = Options()
    chrome_options.add_argument("--headless")
    chrome_options.add_argument("--incognito")

    driver = webdriver.Chrome(options=chrome_options)
    driver.get(search_url)
    time.sleep(5)
    updated_url = driver.current_url
    html = driver.page_source
    driver.quit()
    soup = BeautifulSoup(html, 'html.parser')
    results = []
    if search_engine_name == "Google":
        results = soup.find_all('div', class_='g')
    elif search_engine_name == "Bing":
        results = soup.find_all('li', class_='b_algo')
    elif search_engine_name == "Yahoo":
        results = soup.find_all('div', class_='algo-sr')
    elif search_engine_name == "DuckDuckGo":
        results = soup.find_all('article')
    results_info = []
    for result in results:
        title = ""
        link = ""
        try:
            title_container = None
            if search_engine_name == "Google":
                title_container = result.find('h3')
                link_container = result.find('a')
                if title_container:
                    title = title_container.text
                if link_container:
                    link = link_container['href']
            elif search_engine_name == "Bing":
                title_container = result.find('h2').find('a')
                link_container = result.find('h2').find('a')
                if title_container:
                    title = ''.join(title_container.find_all(string=True, re
                if link_container:
                    link = link_container['href']
            elif search_engine_name == "Yahoo":
                title_container = result.find('a')
                link_container = result.find('a')
                if title_container:
                    title = ''.join(title_container.find_all(string=True, re
                if link_container:
                    link = link_container['href']
            elif search_engine_name == "DuckDuckGo":
                title_container = result.find('h2').find('a').find('span')
                link_container = result.find('a', class_="Rn_JXVtoPVAFyGkcaX
                if title_container:
                    title = ''.join(title_container.find_all(string=True, re
                if link_container:
                        link = link_container['href']
            except:
                pass
            results_info.append({"title": title, "link": link})
    visible_eng_text = soup.get_text(separator=' ')
    visible_eng_text = re.sub(r'[^\x00-\x7F]+', ' ', visible_eng_text)
    visible_eng_text = re.sub(r'\n', ' ', visible_eng_text)
    visible_eng_text = re.sub(r'\s+', ' ', visible_eng_text)


    all_links = []
    for link in soup.find_all('a'):
        all_links.append(link.get('href'))
    search_info = {"search_engine": search_engine_name, "search_phrase": sea
    return search_info
```

In [2]:
```
search_phrases = ["sample search query"]
# search_engine_urls = {"Google": "https://www.google.com/search?q=", "Bing'
search_engine_urls = {"Google": "https://www.google.com/search?q="}
for search_phrase in search_phrases:
    for search_engine_name, search_engine_url in search_engine_urls.items():
        search_info = get_search_results(search_engine_name, search_engine_u
        print(f"{search_engine_name} search for '{search_phrase}' returned {
        for result in search_info['results']:
            print(f"Title: {result['title']}")
            print(f"Link: {result['link']}")
            print()

        # print(f"Visible text: {search_info['visible_eng_text']}")
        # print(f"All links: {search_info['all_links']}")
        print("\n")
```

```
Google search for 'sample search query' returned 14 results

Title: Sample Search Queries | YouTrack Cloud Documentation
Link: https://www.jetbrains.com/help/youtrack/cloud/sample-search-queries.ht
ml

Title: Search Query Definition - Page One Power
Link: https://www.pageonepower.com/search-glossary/search-query#:~:text=For%
20example%2C%20if%20you%20are,specific%20context%20to%20the%20keyword.

Title: What are the best ways to write clear and concise search queries?
Link: https://www.linkedin.com/advice/0/what-best-ways-write-clear-concise-s
earch-queries

Title: Sample queries - IBM
Link: https://www.ibm.com/docs/en/product-master/12.0.0?topic=language-sampl
e-queries#:~:text=Sample%20queries%20are%20provided%20for,text%20box%20in%20
the%20interface.

Title: 3 Types of Search Queries and How to Target Them - WebFX
Link: https://www.webfx.com/blog/seo/types-of-search-queries/

Title: Searching Using Query By Example (Search Developer's Guide)
Link: https://docs.marklogic.com/guide/search-dev/qbe

Title: Examples of simple syntax - Azure AI Search
Link: https://learn.microsoft.com/en-us/azure/search/search-query-simple-exa
mples

Title: Search query examples | HCL Digital Experience
Link: https://help.hcl-software.com/digital-experience/8.5/panel_help/wcm_de
v_search_form_query_examples_2.html

Title: Sample queries
Link: https://www.ibm.com/docs/en/imdm/11.6?topic=language-sample-queries

Title: Elasticsearch Query Examples — Hands-on Tutorial
Link: https://coralogix.com/blog/42-elasticsearch-query-examples-hands-on-tu
torial/

Title: Search Query Samples
Link: https://docs.qualys.com/en/cloudview/latest/search/search_queries.htm

Title: 10 Common Search Queries to Target Your Audience Better
Link: https://learn.g2.com/types-of-search-queries

Title: Query guidelines and sample queries - Search Console Help
Link: https://support.google.com/webmasters/answer/12917174?hl=en

Title: Build a search query
Link: https://doc.sitecore.com/xp/en/users/latest/sitecore-experience-platfo
rm/build-a-search-query.html
```