Software Bros Project Report:
An Analysis of Software Used in Modern Canada

Thomas Sabourin, Jacob Zhang, Kaan Un, Isaac Gabriel
Department of Engineering, University of Ottawa
SDS3386: Data Science Lab
Prof. Tanya Schmah
December 13, 2023

Introduction

Finding job opportunities can be challenging. Companies have various requirements in order to employ new people, and one of the requirements often present in job applications is knowledge of one or many software packages. This can range from knowing Microsoft Excel to programming languages to knowing more niche software packages.

It may feel like a daunting task to understand the software needs landscape around the job market for a student looking to start his or her career for example. Many questions come to mind : what industry corresponds best to my software package arsenal? If I want to be employed in a specific industry, which software should I be learning? For this reason, the following question will be analyzed throughout this report:

What are the popular software packages among Canadian companies from different industry sectors?

A better understanding of the software needs among Canadian companies and different industry sectors is insightful for job searching. Moreover, it can be useful for someone looking to start his or her own company, to know what is the norm in terms of software use in the industry he or she is part of. Finally, it can be useful for someone that creates software packages to know which industry his or her marketing should be focused on.

Literature Review

Landscapes of different software packages were made before. For example, there exists a MAD (ML/AI/Data) landscape of software packages, where the packages are grouped into different sections and subsections, for example Infrastructure - Storage. However, this analysis focuses on specific types of software and groups them, the goal being visualization. The analysis made in this report is focused on studying the software needs of different industries in Canada. It looks to answer questions such as what software packages are used significantly more in one industry over another, looking for what software packages are often together or if software packages tell us anything about which industry a company belongs to. No such analysis was found by our research, and it is important to note that the data used for this analysis consists of data that we scrapped ourselves.

Data Collection and Preprocessing

Dataset:

Pre-Generation:

The process of identifying the precise websites for data generation and developing or adapting the appropriate algorithms for them proved to be more time-consuming than initially anticipated. Consequently, the completion of a test dataset was only feasible following the full development of the scraping algorithms associated with the Dataset Generation Process. This necessitated simultaneous efforts to identify the leading Canadian Companies and the Most Utilized Software Packages during the construction of the comprehensive algorithm.

Generation Stage:
The most significant challenge in this state was Robot Handling Measures such as CloudFlare and Popups. These significantly slowed down the scraping operation and had to be handled manually through numerous workarounds. Some of these include:

- Using Python Packages that specialize in no interruption Web Scraping.

- Patching the Algorithm to adjust to the new packages since their functions varied significantly.

- Utilizing methods to make the Web Scraping loop less recognizable by Cloudflare
    - Giving random wait functions that make the website interactions feel more human-like instead of an algorithm assisted one.
    - Having a trigger and auto-closer for any pop-ups that might come up which would normally completely terminate the Scraping process.
    - Utilizing an Object Oriented Approach to make the overall action loops more human as opposed to a simple scraping algorithm.

- Along with this, adequate error handling was necessary for the project, this would mean whenever there the Data or, in this case, a specific Job Post could not be Scraped it would log it accordingly before being interrupted or crashing completely, the Object Oriented Nature of the Algorithm also helped with this case.

Post-Scraping:
Reducing the skewness of the data set was also a goal, and for this it was needed to either remove the bulk of the data on some industries (Banking and Leasing has been one of the most overused industries in the Dataset so close to 75% of the data tied to that industry had to be removed, making the overall spread of specific software a bit more natural). For this it was needed to tap into the rest of the Company List, thus making the Dataset more varied in the sense of industries and thus Software Packages.

<u>Final Notes on Dataset:</u>

As mentioned in the Presentation, the overall Data is special since the infrastructure to collect it had to be constructed from scratch and along with that, the Dataset was independent of any websites such as Kaggle, thus it can be made to have a more natural or have a less skewed specificity to it.

Another point, and in this point of view, a very valuable part of the Dataset is that it is extremely recent, wherein the Data that we have generated corresponds to the most recent Job Postings by the most successful Canadian Businesses, thus having the full capacity to properly show the current Software Trend (albeit limited to Canada and at most the North American Region). This makes it so there is another layer of exactness to the Dataset, making the efforts to predict and visualize have more accuracy, whereas with a set and obsolete Data, less precision would have been achieved in real life terms.

A note in terms of the limitations that were faced, there was a lack of exactness on finding the relevant Software Packages from the “Requirements” and “Responsibilities” parts of a Job Description from

the Scraping Algorithm. Our list, as it will be mentioned in the next section of "Data Wrangling" has been limited just as the amount of Companies and Industries we have scraped throughout the Project. Although both are very vast, maybe incredibly so, they are not infinite neither are they holistic for the specific question that was worked on, to provide an answer for it. But, the Project has succeeded and showed full capacity of providing a realistic approach and real observations on this very naturally generated Data.

Another important information on the subject matter was that, a sentimental research was going to be initiated based on the most utilized Software Packages, using a popular Software Review site, G2. This site was decided against, since actual ethical issues started arising such as the Extensive Cloudflare Protection that their servers had, which made it very difficult to automate the process of data generation, along with showing that the Website Providers really did not want other users to Scrape Data from their Website without their permission. Another issue was that the review data on hand would not merge well with the current dataset, since it was mostly old reviews if not extremely obsolete on older versions of Software Packages, which has started to deviate from the actual question that the research aspired to find solutions for.

Data Wrangling:

Once the data was collected from web scraping, we had a dataset with the following features: Company, Position, Regional Data, Company Size, Industry, Revenue, Founded, Job Description. For the analysis, we will be keeping Company, Industry and Job Description. Hence we have a dataset, for now, of 4528 rows and 3 columns. However, for the sake of the analysis, we decided to remove the missing values that appeared in Job Description and Industry, as there is no obvious way to impute those values. This unfortunately leaves us with now 2116 observations in our dataset.

However, we still do not have the most crucial information, that is the software packages mentioned in these job descriptions. In order to extract which software packages are present in the job descriptions, it was decided to use the natural language processing model SpaCy. In fact, NLP models are able to identify 'entities' in a text, which can be more than one word. For example, the SpaCy model is able to recognise something like 'Power BI' or 'Microsoft Excel' as two words that go together, something a basic method like string separation could not achieve. Therefore the following algorithm was implemented:

First, every job description is passed through the NLP model. The model is computationally expensive, so this algorithm unfortunately takes a few minutes to run (around 3 to 4 minutes). Then, every entity recognised by the NLP model is extracted and stored in a list. Afterwards, the next step is to look at all these entities and see if they correspond to software packages. Since there is no way for the SpaCy model to know what a software is, these elements need to be compared to some list of already known software packages. Therefore, it was necessary to create such a reference list of Software. Unfortunately, gathering every single software package used in industry is a futile task, since there are too many and there is no existing list of all known software. The approach taken was to use lists of software packages from Wikipedia, for example the list of Microsoft software packages or the list of programming languages. Twenty lists from Wikipedia, which were judged the most pertinent, were gathered together in one big list, which contains 2254 software packages. It is important to note that this choice is subjective, as it would be a daunting task to consider all existing lists of software packages from wikipedia. It was then possible to create a column for each software observed in the data that contains a 1 if the software is present in the job description, 0 otherwise. This

process of creating columns is not dependent on the dataset and is done automatically for the sake of reproducibility. Finally, a final column called ‘software’ is created, which contains the list of mentioned software stored in a string (for example, ‘Excel, Powerpoint,’). It was also decided to drop the observations where no software packages were observed, since the focus of the analysis was on the software used by companies of different industries. The final dataset therefore contains 1067 rows × 244 columns, where the columns are Company, Industry, Job Description, software and the rest are the 240 observed software packages (populated by 0’s and 1’s).

Finally, some difficulties were encountered with this approach. For example, very important software packages like ‘Excel’ were not recognised. This is not surprising, since as expected, only ‘Microsoft Excel’ was present in the reference list. Therefore we were presented with a trade-off. The first option consists of leaving the reference list as it was, which would only contain actual software packages but very important ones would be missed by the algorithm. On the other hand, we could transform multi-word package names into many entries, for example ‘Microsoft Excel’ becomes ‘Microsoft Excel’, ‘Microsoft’, ‘Excel’. However, this would cause many non-software entities to be recognised as software packages. It was decided to go with the second option, as it was judged better to have a list of software containing some noise (non-software entities) rather than missing very important software packages. After the mentioned procedure is completed, it is feasible to manually remove words that are not software packages, but we decided against it since this hinders reproducibility with new data. Moreover, while non-software words are present, they mostly consist of software or relevant subject related words, so they do contain some value.

The final datasets can be accessed here:
https://dataverse.harvard.edu/privateurl.xhtml?token=b45c1d27-f84f-407b-b948-beb97e12610c

Preliminary Data Exploration and Visualization

Some preliminary data visualization will be conducted here. In particular, the top 40 software packages used overall are depicted in Figure 1:

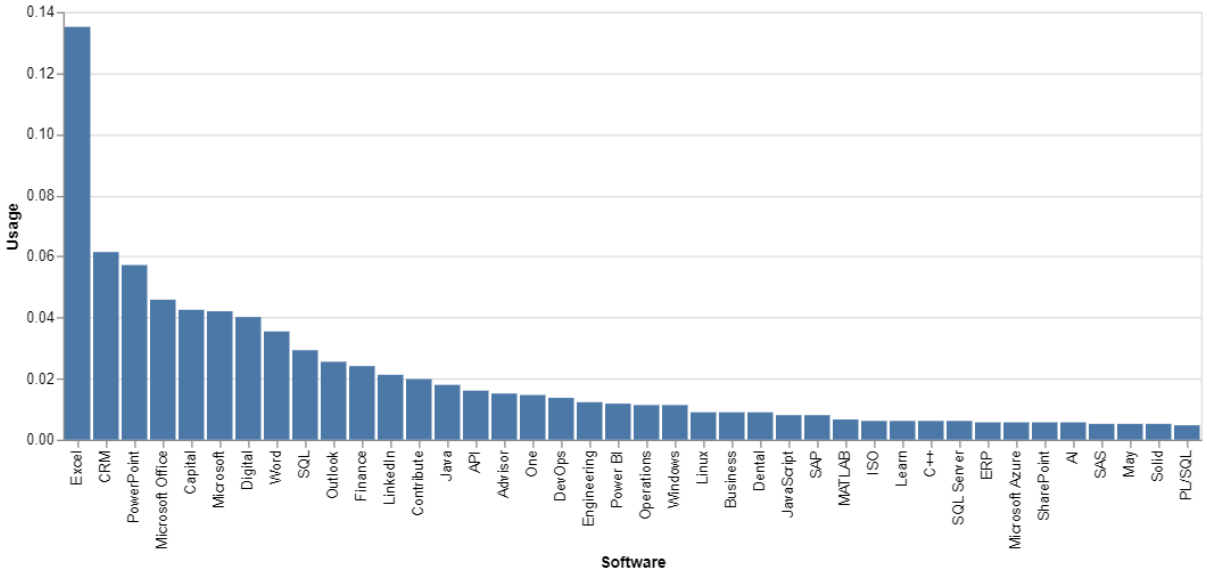

Figure 1: Bar plot of the 40 most popular software packages across all industries

Moreover, the top 25 industries in the dataset are visualized on Figure 2:

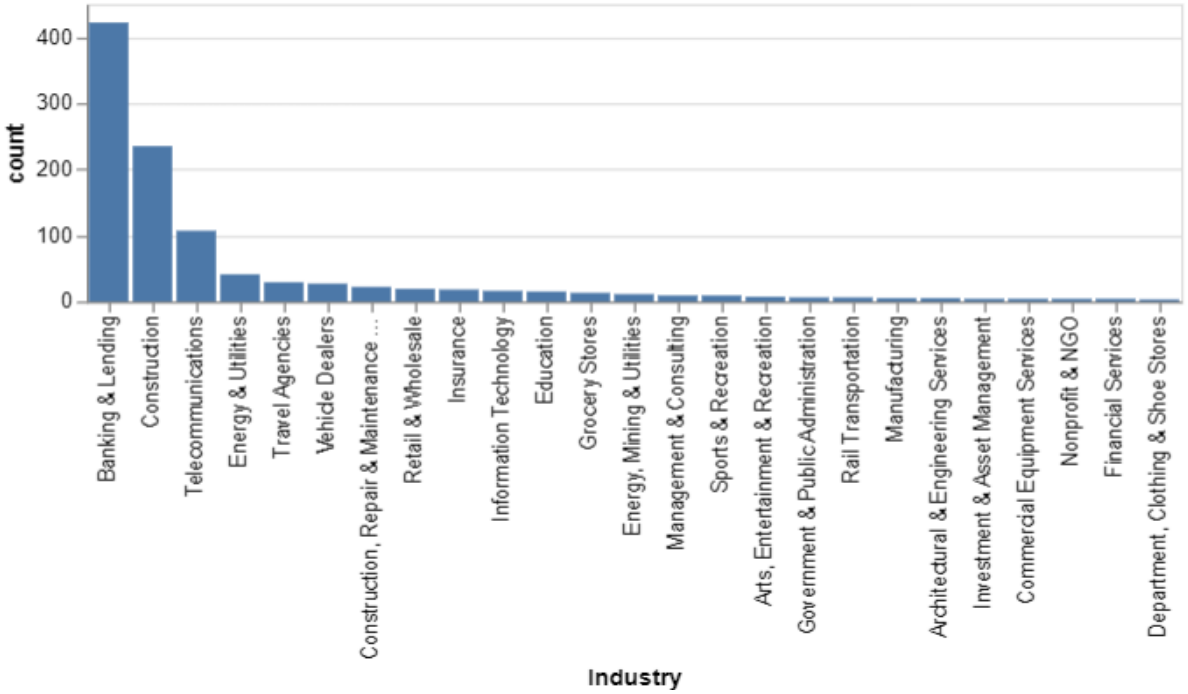

Figure 2: Bar plot of the 25 most popular industries in the dataset

It is unfortunately difficult to capture real life proportions of industries with manually scraped data. This graph sheds light on the imbalance of classes among our dataset, which must be considered while conducting the analysis and interpreting results of the different tests and models.

Finally, we can look at the top 5 most used software packages of the top 5 most popular industries, as seen on Figure 3:

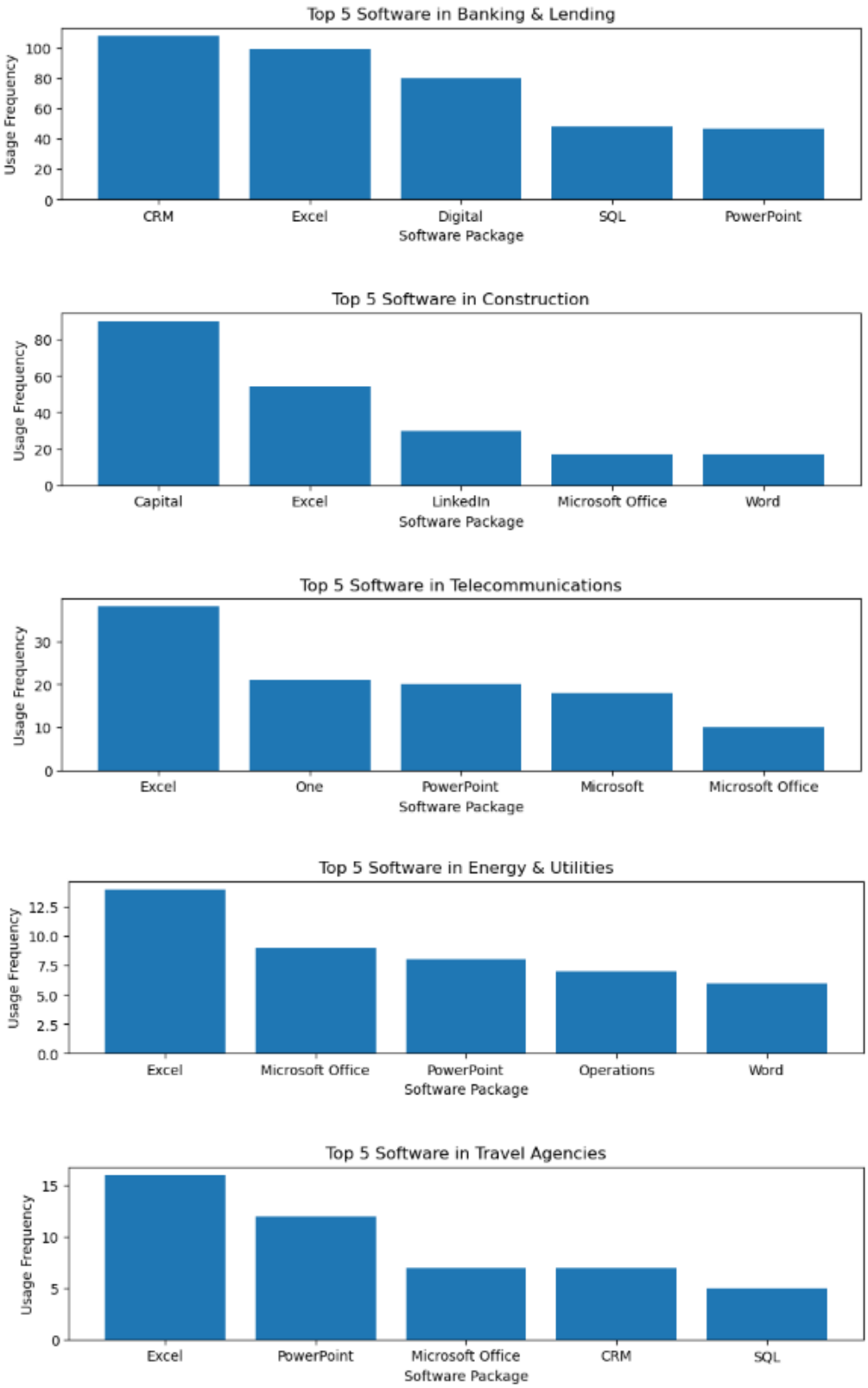

Figure 3: Histograms of the 5 most popular software packages among the leading 5 industries in the dataset

Most dominant software packages in the dataset are unsurprisingly sprinkled on these charts fairly evenly, with Excel being the strongest software leader.

Software Needs Across Industry Sectors

Statistical Tests : Various relationships between industries and software

Hypothesis testing holds significant power for applications in a task of this level of complexity. For instance, hypothesis testing was used to determine whether there were significant differences between usage of individual pieces/families of software across all industries in the dataset. Specifically, for every company in every industry, a one-way F test can be performed against all binary vectors representing usage of said software for each industry; each vector is filled with 1s for companies that use said software, and 0s for those that do not. As expected, *niche* pieces and families of software (i.e. software that are densely used by a small number of industries) yielded low p-values, for which the null hypothesis was rejected and there was significant difference between their usage. **Excel** (2.213e-08), **PowerPoint** (7.929e-06) and **matplotlib** (1.823e-66) were examples of such software. On the other hand, generic and *mainstream* software (i.e. software that are vastly used among all industries) yielded high p-values, for which the null hypothesis was accepted and there was very little difference between their usage across all industries. **Adobe** (0.946) and **Power BI** (0.808) were examples of such software. All in all, this specific hypothesis testing tool allows for better comprehension of the distinctiveness of a given piece or family of software.

Moreover, hypothesis testing was used to assess whether there existed a significant (linear) correlation between usage of two specific software across all industries. This test computes the binary columns of two selected software and evaluates their Pearson correlation coefficient (and their corresponding p-value). It was easily predicted that *mainstream* software would tend to exhibit strong correlation. For example, **Excel** and **PowerPoint** yielded a 0.51 Pearson correlation coefficient, with a corresponding p-value of 5.213e-72. Furthermore, testing a *niche* software and a *mainstream* one would yield poor correlation. For instance, **matplotlib** and **Microsoft** yielded a -0.009 Pearson correlation coefficient, with a corresponding p-value of 0.76. However, comparing two *niche* software that were naturally associated would exhibit significant correlation, such as **matplotlib** and **Python**, with a Pearson score of 0.33 and a corresponding p-value of 6.947e-29. All in all, this hypothesis testing tool allows for better understanding of pairwise associations between specific software, although prior knowledge of the software's usage context provides better interpretation of this test's results.

Classification : What software packages say about industry membership

In order to get insight on which industry uses which software packages, it was judged interesting to attempt to predict the industry of a company given some list of software. Being able to create such a predictive model which performs better than random guessing would be evidence to support the differences in software needs between different industries, and more specifically associate combinations of different software packages to different industries. Moreover, for individuals seeking employment opportunities, this analysis can help determine the industries for which their software skill set is most relevant.

The dataset used for the analysis consists of 240 binary software columns; they are populated with 1's if the software is mentioned in the corresponding job posting row, and 0's otherwise. This classification model aims to predict the industry given a list of software, i.e. P(industry | vector of software). The dataset makes it easy to compute P(industry), by computing the proportion of industries that appear in the dataset, and P(software | industry) for all software (columns) individually, as they follow a Bernoulli distribution (because it is binary data). The naive assumption of independence of predictors greatly simplifies the approximation of P(vector of software | industry). For these reasons, a Naive Bayes Classifier was judged to be a great candidate for the classification task. Hence the BernoulliNB model from *sklearn* was implemented.

The data was split into training and test sets with a ¾ and ¼ holdout split. The Naive Bayes Classifier had an accuracy score of 52.8%, versus the dummy model accuracy score of 39.3%. This confirms that the predictive model is better than just random chance and holds some significant predictive power. However, an accuracy of 52.8% is arguably not very practical in real-life situations. Considering multiple leading industry predictions constitutes a solution to this problem, as it produces more robust insight. For example, someone could look at the top 3 industries that the classifier outputs in its probability vector instead of just the first entry.

As a final remark, this classification algorithm is influenced by the unevenness of the industries present in the data. Therefore, the leading industries might be overrepresented by the algorithm. However, this does not hinder the fact that the algorithm does perform better than random chance, and looking at the top 3 or 5 predicted industries, given a list of software, allows to find less represented industries too.

Clustering : What software packages are found together

In order to investigate software needs across Canadian industries, attempting to find natural groupings among software could produce insightful results. What companies are similar to one another? What software packages are most used in those clusters? In the context of the analysis, graphs and word clouds were employed both to investigate the different software packages that appear in different interesting clusters and to some extent evaluate the clustering schemes. The latter could be considered as external validation of the clustering algorithms. Clustering will allow for further understanding of the software needs' landscape across the job market.

The data set contains the same information in two different ways : the list of software as a string, and the list of software as binary data, 0 and 1, across the columns representing software packages. To consider both input types, the binary data is first clustered with DBSCAN and spectral clustering algorithms using the Jaccard distance metric, which is the most appropriate for binary data. Then, the same algorithms will be tested using the string list of software with the cosine similarity, which is appropriate for text string comparison. SpaCy has a feature to convert text into a high-dimension word vector.

However, clustering in high-dimensional spaces is not always optimal, since the curse of dimensionality causes loss of locality and affects the ability to compare observations. Therefore, the use of dimension reduction techniques were employed, such as Principal Component Analysis (PCA). However, this technique's efficiency was limited, as a high number of PCs remained necessary to

explain acceptable data variance. Hence, t-SNE and UMAP were used in order to produce two-dimensional representations of the data.

From looking at the t-SNE and UMAP representations, significant clusters were identified and emphasized in the following analysis by using a DBSCAN algorithm. However, it is important to keep in mind that these dimension reduction techniques will conserve local structure adequately, but will have varying global structure depending on the seed that is run. This effect is likely to modify the clustering structure for different t-SNE algorithm runs. The use of DBSCAN, in this instance, was mostly done with the intent of easily labeling the clusters and then studying those that are stable and always appear almost identically in the many t-SNE runs that were tried. Hence, in the analysis, we included clusters which were differentiable, prominent, identifiable, and stable compared to others. Two examples would include the largest cluster which is easily identifiable among multiple runs on the algorithm, and a cluster containing all companies from the same sector, a considerable margin away from other clusters.

Results from clustering:

Before delving into the results, it's crucial to acknowledge that the majority of the data we obtained from companies via our scraping algorithm originates from the Banking, Finance, Construction, and Telecommunications industries. This skewness is an important factor to consider when interpreting our findings.

Our analysis revealed a significant reliance on Microsoft software, specifically **Excel**, **PowerPoint**, and **Word**, across all industries. However, when we examined industry-specific clusters, while Microsoft's prominence remained, the usage of other software increased.

In the banking industry, our findings indicate a strong emphasis on Customer Relationship Management (CRM) software, such as Monday, Salesforce, and Pipedrive, to monitor current and prospective customers. These companies also frequently utilize Business Intelligence (BI) software, including Domo, Wyn, and Yellowfin, to process business data and present it in a more digestible format. The mentioned word cloud can be seen on Figure 4:

Figure 4: Word cloud generated from Banking & Lending clusters

For clusters dominated by the construction industry, there is a substantial usage of Project Management software. This software aids companies in managing their human capital and schedules, allocating tasks to team members, and specifying the duration of each project phase. Examples of such software include Float, BigTime, and Kantata.

Figure 5: Word cloud generated from Construction clusters

These findings pertain to the two most prevalent industries in our dataset. While there are no other clusters composed solely of one industry, clusters with a higher concentration of telecommunications and travel agency companies show a noticeable increase in the use of image processing and graphic design software, such as Photoshop and Adobe.

Figure 6: Word cloud generated from Utilities, Telecommunications and Travel Agency clusters

As there are considerably less observations, these clusters have mixes of these less frequent industries. It can be seen that it relatively accurately depicts the top software used in each industry listed above in the Preliminary Data Exploration and Visualization section.

Conclusion

The analysis conducted in this project gave interesting insights on the question : *What are the popular software packages among Canadian companies across different industry sectors?*

Various basic data visualizations allow us to see which software packages are the most used all around, and which are the most used in the top industries in the dataset at our disposal. It also allows us to see the proportion at which different industries appear in the data, which has to be taken into consideration when interpreting results. Moreover, through statistical tests, it was determined that some selected software packages are used significantly more in some industries than others. This seems especially true for more niche software, while more general and popular software are across all industries.

Additionally, running a classifier showed evidence that software packages hold some power in industry prediction. However, while the accuracy is better than random chance, it is still low, although considering multiple leading classifier prediction renders useful in the general context of employment opportunity searching.

Finally, some clusters of job applications were interesting. The clustering algorithms used were able to identify some clusters with high purity, that is, which mostly consisted of one industry. Moreover, the clusters that appeared recurrently in fact presented interesting and more niche software packages. In other words, the clustering algorithm was able to catch groups of applications with main software different from the overall picture, some of these clusters consisting mainly/only of one industry. Those results confirm once again the relevance between industry and software use, while allowing someone to see which software is required in different industries and which software tends to go together in job requirements.

Further work on the topic could include more attempts at classification and clustering, and mostly more data to analyze. Moreover, similar visualization as the one maid mentioned in literature review could be done.

Ethical Concerns

Web Scraping with and without API Usage from the Website Provider:

- Since this is a research project aiming to be helpful for mainly students and get deeper understanding on Software Packages, the ethical concerns over Web Scraping or not as effective, we are not monetizing it neither are we selling this information
- Information scraped has been generated without using any account on these websites (Namely G2.com and Indeed.com), thus complying with their scraping limitations and policies.

Contributions

All team members contributed well in the project with each member focused on certain aspects:

- Kaan: Mainly focused on web scraping to gather datasets to be analyzed. Also worked on data cleaning to produce functional, less biased data.
- Jacob: Heavy focus on data visualization, generating plots and images to help better understand the results of our analysis. Also helped analyze the results of clustering algorithms to specify interesting clusters to visualize.
- Thomas: Heavy focus on statistical and machine learning insights. Also contributed to data wrangling steps.
- Isaac: Focused on hypothesis testing, clustering, data cleaning and additional deep dive into data wrangling on the constructed dataset

References

- Mattingly, William. *Introduction to spaCy 3*, 2021 (1st ed.). https://spacy.pythonhumanities.com./intro.html.
- Wu, Julia. (2021, April 2). *Using The Cosine Similarity and DBSCAN to Get Clusters from The Housing Data Set in Python*. Medium. https://medium.com/web-mining-is688-spring-2021/assignment-4-5236a8c96167
- Banu, Sameera. (September 25). *Multilingual Language Models in Natural Language Processing (NLP) with Python*. Medium. https://medium.com/@mail4sameera/multilingual-language-models-in-natural-language-processing-nlp-with-python-9a6d1fda4adc
- Oskolkov, Nikolay. (2019, July 23). *How to Cluster in High Dimensions: Automated Way to Detect the Number of Clusters. Medium*. https://towardsdatascience.com/how-to-cluster-in-high-dimensions-4ef693bacc6
- Turck, Matt. (2023, February 21) *The 2023 MAD (Machine Learning, Artificial Intelligence & Data) Landscape*. FirstMark. https://mad.firstmark.com/

Lists of software packages:

- https://en.wikipedia.org/wiki/List_of_programming_languages
- https://en.wikipedia.org/wiki/List_of_information_graphics_software
- https://en.wikipedia.org/wiki/List_of_Apache–MySQL–PHP_packages
- https://en.wikipedia.org/wiki/List_of_Adobe_software
- https://en.wikipedia.org/wiki/List_of_spreadsheet_software
- https://en.wikipedia.org/wiki/List_of_spatial_analysis_software
- https://en.wikipedia.org/wiki/List_of_SAP_products
- https://en.wikipedia.org/wiki/List_of_presentation_programs
- https://en.wikipedia.org/wiki/List_of_PHP_editors
- https://en.wikipedia.org/wiki/List_of_personal_finance_software
- https://en.wikipedia.org/wiki/List_of_PDF_software

- https://en.wikipedia.org/wiki/List_of_free_and_open-source_software_packages
- https://en.wikipedia.org/wiki/List_of_statistical_software
- https://en.wikipedia.org/wiki/List_of_open-source_health_software
- https://en.wikipedia.org/wiki/List_of_charting_software
- https://en.wikipedia.org/wiki/List_of_HTML_editors
- https://en.wikipedia.org/wiki/List_of_Mac_software
- https://en.wikipedia.org/wiki/List_of_Microsoft_software
- https://en.wikipedia.org/wiki/List_of_neuroimaging_software
- https://en.wikipedia.org/wiki/List_of_numerical-analysis_software