Justin Grima

WebCrawler and NLP System

Wednesday, December 13th, 2023

Overview

An overview of the Issue

In the dynamic job market, while exploring opportunities on online job platforms, two persistent challenges surface: deciphering diverse job posting layouts to understand key role characteristics and staying informed about companies' technical skills expectations. These challenges resonate not only from personal experience but is also evident in reputable domain-published articles and discussion forums. From the amount of information stuffed into a typical job description, the endless stream of qualifications and responsibilities to fulfill (Humphry, J. 2022), inconsistency in the job posting formats, where each job description looks different (Gere. 2015), unnecessary jargon and lack of clear direction or insight into the job's fundamental basis, can leave qualified candidates feeling confused, underqualified, and ultimately resulting in the possible deterrent of application submission, creating missed opportunities.

Where the Issue is present on the world wide web

The challenge persists on major job platforms like seek.com.au, au.indeed.com, and linkedin.com/jobs/ to name a few. Job seekers grapple with diverse layouts, inconsistent formats, and unclear expectations. Expanding these platforms entails implementing features to address these challenges, including standardizing job posting for simplified formatting and language consistency, translating complex information into accessible language, and ensuring a uniform experience for job seekers to contribute to job application efficiency, aiming to cut time-to-fill—the duration from requisition to candidate screenings (Kelly, R. 2018). Furthermore, by identifying the current trends in technical skills within the job market and specifically within the user's field of work, users can stay well-informed and up-to-date on sought-after skills. This knowledge can contribute to their success in job applications.

How machine learning can be applied to provide a solution to the Issue

To do this, I propose leveraging Natural Language Processing (NLP), a recognized field in machine learning for handling unstructured text data (MonkeyLearn. n.d.). By implementing NLP tools such as Latent Semantic Analysis (LSA), we can analyze long and 'noisy' job posts containing jargon, complicated terms, and unnecessary information, and provide a standard, simplified job summary, making them more user-friendly and comprehensible to a wider range of potential candidates (TRITON AI PTE LTD, 2023). Additionally, to identify trending technical skills within a job domain we can create an initial technical skills list by applying domain knowledge, and then implement Word2Vec, to find semantic relationships between the initial words and other closely relevant technical skill phrases/ words (Nuri, 2020) to add to our list. We can then iterate through the job postings and identify the most frequently mentioned technical skills within the job market for the desired field of study, which for this report, the sample is Data Science jobs in Brisbane. Collectively, users will have a less overwhelming, easier job search experience, with the additional knowledge of the trending technical skills in the job market currently desired by companies.

Domains

Websites consumed

The selected website for this project is www.seek.com.au; a market leader in online employment marketplaces with a multinational presence spanning Australia, New Zealand, and more (SEEK, 2019). Seek was strategically chosen for its comprehensive coverage of job postings, specifically for this report, on Data Science jobs within Brisbane. By concentrating on Data Science roles in this specific location, the web scraping initiative aims to capture a targeted sample of job listings relevant to the identified issue. This focus enhances the specificity and applicability of the data collected, allowing a more in-depth analysis of challenges and trends specifically within the Data Science job market in Brisbane.

Website/data copyright considerations

The SEEK Terms of Service outline guidelines for using SEEK's employment platforms, and restricting certain activities, including web scraping. Users are granted limited rights, but unauthorized automated data-gathering tools, such as web scraping, are explicitly prohibited. SEEK emphasizes intellectual property ownership, requiring a license for user-generated content. The company reserves the right to review and remove content, disclaims liability for user-generated material, and users compensate SEEK from related claims. However, Australian laws permit web scraping for legitimate purposes if it doesn't infringe on copyrights or cause harm to the target website (Is Web Scraping Legal in Australia? - Web Scraping FYI, n.d.). Being a school project and incorporating robots.txt (refer to Image 1 for Seek robots.txt) in the requests to SEEK, emphasizing a commitment to lawful and ethical web scraping practices, proceeding with the web scraping is acceptable. For a brief breakdown of SEEK Terms of Service, refer to Table 1.

Point	Summary
Ownership of Intellectual Property	SEEK Group asserts ownership of intellectual property rights associated with the website and apps. Users are granted limited rights for personal or employment purposes, and they retain ownership of pre-existing intellectual property. Users grant SEEK a license to use their intellectual property.
Usage Restrictions	Users are restricted from using materials and content on the website for commercial purposes without prior written permission. Activities like data mining and automated data gathering tools are prohibited without SEEK's consent. Web scraping falls under prohibited activities without proper consent.
User Content License	Users grant SEEK a worldwide, non-exclusive, royalty-free license to reproduce, adapt, distribute, and publish User Content. This license is discontinued after the User Content is removed from the website.
Right to Review and Remove Content	SEEK reserves the right to review and remove User Content violating terms, applicable laws, or posing harm.
Liability Limitations	SEEK disclaims responsibility for the truthfulness, accuracy, or reliability of User Content. The platform acts as a venue, and SEEK is not responsible for the quality, safety, or legality of jobs, resumes, or any user-generated content.
Indemnity	Users agree to defend, indemnify, and hold the SEEK Group harmless from claims arising from User Content, content use, or breaches of terms.
Governing Law	The contractual relationship is with SEEK Limited, and the terms are governed by the laws of Victoria, Australia. Users submit to the exclusive jurisdiction of the courts of Victoria.
Robots.txt Directives	The robots.txt file provides directives for web crawlers. Unrestricted access is granted to certain bots. Default directives limit access to specific URLs. Disallowed bots are specified (e.g., Facebook, LinkedIn, Baiduspider), and exceptions (e.g., for GPTBot) are outlined, emphasizing the importance of compliance with web scraping guidelines.
Web Scraping Prohibition	Users are explicitly prohibited from engaging in web scraping activities without proper consent, highlighting the significance of adhering to usage restrictions and seeking approval before conducting automated data gathering.

Table 1: Brief breakdown of SEEK Terms of Service (SEEK - Terms and Conditions, n.d.)

```
# robots.txt file for www.seek.com.au

# Unrestricted access
User-agent: Mediapartners-Google
User-agent: AdIdxBot
Disallow:

# Default directives
User-agent: *
Disallow: */job/
Disallow: *?returnUrl=
Disallow: *?page=

# Disallowed bots
User-agent: facebookexternalhit
User-agent: LinkedInBot
User-agent: Baiduspider
Disallow: /

# Exceptions
User-agent: GPTBot
Disallow: /companies
Disallow: */job/
```

Image 1: Seek robots.txt.

Methodology of applying the web crawler/scraper

The methodology involves using a web crawler to extract job titles and URLs from SEEK's job listings, ensuring compliance with robots.txt and web scraping guidelines. A random User-Agent is generated for bot detection avoidance (Vanderwillik, J. n.d.). Headers play a crucial role in ethical web scraping practices and delays simulate human-like behavior (Zenrows. 2023), respecting access rules. The crawler iterates through multiple pages, gathering job titles and the job postings URLs (refer to Images 2 & 3) using BeautifulSoup to parse the HTML content to extract the details. These collected URLs are used for a second scrape to locate and extract job details using BeautifulSoup: company, location, department, position type, salary, and description (refer to Images 5 & 6), which are then organized into a data frame for structured presentation (refer to Table 2 for the raw code, Table 3 for a breakdown of important code features, and Table 4 for desired web scrapped features from Seek website).

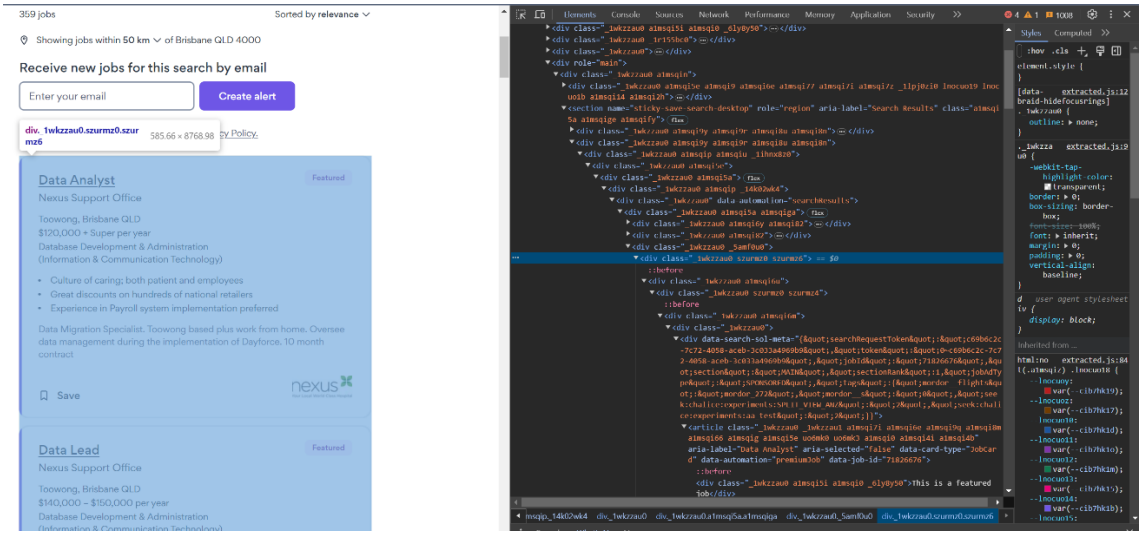


Image 2: Location of jobs from search results in Seek website HTML page to extract job title.

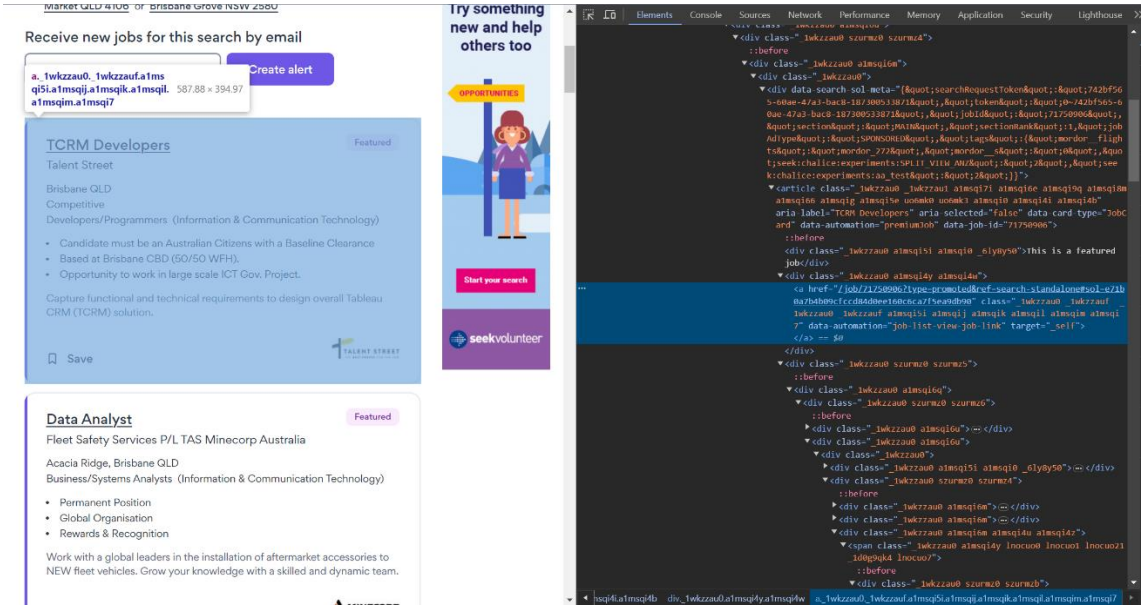


Image 3: Location of Job Posting URLs in Seek website HTML Page.

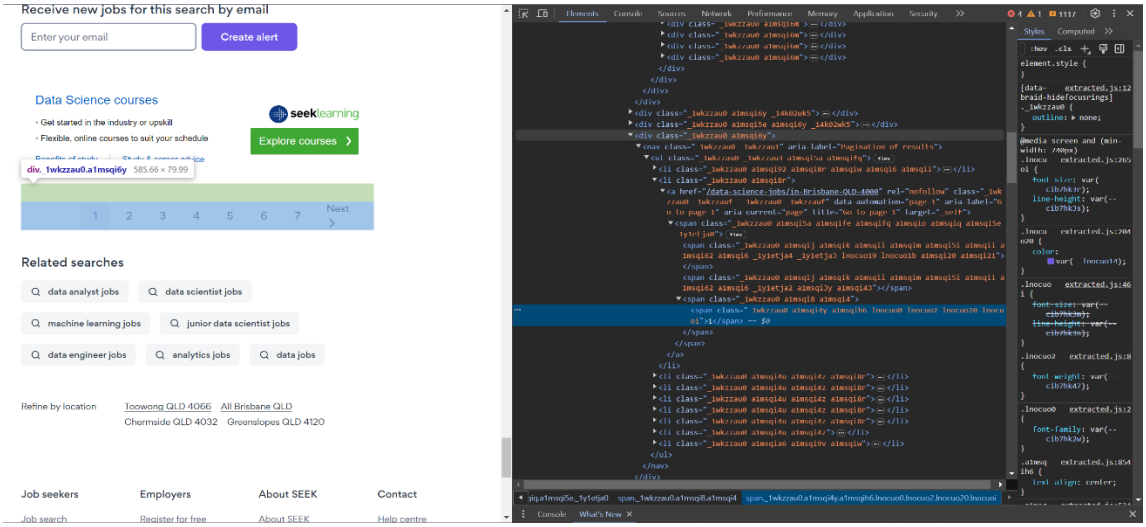


Image 4: Location of Page Buttons in Seek website HTML Page.

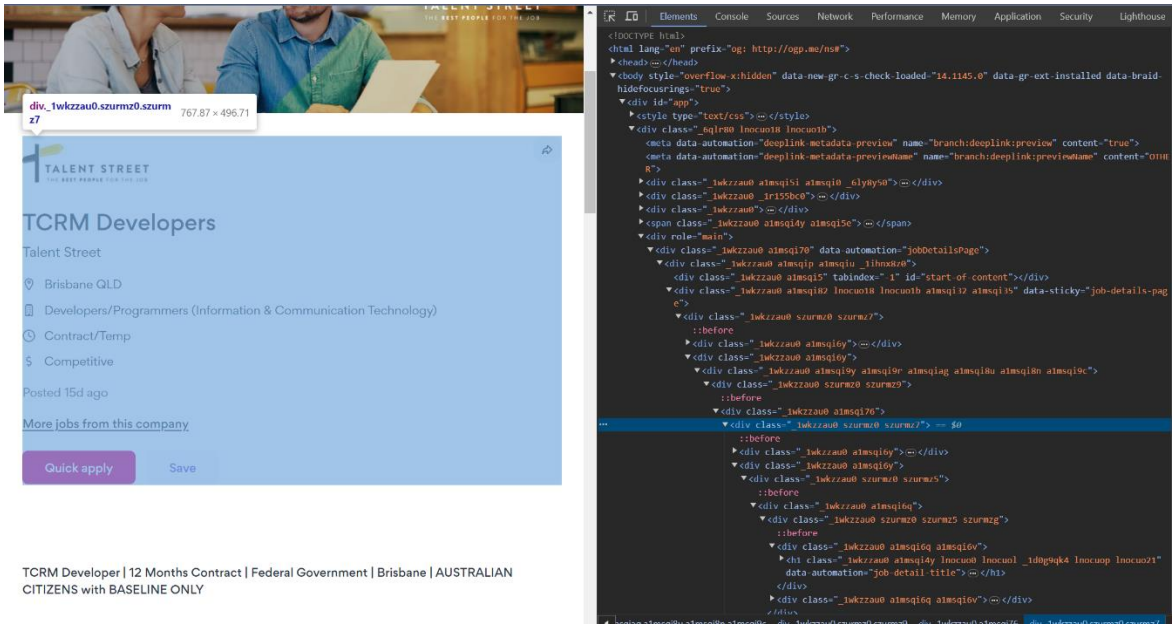


Image 5: Location of Company, Department, Position Type, and Salary features for a Job Posting on the Seek website HTML Page.

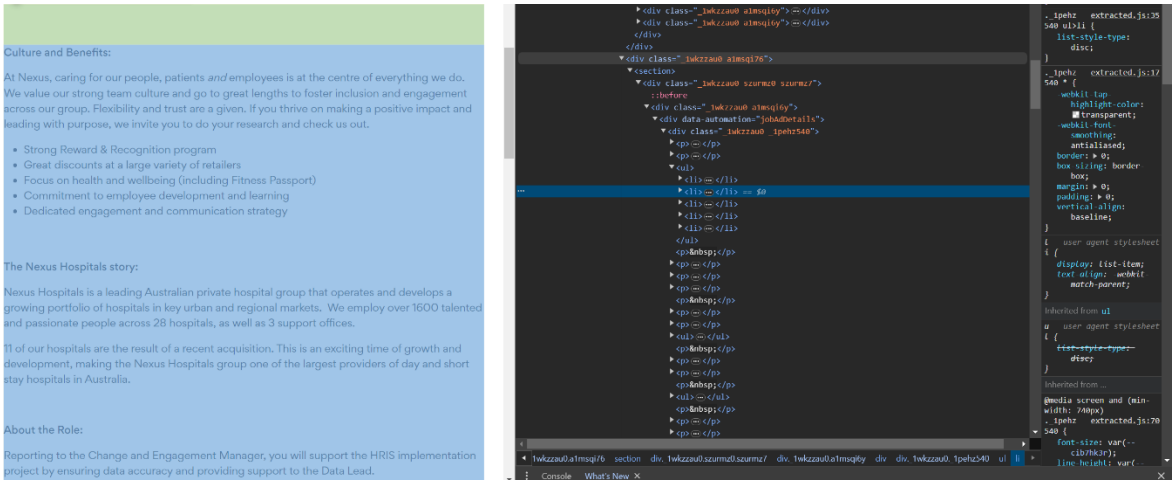


Image 6: Location of Job Description for a Job Posting in Seek website HTML Page.

Task	Code

First web scrape using seek URL with desired filters to extract Job Titles and Job posting URLs from search results.	Def scrape_job_titles_and_urls(url):
	  # Initialize robot parser
	  rp = robotparser.RobotFileParser()
	  rp.set_url("https://www.seek.com.au/robots.txt")
	  rp.read()
	
	  # Generate a random User-Agent to avoid detection as a bot
	  user_agent = UserAgent()
	  headers = {
	  'User-Agent': user_agent.random,
	  'Referrer': 'https://www.google.com',
	  'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
	  'Accept-Language': 'en-US,en;q=0.5',
	  'Accept-Encoding': 'gzip, deflate, br',
	  'Connection': 'keep-alive',
	  'Upgrade-Insecure-Requests': '1',
	  }
	
	  # Create a list to hold extracted data
	  all_job_data = []
	  page = 1
	
	  # Send request to Seek
	  while True:
	  params = {'page': page}
	
	  # Check if the URL is allowed by robots.txt
	  if rp.can_fetch(user_agent.random, url):
	  response = requests.get(url, headers=headers, params=params)
	  print(response.status_code)
	
	  # Inform if there is an access error
	  if response.status_code == 403:
	  logging.warning("Access denied. Consider adjusting your scraping strategy.")
	  return None
	
	  soup = BeautifulSoup(response.text, 'html.parser')
	
	  job_data = soup.find_all('a', class_='_1wkzzau0 _1wkzzauf _1rct8jy2 _1rct8jy4 _1rct8jy7 lnocuo2 _1rct8jy8 _1rct8jyb _1wkzzau0 _1wkzzauf a1msqih', href=True)
	
	  job_titles = [job.text.strip() for job in job_data]

```
        job_urls = [job['href'] for job in job_data]

        print("Job Titles:", job_titles)
        print("Job URLs:", job_urls)

        all_job_data.extend(zip(job_titles, job_urls))

        # Check for the existence of the next page button
        next_button = soup.find('a', {'data-automation': f'page-{page + 1}'})
        if not next_button:
            break  # Exit the loop if there's no next page button

        page += 1

        # Introduce a delay to mimic human behavior
        time.sleep(10)
    else:
        print(f"URL: {url} is disallowed by robots.txt")
        break

  return all_job_data

# URL for the initial search
initial_url = 'https://www.seek.com.au/data-science-jobs/in-Brisbane-QLD-4000'
job_data = scrape_job_titles_and_urls(initial_url)

for title, URL in job_data:
  print(f"Title: {title}, URL: {url}")
```

Creating a list of URLs to iterate through for the second web scrape.

```
# Identify and drop rows with duplicate URLs
job_url_data = job_url_data[~job_url_data['URL'].duplicated(keep='first')]

# Display the DataFrame with duplicates removed
display(job_url_data)

#Create list of urls to used for second webscrape of individual jobs
url_list = []
for urls in job_url_data["URL"]:
  url_list.append(urls)
```

Second web scrape using job URLs to extract job postings company name, location, department, position type, salary, and description.

```
# Initialize robot parser
rp = robotparser.RobotFileParser()
rp.set_url("https://www.seek.com.au/robots.txt")
rp.read()

# List to store dictionaries for each iteration
data_list = []

# User agent setup
user_agent = UserAgent()

# Iterate through each job URL
for job_url in job_url_data["URL"]:
  # Create a full URL
  full_url = f'https://www.seek.com.au{job_url}'

  # Check if the URL is allowed by robots.txt
  if rp.can_fetch(user_agent.random, full_url):
    # Headers for the request
    headers = {
      'User-Agent': user_agent.random,
```

```
        'Referrer': 'https://www.google.com',
        'Accept':
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
        'Accept-Language': 'en-US,en;q=0.5',
        'Accept-Encoding': 'gzip, deflate, br',
        'Connection': 'keep-alive',
        'Upgrade-Insecure-Requests': '1',
    }

    # Send a request to Seek
    response = requests.get(full_url, headers=headers)
    print(f"{full_url}, has been approved for search: {response.status_code}")

    # Pause for a few seconds to avoid overloading the server
    time.sleep(4)

    soup = BeautifulSoup(response.content, 'html.parser')
    try:
        company = soup.find('span', {'data-automation': 'advertiser-
name'}).text.strip()
        print(company)
    except AttributeError:
        company = 'None'
        print("Company not found")

    # Extract information from the specific div
    job_info_div = soup.find('div', class_='_1wkzzau0 a1msqi76')

    # Check if job_info_div is not None
    if job_info_div:
        # Find all spans with the specified class within job_info_div
        spans = job_info_div.find_all('span', class_='_1wkzzau0 a1msqi4y
a1msqir')

        # Check if spans is not empty before accessing its elements
        if spans:
            # Extract location, department, position type, salary and description
from job posting.
            location = spans[0].text.strip()
            print(location)
            department = spans[1].text.strip() if len(spans) > 1 else 'None'
            print(department)
            position_type = spans[2].text.strip() if len(spans) > 2 else 'None'
            print(position_type)

            # Try-except block for handling salary
            try:
                salary = spans[3].text.strip()
                print(salary)
            except IndexError:
                salary = 'None'
                print("Salary not found")

            # Extract description
            description_element = soup.find('div', {'data-automation':
'jobAdDetails'})
            description = description_element.text.strip() if description_element
else 'None'
            print(description)
        else:
            # Handle the case where spans is empty
            print(f"Error: Unable to find spans with the specified class for URL:
```

```
{full_url}")
            location, department, position_type, salary, description = 'None',
'None', 'None', 'None', 'None'
        else:
            # Handle the case where job_info_div is None
            print(f"Error: Unable to find job_info_div for URL: {full_url}")
            location, department, position_type, salary, description = 'None', 'None',
'None', 'None', 'None'

        # Create a dictionary for the current iteration
        job_data_dict = {
            'URL': full_url,
            'Company': company,
            'Location': location,
            'Department': department,
            'Position Type': position_type,
            'Salary': salary,
            'Description': description,
        }

        # Append the dictionary to the list
        data_list.append(job_data_dict)
    else:
        print(f"URL: {full_url} is disallowed by robots.txt")

# Create a DataFrame from the list of dictionaries
job_data_df = pd.DataFrame(data_list)

# Display the DataFrame
print(job_data_df)
```

Table 2: Raw code for both web scrappers to obtain desired features.

Features	Code	Description
Web Scraping Permission Check	rp = robotparser.RobotFileParser() rp.set_url("https://www.seek.com.au/robots.txt") rp.read() ……. # Check if the URL is allowed by robots.txt if rp.can_fetch(user_agent.random, full_url):	A mechanism to check whether a specified URL is allowed for web scraping based on the rules defined in the robots.txt file.

User Agent Configuration	# User agent setup
	user_agent = UserAgent()
		Create user agent strings, to avoid bot detection.

HTTP Headers Configuration	headers = {
	'User-Agent': user_agent.random,
	'Referrer': 'https://www.google.com',
	'Accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8',
	'Accept-Language': 'en-US,en;q=0.5',
	'Accept-Encoding': 'gzip, deflate, br',
	'Connection': 'keep-alive',
	'Upgrade-Insecure-Requests': '1',
	}
		Creating a HTTP header for web requests, including a randomly generated User-Agent, a referrer, and other characteristics, making certain a well-configured and diversified request setup to enhance web scraping efficiency and avoid detection.

URL	# URL for the initial search
	initial_url = 'https://www.seek.com.au/data-science-jobs/in-Brisbane-QLD-4000'
		URL used for initial search.

Web Request	response = requests.get(full_url, headers=headers) print(f"{full_url}, has been approved for search: {response.status_code}")	Send request to url.

Server Load Management	# Pause for a few seconds to avoid overloading the server
	time.sleep(4)
		introducing a delay to prevent potential

		overloading and demonstrating a responsible web scraping approach.
HTML Parsing with BeautifulSoup	soup = BeautifulSoup(response.content, 'html.parser')	Parse HTML content to allow structured and navigable access to the web page elements for further data extraction.

Feature Extraction	spans = job_info_div.find_all('span', class_='_1wkzzau0 a1msqi4y a1msqir')
	
	# Check if spans is not empty before accessing its elements
	if spans:
	# Extract location, department, position type, salary and description from job posting.
	location = spans[0].text.strip()
	print(location)
	department = spans[1].text.strip() if len(spans) > 1 else 'None'
	print(department)
	position_type = spans[2].text.strip() if len(spans) > 2 else 'None'
	print(position_type)
		Extract desired features from job postings and ensure robustness by checking if the spans list is not empty before accessing its elements.

Try, Except handling	# Try-except block for handling salary try: salary = spans[3].text.strip() print(salary) except IndexError: salary = 'None' print("Salary not found")	Employs a try-except block to handle potential IndexError when attempting to extract certain features.

Table 3: Breakdown of important code features from web scrappers

Features	Description	HTML location	Code
Job Title	Title of job in posting.	Physics and Science Teacher	job_data = soup.find_all('a', class_='_1wkzzau0 _1wkzzauf _1rct8jy2 _1rct8jy4 _1rct8jy7 lnocuo2 _1rct8jy8 _1rct8jyb _1wkzzau0 _1wkzzauf a1msqih', href=True) job_titles = [job.text.strip() for job in job_data]
URL	Url associated with job posting.	Physics and Science Teacher	job_data = soup.find_all('a', class_='_1wkzzau0 _1wkzzauf _1rct8jy2 _1rct8jy4 _1rct8jy7 lnocuo2 _1rct8jy8 _1rct8jyb _1wkzzau0 _1wkzzauf a1msqih', href=True) job_urls = [job['href'] for job in job_data]
Company	Company posting the job.	Mount Alvernia College	try: company = soup.find('span', {'data-automation': 'advertiser-name'}).text.strip() print(company) except AttributeError: company = 'None' print("Company not found")
Location	Location of job position.	Kedron, Brisbane QLD	# Extract information from the specific div

			job_info_div = soup.find('div', class_='_1wkzzau0 a1msqi76')
			
			# Check if job_info_div is not None
			if job_info_div:
			# Find all spans with the specified class within job_info_div
			spans = job_info_div.find_all('span', class_='_1wkzzau0 a1msqi4y a1msqir')
			
			# Check if spans is not empty before accessing its elements
			if spans:
			# Extract location, department, position type, salary and description from job posting.
			location = spans[0].text.strip()
			print(location)

Department	Department in company.	Assistant Accountants (Accounting)	# Check if job_info_div is not None
			if job_info_div:
			# Find all spans with the specified class within job_info_div
			spans = job_info_div.find_all('span', class_='_1wkzzau0 a1msqi4y a1msqir')
			
			# Check if spans is not empty before accessing its elements
			if spans:
			……
			# Extract location, department, position type, salary and description from job posting.
			department = spans[1].text.strip() if len(spans) > 1 else 'None'
			print(department)

Position Type	Type of position offered.	Full time	# Check if job_info_div is not None
			if job_info_div:
			# Find all spans with the specified class within job_info_div
			spans = job_info_div.find_all('span', class_='_1wkzzau0 a1msqi4y a1msqir')
			
			# Check if spans is not empty before accessing its elements
			if spans:
			……
			# Extract location, department, position type, salary and description from job posting.
			position_type = spans[2].text.strip() if len(spans) > 2 else 'None'
			print(position_type)

Salary	Salary offered for job position.	$70,000 – $80,000 per year	*# Check if job_info_div is not None*
			if job_info_div:
			  # Find all spans with the specified class within job_info_div
			  spans = job_info_div.find_all('span', class_='_1wkzzau0 a1msqi4y a1msqir')
			
			  # Check if spans is not empty before accessing its elements
			  if spans:
			  .....
			  # Try-except block for handling salary
			  try:
			  salary = spans[3].text.strip()
			  print(salary)
			  except IndexError:
			  salary = 'None'
			  print("Salary not found")

Description	Description of job.	<section> About the Company and the Role CIS Control Union Australia is a leading Testing, Inspection and Certification company (TIC) operating under the globally recognised Royal Peterson Control Union group. We're looking for an energetic person with commercial accounting or bookkeeping experience to support the Accounts Manager with end-to-end accounting in our Sinnamon Park offices. Key responsibilities - End to end bookkeeping processes - Operate in conjunction with our external accountants - Email and phone communication related with accounts - Client invoicing and AR administration - Accounts payable Are you who we are looking for? - Ability to seek out and solve problems - Customer service values - Accurate in data entry - Attention to detail - An inquiring mindset - Positive and supportive team player, development oriented and proactive in achieving goals Qualifications and Experience - Qualifications in bookkeeping or accounting - Relevant industry experience will be highly regarded - Experience in using accounts programs, CRM or data management systems and the capacity to learn new programs and procedures - Experience with data entry - Minimum intermediate level Microsoft Office and 365 Have a look at what we do; globally: petersoncontrolunion.com and locally : cis-	*# Check if spans is not empty before accessing its elements* if spans: ..... *# Extract description* description_element = soup.find('div', {'data-automation': 'jobAdDetails'}) description = description_element.text.strip() if description_element else 'None' print(description)

		controlunion.com If you're you're suited to this role and want to be part of our fantastic company, apply to be part of our team. Please submit a cover letter outlining your interest in the position along with your resume. <iframe src="https://www.youtube.com/embed/dIPMoOUy_h8" class="sp273c1" allowfullscreen=""></iframe> </section>	

Table 4: Desired web scrapped features from Seek website.

Limitations of the WebCrawler and the harvested data.

Conducting web crawling and harvesting data can have several limitations. Initially, there is a learning curve to comprehend the tactics used to ensure you're not blocked, and understand in our case, the HTML structure of the website to pull the desired data which is subject to change as the website may change its layout (Barrett, A. 2023). This required extensive time to investigate the HTML format of the websites to know where exactly the desired data lies to then pull. Additionally, websites may embed complex features, requiring users to need more functional scraping tools (Barrett, A. 2023). In my case, the biggest challenge was implementing code to overcome the 'next' button to iterate through all pages (refer to Table 2) of results from our filtered query. Furthermore, scraping a website extensively brings heavy traffic, which may overload a web server (Barrett, A. 2023), possibly being detected and resulting in an IP ban by the website. Referring to the methodology above, tools were used to simulate normal human browsing behavior (refer to Table 3). Finally, we consider the legalities of Seeks Terms of Services which were previously discussed and analyzed (refer to Table 1), acknowledging the prohibition of any web scraping, but under Australian regulations, web scraping for academic purposes is permitted.

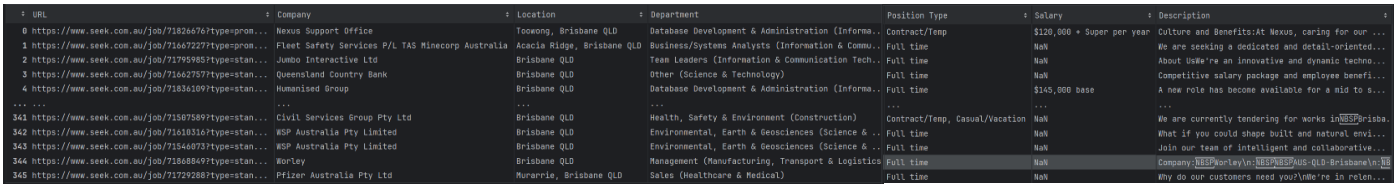
Methodology of storing harvested data

The data harvested from SEEK's job listings is stored through a multi-step process. Initially, a web crawler is employed to extract job titles and URLs, and the collected data is organized into a list. Next, this list is converted into a data frame using pandas for structured presentation. Duplicate URLs are identified and removed to ensure data accuracy. Following this, the list of URLs obtained was prepared for a second web scrape to extract job details mentioned in detail in Table 4, for each job posting. The obtained data from the second web scrape is added to a dictionary and then converted into a data frame. The resulting data frame is then further refined by resetting the index, handling 'None' inputs, dropping missing descriptions, and ensuring the removal of any URL duplicates (refer to Images 7 & 8 for final datasets).

11 rows 346 rows × 2 columns pd.DataFrame

	Job Title	URL
0	Data Analyst	/job/71826676?type=promoted&ref=search-standalone
1	Data Analyst	/job/71667227?type=promoted&ref=search-standalone
2	Data Lead	/job/71795985?type=standout&ref=search-standalone
3	Senior Data Scientist	/job/71662757?type=standout&ref=search-standalone
4	Data Engineer	/job/71836109?type=standard&ref=search-standalone
...	...	...
342	Environmental Officers - BNE - Gen	/job/71507589?type=standard&ref=search-standalone
343	Hazmat & Occupational Hygiene ConsultantHybrid	/job/71610316?type=standard&ref=search-standalone
344	Senior Geochemist - East CoastFlexible	/job/71546073?type=standard&ref=search-standalone

Image 7: Data frame of results from first web scrape.



	URL	Company	Location	Department	Position Type	Salary	Description
0	https://www.seek.com.au/job/71826676?type=prom...	Nexus Support Office	Toowong, Brisbane QLD	Database Development & Administration (Informa..	Contract/Temp	$120,000 + Super per year	Culture and Benefits:At Nexus, caring for our ...
1	https://www.seek.com.au/job/71667227?type=prom...	Fleet Safety Services P/L TAS Minecorp Australia	Acacia Ridge, Brisbane QLD	Business/Systems Analysts (Information & Commu..	Full time	NaN	We are seeking a dedicated and detail-oriented...
2	https://www.seek.com.au/job/71795985?type=stan...	Jumbo Interactive Ltd	Brisbane QLD	Team Leaders (Information & Communication Tech..	Full time	NaN	About UsWe're an innovative and dynamic techno...
3	https://www.seek.com.au/job/71662757?type=stan...	Queensland Country Bank	Brisbane QLD	Other (Science & Technology)	Full time	NaN	Competitive salary package and employee benefi...
4	https://www.seek.com.au/job/71836109?type=stan...	Humanised Group	Brisbane QLD	Database Development & Administration (Informa..	Full time	$145,000 base	A new role has become available for a mid to s...
...	...	...	...	...	...	...	...
341	https://www.seek.com.au/job/71507589?type=stan...	Civil Services Group Pty Ltd	Brisbane QLD	Health, Safety & Environment (Construction)	Contract/Temp, Casual/Vacation	NaN	We are currently tendering for works inNBSPBrisba.
342	https://www.seek.com.au/job/71618316?type=stan...	WSP Australia Pty Limited	Brisbane QLD	Environmental, Earth & Geosciences (Science & ..	Full time	NaN	What if you could shape built and natural envi...
343	https://www.seek.com.au/job/71546073?type=stan...	WSP Australia Pty Limited	Brisbane QLD	Environmental, Earth & Geosciences (Science & ..	Full time	NaN	Join our team of intelligent and collaborative...
344	https://www.seek.com.au/job/71868849?type=stan...	Worley	Brisbane QLD	Management (Manufacturing, Transport & Logistics	Full time	NaN	Company:NBSPWorley\n:NBSPNBSPAUS-QLD-Brisbane\n:NB
345	https://www.seek.com.au/job/71729288?type=stan...	Pfizer Australia Pty Ltd	Murarrie, Brisbane QLD	Sales (Healthcare & Medical)	Full time	NaN	Why do our customers need you?\nWe're in relen...

Image 8: Data frame of results from second web scrape.

Data Wrangling

Cleaning, normalization, and feature extraction of the sourced data. Normalization may include applying a word embedding algorithm

In the data wrangling phase, we first merged the two data frames from our web scraping (Images 7 & 8) using a left join on image 7 to image 8 based on using the index column and incorporating 'Job Title' as well (refer to table 5). Following this we conducted comprehensive feature extraction to build a corpus that concatenates job title, company, location, department, position type, salary, and description (refer to Table 5 for a detailed summary) into a single string. Due to a significant portion of missing salary data (210/346), 'Salary' was omitted from the corpus. To enhance the quality of the text data, we performed text preprocessing using the NLTK library. This involved removing special characters, tokenizing, and lemmatizing words to reduce them to their base form.

With that said, to conduct two separate NLP tasks I created two variations of the corpus. The first corpus is designed for the unsupervised machine learning (ML) method for extractive text summarization (Shrivarsheni. 2020). For the objective of text summarization, during the text pre-processing, I decided to keep numbers, stop words, parenthesis, periods, and commas in the corpus (refer to Table 5 for raw code). The reason for this is to ensure proper sentence formatting and coherence when conducting the summarizations. In our second unsupervised ML task, we use Word2Vec (Data Basecamp, 2023) to help extend/improve our list of technical Data Science skills to identify popular skills trends in job postings. For this aspect, we create a corpus using the same process but incorporate the removal of stop words, parenthesis, period, and commas (refer to Table 5 for raw code and images 9 & 10 for raw and processed corpus outputs). The results are refined corpora (refer to Table 5 for corpus creation and cleaning) ready for natural language processing tasks. Both ML methods will be further analyzed and discussed later in the report.

We carefully considered hyperparameters during text preprocessing, utilizing WordNet lemmatization to capture the semantic meaning of words. The corpus was then tokenized to individual words, and lemmatization ensured standardized representations for analysis. Moreover, we applied a technique to identify words common across all documents, providing insights into shared vocabulary, and words unique to individual documents, highlighting distinct terminology. This process sets the foundation for building an effective model by transforming raw textual data into a normalized, feature-rich corpus contributing to successive natural language processing tasks.

Method	Code
Merging web scraped datasets	*# Merge data on the left join* job_df = pd.merge(job_data_df, job_url_data[['index', 'Job Title']], on='index', how='left') *# Print the result* display(job_df)
Creating raw corpus	*#Append features to 14reate corpus* job_tech_skills_df['Corpus_raw'] = (job_tech_skills_df['Job Title'].fillna('') + ' ' + job_tech_skills_df['Company'].fillna('') + ' ' +

```
    job_tech_skills_df['Location'].fillna('') +
    ' ' +
    job_tech_skills_df['Department'].fillna('') +
    ' ' +
    job_tech_skills_df['Position Type'].fillna('') +
    ' ' +
    job_tech_skills_df['Description'].fillna('')
)

display(job_tech_skills_df)
```

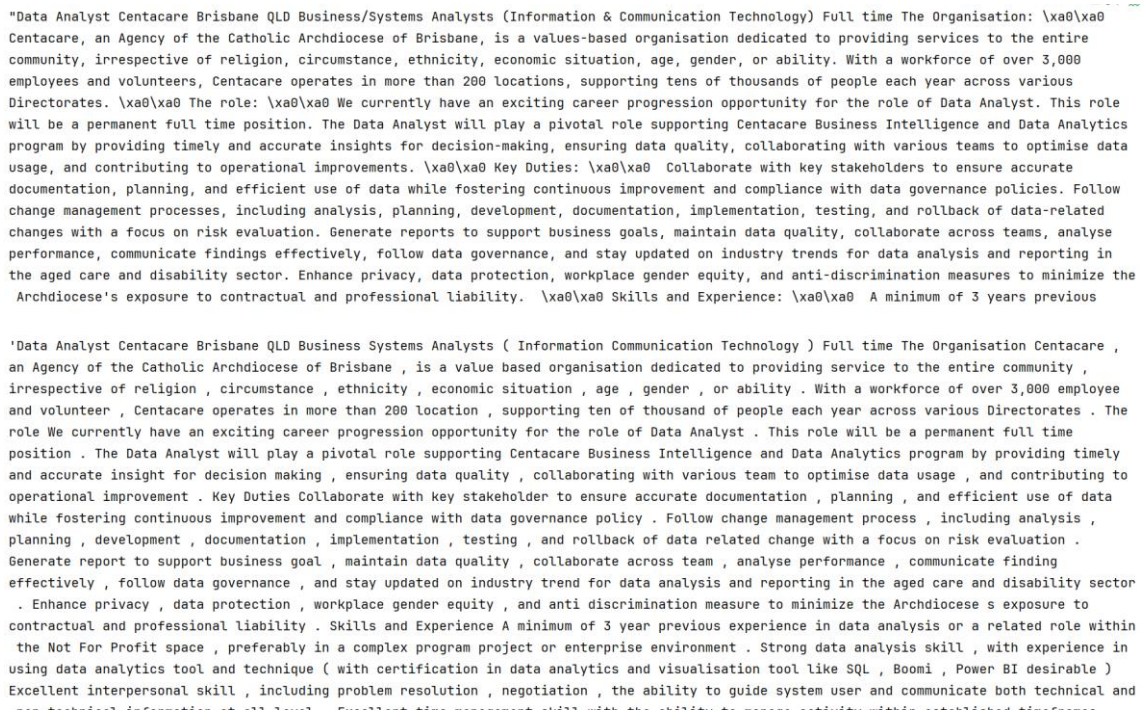
Corpus text pre-processing for LSA text summarization

```
def text_preprocessing(uncleaned_text_corpus):
    lemmatizer = WordNetLemmatizer()
    corpus = []

    for I in range(0, len(uncleaned_text_corpus)):
        # Remove special characters, including '\xa0'
        text = re.sub('[^a-zA-Z0-9.,()]', ' ', uncleaned_text_corpus[i])

        # Tokenize the text
        words = word_tokenize(text)

        # Lemmatize each word
        words = [lemmatizer.lemmatize(word) for word in words]

        # Join the lemmatized words back into a sentence
        text = ' '.join(words)
        corpus.append(text)

    return corpus

job_df['Corpus'] = text_preprocessing(job_df['Corpus_raw'])
display(job_df) #353 rows x 11 columns
```

Corpus text pre-processing for Word2Vec

```
#Clean corpus
stop_words = set(stopwords.words('english'))

def text_preprocessing(uncleaned_text_corpus):
    lemmatizer = WordNetLemmatizer()
    corpus = []

    for I in range(0, len(uncleaned_text_corpus)):
        text = re.sub("\':", "", uncleaned_text_corpus[i])
        text = re.sub('[^a-zA-Z]', ' ', text)
        text = text.lower()

        # Tokenize the text
        words = word_tokenize(text)
        all_stopwords = set(stopwords.words('english'))

        # Lemmatize each word and remove stopwords
        words = [lemmatizer.lemmatize(word) for word in words if word not in all_stopwords]

        # Join the lemmatized words back into a sentence
        text = ' '.join(words)
        corpus.append(text)

    return corpus

#Apply text pre-processing
job_tech_skills_df['Corpus'] = text_preprocessing(job_tech_skills_df['Corpus_raw'])
```

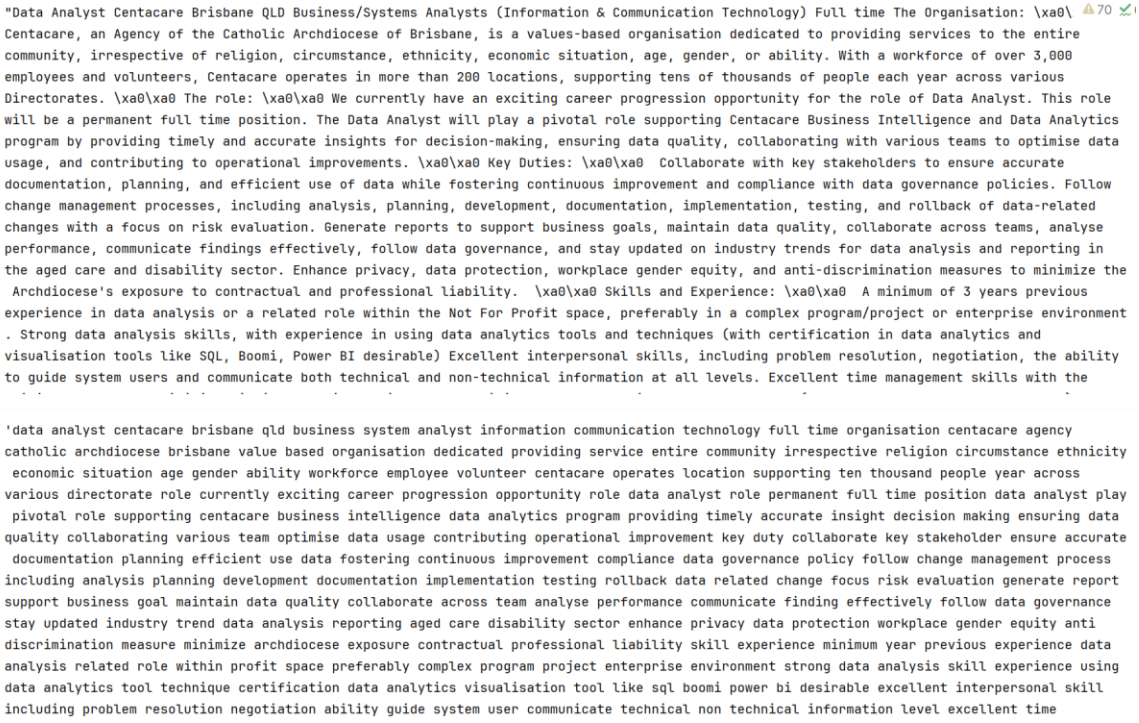
Table 5: Corpus creation and cleaning.



"Data Analyst Centacare Brisbane QLD Business/Systems Analysts (Information & Communication Technology) Full time The Organisation: \xa0\xa0 Centacare, an Agency of the Catholic Archdiocese of Brisbane, is a values-based organisation dedicated to providing services to the entire community, irrespective of religion, circumstance, ethnicity, economic situation, age, gender, or ability. With a workforce of over 3,000 employees and volunteers, Centacare operates in more than 200 locations, supporting tens of thousands of people each year across various Directorates. \xa0\xa0 The role: \xa0\xa0 We currently have an exciting career progression opportunity for the role of Data Analyst. This role will be a permanent full time position. The Data Analyst will play a pivotal role supporting Centacare Business Intelligence and Data Analytics program by providing timely and accurate insights for decision-making, ensuring data quality, collaborating with various teams to optimise data usage, and contributing to operational improvements. \xa0\xa0 Key Duties: \xa0\xa0 Collaborate with key stakeholders to ensure accurate documentation, planning, and efficient use of data while fostering continuous improvement and compliance with data governance policies. Follow change management processes, including analysis, planning, development, documentation, implementation, testing, and rollback of data-related changes with a focus on risk evaluation. Generate reports to support business goals, maintain data quality, collaborate across teams, analyse performance, communicate findings effectively, follow data governance, and stay updated on industry trends for data analysis and reporting in the aged care and disability sector. Enhance privacy, data protection, workplace gender equity, and anti-discrimination measures to minimize the Archdiocese's exposure to contractual and professional liability. \xa0\xa0 Skills and Experience: \xa0\xa0 A minimum of 3 years previous

'Data Analyst Centacare Brisbane QLD Business Systems Analysts (Information Communication Technology) Full time The Organisation Centacare , an Agency of the Catholic Archdiocese of Brisbane , is a value based organisation dedicated to providing service to the entire community , irrespective of religion , circumstance , ethnicity , economic situation , age , gender , or ability . With a workforce of over 3,000 employee and volunteer , Centacare operates in more than 200 location , supporting ten of thousand of people each year across various Directorates . The role We currently have an exciting career progression opportunity for the role of Data Analyst . This role will be a permanent full time position . The Data Analyst will play a pivotal role supporting Centacare Business Intelligence and Data Analytics program by providing timely and accurate insight for decision making , ensuring data quality , collaborating with various team to optimise data usage , and contributing to operational improvement . Key Duties Collaborate with key stakeholder to ensure accurate documentation , planning , and efficient use of data while fostering continuous improvement and compliance with data governance policy . Follow change management process , including analysis , planning , development , documentation , implementation , testing , and rollback of data related change with a focus on risk evaluation . Generate report to support business goal , maintain data quality , collaborate across team , analyse performance , communicate finding effectively , follow data governance , and stay updated on industry trend for data analysis and reporting in the aged care and disability sector . Enhance privacy , data protection , workplace gender equity , and anti discrimination measure to minimize the Archdiocese s exposure to contractual and professional liability . Skills and Experience A minimum of 3 year previous experience in data analysis or a related role within the Not For Profit space , preferably in a complex program project or enterprise environment . Strong data analysis skill , with experience in using data analytics tool and technique (with certification in data analytics and visualisation tool like SQL , Boomi , Power BI desirable) Excellent interpersonal skill , including problem resolution , negotiation , the ability to guide system user and communicate both technical and

Image 9: Raw corpus and cleaned corpus used for NLP text summarisation task.



"Data Analyst Centacare Brisbane QLD Business/Systems Analysts (Information & Communication Technology) Full time The Organisation: \xa0\ Centacare, an Agency of the Catholic Archdiocese of Brisbane, is a values-based organisation dedicated to providing services to the entire community, irrespective of religion, circumstance, ethnicity, economic situation, age, gender, or ability. With a workforce of over 3,000 employees and volunteers, Centacare operates in more than 200 locations, supporting tens of thousands of people each year across various Directorates. \xa0\xa0 The role: \xa0\xa0 We currently have an exciting career progression opportunity for the role of Data Analyst. This role will be a permanent full time position. The Data Analyst will play a pivotal role supporting Centacare Business Intelligence and Data Analytics program by providing timely and accurate insights for decision-making, ensuring data quality, collaborating with various teams to optimise data usage, and contributing to operational improvements. \xa0\xa0 Key Duties: \xa0\xa0 Collaborate with key stakeholders to ensure accurate documentation, planning, and efficient use of data while fostering continuous improvement and compliance with data governance policies. Follow change management processes, including analysis, planning, development, documentation, implementation, testing, and rollback of data-related changes with a focus on risk evaluation. Generate reports to support business goals, maintain data quality, collaborate across teams, analyse performance, communicate findings effectively, follow data governance, and stay updated on industry trends for data analysis and reporting in the aged care and disability sector. Enhance privacy, data protection, workplace gender equity, and anti-discrimination measures to minimize the Archdiocese's exposure to contractual and professional liability. \xa0\xa0 Skills and Experience: \xa0\xa0 A minimum of 3 years previous experience in data analysis or a related role within the Not For Profit space, preferably in a complex program/project or enterprise environment . Strong data analysis skills, with experience in using data analytics tools and techniques (with certification in data analytics and visualisation tools like SQL, Boomi, Power BI desirable) Excellent interpersonal skills, including problem resolution, negotiation, the ability to guide system users and communicate both technical and non-technical information at all levels. Excellent time management skills with the

'data analyst centacare brisbane qld business system analyst information communication technology full time organisation centacare agency catholic archdiocese brisbane value based organisation dedicated providing service entire community irrespective religion circumstance ethnicity economic situation age gender ability workforce employee volunteer centacare operates location supporting ten thousand people year across various directorate role currently exciting career progression opportunity role data analyst role permanent full time position data analyst play pivotal role supporting centacare business intelligence data analytics program providing timely accurate insight decision making ensuring data quality collaborating various team optimise data usage contributing operational improvement key duty collaborate key stakeholder ensure accurate documentation planning efficient use data fostering continuous improvement compliance data governance policy follow change management process including analysis planning development documentation implementation testing rollback data related change focus risk evaluation generate report support business goal maintain data quality collaborate across team analyse performance communicate finding effectively follow data governance stay updated industry trend data analysis reporting aged care disability sector enhance privacy data protection workplace gender equity anti discrimination measure minimize archdiocese exposure contractual professional liability skill experience minimum year previous experience data analysis related role within profit space preferably complex program project enterprise environment strong data analysis skill experience using data analytics tool technique certification data analytics visualisation tool like sql boomi power bi desirable excellent interpersonal skill including problem resolution negotiation ability guide system user communicate technical non technical information level excellent time

Image 10: Raw corpus and cleaned corpus used for NLP word2vec task.

Summary and visualization of the harvested data. Preliminary EDA is acceptable in this section as well.

After conducting our web scraping and refining our corpora we have our final Data Frame to conduct our NLP tasks (refer to image 11 for data frame); 346 rows (job posts) and 11 columns consisting of the desired job features, job URLs, raw and cleaned corpora. Referring to Table 5, there are two variations of the corpus used, the only difference is the clean corpuses formatting, which was discussed earlier, due to the differing ML tasks conducted. Furthermore, we conducted an extensive exploratory data analysis (EDA) on the sample distributions (refer to images 12 to 32) and corpora (refer to images 33 to 36) with the summaries found in Table 6.

346 rows × 11 columns pd.DataFrame

	index	URL	Company	Location
2	2	https://www.seek.com.au/job/71795985?type=stan...	Jumbo Interactive Ltd	Brisbane QLD
3	3	https://www.seek.com.au/job/71662757?type=stan...	Queensland Country Bank	Brisbane QLD
4	4	https://www.seek.com.au/job/71836109?type=stan...	Humanised Group	Brisbane QLD
...	...	...	...	...
341	341	https://www.seek.com.au/job/71507589?type=stan...	Civil Services Group Pty Ltd	Brisbane QLD
342	342	https://www.seek.com.au/job/71610316?type=stan...	WSP Australia Pty Limited	Brisbane QLD
343	343	https://www.seek.com.au/job/71546073?type=stan...	WSP Australia Pty Limited	Brisbane QLD
344	344	https://www.seek.com.au/job/71868849?type=stan...	Worley	Brisbane QLD
345	345	https://www.seek.com.au/job/71729288?type=stan...	Pfizer Australia Pty Ltd	Murarrie, Brisbane QLD

346 rows × 11 columns pd.DataFrame

Department	Position Type	Salary	Description
Team Leaders (Information & Communication Tech...	Full time	NaN	About UsWe're an innovative and dynamic techno...
Other (Science & Technology)	Full time	NaN	Competitive salary package and employee benefi...
Database Development & Administration (Informa...	Full time	$145,000 base	A new role has become available for a mid to s...
...	...	...	...
Health, Safety & Environment (Construction)	Contract/Temp, Casual/Vacation	NaN	We are currently tendering for works inNBSPBrisba.
Environmental, Earth & Geosciences (Science & ...	Full time	NaN	What if you could shape built and natural envi...
Environmental, Earth & Geosciences (Science & ...	Full time	NaN	Join our team of intelligent and collaborative...
Management (Manufacturing, Transport & Logistics)	Full time	NaN	Company:NBSPWorley\n:NBSPNBSPAUS-QLD-Brisbane\n:NB
Sales (Healthcare & Medical)	Full time	NaN	Why do our customers need you?\nWe're in relen...

1 columns pd.DataFrame

Job Title	Corpus_raw	Corpus
Data Lead	Data Lead Jumbo Interactive Ltd Brisbane QLD T...	Data Lead Jumbo Interactive Ltd Brisbane QLD T...
Senior Data Scientist	Senior Data Scientist Queensland Country Bank ...	Senior Data Scientist Queensland Country Bank ...
Data Engineer	Data Engineer Humanised Group Brisbane QLD Dat...	Data Engineer Humanised Group Brisbane QLD Dat...
...	...	...
Senior Public Health Officer	Senior Public Health Officer Civil Services Gr...	Senior Public Health Officer Civil Services Gr...
Environmental Officers - BNE - Gen	Environmental Officers - BNE - Gen WSP Austral...	Environmental Officers BNE Gen WSP Australia P...
Hazmat & Occupational Hygiene ConsultantHybrid	Hazmat & Occupational Hygiene ConsultantHybrid...	Hazmat Occupational Hygiene ConsultantHybrid W...
Senior Geochemist - East CoastFlexible	Senior Geochemist - East CoastFlexible Worley ...	Senior Geochemist East CoastFlexible Worley Br...
Logistics Manager	Logistics Manager Pfizer Australia Pty Ltd Mur...	Logistics Manager Pfizer Australia Pty Ltd Mur...

Image 11: Final data frame to conduct NLP tasks.

1-10 224 rows × 1 columns np.ndarray

	0
0	Nexus Support Office
1	Fleet Safety Services P/L TAS Minecorp Australia
2	Jumbo Interactive Ltd
3	Queensland Country Bank
4	Humanised Group
5	Centacare
6	Department of Health - Queensland
7	Australian Retirement Trust
8	Wesley Mission Queensland
9	Hatch Pty Ltd

Image 12: Unique Companies identified.

11 rows × 2 columns pd.DataFrame

	Company	Frequency
0	The University of Queensland	15
1	Queensland University of Technology	9
2	Department of Health - Queensland	7
3	WSP Australia Pty Limited	5
4	Mater Group	5
5	Hatch Pty Ltd	4
6	Suncorp	4
7	Sanofi CHC	3
8	Rio Tinto	3
9	Robert Half	3
10	UnitingCare Qld	3

5 rows × 2 columns pd.DataFrame

	Company	Frequency
11	NES Fircroft	3
12	Energy Queensland Group	3
13	Virgin Australia	3
14	Seqwater	3

Image 13: Frequency count of Companies.

Image 14: WordCloud of unique Companies frequency count.

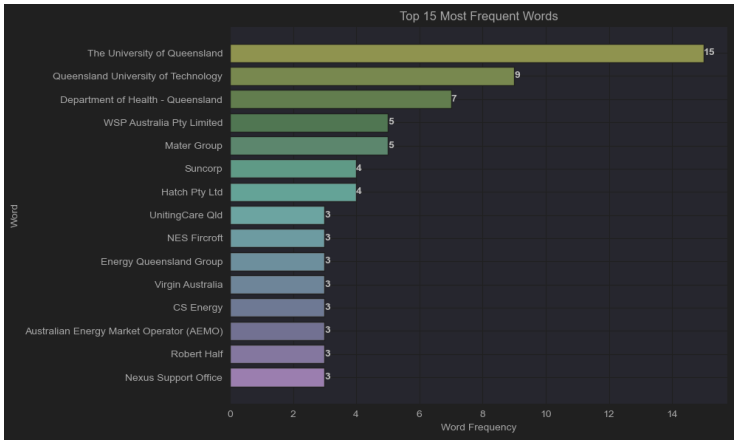


Image 15: Distribution of Company variable.

1-10 61 rows × 1 columns np.ndarray

	0
0	Toowong, Brisbane QLD
1	Acacia Ridge, Brisbane QLD
2	Brisbane QLD
3	Bowen Hills, Brisbane QLD
4	Chermside, Brisbane QLD
5	Greenslopes, Brisbane QLD
6	Coorparoo, Brisbane QLD
7	South Brisbane, Brisbane QLD
8	Fortitude Valley, Brisbane QLD

Image 16: Unique Locations identified.

11 rows × 2 columns pd.DataFrame

	Location	Frequency
0	Brisbane QLD	188
1	South Brisbane, Brisbane QLD	13
2	Murarrie, Brisbane QLD	10
3	St Lucia, Brisbane QLD	9
4	Newstead, Brisbane QLD	8
5	Fortitude Valley, Brisbane QLD	8
6	Toowong, Brisbane QLD	7
7	Eight Mile Plains, Brisbane QLD	7
8	Kedron, Brisbane QLD	4
9	Indooroopilly, Brisbane QLD	4
10	Ipswich, Brisbane QLD	4

5 rows × 2 columns pd.DataFrame

	Location	Frequency
11	Wacol, Brisbane QLD	4
12	Milton, Brisbane QLD	4
13	Kenmore, Brisbane QLD	4
14	Woolloongabba, Brisbane QLD	4
15	Coopers Plains, Brisbane QLD	3

Image 17: Frequency count of Locations.

Image 18: WordCloud of unique Locations frequency count.

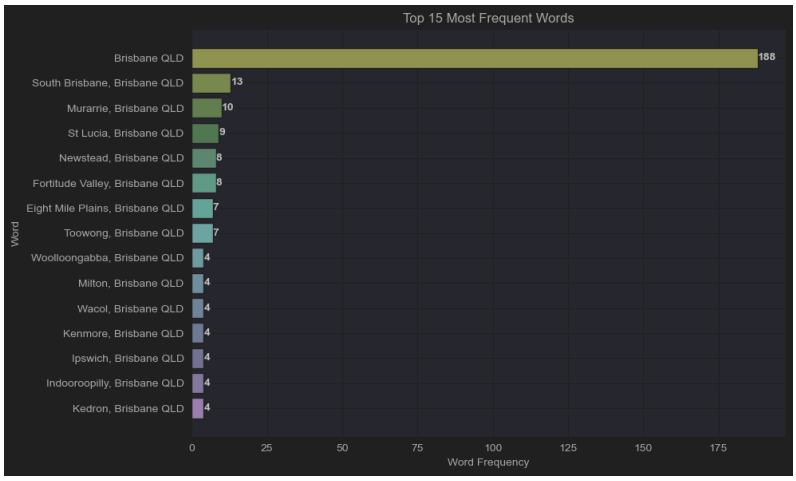


Image 19: Distribution of Location variable.

1-10 130 rows × 1 columns np.ndarray

	0
0	Database Development & Administration (Information & Communication Technology)
1	Business/Systems Analysts (Information & Communication Technology)
2	Team Leaders (Information & Communication Technology)
3	Other (Science & Technology)
4	Analysis & Reporting (Banking & Financial Services)Government - State (Government & Defence)
5	Business/Systems Analysts (Information & Communication Technology)Government - State (Government & Defence)
6	Engineering - Software (Information & Communication Technology)
7	Mathematics, Statistics & Information Sciences (Science & Technology)Government - Federal (Government & Defence)
8	Other, Government - Federal (Government & Defence)
9	Analysts (Consulting & Strategy)

Image 20: Unique Departments identified.

11 rows × 2 columns pd.DataFrame

	Department	Frequency
0	Business/Systems Analysts (Information & Commu...	24
1	Database Development & Administration (Informa...	18
2	Engineering - Software (Information & Communic...	15
3	Laboratory & Technical Services (Science & Tec...	15
4	Environmental, Earth & Geosciences (Science & ...	14
5	Other (Information & Communication Technology)	10
6	Research & Fellowships, Other (Education & Tra...	9
7	Teaching - Secondary (Education & Training)	8
8	Developers/Programmers (Information & Communic...	8
9	Government - Federal (Government & Defence)	7
10	Consultants (Information & Communication Techn...	6

5 rows × 2 columns pd.DataFrame

	Department	Frequency
11	Agronomy & Farm Services (Farming, Animals & C...	5
12	Business/Systems Analysts (Information & Commu...	5
13	Process Engineering (Engineering)	4
14	Analysts (Consulting & Strategy)	4
15	Environment & Sustainability Consulting (Consu...	4

Image 21: Frequency count of Departments.

Image 22: WordCloud of unique Departments frequency count.

Image 23: Distribution of Department variable.

8 rows × 2 columns pd.DataFrame

	Position Type	Frequency
0	Full time	277
1	Contract/Temp	51
2	Part time	5
3	Casual/Vacation	5
4	Contract/Temp, Casual/Vacation, Full time, Par...	3
5	Contract/Temp, Casual/Vacation, Full time	2
6	Contract/Temp, Full time	2
7	Contract/Temp, Casual/Vacation	1

Image 24: Frequency count of Position Types.

Image 25: WordCloud of unique Position Types frequency count.

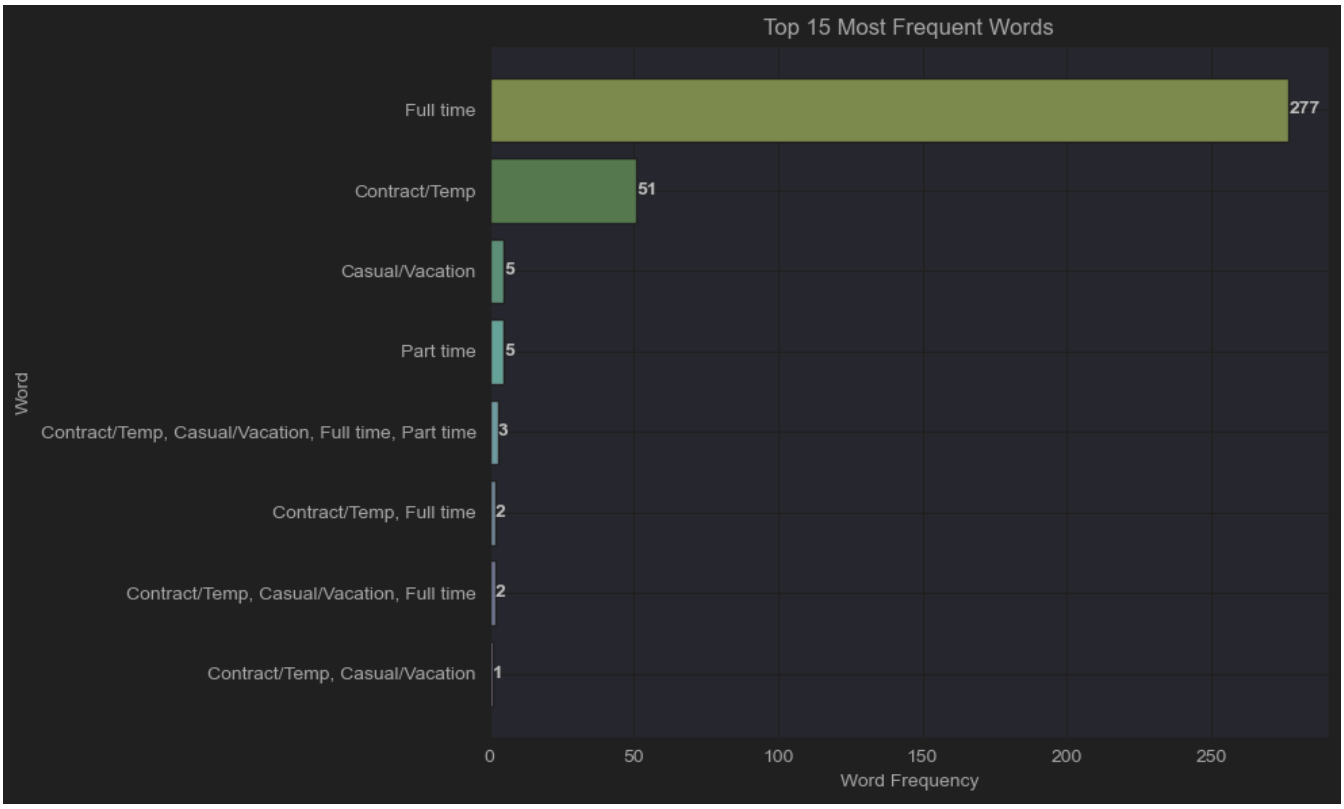


Image 26: Distribution of Position Types variable.

	Salary	Frequency
0	$70,000 - $80,000 per year	3
1	$120,000 + Super per year	2
2	$117,448-$128,664 + 12.75% super dependant on ...	2
3	$93,865.73-$98,107.66 + super + salary packaging	2
4	$87,311 - $93,724 + super + salary packaging	2
5	$60,000 - $70,000 per year	2
6	+ Higher Super, Great Bonus and Share Options	2
7	Attractive salary + fantastic benefits with SNP!	2
8	$37 - $46 per hour	2
9	$100,000 - $120,000 per year	2
10	$71.16 per hour + Super	2

Image 27: Frequency count of Salaries.

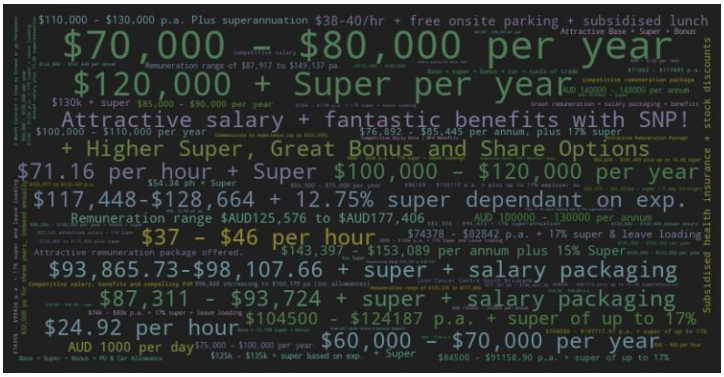


Image 28: WordCloud of unique Salaries frequency count.

1-10 298 rows × 1 columns np.ndarray

	0
0	Data Analyst
1	Data Lead
2	Senior Data Scientist
3	Data Engineer
4	Data and Analytics Lead
5	Principal HR Data Analyst
6	Data Engineer - Investment Resilience & Planning
7	Data and Insights Analyst
8	Analytics and Decision Solutions Regional Lead
9	Safety Data & Insights Lead

Image 29: Unique Job Titles identified.

11 rows 11 rows × 2 columns pd.DataFrame

	Job Title	Frequency
0	Data Analyst	10
1	Data Engineer	5
2	Laboratory Technician	5
3	Senior Data Engineer	3
4	Lead Developer - Brisbane, Australia	3
5	Accounts Assistant	2
6	Software Engineer	2
7	Farm Systems Agronomist - Soil Carbon	2
8	Safe and Connected Futures Senior Practitioner	2
9	Behaviour Support Practitioner	2
10	Physics and Science Teacher	2

5 rows 5 rows × 2 columns pd.DataFrame

	Job Title	Frequency
11	Manager Sustainability	2
12	Central Laboratory Assistant	2
13	Warehouse Supervisor	2
14	Research Fellow - ISSR	2
15	Sample Receipt Officer	2

Image 30: Frequency count of Job Titles in the data frame.

Image 31: WordCloud of unique Job Titles frequency count.

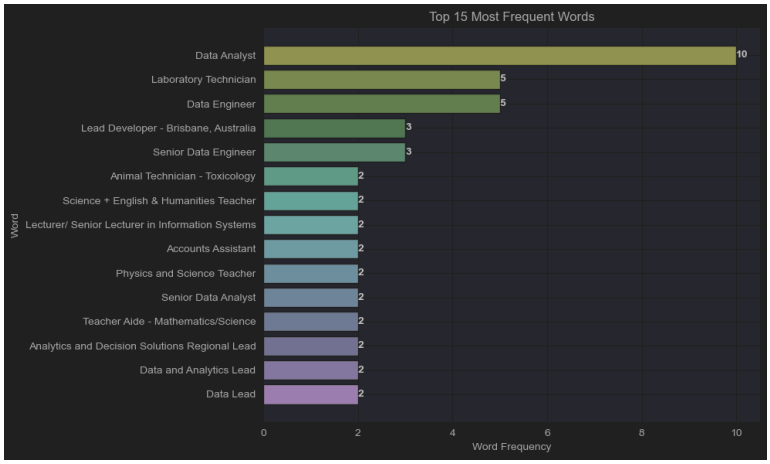


Image 32: Distribution of Job Titles variable.

Image 33: WordCloud of unique word frequency count in the corpus for text summarization NLP task.

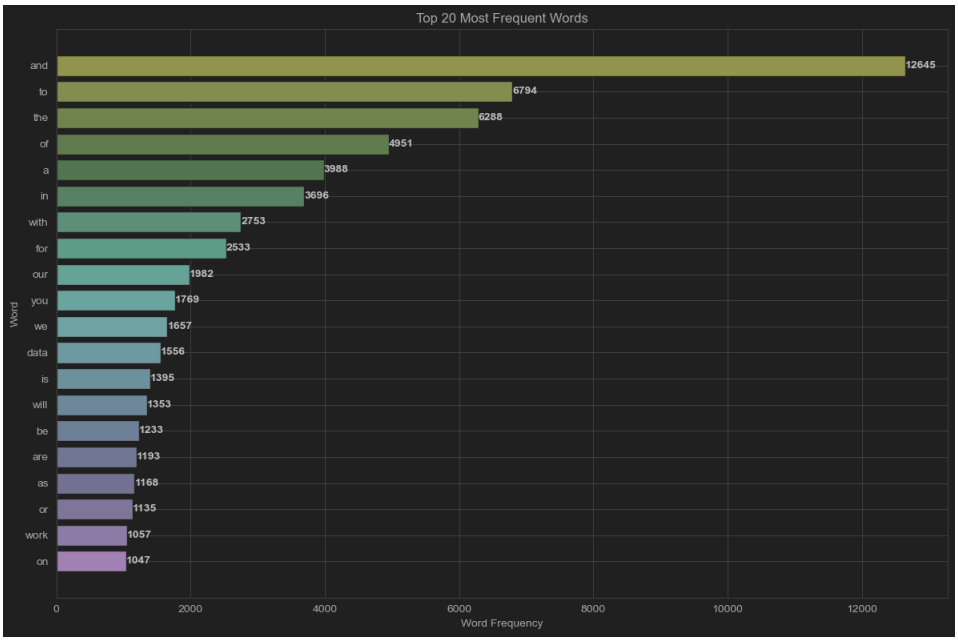


Image 34: Distribution of words in the corpus for text summarization NLP task.

Image 35: WordCloud of unique word frequency count in the corpus for the Word2Vec NLP task.

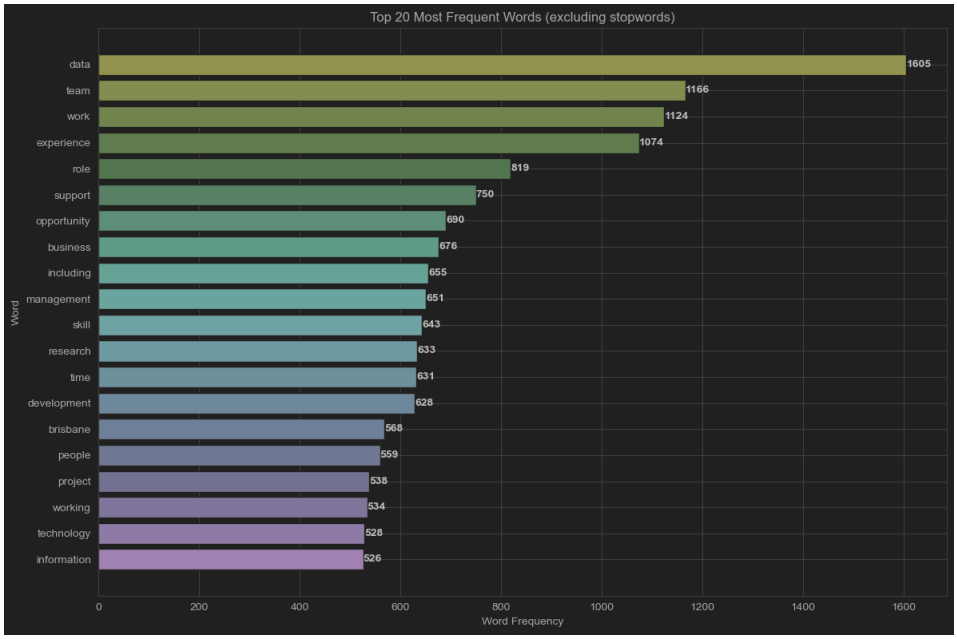


Image 36: Distribution of words in the corpus for Word2Vec NLP task.

Sample	Analysis
Company	In our examination of the 'Company' variable images 12 to 15, we identified 244 unique 'Companies' recruiting based on our query. The 'University of Queensland secures the top two positions, contributing a combined total of 24 job postings. Following closely is the 'Queensland Department of Health' with 9 postings, and 'WSP Australia' with 5. The remaining 'Companies' each post 4 or fewer job advertisements on Seek.
Location	In our examination of the 'Location' variable images 16 to 19, we identified 61 unique 'Locations' based on our query. General 'Brisbane' secures the top positions, contributing a combined total of 188 job postings. Following closely is the 'South Brisbane' with 13 postings, and 'Murarrie' with 10. The remaining 'Locations' each have 9 or fewer job advertisements on Seek.
Department	In our examination of the 'Department' variable images 20 to 23, we identified 130 unique 'Departments' recruiting based on our query. 'Business/Systems Analysts' secures the top positions, contributing a combined total of 24 job postings. Following closely is the 'Database Development & Administration' with 18 postings, and 'Engineering – Software' and 'Laboratory & Technical Services' with 15. The remaining 'Departments' each post 14 or fewer job advertisements on Seek.
Position Type	In our examination of the 'Position Type' variable images 24 to 26, we identified 8 unique 'Positions Types' based on our query. 'Full Time' secures the top positions, contributing a combined total of 277 job postings. Following closely is the 'Contract/ Temp' with 51 postings, and 'Casual/ Vacation' and 'Part Time' with 5. The remaining 'Position Types' each have 3 or fewer job advertisements on Seek.
Salary	Due to the lack of data on salary that was web-scrapped, we will not be using this variable, and will not be discussed. (refer to images 27 & 28)
Job Title	In our examination of the 'Job Tite' variable images 29 to 32, we identified 298 unique 'Job Titles' based on our query. General 'Data Analyst' secures the top positions, 10 job postings. Following closely is the 'Data Engineer with 5 and 'Lab Technician' with 5. The remaining 'Job Titles' each have 3 or fewer job advertisements on Seek. Interestingly, the job titles lack direct mention of 'Data Science,' possibly indicating a growing understanding of the field. It suggests that companies posting these jobs may not fully recognize that the roles and responsibilities align with Data Science, despite the absence of specific terms in the job titles (Frikha, 2018)
LSA Cleaned Corpus	In our examination of the 'LSA Cleaned Corpus', we evaluate the frequency of the word in the corpus. From images 33 and 34, we can conclude that most of the top words are indeed stop words. Referring to our corpus discussion for the LSA ML method, we are keeping the stop wards to provide proper, coherent text summarisations. Therefore, we have 'and' with 12,645 occurrences, followed by 'to' with 6,794 and 'the' with 6,288. The remaining top 10 words that follow are 'a'; 3,988, 'in'; 3,696, 'with'; 2,753, 'for'; 2,533, 'our'; 1,982, 'you'; 1,769, 'we'; 1,657 and 'data'; 1605. Please refer to Image 34 for the remaining words to analyze.
Word2Vec Cleaned Corpus	In our examination of the 'Word2Vec Cleaned Corpus', we evaluate the frequency of words in the corpus. In this instance, after pre-processing which includes removing all stop words, we get a glimpse into the top keywords relevant to our query for Data Science. We can observe in images 35 & 36 that the top word is 'data' with 1,605 occurrences, followed by 'team' with 1,168 and 'work' with 1,124. The

	remaining top 10 are ‘experience’; 1,074, ‘role’; 519, ‘support’; 750, ‘opportunity’; 690, ‘business’; 676, ‘including’; 655 and ‘management’; 651. Please refer to Image 36 for the remaining words to analyze.

Table 6: Summary of EDA of sample distributions and distributions of words in both corpora.

Several biases come to light in defining my web scraping parameters and examining the job postings. In my search criteria, I specifically target Data Science Jobs within a 50 km radius of Brisbane. These biases are introduced by my focus on Data Science positions, potentially overlooking other job categories on Indeed and Seek. Similarly, confining the search to Brisbane and those jobs within a 50 km radius neglects opportunities in other Australian locations and beyond the 50 km radius boundary, and even a global perspective is disregarded. Examining the job postings themselves reveals biases, including preferences for Australian citizens or permanent residents, potentially excluding individuals on other visas. Moreover, subtle biases may arise from certain phrases, potentially involving aspects like Racial, Religious, Age, Gender, and Ability Biases in the job descriptions themselves.

In the context of job data extraction for this assignment, text data limitations can relate to data quality, representation, extraction, and interpretation. Ensuring high-quality text data is important. With that said upon initial web scrapping there often contains noise, inconsistencies, and errors. We can also expect that the job descriptions are unstructured and have high dimensionality which imposes its challenge. Furthermore, ambiguity, language variations, and contextual nuances make accurate data extraction challenging. Overcoming these challenges will involve meticulous text cleaning and preprocessing, strategic representation choices, and careful selection of extraction methods. Understanding text data limits guides an effective approach to sampling jobs, ensuring a robust data extraction process. Refer to Machine Learning: *Specification and justification of the implementation of the ML model* for discussion on train and test split for our NLP tasks.

Machine Learning

Specification and justification of the implementation of the ML model

In our introduction to our report, we introduce our issues of deciphering diverse job posting layouts to understand key role characteristics and staying informed about companies' technical skills expectations. This introduces two NLP tasks, the first task implemented an unsupervised ML extractive text summarisation model method to provide a standard, simplified job summary, making job postings more user-friendly and accessible to a wider range of potential candidates. The second task used another unsupervised ML model to assist in the creation of a diverse list of technical Data Science skills to use and iterate through our corpus to identify which skills are mentioned and how frequently. Based on this we can identify trending technical skills in the job market, specifically from job postings found on Seek.

After investigating several diverse text summarisation models, I have chosen four popular, effective unsupervised ML models; TextRank, LSA, BART, and T5 (Shrivarsheni, 2020), to implement, compare, and determine which model is the most effective. Further, to help in our list creation I will implement the unsupervised ML model, Word2Vec, to find semantic relationships between our initial technical skills words and other closely relevant words (Nuri, 2020) to then enhance our Data Science technical skills list (refer to Table 7 for a full summary of computational environments, each model, the tools implemented in each model and the raw code).

Computational Environment	Specifics	Description
PC: Omen 17.3-inch gaming laptop pc 17-ck0000	Operating system: Windows 11 Home 64-bit Version: 22631.2715	I operated on the Omen 17.3-inch gaming laptop for running my code throughout this report. Every aspect of the project, including data size, computations, processing, web scraping, and data wrangling, seamlessly executed on this computer without requiring any adjustments. The strong computational power of the system effortlessly handled the scaling up of datasets with the selected machine learning models, eliminating the need for any modifications to the computation environment.

	Microprocessor: 11th Gen Intel(R) Core(TM) i9-11900H @ 2.50GHz System memory: 32 GB (two 16GB Samsung 3200MHz) System board: 88FE 86.43 System BIOS: F.44		
PyCharm	System: PyCharm 2023.2.5 (Professional Edition) Runtime version: 17.0.9+7-b1000.46 amd64 VM: OpenJDK 64-Bit Server VM by JetBrains s.r.o. Windows 11.0 GC: G1 Young Generation, G1 Old Generation Memory: 4050M Cores: 16	The Python environment used for this project is on PyCharm version 2023.2.5 (Professional Edition), running on Windows 11. The runtime details include OpenJDK 64-Bit Server VM by JetBrains, with 16 cores and 4050M memory. The implemented packages involve NLTK and Gensim for natural language processing, along with Hugging Face's Transformers for transformer models such as BART and T5. PyTorch is employed for deep learning tasks, and Gensim's Word2Vec facilitates learning dense vector representations of words. Summarization techniques are diverse, including TextRank and Latent Semantic Analysis (LSA) for extractive summarization, BART for extractive summarization with a chunking strategy, and T5 for abstractive summarization. Additionally, Word2Vec is utilized to analyze data science technical skills, creating a list, and ranking them based on frequency in the corpus. The Python environment demonstrates proficiency in handling various natural language processing and machine learning tasks, making it well-suited for the project's requirements. The summarization techniques, coupled with the analysis of data science skills, showcase a comprehensive and capable computational setup.	
Unsupervised Model	**Computational Environments**	**Implemented package tools**	**Code**
TextRank: Extractive Summarisation	NLTK (Natural Language Toolkit): used for natural language processing tasks, including tokenization and parsing. Gensim: used for topic 23odelling and document	Word_tokenize (NLTK): tokenize the job description into words. PlaintextParser (NLTK): parsing the tokenized text. TextRankSummarizer (Gensim): used to	Def summarize_job_description_TextRank(job_description, num_sentences=5): *# Tokenize and parse the job description* words = word_tokenize(job_description) sentences = ‘ ‘.join(words) parser = PlaintextParser.from_string(sentences, Tokenizer(“23odelli”)) *# Use TextRank for summarization* summarizer = TextRankSummarizer() summary = summarizer(parser.document, num_sentences) *# Combine sentences to form the summary text* summary_text = “ “.join(str(sentence) for sentence in summary)

	similarity analysis.	implement the TextRank algorithm; a graph-based algorithm for extractive summarization.	return summary_text job_summary_df['Job_Summaries_TextRank'] = job_summary_df['Corpus'].apply(summarize_job_description_TextRank) print(job_summary_df['Job_Summaries_TextRank'][5])
Latent Semantic Analysis (LSA): Extractive Summarisation	NLTK (Natural Language Toolkit): used for natural language processing tasks, including tokenization and parsing. Gensim: used for topic 24odelling and document similarity analysis.	Word_tokenize (NLTK): tokenize the job description into words. PlaintextParser (NLTK): parsing the tokenized text. LsaSummarizer (Gensim): used to implement Latent Semantic Analysis (LSA), a technique for dimensionality reduction and summarization.	Def summarize_job_description_lsa(job_description): *# Tokenize and parse the job description* words = word_tokenize(job_description) sentences = ' '.join(words) parser = PlaintextParser.from_string(sentences, Tokenizer("24odelli")) *# Use LSA (Latent Semantic Analysis) for summarization* summarizer = LsaSummarizer() summary = summarizer(parser.document, 8) *# Adjust the number of sentences as needed* *# Combine sentences to form the summary text* summary_text = " ".join(str(sentence) for sentence in summary) return summary_text *# Add the summaries to the DataFrame* job_summary_df['Job_Summaries_lsa'] = job_summary_df['Corpus'].apply(summarize_job_description_lsa)

BART: Extractive Summarisation	Transformer (from Hugging Face): provides pre-trained transformer models, in this case, Bart, and utilities for natural language processing tasks, including summarization.	Pipeline (transformer): used to load the BART-based summarization model (facebook/bart-large-cnn)
		
		bart_summarizer (transformer): employed by bart_summarizer to generating summaries.
		
		**chunk_size: the maximum length of text for the bart_summarizer is 1024. For corpus documents that are longer than 1024 are split into chunks and the summaries are made for each chunk and then
			# Load the BART-based summarization model
			bart_summarizer = pipeline("summarization", model="facebook/bart-large-cnn")
			
			# Create an empty list to store the summaries
			summaries = []
			
			# Set chunk size to the maximum sequence length
			chunk_size = 1024
			
			# Iterate through each document in the corpus
			for idx, job_description in enumerate(job_summary_df['Corpus']):
			# Split the job description into chunks
			chunks = [job_description[i:I + chunk_size] for I in range(0, len(job_description), chunk_size)]
			
			# Generate a summary for each chunk
			chunk_summaries = []
			for chunk in chunks:
			summary = bart_summarizer(chunk, max_length=120, min_length=50, length_penalty=1.0, num_beams=6)
			chunk_summaries.append(summary[0]['summary_text'])
			
			# Concatenate the chunk summaries to form the final summary
			final_summary = ' '.join(chunk_summaries)
			
			# Print progress
			print(f"Summarization {idx + 1} completed")
			
			# Append the final summary to the list
			summaries.append(final_summary)

		concatenated to form a final summary.	*# Add the summaries to the DataFrame* job_summary_df['Job_Summaries_Bart'] = summaries

T5: Abstractive Summarisation	Transformer (from Hugging Face): provides pre-trained transformer models, in this case, T5, and utilities for natural language processing tasks, including summarization.
	
	PyTorch: open-source deep learning library used for building neural network models.
		Tokenizer.encode (T5Tokenizer.from_pretrained): used to tokenize the input text; converts the input text into a format suitable for the T5 model.
		
		Model.generate (T5ForConditionalGeneration.from_pretrained): used to generate the summary based on the encoded input; takes the tokenized input and produces the summary.
			# Load pre-trained T5 model and tokenizer
			tokenizer = T5Tokenizer.from_pretrained("t5-small")
			model = T5ForConditionalGeneration.from_pretrained("t5-small")
			
			def t5_summarizer(text, max_length=300, min_length=50):
			  # Tokenize the input text
			  inputs = tokenizer.encode("summarize: " + text, return_tensors="pt", max_length=1024, truncation=True)
			
			  # Generate the summary
			  summary_ids = model.generate(inputs, max_length=max_length, min_length=min_length, length_penalty=1.0, num_beams=6, early_stopping=False)
			
			  # Decode the summary
			  summary = tokenizer.decode(summary_ids[0], skip_special_tokens=True)
			
			  return summary
			
			# Apply T5 summarization to each document
			summaries = []
			
			for idx, job_description in enumerate(job_summary_df['Corpus']):
			  summary = t5_summarizer(job_description)
			  print(f"Summarization {idx + 1} completed")
			  summaries.append(summary)
			
			# Add the summaries to the DataFrame
			job_summary_df['Job_Summaries_T5'] = summaries

Word2Vec	NLTK (Natural Language Toolkit): used for natural language processing tasks, including tokenization and parsing. Gensim: used for topic 25odelling and document similarity analysis.	Word_tokenize (NLTK): used for tokenization to preprocess the text data. Word2Vec (Gensim): used to learn dense vector representations of words, capturing semantic relationships by placing words with similar meanings closer together in a continuous vector space using cosine similarity. Word2vec_model.wv (Gensim): provides access to the vectors	Corpus = job_tech_skills_df['Corpus'].apply(word_tokenize).tolist() *# Define and train the Word2Vec model* word2vec_model = Word2Vec(sentences=corpus, vector_size=1000, window=50, min_count=1, workers=4) *# Save the trained model to a file for future use* word2vec_model.save("word2vec_model.bin") *# Get the vector for each skill* python_vector = word2vec_model.wv['python'] sql_vector = word2vec_model.wv['sql'] azure_vector = word2vec_model.wv['azure'] power_bi_vector = word2vec_model.wv['powerbi'] tableau_vector = word2vec_model.wv['tableau'] boomi_vector = word2vec_model.wv['boomi'] *# Find similar words for each skill* similar_python = word2vec_model.wv.most_similar('python', topn=30) similar_sql = word2vec_model.wv.most_similar('sql', topn=30) similar_azure = word2vec_model.wv.most_similar('azure', topn=30) similar_power_bi = word2vec_model.wv.most_similar('powerbi', topn=30) similar_ability = word2vec_model.wv.most_similar('tableau', topn=30) similar_boomi = word2vec_model.wv.most_similar('boomi',

		associated with individual words in the vocabulary. Word2vec_model.wv.most_similar (Gensim): finds the words most like the word provided based on cosine similarity.	topn=30) *# Print the results that display associated words and their cosine similarities to the searched word* display("Similar words for Python:", similar_python) display("Similar words for SQL:", similar_sql) display("Similar words for Azure:", similar_azure) display("Similar words for Power BI:", similar_power_bi) display("Similar words for Tableau:", similar_ability) display("Similar words for Boomi:", similar_boomi) *#Craete Data Science technical skills list based on Word2Vec and domain knowledge* data_science_skills = ['python', 'sql', 'machine learning', 'data analysis', 'data visualization', 'azure', 'power bi', 'deep learning', 'devops', 'linux', 'javascript', 'web development', 'oracle', 'snowflake', 'git', 'analytics', 'gcp', 'analytical skills', 'bi', 'qgis', 'communication skills','boomi', 'tableau', 'devops', 'programming', 'qlik', 'scada', 'aws', 'jenkins', 'waterfall', 'transact', 'gitlab', 'frontend', 'backend', 'database', 'sharepoint', 'qlik', 'llm', 'uml','linq', 'tensorflow', 'pyspark', 'mstest', 'pentaho'] *# Search for occurrences of skills in the corpus* for skill in data_science_skills: job_tech_skills_df[skill] = job_tech_skills_df['Corpus'].str.contains(skill, case=False) *# Rank the skills based on frequency* skill_frequency = job_tech_skills_df[data_science_skills].sum().sort_values(ascending =False) *# Convert to DataFrame* skills_df = pd.DataFrame({'Skills': skill_frequency.index, 'Frequency': skill_frequency.values}) *# Display the ranked skills DataFrame* display(skills_df)

Table 7: Overview of unsupervised models used for text summarisation and technical skills list enhancement NLP tasks.

Insightful knowledge is demonstrated through a comprehensive exploration of limitations and challenges encountered during hyperparameter tuning, showcasing an understanding of model behavior and optimization challenges. To determine the hyperparameter values for each model, I manually implemented different values to analyze the outputs. After many iterations, the final hyperparameter values were chosen based on the summaries produced that met the criteria that I set; the summaries must mention the job title, company details, technical skills, and soft skills. Additionally, the summaries were to be the shortest length that would allow the inclusion of these features. Table 8 provides an overview of the hyperparameters and the final values chosen for each model. During this process, the summary output for the T5 was not producing the desired output summaries (refer to Image 40), regardless of the multiple hyperparameter tuning attempts. Therefore, the T5 model was excluded from the remainder of the report.

Unsupervised Model	Hyperparameters	Code
TextRank	num_sentences: determines the number of sentences in the summary.	def summarize_job_description_TextRank(job_description, num_sentences=5)

Latent Semantic Analysis (LSA)	summarizer(document, number of sentences): determines the number of sentences in the summary	*# Use LSA (Latent Semantic Analysis) for summarization* summarizer = LsaSummarizer() summary = summarizer(parser.document, 8) *# Adjust the number of sentences as needed*
BART	max_length: maximum length of the output summary, min_length: minimum length of the output summary, length_penalty: controls the penalty for generating longer summaries; A higher value, means that longer summaries are penalized less compared to shorter ones, num_beam: explores multiple possible sequences during generation and keeps the top-n sequences based on their likelihood.	summary = bart_summarizer(chunk, max_length=120, min_length=50, length_penalty=1.0, num_beams=6)
T5	max_length: maximum length of the output summary, min_length: minimum length of the output summary, max_length(chunki): maximum length of chunk text to conduct summary on.	def t5_summarizer(text, max_length=300, min_length=50): *# Tokenize the input text* inputs = tokenizer.encode("summarize: " + text, return_tensors="pt", max_length=1024, truncation=True)
Word2Vec	vector_size: dimensionality of the word vectors, window: maximum distance between the current and predicted word within a sentence; considers the context of words to the left and the right of the target word, min_count: minimum count of a word in the corpus for it to be included in the model, workers: number of CPU cores to use for training the model, word2vec_model.wv: extracts the word vector for the specified word from the trained Word2Vec model,	*# Define and train the Word2Vec model* word2vec_model = Word2Vec(sentences=corpus, vector_size=1000, window=50, min_count=1, workers=4) *# Get the vector for each skill* python_vector = word2vec_model.wv['python'] *# Find similar words for each skill* similar_python = word2vec_model.wv.most_similar('python', topn=30)

	word2vec_model.wv.most_similar; finds the top set number of words that are most like the specified word based on their word vectors.	

Table 8: Overview of unsupervised models hyperparameters for conducting text summarisation and technical skills list enhancement NLP tasks.

Given the unsupervised nature of the machine learning tasks in my report, which involve text summarization and semantic relationship analysis, there is no requirement for a train and test split (Train-Test Datasets in Machine Learning, n.d.). Unsupervised models, such as TextRank, LSA, BART, T5, and Word2Vec, learn patterns directly from the entire dataset without relying on labeled examples, making traditional train/test evaluations unnecessary for the specified tasks. Overall, based on the extensive research into text summarisation for creating a standard, concise, and simplistic summary of Seek job postings, I have chosen to implement TextRank, LSA, and Bart. Following their implementation and execution, I used Recall-Oriented Understudy for Gisting Evaluation (ROUGE), in addition to my evaluation of a proper summarization, to determine which model performed the best. Additionally, I implemented Word2Vec to assist in building a Data Science technical skills list where we used initial technical skills phrases obtained from domain knowledge to find semantic relationships between those and other associated words that are then hand-picked to add to our list to use for trend analysis (refer to images 37 to 40 for example summary outputs for each model and image 41 for Word2Vect output).

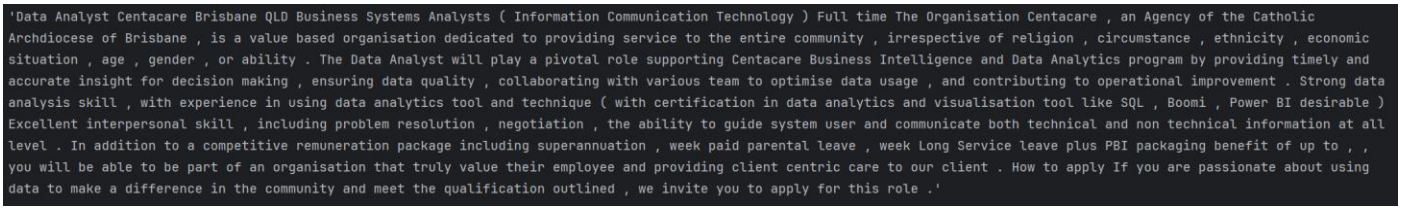


Image 37: Text summarisation output example using TextRank.

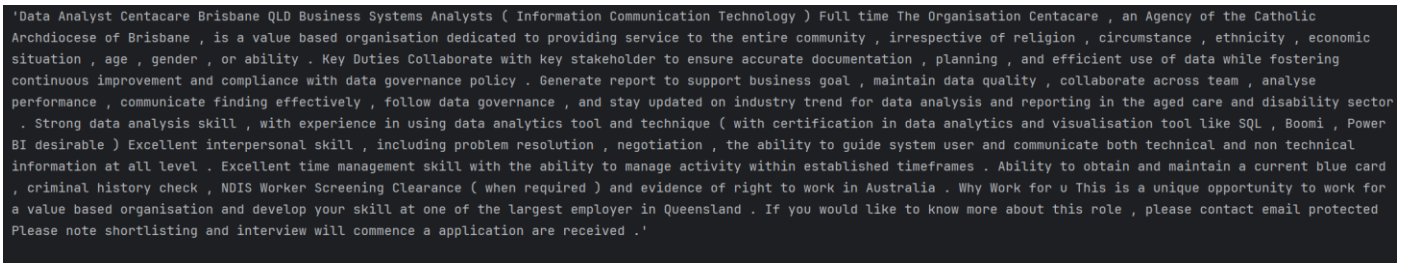


Image 38: Text summarisation output example using LSA.

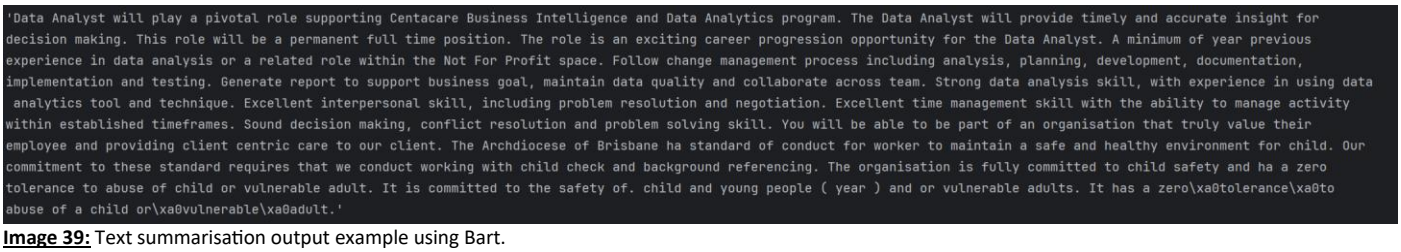


Image 39: Text summarisation output example using Bart.

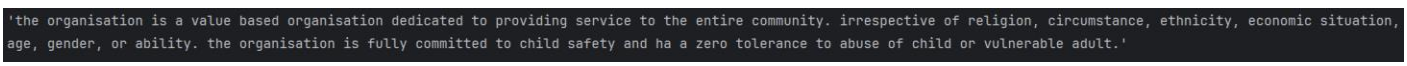


Image 40: Text summarisation output example using T5.

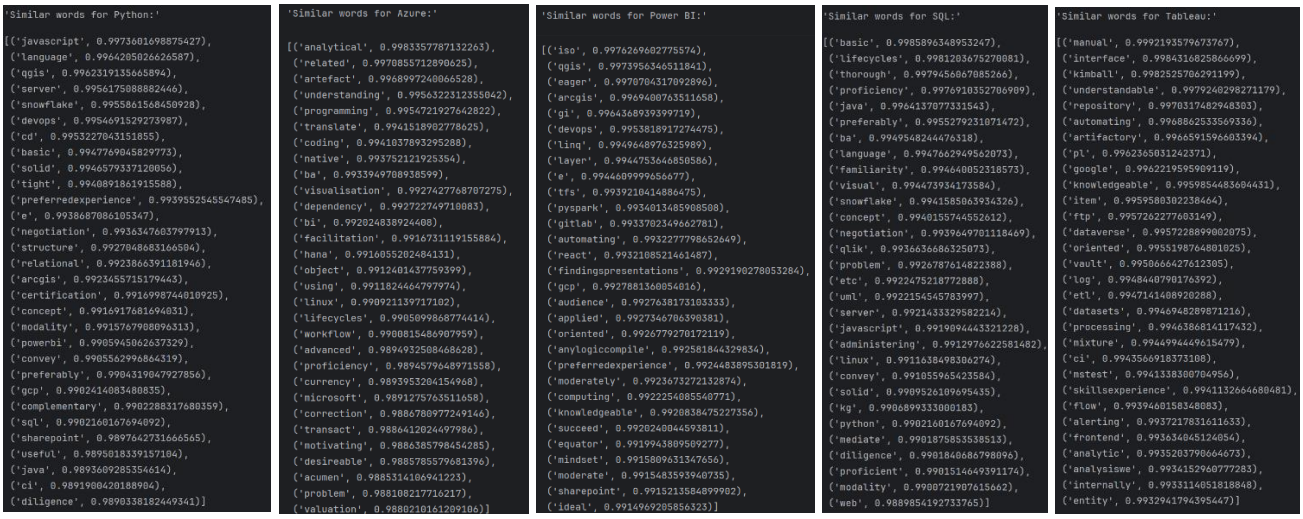


Image 41: Word2Vec output for semantic relationships between the initial technical skill words and other associated words.

Evaluation and visualization of the machine learning model performance

In evaluating the performance of the unsupervised machine learning models for text summarization (TextRank, LSA, and Bart), we implement ROUGE metrics: a set of metrics for evaluating automatic summarization of texts as well as machine translations, by comparing an automatically produced summary or translation; by the model, against a set of reference summaries, typically human-produced. (Ganesan, 2017). Specifically, by implementing ROUGE, we evaluate the performance of each summary by using 'precision', 'recall', and 'F1' scores (refer to Table 9 for score metric explanation and raw code). The methodology for ROUGEs evaluations is the measured overlap between the n-grams, or tokens, in the reference (job postings) and hypothesis (model made) summaries. The matching can be done based on exact matches or partial matches. In our case, we conduct matches based on unigrams, bigrams, and trigrams between the reference and hypothesis summaries.

Therefore, at the most granular layer, we have a precision, recall, and f1 score for each n-gram case, for each document in the corpus, based on each of the summarization models. We then take the n-gram scores for each document and combine them to take an average n-gram precision, recall, and f1 score for each document associated with each summarization model type. Finally, we take the average scores for each summarization model to determine the best-performing text summarisation unsupervised ML model (refer to Images 42 & 43 for sample output of average document scores and Images 44 & 45 for results). From Images 44 & 45, we can see that TextRank and LSA are tied for best evaluation scores. As a result, further analysis was done by randomly selecting summaries from both models and evaluating them based on my own criteria mentioned earlier. In the end, the best text summarization model, in my opinion, was LSA and was chosen as the best-unsupervised ML extractive text summarisation model for Data Science job positions on Seek within Brisbane.

Evaluation Metric	**Description**	**Equation**	**N-Grams**	**Models**	**Code**

Precision	Determines the relevance of the generated summary. Instead of dividing by the reference n-gram count, we divide by the output n-gram count when	Precision = Word count in generated summary/Word count of overlapping words in the generated summary.	Unigram
			Bigram
			Trigram
				TextRank
				LSA
				Bart
					#Create a list of methods used
					methods = ['TextRank', 'lsa', 'Bart']
					references = job_summary_df['Corpus']
					
					# Iterate over each summarization method
					for method in methods:
					  hypotheses = job_summary_df[f'Job_Summaries_{method}']
					
					  # Set up the data for Rouge
					  for idx, (hypothesis, reference) in enumerate(zip(hypotheses, references)):
					  # Run Rouge for unigrams (ROUGE-1)

	calculating precision. (ROUGE Metric, n.d.)				
Recall	The model output and reference are compared for the number of overlapping n-grams, and then this number is divided by the total number of n-grams in the reference. (ROUGE Metric, n.d.)	Recall = Word count in reference summary / Word count of overlapping words in the reference.	Unigram Bigram Trigram	TextRank LSA Bart	
F1	Measure of the performance of generated output based on recall and precision values. (ROUGE Metric, n.d.)	F1 score = 2 ((precision recall) / (precision + recall))	Unigram Bigram Trigram	TextRank LSA Bart	

```
    scores_unigram = rouge_score.rouge_n(hypothesis, reference, 1)

    # Run Rouge for bigrams (ROUGE-2)
    scores_bigram = rouge_score.rouge_n(hypothesis, reference, 2)

    # Run Rouge for trigrams (ROUGE-3)
    scores_trigram = rouge_score.rouge_n(hypothesis, reference, 3)

    # Calculate average scores for precision, recall, and F1 across all n-gram types
    avg_precision = (scores_unigram['p'] + scores_bigram['p'] + scores_trigram['p']) / 3
    avg_recall = (scores_unigram['r'] + scores_bigram['r'] + scores_trigram['r']) / 3
    avg_f1 = (scores_unigram['f'] + scores_bigram['f'] + scores_trigram['f']) / 3

    # Print the results for each document and method
    print(f"\nROUGE Scores for Document {idx + 1} - {method} (Average):")
    pprint({
      'precision': avg_precision,
      'recall': avg_recall,
      'f': avg_f1
    })

    # Add average scores to job_summary_df
    job_summary_df.at[idx, f'{method}_Avg_Precision'] = avg_precision
    job_summary_df.at[idx, f'{method}_Avg_Recall'] = avg_recall
    job_summary_df.at[idx, f'{method}_Avg_F1'] = avg_f1

print("\nUpdated DataFrame with ROUGE Scores:")
print(job_summary_df)

#Take the average score for each evaluation metric for the TextRank summarization method
tr_p_average_value = job_summary_df['TextRank_Avg_Precision'].mean()
tr_r_average_value = job_summary_df['TextRank_Avg_Recall'].mean()
tr_f1_average_value = job_summary_df['TextRank_Avg_F1'].mean()

# Round the average values to two decimal places
tr_p_average_rounded = round(tr_p_average_value, 2)
tr_r_average_rounded = round(tr_r_average_value, 2)
tr_f1_average_rounded = round(tr_f1_average_value, 2)

print(f"Average evaluation scores of TextRank are: Precision - {tr_p_average_rounded}, Recall - {tr_r_average_rounded}, and F1 - {tr_f1_average_rounded}.")
```

```
#Take the average score for each evaluation metric for the LSA summarization method
lsa_p_average_value = job_summary_df['lsa_Avg_Precision'].mean()
lsa_r_average_value = job_summary_df['lsa_Avg_Recall'].mean()
lsa_f1_average_value = job_summary_df['lsa_Avg_F1'].mean()

# Round the average values to two decimal places
lsa_p_average_rounded = round(lsa_p_average_value, 2)
lsa_r_average_rounded = round(lsa_r_average_value, 2)
lsa_f1_average_rounded = round(lsa_f1_average_value, 2)

print(f"Average evaluation scores of LSA are: Precision - {lsa_p_average_rounded}, Recall - {lsa_r_average_rounded}, and F1 - {lsa_f1_average_rounded}.")

#Take the average score for each evaluation metric for the Bart summarization method
bart_p_average_value = job_summary_df['Bart_Avg_Precision'].mean()
bart_r_average_value = job_summary_df['Bart_Avg_Recall'].mean()
bart_f1_average_value = job_summary_df['Bart_Avg_F1'].mean()

# Round the average values to two decimal places
bart_p_average_rounded = round(bart_p_average_value, 2)
bart_r_average_rounded = round(bart_r_average_value, 2)
bart_f1_average_rounded = round(bart_f1_average_value, 2)

print(f"Average evaluation scores of Bart are: Precision - {bart_p_average_rounded}, Recall - {bart_r_average_rounded}, and F1 - {bart_f1_average_rounded}.")
```

Table 8: Evaluation score metric explanation and raw code.

```
ROUGE Scores for Document 1 - TextRank (Average):
{'f': 0.9645847998327644, 'precision': 1.0, 'recall': 0.9326086791285684}

ROUGE Scores for Document 2 - TextRank (Average):
{'f': 0.9571946598989376, 'precision': 1.0, 'recall': 0.9182310116238189}

ROUGE Scores for Document 3 - TextRank (Average):
{'f': 0.8646578110682847, 'precision': 1.0, 'recall': 0.7699575804735439}

ROUGE Scores for Document 4 - TextRank (Average):
{'f': 0.9249626866678158, 'precision': 1.0, 'recall': 0.8637357454478076}

ROUGE Scores for Document 5 - TextRank (Average):
{'f': 0.999999995, 'precision': 1.0, 'recall': 1.0}

ROUGE Scores for Document 6 - TextRank (Average):
{'f': 0.8392473114656628, 'precision': 1.0, 'recall': 0.7310227141784139}

ROUGE Scores for Document 7 - TextRank (Average):
{'f': 0.9849339691036686, 'precision': 1.0, 'recall': 0.9705959183673469}
```

```
ROUGE Scores for Document 1 - lsa (Average):
{'f': 0.9892822380707308, 'precision': 1.0, 'recall': 0.9789904571852533}

ROUGE Scores for Document 2 - lsa (Average):
{'f': 0.987160174179566, 'precision': 1.0, 'recall': 0.9748350525044142}

ROUGE Scores for Document 3 - lsa (Average):
{'f': 0.8705971870532684, 'precision': 1.0, 'recall': 0.7761114303329503}

ROUGE Scores for Document 4 - lsa (Average):
{'f': 0.9284392566779616, 'precision': 1.0, 'recall': 0.8680751075897502}

ROUGE Scores for Document 5 - lsa (Average):
{'f': 0.999999995, 'precision': 1.0, 'recall': 1.0}

ROUGE Scores for Document 6 - lsa (Average):
{'f': 0.8876441191972019, 'precision': 1.0, 'recall': 0.8039225181598063}

ROUGE Scores for Document 7 - lsa (Average):
{'f': 0.999999995, 'precision': 1.0, 'recall': 1.0}

ROUGE Scores for Document 8 - lsa (Average):
{'f': 0.8925240162360897, 'precision': 1.0, 'recall': 0.8157199752652491}
```

```
ROUGE Scores for Document 1 - Bart (Average):
{
    'f': 0.8300379234253302,
    'precision': 0.9817972851240953,
    'recall': 0.7279157555518433,
}

ROUGE Scores for Document 2 - Bart (Average):
{
    'f': 0.7867730744812084,
    'precision': 0.9010434545996787,
    'recall': 0.7123753024226956,
}

ROUGE Scores for Document 3 - Bart (Average):
{
    'f': 0.8240985326465905,
    'precision': 0.9757187537591724,
    'recall': 0.7217994232851366,
}
```

Image 42: Sample output of average ROUGE evaluation scores for each document in the corpus for each text summarisation method used, part 1.

	TextRank_Avg_Recall	TextRank_Avg_F1	lsa_Avg_Precision	lsa_Avg_Recall	lsa_Avg_F1	Bart_Avg_Precision	Bart_Avg_Recall	Bart_Avg_F1
0	0.932601	0.964585	1.0	0.978990	0.989282	0.981797	0.727916	0.830038
1	0.918231	0.957195	1.0	0.974835	0.987160	0.901043	0.712375	0.786773
2	0.769976	0.864658	1.0	0.776111	0.870597	0.975719	0.721799	0.824091
3	0.863736	0.924963	1.0	0.868075	0.928439	0.985308	0.702469	0.813424
4	1.000000	1.000000	1.0	1.000000	1.000000	0.956360	0.729899	0.823385
..	...	...	...	...	...	...	...	...
341	0.869546	0.926845	1.0	0.920978	0.957870	0.936078	0.744503	0.819265
342	0.810682	0.894149	1.0	0.960460	0.979486	0.982186	0.687070	0.804128
343	0.529982	0.687570	1.0	0.738404	0.838341	0.980596	0.770105	0.854140
344	0.764602	0.864730	1.0	0.890831	0.940054	0.979864	0.729910	0.826545
345	0.745349	0.847151	1.0	0.789464	0.872352	0.964491	0.741611	0.829409

Image 43: Sample output of average ROUGE evaluation scores for each document in the corpus for each text summarisation method used, part 2.

```
Average evaluation scores of TextRank are: Precision - 1.0, Recall - 0.86, and F1 - 0.92.
Average evaluation scores of LSA are: Precision - 1.0, Recall - 0.86, and F1 - 0.92.
Average evaluation scores of Bart are: Precision - 0.97, Recall - 0.72, and F1 - 0.82.
```

Image 44: Results for ROUGE evaluation metric scores for text summarisation models.

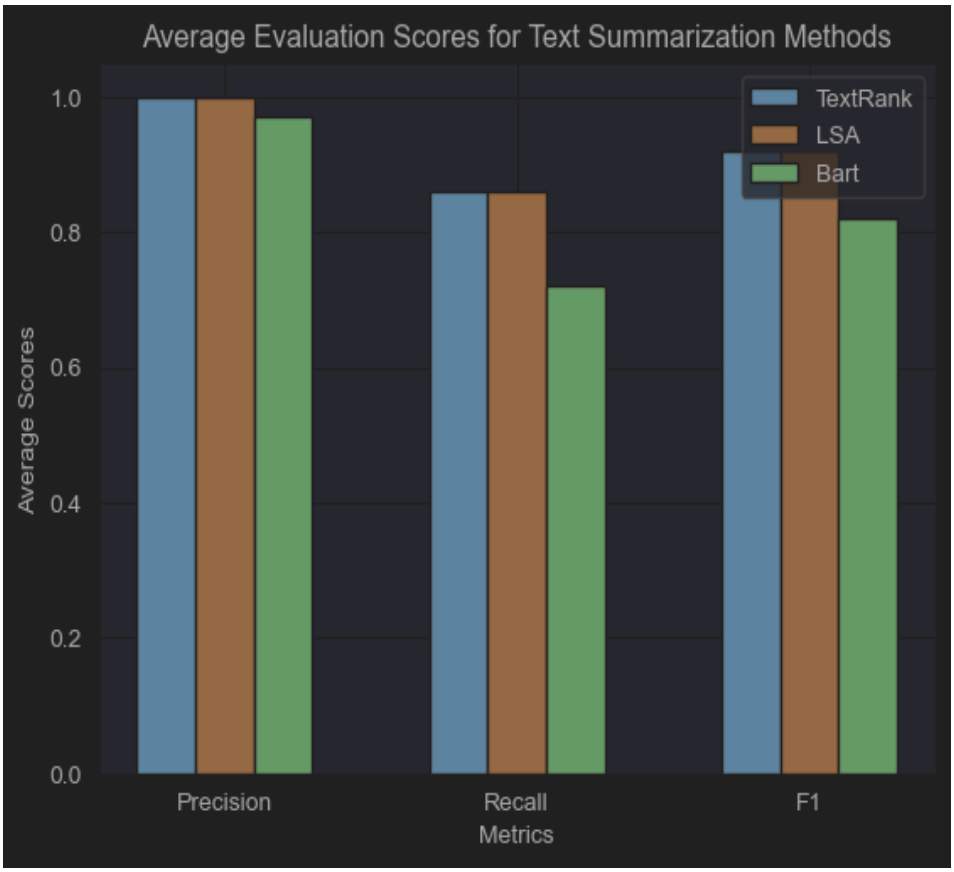


Image 45: Visual representation of results for ROUGE evaluation metric scores for text summarisation models.

Word2Vec enhances our technical skills list by identifying semantically related words to the initial set. Unlike traditional supervised models, evaluation isn't needed for this Word2Vec application, as we aren't predicting or assessing labeled data. Word2Vec focuses on exploring semantic relationships through

cosine similarity, refining, and expanding our skills list. This approach yields a more comprehensive set for technical skill trend analysis in the Data Science job market. Our results (refer to Images 46 to 48) highlight the top 11 trending technical skills, led by 'bi' (Business Intelligence: analytical skills for effective data evaluation and sharing), followed by 'analytics', 'git', 'database', 'SQL', 'data analysis', 'python', 'programming', 'azure', 'power bi', and 'aws'.

1-11 44 rows × 2 columns pd.DataFrame

	Skills	Frequency
0	bi	340
1	analytics	81
2	git	80
3	database	71
4	sql	63
5	data analysis	62
6	python	50
7	programming	45
8	azure	39
9	power bi	29
10	aws	23

Image 46: Frequency count of technical skills identified in the corpus.

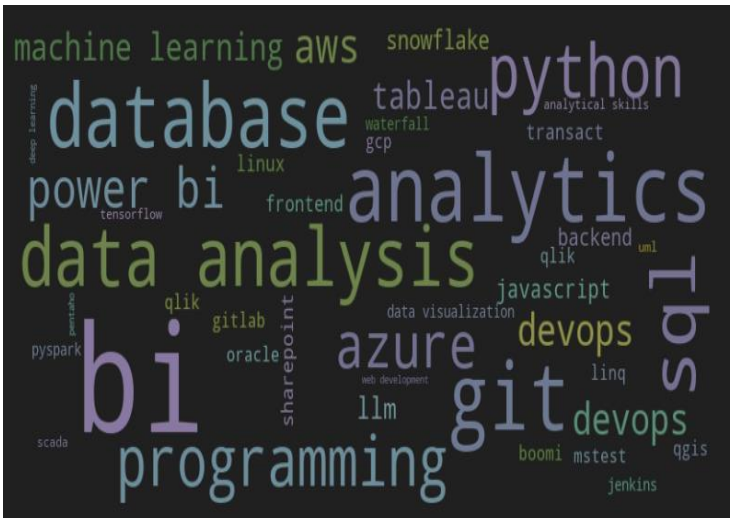


Image 47: WordCloud top trending technical skills identified in the corpus.

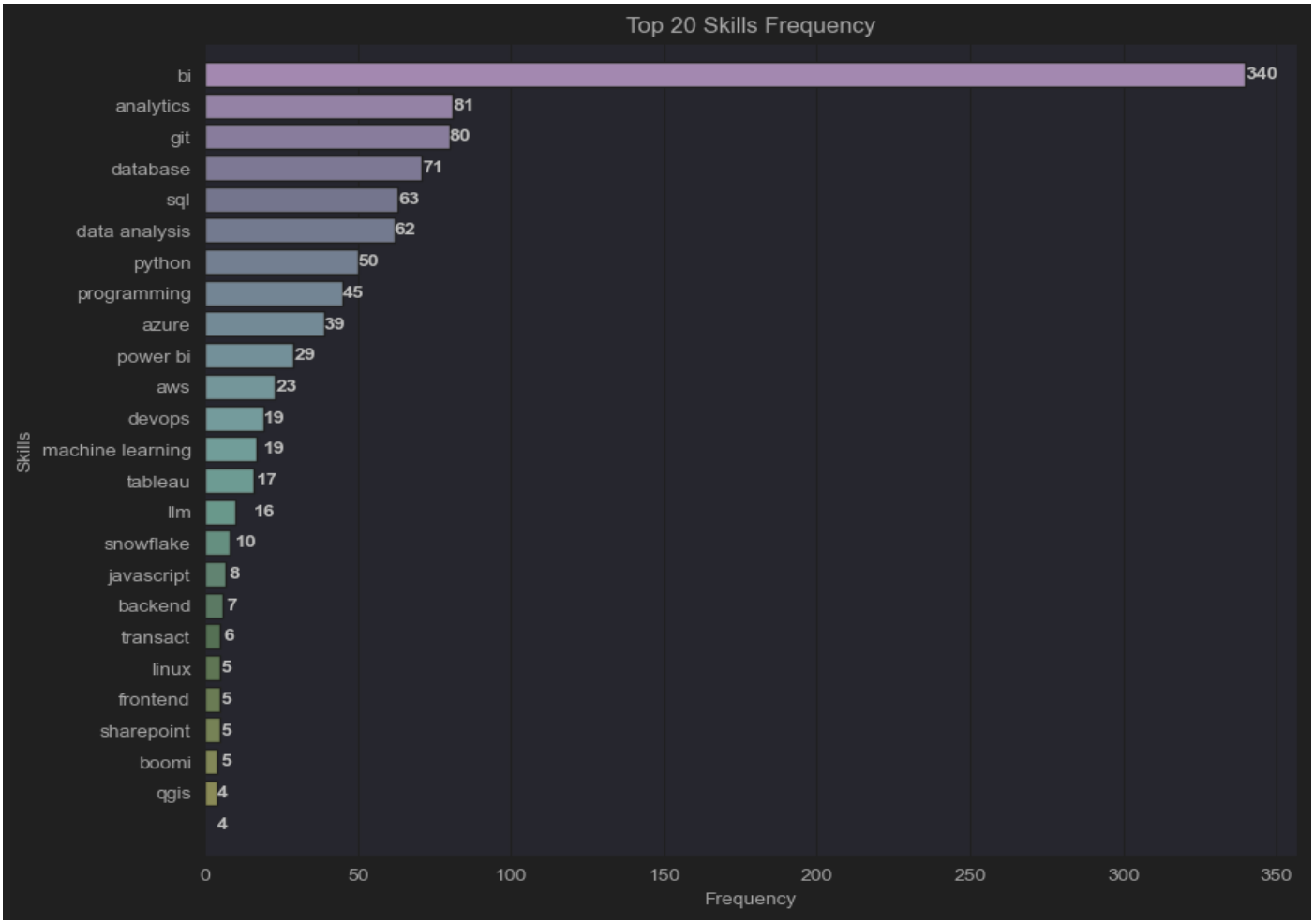


Image 48: Visual distribution of top trending technical skills identified in the corpus.

Effect of the data limitations and sampling biases on the machine learning performance

In the context of Data Science jobs in Brisbane within a 50 km radius, the impact of data limitations and sampling biases on machine learning model performance is something to acknowledge. The geographical

constraint of the dataset introduces limitations, potentially restricting the models' ability to generalize beyond the specified region. If the training data predominantly represents certain types of Data Science roles or specific industries within Brisbane, the models may exhibit biases towards those sectors, influencing the summarization of job descriptions. The localized sampling bias could result in summaries that are more tailored to the characteristics of Data Science positions prevalent in the Brisbane area, potentially limiting the models' adaptability to job descriptions from other regions or other job positions. Mitigating these challenges requires careful consideration of the geographic constraints and a deliberate effort to diversify the dataset within the specified region. Strategies should be implemented to ensure that the summarization models capture a comprehensive representation of the diverse Data Science roles and industries relevant to the local job market dynamics, promoting more robust and inclusive machine learning model performance.

References

Barrett, A. (2023, January 14). *7 Web Scraping Limitations You Should Know*. (n.d.). Www.octoparse.com. Retrieved December 7, 2023, from https://www.octoparse.com/blog/web-scraping-limitations

BasuMallick, C. (2022, December 19) *BI Analyst Job Description, Key Skills, and Salary 2022*. Spiceworks. Retrieved December 9, 2023, from https://www.spiceworks.com/tech/it-careers-skills/articles/business-intelligence-analyst/#:~:text=Business%20Intelligence%20Analyst%3F-

Data Basecamp. (2023, August 5). *What is Word2Vec? | Data Basecamp*. Retrieved December 7, 2023, from https://databasecamp.de/en/ml/word2vec-en

Frikha, S. (2018, September 14). *Data jobs, why you shouldn't give a f*** about job titles!* Medium. Retrieved December 7, 2023, from https://medium.com/@slimfrikha/fake-data-jobs-9b1c74bf7da8

Ganesan, K. (2017, January 26). *An intro to ROUGE, and how to use it to evaluate summaries*. FreeCodeCamp.org. Retrieved December 7, 2023, from https://www.freecodecamp.org/news/what-is-rouge-and-how-it-works-for-evaluation-of-summaries-e059fb8ac840/

Gere, A. (2015, March 11). *The Six Common Problems With Job Descriptions*. Gere Consulting Associates. Retrieved December 8, 2023, fromhttps://gereconsulting.com/the-six-common-problems-with-job-descriptions/

Humphry, J. (2022, March 3) *7 things companies get wrong when writing job descriptions.* www.factcompany.com. Retrieved December 8, 2023 from, https://www.fastcompany.com/90736047/7-things-companies-get-wrong-when-writing-job-descriptions

Is Web Scraping Legal in Australia? - Web Scraping FYI. (n.d.). Webscraping.fyi. Retrieved December 7, 2023, from https://webscraping.fyi/legal/AU/

Kelly, R (2018, May 8). *5 Easy Ways to Have More Consistent Job Descriptions*. Ongig Blog. Retrieved December 8, 2023, from https://blog.ongig.com/job-descriptions/5-easy-ways-to-have-more-consistent-job-descriptions/

MonkeyLearn (n.d.) *Natural Language Processing (NLP): What Is It & How Does it Work?* MonkeyLearn. Retrieved December 9, 2023, from https://monkeylearn.com/natural-language-processing/#:~:text=Natural%20Language%20Processing%20helps%20machines

Nuri. (2020, December 31). *Word2Vec for Talent Acquisition*. Medium. Retrieved December 9, 2023, from https://medium.com/@nuripurswani/word2vec-for-talent-acquisition-ab20a23e01d8

ROUGE Metric. (n.d.). Www.linkedin.com. Retrieved December 8, 2023, from https://www.linkedin.com/pulse/rouge-metric-diya-mathew/

SEEK. (2019). *About SEEK*. Seek. Retrieved December 8, 2023, from, https://www.seek.com.au/about/

SEEK - Terms and Conditions. (n.d.). SEEK. Retrieved December 8, 2023, from, https://www.seek.com.au/terms/

Shrivarsheni (2020, October 24). *Text Summarization Approaches for NLP - Practical Guide with Generative Examples*. Machine Learning Plus. Retrieved December 7, 2023, from https://www.machinelearningplus.com/nlp/text-summarization-approaches-nlp-example/

Train-Test Datasets in Machine Learning. (n.d.). Finance Train. Retrieved December 7, 2023, from, https://financetrain.com/train-test-datasets-in-machine-learning

TRITON AI PTE LTD (2023, May 16) *Job Description Optimization: Using AI to Attract the Right Candidates*. (n.d.). Www.linkedin.com. Retrieved December 7, 2023, from https://www.linkedin.com/pulse/job-description-optimization-using-ai-attract-right-candidates/

Vanderwillik, J. (n.d.). *User Agent for Web Scraping*. Bright Data. Retrieved December 7, 2023, from https://brightdata.com/blog/how-tos/user-agents-for-web-scraping-101#:~:text=A%20user%20agent%20(UA)%20string

Zenrows. (2023, March 6) *Most Common HTTP Headers for Web Scraping.* Zenrows blog. Retrieved December 7, 2023, from https://www.zenrows.com/blog/web-scraping-headers

Github: https://github.com/JustinGrima/A3_Justin_Grima.ipynb