School of Computing

BT4013 Capstone Project Final Report

Exploring Portfolio Decarbonisation Using AI

Group 08 - Natwest Markets

Aw Xin Min (A0190383A)

Chia Ai Fen (A0188215B)

Lee Jun Hui Sean (A0189893B)

Lim Jermaine (A0187866E)

Wong Jao Kuean (A0200290Y)

Github Repository: https://github.com/jaokuean/team08-capstone

Complete Project Code (including data files and saved models):

https://drive.google.com/drive/folders/1ce9L5dHZXrWLzpRNf3iq6cdK5KU_QF0a?usp=sharing

Dashboard Demo: https://youtu.be/VApBSNr_FFg

Executive Summary

Climate change is a major threat to the world. A key challenge to address in today's world is to manage climate-related risks and support the transition to a sustainable low carbon economy. The finance sector plays a key role in the monitoring, measuring and reporting of investment flows and the development of financial services activity to reduce emissions. Financial institutions can play a part in the decarbonisation of the global economy by engaging with their portfolio companies to support sustainable investing and divest from high emitting industries such as thermal coal production. With the commitment to reduce carbon emissions, we see more companies reporting their climate risks, metrics, and targets. Financial institutions including asset managers and pension funds have also listed strategies and targets in their efforts to decarbonise their investment portfolios.

However, decarbonisation-related information in these reports are often presented in unstructured text format, buried in hundreds of pages of financial documents that require intensive human labour to manually read through. In the past decade, disclosing of climate-related risks and targets has remained predominantly voluntary, resulting in the lack of a standardized reporting format with regards to the subjects covered and quantifiable metrics used.

In this project, we have partnered with NatWest Markets to propose a solution to automate the retrieval and analysis of decarbonisation-related information in companies' reports. NatWest Markets is the investment banking arm of NatWest Group, the largest business and commercial bank in the UK. It actively supports customers and investors in sustainable investing.

In recent years, researchers have explored advancements in Natural Language Processing (NLP) to automatically analyse sustainability disclosures in reports. Different layouts that companies present their metrics in pose a significant challenge in applying text mining approaches alone. As such, our group proposes a hybrid machine learning approach involving the use of text mining techniques together with image processing techniques to extract all information related to decarbonisation from companies' reports. We have also built a dashboard as our end product to capture the decarbonisation metrics and textual information in a visually appealing manner.

Table of Contents

1 Introduction

1.1 Description of the Problem

Environmental, Social, and Governance (ESG) investing is gaining traction in the financial sector and has increasingly become an important part of the investment process. For investors and financial institutions, ESG integration is seen as a means to improve the risk–return characteristics of a portfolio as ESG factors are believed to influence the operations of a firm and its ability to generate profits in the long run. Increasingly, we see investors incorporating ESG data alongside the conventional financial data to identify risks and potential opportunities. In fact, Bloomberg Intelligence (Bloomberg Intelligence, 2021) has reported that global ESG assets are on track to exceed $53 trillion by 2025, representing more than a third of the $140.5 trillion in projected total assets under management. Among the three pillars, E which represents the environmental criteria is emerging as a leader in attracting investments, with related investments growing at 58% since 2018 (Nason, 2020) and showing no sign of slowing down.

However, problems still remain around data accessibility and quality. In the area of ESG investing, there exists a lack of mandatory and consistent reporting of ESG performance data, which makes it challenging for investors to obtain curated data for analysis. In particular, corporate disclosure of ESG metrics are often scattered across long reports in unstructured formats, making data retrieval labour intensive and prone to human errors. Therefore, this project aims to aid our partnered financial institution, NatWest Markets, in improving its efficiency and information quality in ESG-related workflows so that they can better support clients in their transition to achieving broader environmental and societal goals.

1.2 Proposed Solution

For our project, we partnered with NatWest Markets, the investment banking arm of NatWest Group. It is the largest business and commercial bank in the United Kingdom (UK). NatWest Markets offers financing and risk management solutions for corporate and institutional clients and is committed to acting sustainably and responsibly, actively supporting customers in their transition to achieving environmental goals and working with investors to develop holistic sustainability strategies. Therefore, it is important for NatWest Markets to monitor how Financial Institutions, in their portfolio, manage ESG-related risks. For this project, we will be focusing specifically on the “E” aspect of ESG, tracking how FIs are progressing towards their decarbonisation targets and managing climate-related risks.

As aforementioned, to deal with the current challenge of investors having to manually analyze sustainability reports to source for climate-related information, we aim to employ machine learning techniques to automate this manual and tedious process. With increased interest in ESG in recent years, there has been more research and work done on analysing sustainability reports. However, existing research has mainly employed Natural Language Processing (NLP) techniques to conduct sentiment analysis as well as topic modelling on sustainability reports. To the best of our knowledge, limited research and work has been done in the area of information extraction involving the extraction of sentences, tables and charts from such reports. This could possibly be due to the heterogeneity and lack of standardisation in the structure of sustainability reports. Not only do different companies present their information differently, but they also showcase all this information in different layouts and formats, which can be in the form of text,

tables, charts and images. As a result, information extraction from this kind of reports is a challenging task. Therefore, we present a holistic and customised approach that aims to extract information related to decarbonisation in all forms. We leverage advancements in NLP techniques for text processing and Computer Vision (CV) techniques for image processing, to not only identify decarbonisation-related information in textual format, but also key metrics presented in tabular and graphical formats. The main objective of this project is to employ machine learning techniques to create a tool that allows for more efficient and structured analysis of sustainability reports to extract decarbonisation-related information. We hope that this project can help to significantly reduce the amount of manual labour required to sieve out relevant information buried in these reports. Our end product is an interactive dashboard that will present all decarbonisation related information and metrics extracted by our pipeline, in a clear, comprehensive and interactive manner.

2 Data

2.1 Data Collection

Our project sponsor, Mr. Gupta, has kindly provided us with a list of companies across 4 main types of FIs to focus on, namely Asian Banks, Asset Managers, Insurance, and Pension Funds. Examples of FIs include DBS Bank, BlackRock, Allianz Global Investors and Ping An Insurance. With the list of FIs, we manually collected a total of 474 publicly available sustainability PDF report URLs across 167 firms. These reports were sourced from individual corporate websites and collated across the years, depending on the year in which the company started reporting its sustainability-related disclosures.

Using the PDF report URLs collected, we used the Python *requests* package as well as an open source PDF text parser, *pdfminer3*, to extract textual information from the PDF reports. Among the 474 report URLs collected, we were able to extract information from 230 reports from 123 firms for subsequent downstream tasks, as the other reports were either not in English or were unable to be parsed by the *pdfminer3*.

2.2 Database Schema

The final database for this project consists of a JSON file, a pickle file and folders containing extracted chart images, table images and word cloud images to be displayed on the dashboard - all of which are information extracted by our pipeline from all 230 reports. Due to the lack of access to a hosted database, data was hosted and stored locally. JSON was chosen to be the main data storage type due to the unstructured nature of our extracted data as well as its scalability. Pickle was chosen as it could support binary serialization format for any arbitrary Python object, making it useful for storing dataframes of extracted tables.

The JSON file contains information from all reports with each report having the fields: company name, year of report, report URL, details of relevant decarbonisation related text such as its carbon category, details of chart images and lastly table images extracted from our pipeline. The detailed description of the JSON schema can be found in Appendix A. Meanwhile, the pickle file contains the text-based dataframe tables extracted from all reports with each record having the fields: company name, year of report, report URL and dataframes of relevant decarbonisation tables. The primary key of both the JSON and pickle files is the report URL. When a new PDF

report URL is inputted by the user, the new report and the relevant extracted information will be appended to the existing database and stored for future use.

2.3 Data Labelling

Preprocessed text from each report was put through our preliminary page and secondary sentence filtering pipeline to obtain a set of sentences that were more likely to be relevant to decarbonisation. A total of 8470 sentences were obtained from all 231 reports after these steps. More details on these steps can be found in Section 4 and 5.1. For the prediction of highly relevant sentences and carbon categories of these sentences, we used supervised machine learning methods. Thus, we had to derive labels for supervised learning. We randomly selected 5000 sentences and each sentence was labelled with a relevance score - relevant (1) and not relevant (0). The final labelled dataset was highly imbalanced, with class 1 being the minority class and class 0 being the majority class. Class distribution was about 10% for class 1 and 90% for class 0. Thus, to deal with this imbalance, oversampling was later conducted on the underrepresented class (1) using various techniques. Next, for each sentence labelled as relevant (1), we further classified and labelled them with a carbon class - carbon/greenhouse gas emissions (0), energy consumption/renewables (1), waste (2), sustainable financing/investing (3) & others (4). We chose these 5 categories for our carbon class as these were the most frequently observed decarbonisation categories during the labelling process. Class distribution for carbon classes is as follows : 30%, 18%, 5%, 29% and 18% for classes 0, 1, 2, 3 and 4 respectively.

3 Pipeline

Our entire information extraction pipeline is shown in Fig 1. Details of each step will be explained in subsequent sections.

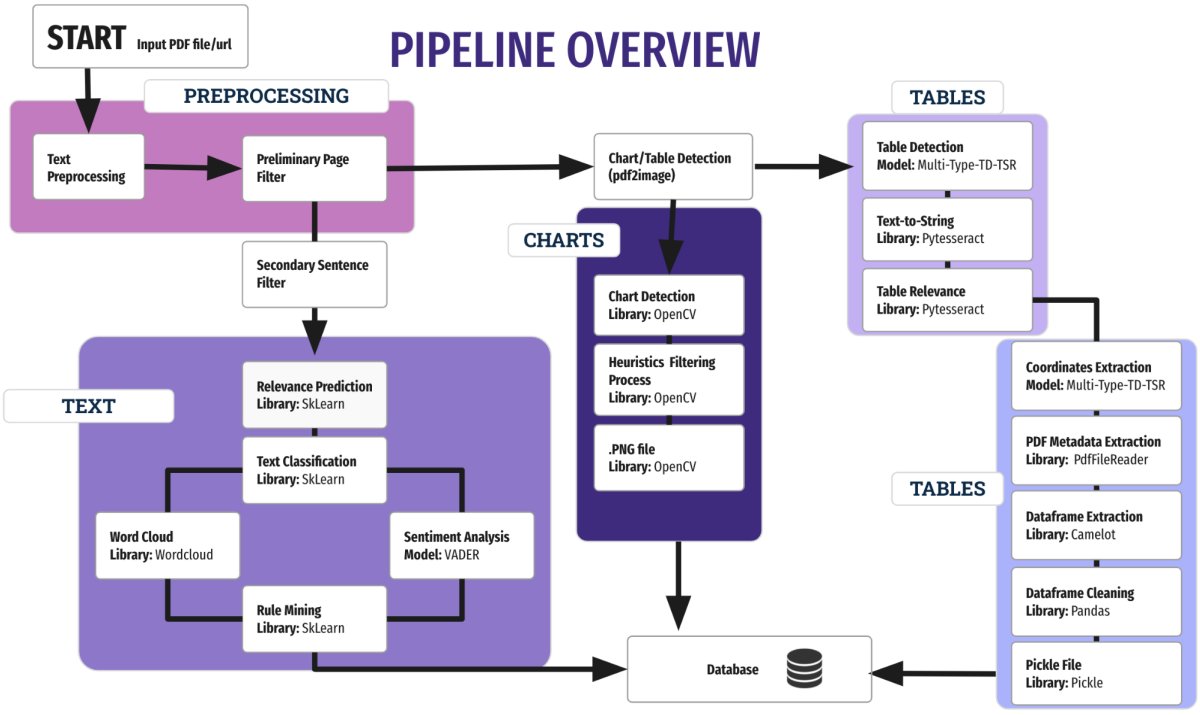


Fig 1. Our proposed information extraction pipeline

4 PDF Text Preprocessing & Preliminary Page Filtering

As the raw text parsed using *pdfminer3* was highly unstructured, text preprocessing had to be conducted to obtain structured text that we could work with. The following steps were taken:

1. Remove header number.
2. Remove trailing spaces.
3. Remove emails and URLs mentioned.
4. Concatenate words to form proper sentences.
5. Concatenate sentences to form proper paragraphs.
6. Lemmatize and lowercase the text in each report.
7. Split each report and store them by pages.
8. Split each page and store them by sentences.

Next, a preliminary page filtering step was carried out to filter each report for pages that contained information related to decarbonisation. This step was necessary as sustainability reports usually contain all environment, social and governance (ESG) related information. However not all of this information was relevant for the scope of our project as we only want to extract the environmental, decarbonisation related information. Therefore, this page filtering step was done so that time and computational resources can be saved by processing only relevant pages. This step was carried out using a simple word filter, where only pages that contained words defined in a list of decarbonisation related words and fulfilled certain filtering criteria were retained. This list of words was constructed by research as well as manually reading through sustainability reports. Examples of words include: greenhouse gas emissions, carbon footprint, decarbonisation and sustainable financing etc. The full list of words and filtering criteria can be found in Appendix B. Words in the list were lowercased and lemmatized before comparing to the text in the report to filter them.

With this step, we were able to filter away an average of about 80% of irrelevant pages from each report and the remaining 20% were used for the subsequent text, chart and table extractions.

5 Text Extraction

For text extraction, there are 4 main steps: (1) Relevance Prediction; (2) Text Classification; (3) Rule Mining; and (4) Sentiment Analysis. The text extraction pipeline can be found in Appendix C.

5.1 Relevance Prediction

For the extraction of decarbonisation related text information, we aim to obtain highly relevant sentences from each report in 2 main steps. First, using the pages of the report obtained after the preliminary page filtering step, a secondary sentence filter was applied to all sentences in those pages. This was essential to minimize the noise in the dataset as not all sentences from the filtered pages contained relevant sentences. However, due to the unsupervised nature of the first step, the set of sentences obtained is still likely to contain a significant amount of irrelevant sentences. Thus, to obtain the final set of relevant sentences, supervised machine learning techniques were applied on the sentences obtained in the first step. The use of supervised

machine learning techniques allows us to identify underlying patterns and relationships that exist within the corpus, which allows us to obtain the final set of highly relevant sentences.

5.1.1 Secondary Sentence Filter: BERT Embeddings

For this step, we utilise an NLP model, Bidirectional Encoder Representations from Transformers (BERT). BERT is a deep learning language representation model developed by Google AI Language researchers and pretrained on an enormous amount of publicly available annotated text (Devlin et al., 2018). It is derived from the Transformer model architecture, and is an attention mechanism that learns contextual relationships between words in a sentence. One of its use cases is text vectorisation. It can be used to encode words or sentences for many downstream NLP tasks, such as sentiment analysis and text classification. For text vectorisation, BERT offers an advantage over context-free models like word2vec as it is able to generate contextualised text vectors. Unlike context-free models such as word2vec or GloVe, which generate a single word embedding representation for each word in the vocabulary, contextual models like BERT can generate a representation of each word that is based on the other words in the sentence by capturing relationships between words in a bidirectional way. Thus, we aim to make use of BERT's abilities to understand the context of words and sentences to help us filter for relevant sentences, by comparing all sentences with a list of relevant sentences.

We first manually obtain and collate a list of sentences and phrases that we deem relevant from sustainability reports and would want displayed on our final dashboard. An example would be a sentence like "*The equity portfolio's carbon intensity was 9 percent below that of the benchmark index.*". All the sentences used can be found in Appendix D. Then, we encode these sentences using BERT to obtain their sentence embeddings. We also encode all sentences from the filtered pages using BERT. We then proceed to compare each sentence vector from the filtered pages with each sentence vector of the relevant sentences using cosine similarity. Cosine similarity calculates the similarity of 2 vectors by measuring the cosine of the angle between the 2 vectors. Cosine similarity will be 1 for 2 identical vectors and 0 for 2 vectors oriented at 90° to each other. Thus for 2 highly similar sentences, the cosine similarity should be close to 1. Using this property, we determine a threshold that would indicate a high enough cosine similarity, which implies a high enough probability of a sentence being relevant, to retain relevant sentences and filter out irrelevant sentences. To determine this threshold, the cosine similarities between the list of sentences collated and the sentences from the filtered pages were calculated. The statistics for the cosine similarities are shown in Fig 2 & 3 and the 75th percentile value of 0.7357 was chosen as the threshold as it would allow for sufficient sentences to be retained for subsequent supervised machine learning. Cosine similarity was chosen to compare text similarity as it performs the best when used for BERT embeddings[1] comparison. With this step, we reduced the number of sentences from 18465 to 8470 which will be used for our subsequent supervised learning.

[1] https://medium.com/@adriensieg/text-similarities-da019229c894

	cos_similarity
count	249264.000000
mean	0.681896
std	0.079136
min	0.206599
25%	0.641717
50%	0.693079
75%	0.735703
max	1.000000

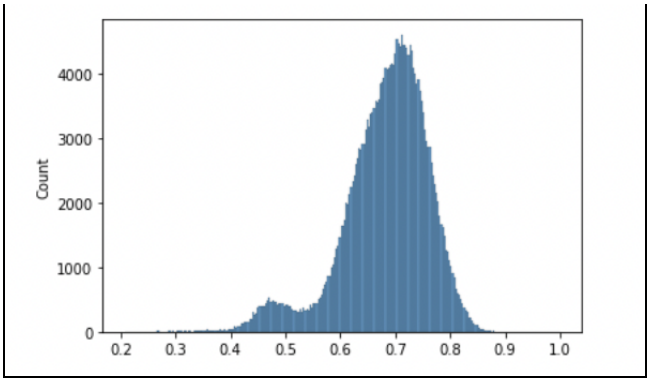


Fig 2&3. Statistics and Distribution of Cosine Similarities

As BERT embeddings are not affected by capitalisation and should not be stemmed or lemmatized, none of these text preprocessing steps were carried out. However since BERT is affected by punctuations, all sentences were only preprocessed by removing punctuations, except for "%,$,&" as highly relevant sentences usually contain these, before encoding the sentences and calculating cosine similarities. BERT-as-service[2], an API version of BERT, was chosen to encode sentences into fixed length vectors over the traditional BERT model for two reasons. Firstly, BERT-as-service is able to directly generate embeddings at a sentence level as compared to a word level for BERT. This means that embeddings do not have to be aggregated up from words to sentences. Secondly, testing on our end has also shown that BERT-as-service can generate embeddings at 10 times the speed of traditional BERT. This enables our eventual dashboard to be able to process new PDFs at faster speeds without sacrificing performance.

5.1.2 Supervised Machine Learning

To support our supervised machine learning models, we first split the labelled dataset into train, validation and test sets of 60%, 20% and 20% respectively. Distribution of classes can be found in Appendix E. We then vectorised the sentences to convert them into suitable inputs for our models. We first performed a grid search on the training dataset to obtain the best hyperparameters for both the text vectorizer and model, before selecting the combination of parameters with the best performance on the validation data to prevent overfitting. The tuning for the parameters of the text vectorizer, model, oversampling method, and text cleaning were done in tandem as different models may be more suited to different processing approaches.

For the metric chosen to evaluate model performance, weighted F1 Class 1 score was chosen for 2 reasons. Firstly, F1 score is used as it helps to balance the trade offs that exist between precision and recall. Secondly, for the purpose of this project, we deemed class 1 predictions to be more important than class 0 predictions as we want to capture as many class 1 (relevant) sentences as possible. Thus, we placed more importance on the Weighted F1 (Class 1) as compared to the macro F1 and Weighted F1 (Class 0) scores.

[2] See github code implementation of bert-as-service

5.1.2.1 Text-Vectorization

We tested three approaches, Bag-of-Words (BoW), Term Frequency - Inverse Document Frequency (TFIDF) and BERT embeddings, to which the last approach was mentioned in the previous section.

BoW, which is quick to run and generally effective, builds a vocabulary from the train corpus and counts the number of occurrences of each word in a document (sentence). Meanwhile, the TFIDF method captures the importance of a word to a document in the corpus, proportional to the number of times a word appears in the document and inversely proportional to the number of documents containing the word. While both methods do not take into account the different lengths of documents, we felt that this limitation was not very significant in our problem since our derived sentences are usually of approximately the same length. For these 2 vectorizers, we performed hyperparameter tuning on the parameters found in Table 1.

Parameter	Definition	Purpose
min_df	Floor for number of word occurrence within sentences	Tuned to remove words that appears in too few sentences which may be outliers
max_df	Ceiling for number of word occurrence within sentences	Remove words that appear in too many sentences, which can sieve out corpus specific stop words
ngram_range	Lower and upper boundary for different n-grams vectorization	Incorporates word order information, which is important for negation

Table 1: Details of parameters tuned for BoW and TFIDF Vectorizers

5.1.2.2 Oversampling

Due to the class imbalance that exists in our labelled dataset, as mentioned in Section 2.3, we tested 2 oversampling techniques, Random Over Sampling (ROS) and Synthetic Minority Oversampling Technique (SMOTE) before training and tuning our models. This is to ensure that our models will have sufficient predicting power even on the minority class (relevant sentences) when used in the production pipeline. ROS is an oversampling method where additional data points are added to the minority class by selecting data points from the minority class at random and with replacement.[3] Meanwhile, SMOTE is an oversampling technique where new minority synthetic data points are created by the following algorithm: First, a random minority data point is selected. Second, k (default is 5 in imbalanced-learn implementation) nearest minority neighbours are derived through the use of distance measures, the most common one being the euclidean distance. Third, one of these k points are selected. Finally, a new synthetic data point is created through a convex combination of the data point chosen in the first and third step. This process is repeated until the class imbalance is corrected. SMOTE differs from ROS as it provides new information through the creation of synthetic data points in the feature space

[3] See official imbalance learning documentation on RandomOverSampler class

(Chawla et. al, 2002). Across our models, we have observed that SMOTE produced better results on the validation data as compared to ROS; which reinforces the claims that the inventors of the algorithm have made in their paper.

5.1.2.3 Sentence Cleaning

We also attempted two types of input sentences, raw and clean. Raw sentences refer to sentences in their original form without any processing done. For clean sentences, we removed stopwords, punctuations, numbers, and also lemmatized all the words within them. This could potentially help to reduce any commonly occurring features that serve no purpose other than being noisy features. This difference in input processing was explored only for the BoW and TFIDF vectorizer approaches. For BERT embeddings, the preprocessing step mentioned in Section 5.1.1 was used to clean sentences.

5.1.2.4 Machine Learning Models

For this section, we will discuss the 4 supervised machine learning models explored, namely Logistic Regression, Multinomial Naive Bayes, Support Vector Machine and Random Forest. All text vectorizers were attempted for Logistic Regression and Support Vector Machine. However BERT embeddings were not attempted on Multinomial Naive Bayes and Random Forest as the embeddings generated were not compatible with both models. A description of the workings of each model and the parameters that we tuned can be found below.

Logistic Regression (LR)

The first model we attempted is the Logistic Regression classifier, which was selected as a candidate model due to its simplicity and ease of interpretation. The LR model is essentially a regression analysis method that is suitable for binary classification problems such as in our case. The model is fitted using log-loss, and transforms the output of a linear regression model using a sigmoid function. This then outputs the probability of a sentence belonging to the positive class, which ranges between 0 and 1.

The parameters tuned during hyperparameter tuning can be found in Table 2.

Parameter	Definition	Purpose
C	Gives the inverse of regularisation strength	Tuned so as to control for overfitting by introducing a penalty term in the log-loss function penalising large model coefficients[4]
penalty	Defines the regularisation method used	
solver	Denotes the algorithm that is used to solve the optimization problem	As our data set was not that large, the lbfgs and newton-cg solvers were tested as they tend to perform better despite being more computationally expensive.

[4] https://towardsdatascience.com/regularization-in-machine-learning-76441ddcf99a

class_weight	Defines if the weights associated with each class are to be inversely proportional to the class frequencies in the model input	Tuned so that we can test if assigning different importances to each class could result in a better overall model.

Table 2: Details of parameters tuned for LR

Multinomial Naive Bayes (NB)

The second method we attempted is the Multinomial Naive Bayes classifier. The Naive Bayes is computationally simple when compared to the other tested models. Despite this simplicity, it is used rather extensively across Natural Language Processing applications and has been shown to perform quite well. In terms of the model's workings, it calculates the posterior probability of a sentence being in an output class by leveraging Bayes' Theorem and assuming that each token is independent from the others. This is done by multiplying the prior probability of a sentence being in a class with the likelihood of observing each sentence token given the particular class. In our context, this means that two probabilities will be computed, one for class relevant and the other for class non-relevant. For hyperparameter tuning, the parameters tuned can be found in Table 3.

Parameter	Definition	Purpose
alpha	Gives the additive smoothing parameter	Tuned to reduce overfitting

Table 3: Details of parameters tuned for NB

Support Vector Machine (SVM)

SVM is the third model we chose to test as it provides a non-linear modelling perspective which performs well in a high dimensional feature space[5] and is appropriate for our problem due to our relatively large text corpus. The model works by searching for the most optimal hyperplane that creates the largest margin between data points from the two classes. Different kernels can also be applied to transform the input data into a higher dimension for improved classification results. Specific details of the hyperparameters we tuned can be found in Table 4.

Parameter	Definition	Purpose
C	Similar to the definitions and purpose defined in Table 1 for LR	
penalty		
kernel	Function that is being used to transform the input data to a higher dimension	Mostly used when it is difficult to find an optimal hyperplane in a lower (e.g. linear) dimension

[5] https://scikit-learn.org/stable/modules/svm.html

gamma	Refers to whether we chose to scale the kernel coefficient depending on the variance of the input data	Gives the extent of influence that we allow a single training data point to have, with low gamma value meaning 'far' and high gamma value meaning 'close'.

Table 4: Details of parameters tuned for SVM

Random Forest (RF)

The fourth model that we chose to test is Random Forest. This is because the Random Forest has been shown to be able to work well with non-linear data and is robust to outliers. By combining the prediction of various base decision tree classifiers, the model also reduces the chance of overfitting. Within each node in a decision tree classifier, data points are split according to different decision criteria that aim to maximise the information gain at each split. The size of the tree and metric used in the information gain calculation is tuned during hyperparameter selection (Table 5).

Parameter	Definition	Purpose
min_samples_split	Minimum number of samples required before a node is split	Tuned to reduce overfitting through limiting the size of each decision tree classifier that is grown
max_features	Maximum number of features to consider when making a split	
min_samples_leaf	Minimum number of samples required in each leaf node	
criterion	Function used to determine the quality of a node split	Tuned as different criterion may fit our corpus differently, allowing for potentially better results
class_weight	Similar to the definitions and purpose defined in Table 1 for LR	

Table 5: Details of parameters tuned for RF

5.1.3 Model Ensemble

For our final predictions, we attempted various ensemble methods in an attempt to improve overall model performance as ensembles tend to achieve better predictive performance (lower bias) than a single predictive model. They also improve robustness (reduce variance) of the average performance of a model. As we utilised heterogenous machine learning models for relevance predictions, two ensemble methods, namely stacking and voting were used to combine the predictions of our models. The architecture of our ensembles involved 4 and 5 base models for voting and stacking respectively, all of which are optimal stand-alone models (highlighted in purple in Appendix F) that have undergone hyperparameter tuning as mentioned previously. For

our ensembles, we generated 5 folds from our training data, and used the output predictions of our base models on these folds as inputs for the ensembles. Specific details of each ensemble model can be found as follows.

Stacking

For our stacker, we fitted the training predictions from the base model on a meta-model. The meta-model, chosen based on model performance, then learns how to best combine such predictions.

Base-Models: Logistic Regression (TFIDF & BERT), Naive Bayes (BoW), SVM (TFIDF) and Random Forest (TFIDF). Each model's predicted probability of a sentence being relevant and not relevant were extracted and used as input for the meta model.

Meta-Model: Two simple models, Logistic Regression and SVM, were candidate choices for the final meta model. Due to the low dimensionality of the meta-model input, simple models would likely perform well and be more interpretable. Hyperparameter tuning and model training was conducted for each of the meta models and the model that achieved the best performance (SVM) on the validation dataset was chosen as the final meta-model.

Voting Ensemble

The second ensemble that we tried was the voting ensemble. Two variations were tried, namely the soft voting ensemble and the hard voting ensemble. The importance of each base model's predictions were kept equal for this part as we wanted to fit the voting ensemble in a different feature space than the stacker, where the latter already employs a meta model to optimise the weights of the base model predictions.

Soft-Voting: The 2 predicted probabilities of a sentence belonging to the relevant and non-relevant class from the base models were summed. Afterwards, the voting ensemble outputs a prediction for whichever class has a higher total summed probability.

Hard-Voting: In this method, also known as majority voting, we take the mode of the base model class predictions as the final prediction.

5.1.4 Model Results

The full table of results are in Appendix F. We observe that the stacked model was able to obtain the best performance across all metrics except the Weighted F1 (Class 0). However, since we place more emphasis on predicting Class 1 sentences and the stacked model does the best for Weighted F1 (Class 1), it was thus chosen as our final model.

5.2 Carbon Class Text Classification

We sought to extract further value that would be valuable for investors from the relevant sentences predicted by classifying each sentence into one of the 5 broad decarbonisation categories that were mentioned in Section 2.3. This classifier would also support text filtering in the dashboard.

Similar to relevance predictions, we split the labelled dataset into train, validation and test sets of ratio 60:20:20. The vectorisation methods and machine learning models were chosen to cater to the small dataset we had for this task, as mentioned in Section 2.3. During the hyperparameter tuning phase, we used the weighted average F1 score to evaluate the various model parameter combinations. This metric was chosen as we wanted to balance recall and precision, account for size of the various classes within our data set while not prioritising any class for this modelling portion.

5.2.1 Vectorization Methods

For this classification, we tested two approaches, BoW and TFIDF. Details of these vectorizers can be found in Section 5.1.2.1. We did not attempt BERT embeddings for this task as BERT is a complex model and may not be suitable for using on such a small dataset. Thus, only simpler vectorisation methods were used. In addition, we did not want to further increase the time spent on generating the outputs from the entire pipeline. This is especially as BERT embeddings do take a significantly longer time to generate.

5.2.2 Machine Learning Models

For the text classification problem, we similarly employed the help of the 4 models that can be found in Section 5.1.2.4 (LR, NB, SVM and RF). Details of the parameters and the models can be found in the aforementioned section. Additionally, we also attempted CatBoost to determine if gradient boosting on decision trees can help to improve the results as the base results were less promising than in the relevance model. Details of the model and the hyperparameters we tuned can be found below.

CatBoost (CB)

CatBoost is a boosting algorithm, which means that it is a weighted ensemble method as compared to the bagging method in Random Forest. This means that as the model is being trained, a number of decision trees will be built in tandem, with each following tree being built with a lower loss when compared to the previous tree. This is done through updating the weights to pay more attention to misclassified data points. The hyperparameters that we experimented with can be found in Table 6.

Parameter	Definition	Purpose
depth	Controls the height of the grown trees	Control for overfitting
iteration	Control for the number of grown trees	Influences gradient descent minimal convergence and overall training time
learning_rate	Control the gradient step at each iteration	Choice of the correct learning rate is important as it can either speed up convergence if chosen appropriately, prevent loss function convergence to the minimum if it is too large, or cause training to take an impossibly

		long time if the learning rate is too low.

Table 6: Details of parameters tuned for CB

5.2.3 Word Heuristics

Due to the small size of the training sample, we also tried to use a simpler word filtering method to predict the carbon classes for each sentence. For this method, when a sentence contains certain words, it will be predicted as a certain class. The word filter for each carbon class was created by examining the top 10 words with the highest TFIDF scores from each carbon class. Words that were unique to each class were chosen to create the filter. Fig 4. shows the exact word filters used for each class. If a sentence does not contain any words from "*class_zero*", it will be checked to see if it contains words from the other lists. If it does, it will be predicted as that class, else it will be predicted as class 4 (others).

```
class_zero = ["emissions","footprint","ghg", "coal"]
class_one = ["energy","renewable","electricity","power", "solar", "kwh"]
class_two = ["waste","paper", "office","recycled","environmental"]
class_three = ["sustainable","investment","investments","bonds", "portfolio", "finance"]
```

Fig 4. Word Filters Used

5.2.4 Model Ensemble

We also attempted the voting ensemble, mainly for two reasons. Firstly, even though the heuristics method mentioned in the previous section is appropriate for our small dataset size, its validation weighted F1 pales in comparison to that of the machine learning models by around 0.09, which is rather significant (see Appendix G). Secondly, given our small dataset, we were also relatively uncomfortable with using a single machine learning model in the production pipeline, which could hinder the predictive power of our model on unseen data points. Therefore, after research and with the green light by our advisor, Mr Gupta, we decided to go with a hybrid approach of a voting ensemble. Details of the general workings of the voting ensemble can be found in Section 5.1.3.

For our text classification voting ensemble, we used 6 base classifiers, which are the LR, NB, SVM, RF, CB and the heuristics classifier. In addition, as we were unable to output probabilities for the heuristics classifier, we explored a modified soft ensemble approach where we added a float between 0 and 1 to the total class probability of the heuristics classifier's predicted class. This float was tuned to ensure that we had the most optimal weight for the heuristics classifier.

5.2.5 Model Results

The text classification results can be found in Appendix G. It is observed that the modified soft voting ensemble shared the highest validation weighted average F1 score and validation accuracy as the RF model. However, as the random forest is prone to overfitting on small datasets, and

given our preference for the ensemble method as mentioned in Section 5.2.4, we chose the modified soft voting ensemble as the final production text classifier.

5.3 Rule Mining

As we would be displaying the relevant sentences in the dashboard, we wanted to retrieve and highlight the most important sections of each sentence, which are sections that quantify decarbonisation progress within each predicted relevant sentence. This will allow our dashboard to leverage visual cues such as color and size to draw attention to the most important parts of a sentence. This was achieved by using rules to mine for important sections of each sentence.

5.3.1 Algorithm

An overview of our rule mining process and algorithm is shown below (Fig 5).

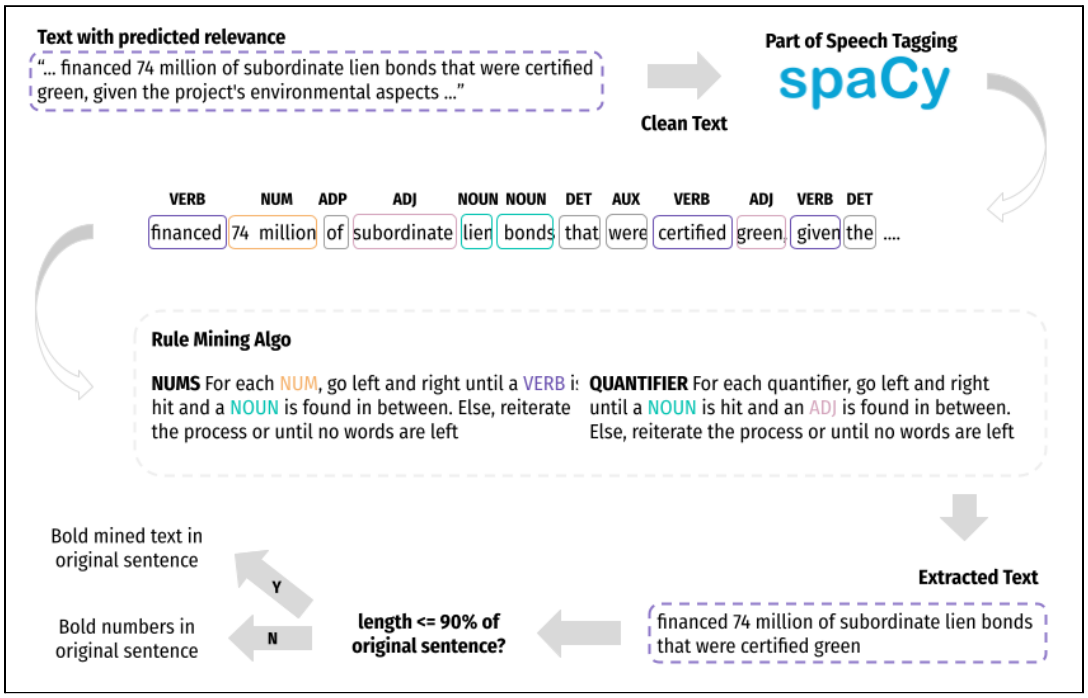


Fig 5. Rule Mining Process and Algorithm

Firstly, for all the sentences predicted as relevant, we will first clean the sentences by removing date combinations and also mentions of 'co2' (carbon dioxide). This was done so that the rule mining extraction algorithm for words would not be triggered for these context noisy tokens (words). Afterwards, we used the spaCy library for part of speech (POS) tagging to retrieve the coarse and fine grain POS for each token in the cleaned sentence. Using the retrieved POS tags for each token, we then run our rule mining algorithm which was formulated by observing POS patterns within the ground zero set. The algorithm runs whenever there are numbers and quantifiers observed in a sentence and both of these are extracted using different rules as seen in Fig 5. After the extracted text(s) is/are retrieved, an additional step was conducted before

bolding these extracted portions within the sentence in the dashboard. This step checks if the extracted text is less than 90% of the original text length. If it is, all extracted text(s) will be bolded, else only numbers in the sentence would be bolded. This is because rule mining is unsupervised and might sometimes extract noisy data which could potentially be the whole sentence. In such cases, we felt that bolding too much of the text would not add value to the end users and thus we added the last step before bolding the mined text in the dashboard. The rules were fine-tuned by testing them against our ground zero data set, as described in the next section, and the rules with the best performance were chosen.

5.3.2 Ground Zero Data Set

We labelled a ground zero data set of 60 sentences, with the text that should ideally be extracted, to evaluate the performance of the eventual rule mining algorithm. This set consists of sentences from Appendix D and was further supplemented with relevant sentences from the data labelled in Section 2.3. To ensure that a representative set was constructed, we ensured that sentences from all the decarbonisation classes described earlier were included in the ground zero data set. In addition, we took steps to ensure that a suitable number of sentences containing quantifiers such as ‘double’, ‘triple’ and ‘half’ were also represented despite being a rarer occurrence amongst the entire text corpus. In total, 80 tokens were labelled with the corresponding text that should be extracted in the ideal case.

5.3.3 Evaluation Metric

We developed a new metric, matched accuracy percentage (%) (Fig 6), to quantify the rule mining accuracy as conventional metrics such as precision, recall and weighted F1 would not work in this context. For each token, we derived the absolute difference in the number of words extracted versus the number of words that should have been extracted. This is then summed across all tokens to retrieve the total absolute differences in extracted and expected text across the ground zero corpus. Afterwards, we divided this sum by the total number of expected words and subtracted this from 1 to get the percentage of words correctly matched.

$$Matched\ Accuracy\ \% \ = \ 100 * (1 - \frac{Total\ absolute\ differences\ in\ extracted\ and\ expected\ text}{Total\ number\ of\ words\ in\ ground\ zero\ set})$$

Fig 6. Matched Accuracy Percentage Formula

5.3.4 Results

For the rule mining algorithm, we experimented with lemmatization and non lemmatizing the sentence. Lemmatization refers to the process where we convert many variants of a token into its base dictionary form. For example, under lemmatization, tokens such as ‘reduces’, ‘reducing’, ‘reduced’ will all be converted to the base form of ‘reduce’. Another parameter we optimised was the verb fine grain POS that does not induce stopping in the numbers extraction algorithm.

This was attempted as we found that simply inducing stopping once all verbs are reached does not give the most optimal result.

A summary of the matched accuracy % under the different parameter combinations can be found in Appendix H. Amongst all the combinations, we found that the combination of lemmatizing the input text along with excluding fine grain POS gives the highest matched accuracy % of 77.1%, which was what we used in the final production rule mining code.

5.4 Sentiment Analysis - VADER

We also conducted sentiment analysis on the relevant sentences from each carbon class to obtain the overall sentiments for a specific carbon class within the text corpus. This was done so as to provide a quantitative measure that could shed light into an FI's current decarbonisation progress across the various carbon categories.

For sentiment analysis, we chose to use VADER, a lexicon and rule-based sentiment analysis tool. VADER relies on a dictionary that maps lexical features (words) to sentiment scores. One key advantage of VADER is that it not only analyses words, but it also takes into account the emotions of the words used when creating the sentiment scores. Most importantly, its rule-based approach requires little pre-work, in contrast to machine learning methods which require large amounts of data for training, making it suitable for our use since our dataset of relevant sentences is small. This is because we not only have limited labelled sentences, but these labelled sentences are also subsequently further filtered during the relevance prediction step mentioned in section 5.1, further decreasing the size of our training data for carbon class predictions and sentiment analysis. We first used the pretrained VADER dictionary as the baseline model and the "compound" score, which is the aggregated version of the "neg", "neu" and "pos" sentiment, was extracted as the sentiment score for each sentence. We further customised the VADER model by adding in decarbonisation specific terms (Appendix I) to allow it to deal with the sentences in our dataset, to enhance our model performance. An example of improvements can be seen in Fig 7 below, where updating the VADER model for 'reduced' to 1 from 0 gives a more accurate depiction of the model's general sentiment. To display the overall sentiments for each carbon class in the dashboard, the sentiment scores for each sentence in a specific class was aggregated by averaging.

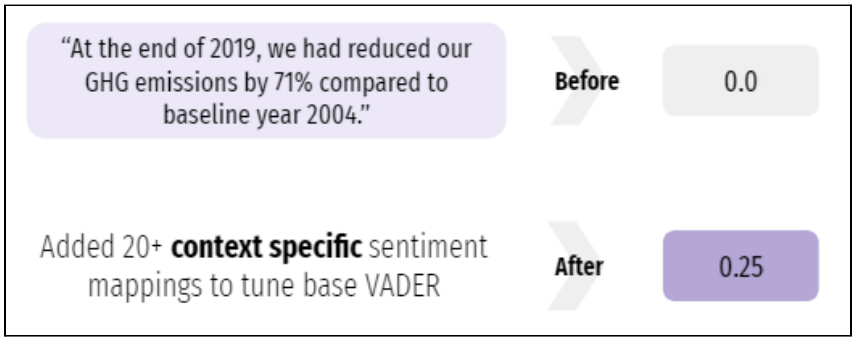


Fig 7. Improvements in VADER sentiment analysis model

6 Chart Detection and Extraction

As graphs and charts are often used to report decarbonisation-related information in sustainability reports, we saw immense value in including the detection and subsequent extraction of chart and graph images (Appendix J) in our information extraction pipeline. By extending existing Optical Character Recognition (OCR) techniques and Computer Vision libraries, we are able to capture image-based charts and graphs related to decarbonisation.

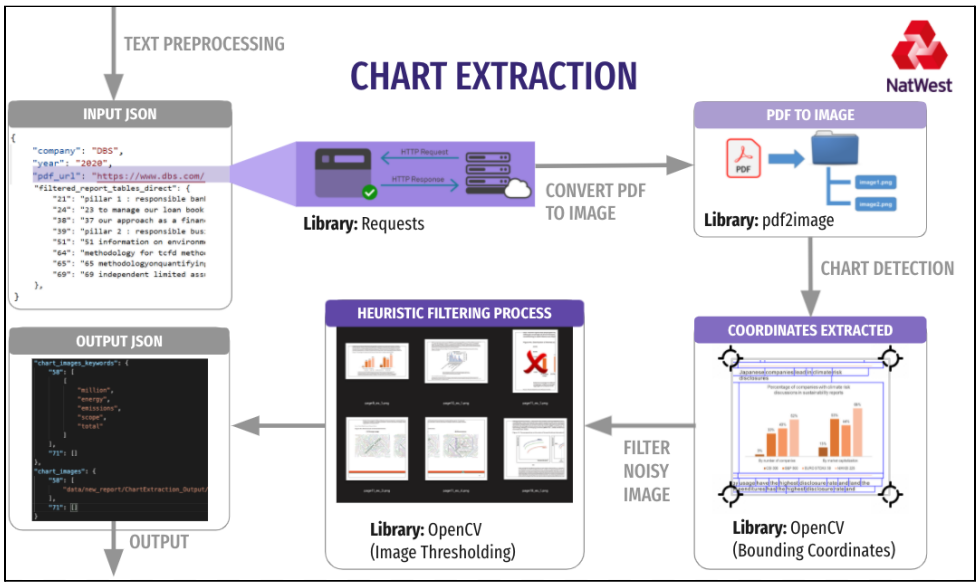


Fig 8. Chart Detection Pipeline

For the chart extraction pipeline, we utilise the list of filtered pages obtained from the preliminary page filtering step, in Section 4, as we believe that using the relevant filtered pages will yield better extraction results. In addition, this allowed us to easily scale up our code as working with images can be rather computationally expensive. From there, we retrieved the URL link for the PDF and converted each PDF page into an image using the *pdf2image* Python package and stored the images for subsequent processing. Our chart extraction pipeline consists of 3 main phases: chart detection, heuristic filtering process and storage of the extracted information in the final database json file (Fig 8).

6.1 Chart Detection

For chart detection, the main task is to correctly detect charts in reports and identify the coordinates of the rectangular box that outlines the borders of the charts. This may seem straightforward, but in reality, automatic chart detection is complex due to variations not only between the different chart types, but also between charts of the same kind, which may differ in data distribution, layout, or presence of noise. Furthermore, the limited resources and academic studies done on chart detection in PDF reports for us to make reference to further increases the challenge of this task. Thus, in our attempt to detect charts, we devised an approach that uses specific image properties that can intuitively suggest that an image and/or a specific part of an image (i.e. image of PDF page) is a chart. This approach was chosen and developed through an intensive examination of our dataset.

In the first phase, we start by identifying the orientation of a pdf page image by comparing the height and width of the pdf image of the report. A report is determined to be portrait if the height is greater than its width and landscape if the width is greater than its height. This step is necessary as the package used, OpenCV, can only process portrait images, compressing the image size for landscape images pages. This has been observed to negatively affect image detection and thus there is a need to identify the orientations of a page image and process different orientations differently. For landscape orientations, we will split the page into two and process each page individually as a pdf page while for portrait orientation, we skip the splitting step.

Next, we leverage the computer vision library, *OpenCV*'s functionalities to automatically detect all elements in a PDF image to aid our chart detection. *OpenCV* detects PDF elements by identifying the xy-coordinates of all bounding boxes that may contain text or chart PDF images. An example can be shown in Fig 9. Using these xy-coordinates identified, we split each pdf image into pdf elements. We then proceed to iterate through each pdf element image and filter images by size. Images that are too small based on dimensions such as height, width and area of the image will be dropped. This is because such images are highly likely to contain texts and not charts, allowing us to eliminate irrelevant images. As this step is not foolproof, the images retained will be stored in a folder for further filtering in the next section.

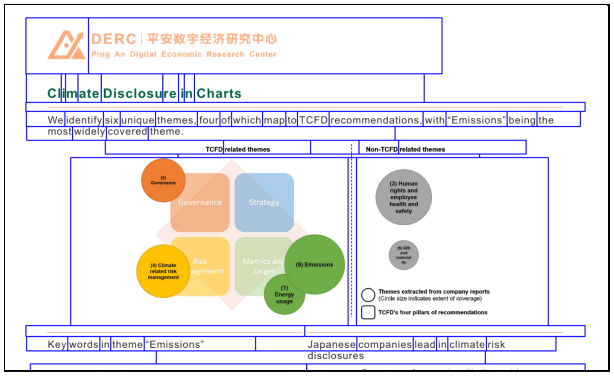


Fig 9. Sample output after running a PDF image through OpenCV

6.2 Heuristic Filtering Process

The next phase aims to filter out noisy images which have low probabilities of being charts from the images extracted in the first phase by using heuristic rules. We identified 15 image properties that could potentially distinguish those with charts from those without charts. The properties, description and rationale behind choosing them can be found in Appendix K. For all images extracted from all reports in the test set (Section 8) during the first phase, the values for all 15 properties were computed and stored in an Excel sheet that can be found in Appendix L.

To filter out irrelevant images, all images will be passed through 2 filtering steps. After which, the images will be deemed as highly relevant and will be stored to be displayed in our dashboard.

1. Firstly, filter out images using pixel properties. Images that have minimal color pixels will be filtered out as images of chart visualisations usually contain multiple colors.

- Image properties and rules used : (7 < *dilated_region* < 26) AND (7000 < *white_pix* < 90000) AND (*total_len* < 68) AND (*ta_ratio* < 10)

2. Next, for the images filtered in the first step, further filter out images based on the properties of an image and the relevance of text in an image. For this step, images that contain too much text or do not contain keywords from a list of decarbonisation related words in Appendix M will be filtered out.
- Image properties and rules used : (*keywords* > 0) AND (*tt_ratio* < 0.99) AND (*bw_ratio* > 14) AND (*textonly_len* < 90) AND (*numonly_len* > 1) AND (*na_ratio* > 0.45)

These 2 filtering steps were first developed by examining properties of images with and without charts. Subsequently, all possible combinations and threshold values for the properties were attempted to filter images. The final combination of properties and threshold values were chosen based on the heuristic rule performance on the test set (details in Section 8).

6.3 Limitations

As the process taken to detect and extract charts mainly utilise Image properties and heuristics that we identified and fine tuned by looking through several sustainability reports in our generated test set (Section 8), they are generalised and are not personalised to the myriads of chart layouts and structures that exist in PDF reports. As such, images that are extracted would (1) not be fully clean and may include text from the report together with the image itself; (2) may be totally irrelevant. Thus, despite our efforts, we recognise that improvements can still be made.

7 Table Detection and Extraction

From the sustainability reports collected, another key observation was that a majority of firms presented decarbonisation-related metrics in tabular format. Therefore, we saw value in extending our data collection pipeline beyond text and chart extraction to include the detection and extraction of tabular data. By building on existing OCR techniques and PDF table extraction libraries, we developed a pipeline that is able to capture both image-based and text-based tabular data related to decarbonisation. Through our tabular data pipeline, we are able to offer our client a more comprehensive set of decarbonisation data and better value-add to their workflows.

Our tabular data extraction pipeline consists of 3 main phases: table detection, tabular data extraction as well as table cleaning and filtering as shown in Fig 10. First, a table detection algorithm is needed to correctly detect the regions in a PDF page that contain tables and identify the coordinates of those tables. Next, a tabular data extraction algorithm is required for proper identification of rows and columns as well as for extraction of content within each cell. The table is eventually extracted as a dataframe and undergoes a series of data cleaning and filtering steps to assess for its ESG relevance. Only dataframes identified as relevant are stored alongside with its corresponding table image. Reasons as to why we decided to store both dataframes and images can be found in Section 8.3.

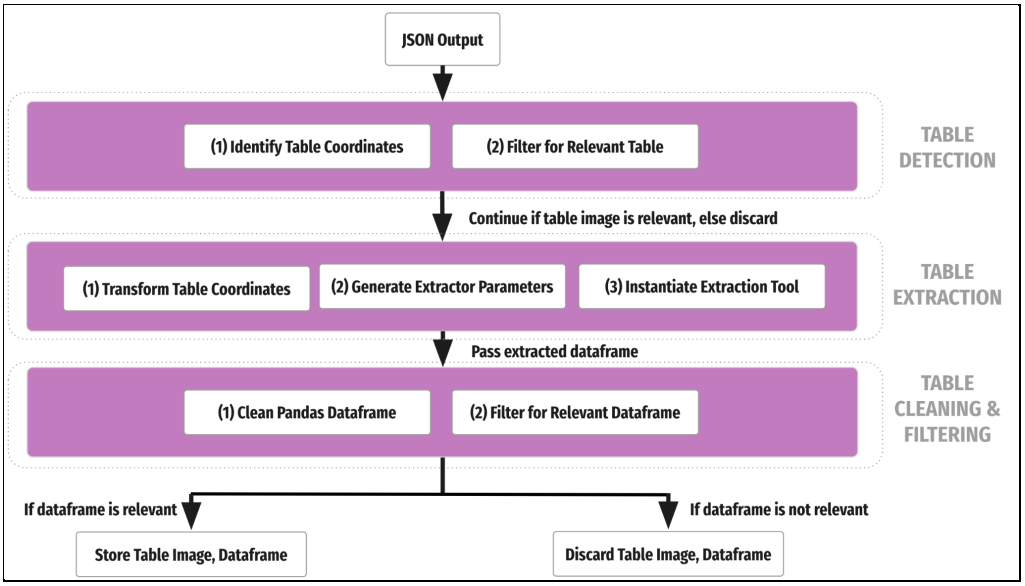
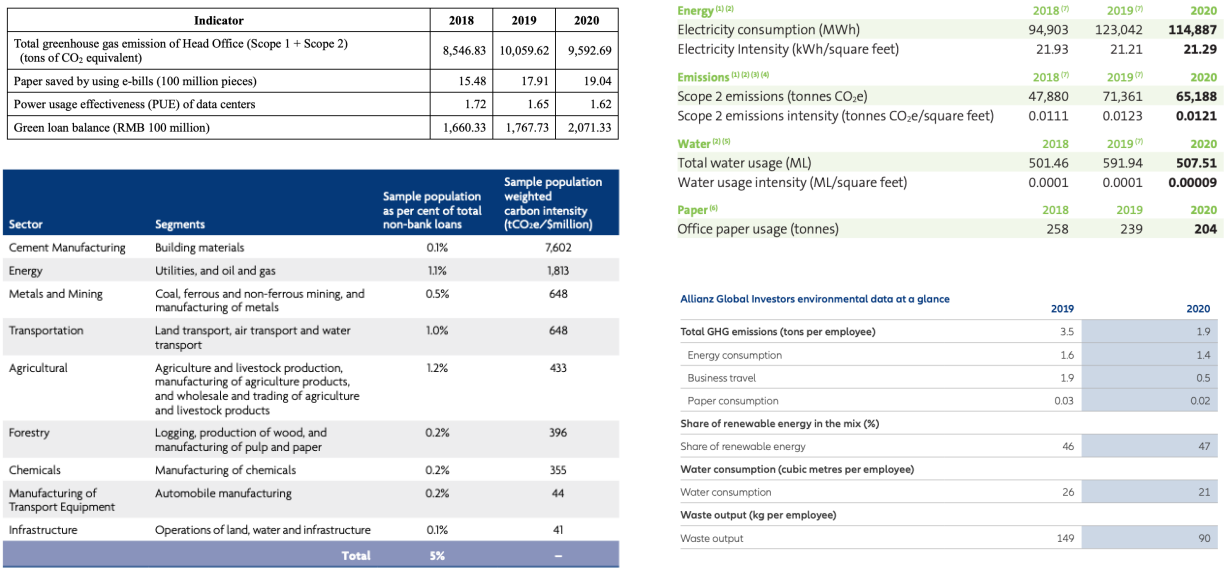


Fig 10. Flow of the tabular detection and extraction pipeline

From the pages of the report filtered after the page filtering step in Section 4, we further refined the page selections to aid the table detection process. If a page contains at least 1 unit of measurement (Appendix N) and 10 numbers that are not identified as dates, we include the page number into the list of pages to be processed for the subsequent tasks in our tabular pipeline. This step is necessary despite the preliminary page filter as we observe that not all filtered pages contain tables and pages with tables usually contain at least 1 unit of measurement. Thus by adding this step, we are able to reduce and focus on a set of pages that are highly likely to contain tables. Without minimising the area of search in these lengthy reports, the table detection algorithm will have to process every page, which is an extremely computationally intensive task that cannot scale up easily.

7.1 Table Detection

The first stage of the tabular data pipeline involves table detection from both text-based and image-based (scanned) PDFs. The main task is to correctly detect tables in reports and identify the coordinates of the rectangular box that outlines the borders of the table. This may seem like a straightforward task, but in reality, automatic table detection is complex due to the primary reason that tables come in various structures with different formatting and border styles (Fig 11). Therefore, we should be aware of these challenges when selecting a robust model.



Indicator	2018	2019	2020
Total greenhouse gas emission of Head Office (Scope 1 + Scope 2) (tons of CO_2 equivalent)	8,546.83	10,059.62	9,592.69
Paper saved by using e-bills (100 million pieces)	15.48	17.91	19.04
Power usage effectiveness (PUE) of data centers	1.72	1.65	1.62
Green loan balance (RMB 100 million)	1,660.33	1,767.73	2,071.33

Sector	Segments	Sample population as per cent of total non-bank loans	Sample population weighted carbon intensity (tCO_2e/$million)
Cement Manufacturing	Building materials	0.1%	7,602
Energy	Utilities, and oil and gas	1.1%	1,813
Metals and Mining	Coal, ferrous and non-ferrous mining, and manufacturing of metals	0.5%	648
Transportation	Land transport, air transport and water transport	1.0%	648
Agricultural	Agriculture and livestock production, manufacturing of agriculture products, and wholesale and trading of agriculture and livestock products	1.2%	433
Forestry	Logging, production of wood, and manufacturing of pulp and paper	0.2%	396
Chemicals	Manufacturing of chemicals	0.2%	355
Manufacturing of Transport Equipment	Automobile manufacturing	0.2%	44
Infrastructure	Operations of land, water and infrastructure	0.1%	41
	Total	5%	–

Energy [1] [2]	2018 [7]	2019 [7]	2020
Electricity consumption (MWh)	94,903	123,042	**114,887**
Electricity Intensity (kWh/square feet)	21.93	21.21	**21.29**
Emissions [1] [2] [3] [4]	**2018** [7]	**2019** [7]	**2020**
Scope 2 emissions (tonnes CO_2e)	47,880	71,361	**65,188**
Scope 2 emissions intensity (tonnes CO_2e/square feet)	0.0111	0.0123	**0.0121**
Water [2] [5]	**2018**	**2019** [7]	**2020**
Total water usage (ML)	501.46	591.94	**507.51**
Water usage intensity (ML/square feet)	0.0001	0.0001	**0.00009**
Paper [6]	**2018**	**2019**	**2020**
Office paper usage (tonnes)	258	239	**204**

Allianz Global Investors environmental data at a glance	2019	2020
Total GHG emissions (tons per employee)	3.5	1.9
Energy consumption	1.6	1.4
Business travel	1.9	0.5
Paper consumption	0.03	0.02
Share of renewable energy in the mix (%)		
Share of renewable energy	46	47
Water consumption (cubic metres per employee)		
Water consumption	26	21
Waste output (kg per employee)		
Waste output	149	90

Fig 11. Relevant table samples of different border styles found in sustainability reports

7.1.1 Choice of Algorithm

One key factor in selecting the best model is its ability to detect tables of all border types - fully bordered, partially bordered, and borderless (Fig 11). *OpenCV* is a traditional and commonly used computer vision technique for table detection. It is an open source CV library that performs processing of images including adjusting the contrast of the image and edge detection. However, it fails to generalize as it is not robust to table layout variations (Huynh-Van et al., 2018). Clearly defined horizontal and vertical lines of a table are required to detect the table region on a page. The performance of *OpenCV* drastically reduces when the table is partially bordered.

In recent years, modern methods using deep learning techniques have surfaced, with the introduction of more robust methods such as *TableNet* (Paliwal et al.,2019), *CascadeTabNet* (Prasad et al., 2020) and *Multi-Type-TD-TSR* (Fischer et al., 2021). They do not require extensive pre-processing, unlike the traditional *OpenCV* model, and results have shown that they are able to outperform state-of-art methods in table detection. The categories that each table detection method works for are listed in Table 7 respectively.

Algorithm	Bordered	Partially-bordered	Borderless
TableNet	✓		✓
CascadeTabNet	✓		✓
Multi-Type-TD-TSR	✓	✓	✓

Table 7. Table styles that each table detection library works for

From Table 7, we observe that *Multi-Type-TD-TSR* offers a more comprehensive approach by distinguishing 3 types of tables - fully bordered, partially bordered and borderless. Furthermore, *Multi-Type-TD-TSR* addresses the issue of possible noise in the image and corrects image alignment. The table detection algorithm then extracts bounding boxes for each table in a page by using Convolutional Neural Network (CNN).

As *CascadeTabNet* and *Multi-Type-TD-TSR* code implementations are publicly available, we experimented with both algorithms to detect varying table types. Undoubtedly, *Multi-Type-TD-TSR* outperforms *CascadeTabNet* as it is able to detect tables of all types and accurately extract the coordinates of the rectangular bounding boxes. Therefore, *Multi-Type-TD-TSR* is the final choice of model to be used for table detection in PDF reports.

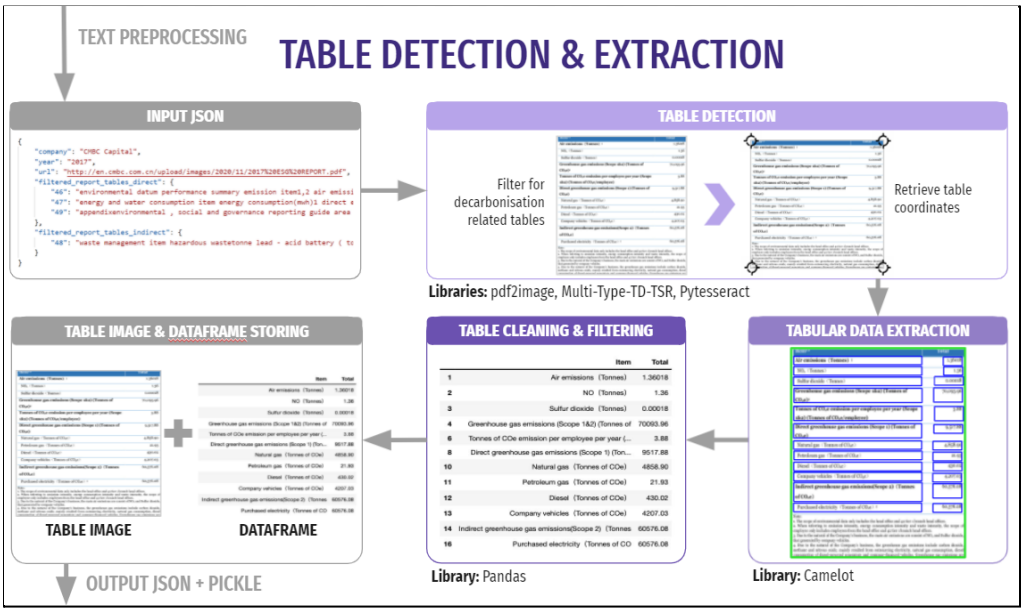
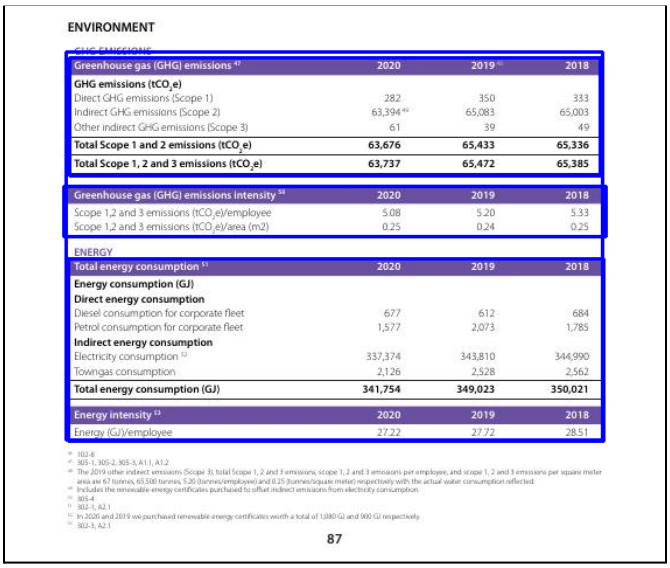


Fig 12. Flow of table detection and extraction pipeline

Fig 12 outlines the table detection pipeline using our chosen algorithm. Similar to the chart extraction pipeline in Section 6, the URL link for the PDF is extracted from the JSON and the sustainability report is retrieved and converted to images (i.e. every page is converted to one image) via *pdf2image* library in order to apply the table detection algorithm which takes in an image as input. We then apply the chosen *Multi-Type-TD-TSR* algorithm, on selected pages of the report as mentioned earlier. For each table detected, the coordinates of the table are retrieved. The table image is enlarged and converted to text using *pytesseract* library. Using the extracted text strings, we perform keyword filtering to check if the detected table contains at least 1 relevant decarbonisation-related keywords such as GHG emissions, energy consumption and 1 unit of measurement such as tonnes of CO_2 (list of keywords and unit of measurement can be found in Appendix N). If the tables meet the above criteria, the table image will be kept for subsequent tabular data extraction (discussed in Section 7.2)

Upon further inspection of the tables extracted from each report, we found that there were overlapping rectangular regions being extracted in a single page. This poses a problem as there will be repetitive parts of an extracted image appearing in another extracted image, resulting in

overlapping information displayed on the dashboard. An example is illustrated in Fig 13. To address this issue, we check if there is an intersection between 2 rectangular bounding boxes by comparing their coordinates. We discard a detected table region should it overlap with existing detected regions in a page. This has reduced the number of tables that the model has extracted per report by 19%, which also means that the subsequent tabular data extraction phase will be more efficient and see reduced noise. We will discuss the performance of our table detection pipeline in Section 8.2.



ENVIRONMENT

Greenhouse gas (GHG) emissions [47]	2020	2019 [48]	2018
GHG emissions (tCO_2e)			
Direct GHG emissions (Scope 1)	282	350	333
Indirect GHG emissions (Scope 2)	63,394 [49]	65,083	65,003
Other indirect GHG emissions (Scope 3)	61	39	49
Total Scope 1 and 2 emissions (tCO_2e)	**63,676**	**65,433**	**65,336**
Total Scope 1, 2 and 3 emissions (tCO_2e)	**63,737**	**65,472**	**65,385**

Greenhouse gas (GHG) emissions intensity [50]	2020	2019	2018
Scope 1,2 and 3 emissions (tCO_2e)/employee	5.08	5.20	5.33
Scope 1,2 and 3 emissions (tCO_2e)/area (m2)	0.25	0.24	0.25

ENERGY

Total energy consumption [51]	2020	2019	2018
Energy consumption (GJ)			
Direct energy consumption			
Diesel consumption for corporate fleet	677	612	684
Petrol consumption for corporate fleet	1,577	2,073	1,785
Indirect energy consumption			
Electricity consumption [52]	337,374	343,810	344,990
Towngas consumption	2,126	2,528	2,562
Total energy consumption (GJ)	**341,754**	**349,023**	**350,021**

Energy intensity [53]	2020	2019	2018
Energy (GJ)/employee	27.22	27.72	28.51

[47] 102-8
[48] 305-1, 305-2, 305-3, A1.1, A1.2
[49] The 2019 other indirect emissions (Scope 3), total Scope 1, 2 and 3 emissions, scope 1, 2 and 3 emissions per employee, and scope 1, 2 and 3 emissions per square meter area are 67 tonnes, 65,500 tonnes, 5.20 (tonnes/employee) and 0.25 (tonnes/square meter) respectively with the actual water consumption reflected.
[50] Includes the renewable energy certificates purchased to offset indirect emissions from electricity consumption.
[51] 305-4
[52] 302-1, A2.1
[53] In 2020 and 2019 we purchased renewable energy certificates worth a total of 1,080 GJ and 900 GJ respectively.
[54] 302-3, A2.1

87

Fig 13. A sample page with overlapping bounding regions identified

7.2 Tabular Data Extraction and Conversion

The second stage of the tabular data pipeline involves utilizing a PDF table extraction library to instantiate an extractor for the identification of the table structure and extraction of its corresponding cell content. Though this may seem like an easy task, the reality is that tabular data extraction is an extremely challenging task because table-specific machine readable markup is missing in PDF format, which often leads to extremely poor and noisy extraction of tabular data. To make matters even more challenging, there is a wide variety of innovative table formats employed across the sustainability reports, which makes developing a general-purpose automated extraction technique extremely difficult. Therefore, noting the severe limitations of a general-purpose extractor for our project, we decided to develop a custom extraction technique that instantiates a customised extractor for each table identified. Though this approach takes a slightly longer processing time, we managed to achieve a huge improvement in the extraction accuracy.

As mentioned, one key factor in selecting the best and most appropriate tabular data extractor is the ability to customise the different parameter settings of the extractor in order to instantiate a table-specific extractor that can caters for individual tabular formats - fully bordered, semi-bordered, and borderless as shown in Section 7.1. There are open source tools such as *pdfplumber*, *Tabula* and *Camelot* that are widely used to extract tables from PDF files. We

experimented with these extraction tools and our observations on its advantages and limitations are listed in Table 8.

Library	Advantage(s)	Limitation(s)	Reason(s)
pdfplumber[6]	Extracts simple bordered tables well	• Low extraction accuracy for complex tables • Limited parameters for tweaking	Limited tweakable parameters
Tabula-py[7]	Offers more parameters such as table region available for tweaking as compared to pdfplumber	• Drop in extraction performance for complex tables • Less customisable parameters in comparison to Camelot • Challenging to tweak parameters due to poor documentation and class definitions (Rosén, n.d.)	More parameters than pdfplumber but still less customisable in comparison to Camelot
Camelot[8]	Offers greatest flexibility control in tweaking parameters	• Unable to extract tables with merged cells well as cell content are extracted line by line	Rich in parameters and most customisable among all 3 models

Table 8. Summary of Advantages and Limitations of Extraction Tools

Fig 12 outlines the table extraction pipeline. In order to create table-specific extractor, the second phase of our tabular data pipeline utilises the coordinates obtained earlier from the *Multi-Type-TD-TSR* algorithm. Since the obtained coordinates are image coordinates, the pipeline first has to transform the coordinates to PDF coordinates due to differences in image and PDF coordinate systems. The PDF coordinates are in the form of (X_1, Y_1) which represents the top left corner and (X_2, Y_2) which represents the bottom right corner of a detected table. With the PDF coordinates, the pipeline then instantiates a table-specific Camelot extractor with parameters such as table_areas, edge_tol and split_text specified. The specified values and corresponding justifications on why these parameters are selected are outlined in Table 9. Once identified, the Camelot extractor extracts the text-based table and converts it to a Pandas dataframe. The extracted dataframe would undergo a series of table cleaning and filtering steps in the final phase of our pipeline, which would be elaborated in Section 7.3.

[6] See GitHub code implementation for *pdfplumber*
[7] See GitHub code implementation for *Tabula*
[8] See GitHub code implementation for *Camelot*

Parameter	Value(s)	Reason for Selection of Parameter
Table Region (table_areas)	(X_1, Y_1, X_2, Y_2)	With the table_areas specified, the extractor tool focuses on analysing the specified region to look for tables, thus reducing the noise from other irrelevant sections of the page.
Edge Tolerance (edge_tol)	X_2-X_1	With the edge_tolerance=table width specified, relatively far apart columns in the table would be extracted in a single table instead of spanning across multiple tables, thus improving the table extraction process.
Split Text (split_text)	True	With the split_text=True specified, text in different cells that are relatively near would be identified and split into different cells in the extracted dataframe instead of being merged into a single cell.

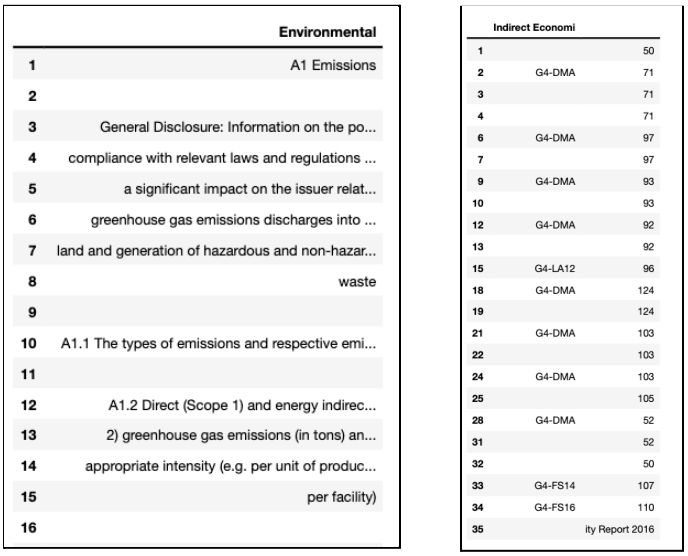
Table 9. Table of relevant parameters we identified for Camelot and their uses

7.3 Table Cleaning and Filtering

The final phase of our tabular data pipeline is the table cleaning and filtering phase. After extracting the table and converting it to a Pandas dataframe, the pipeline sieves out inaccuracies during the extraction process and assesses the relevance of the dataframe. If the dataframe is assessed to be irrelevant at the end of this phase, no dataframe would be stored. On the other hand, if the dataframe is identified as relevant, the curated dataframe would be stored in the Pickle database file. Primarily, the data cleaning pipeline focuses on preserving only tables with useful numerical ESG metrics instead of lengthy text content that are summarised and outputted in our text extraction pipeline. The following table cleaning and filtering steps were taken:

1. Loops through the content of dataframe's cell and converts cell content to NaN if the content length exceeds 3 and/or if there is no numerical value present.
2. Drop column(s) with a high proportion of empty rows.
3. Drop the entire table if there exists only 1 single column or 0 row left.
4. Drop the entire table if the related keywords like "page", "Page" or "PAGE" exist in the dataframe, indicating the numerical values are simply page numbers of limited use.
5. Drop the entire table if there exists a high proportion of empty rows in the 1st column.
6. Drop the entire table if there exists not a single header.

At the end of the phase, we incorporated an additional key step to further eliminate noisy dataframe. On the final curated dataframe, the pipeline conducts a ESG keyword search and attempts to find decarbonisation-related information from the cell content. If no keyword can be found, the dataframe and its corresponding image are dropped as this simply implies the initial detected table is likely not in a tabular structure, and thus should be discarded by our tabular data pipeline. Some samples of these noisy dataframes discarded are shown in Fig 14. On the other hand, if the curated dataframe is identified to be extracted well, both dataframe and table image would be stored and can be viewed by the user in their preferred format on the Dashboard.



	Environmental
1	A1 Emissions
2	
3	General Disclosure: Information on the po...
4	compliance with relevant laws and regulations ...
5	a significant impact on the issuer relat...
6	greenhouse gas emissions discharges into ...
7	land and generation of hazardous and non-hazar...
8	waste
9	
10	A1.1 The types of emissions and respective emi...
11	
12	A1.2 Direct (Scope 1) and energy indirec...
13	2) greenhouse gas emissions (in tons) an...
14	appropriate intensity (e.g. per unit of produc...
15	per facility)
16	

	Indirect Economi	
1		50
2	G4-DMA	71
3		71
4		71
6	G4-DMA	97
7		97
9	G4-DMA	93
10		93
12	G4-DMA	92
13		92
15	G4-LA12	96
18	G4-DMA	124
19		124
21	G4-DMA	103
22		103
24	G4-DMA	103
25		105
28	G4-DMA	52
31		52
32		50
33	G4-FS14	107
34	G4-FS16	110
35		ity Report 2016

Fig 14. Sample outputs of noisy Pandas dataframes extracted by Camelot

8 Chart & Table Extraction Pipeline Evaluation

Unlike the labelled data set used for text classification in Section 5, we do not have ground-truthed target outputs to evaluate the chart and table extraction pipeline as it is extremely challenging to automate the evaluation process. Therefore, we used a standardized set of sustainability reports to form our test set and performed a manual comparison between the expected extraction and the actual extraction outputs by our pipeline. The same test set is used across chart and table extractions to validate the performance of our pipelines. To ensure that our pipelines are robust, we carefully selected test samples based on different criteria and layout variations for charts and tables. In total, 9 PDF reports across all FI types are collated as the standardized test set to evaluate our pipeline for table and chart detection. The details and justifications of the selection of reports are reflected in the table under Appendix O.

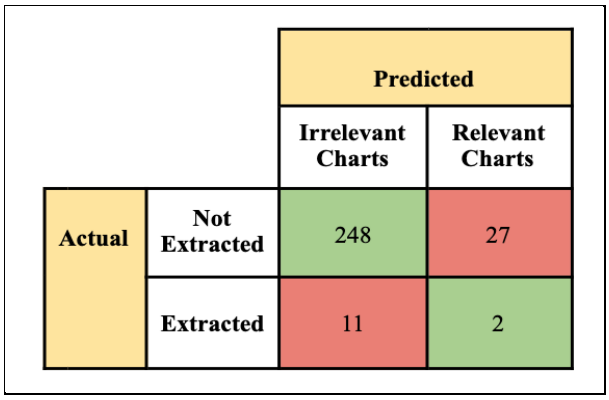
Metrics Used

For pipeline evaluation, we used 2 main metrics, precision and recall. The definitions and formulae of precision and recall used for the chart and tabular data pipeline varies due to the different goals of the respective pipelines, and details can be found in Appendix P.

For the chart extraction pipeline, our goal is to ensure the pipeline is robust in eliminating irrelevant images and be left with charts from our extraction since there is a high number of irrelevant images extracted across the reports.

For tabular data pipeline, since there are relatively fewer tables across the reports, our aim is to extract as many relevant tables as possible while maintaining a relatively low level of noise in our extraction.

8.1 Chart Extraction Results



		Predicted	
		Irrelevant Charts	Relevant Charts
Actual	Not Extracted	248	27
	Extracted	11	2

The chart detection pipeline was evaluated on the 9 reports in our common test set. The confusion matrix and performance metrics are computed as follows:

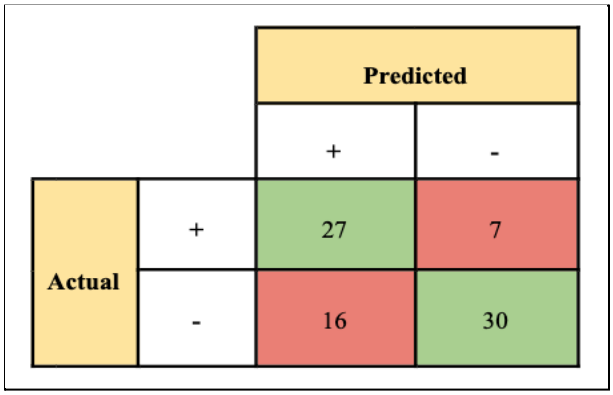
Precision: 0.95
Recall: 0.9
Accuracy: 0.92

The high recall of 0.9 indicates that the heuristic process is able to eliminate an overwhelming 90% of all the irrelevant images correctly in a report. The precision is 0.95, which means that 95% of all the extracted images are irrelevant. The overall accuracy of 92% indicates that the chart detection process is very robust in eliminating irrelevant images extracted.

8.1.1 Limitations & Methods to Overcome

Even though a relatively high recall is achieved, 5% of the images not extracted are relevant since the pipeline achieved a precision of 0.95. This approach can reduce the amount of noisy images displayed in the dashboard but risk the trade off that some charts may potentially be removed due to the strict rules set to reduce filter images. A possible extension, in Section 11.1, can be considered as an effective next step to overcome removal of relevant charts through modelling.

8.2 Table Detection Results



		Predicted	
		+	-
Actual	+	27	7
	-	16	30

The table detection pipeline was evaluated on the 9 reports in our common test set. The confusion matrix and performance metrics are computed as follows:

Precision: 0.63
Recall: 0.79
Accuracy: 0.71

The recall of 0.79 is relatively high, indicating that the table detection process is able to correctly extract 79% of all the relevant tables in a report. The precision is 0.63, which means that 63% of all the extracted images are relevant. The overall accuracy of 71% is satisfactory, although in general, the performance can still be improved.

8.2.1 Limitations & Methods to Overcome

Even though a relatively high recall is achieved, 37% of the extracted images are irrelevant since the pipeline achieved a precision of 0.63. After a thorough visual inspection, we noticed one common problem that lies in the table detection pipeline is that while it is able to detect pages with tables well, but for pages without, it incorrectly detects rectangular images as bordered tables or columns of text as borderless tables at times, hence the false positive rate is high. Therefore, we went a step further to improve the precision by parsing these images into the next phases of the tabular data pipeline that consists of table extraction and table cleaning and filtering (Section 7.2 and 7.3). We monitor any improvements in the precision and discuss them in the next subsection.

8.3 Table Detection Results (Post Filtering and Cleaning)

In Section 8.2, we discussed the issue of low precision of 0.63 (or high false positive rate) in the table detection phase. After processing these table images into the tabular data extraction and cleaning phase, we observe an evident improvement in the table image extraction results, where most images incorrectly extracted as tables from the table detection process have been discarded.

		Predicted	
		+	-
Actual	+	27	7
	-	7	39

Precision: 0.79
Recall: 0.79
Accuracy: 0.83

The precision improved significantly from 0.63 to 0.79, due to 9 previously detected non-table images being successfully discarded after the table extraction and cleaning process.

8.4 Table Extraction Results (Post Filtering and Cleaning)

		Predicted	
		+	-
Actual	+	12	4
	-	4	14

The table extraction pipeline was evaluated on the 9 reports in our common test set. The confusion matrix and performance metrics are computed as follows:

Precision: 0.75
Recall: 0.75
Accuracy: 0.77

The recall of 0.75 is satisfactory, indicating that the tabular data extraction process is able to correctly extract 75% of all the relevant tables detected from the table detection phase and output

them in cleaned, curated dataframes. The precision of 0.75 indicates that 75% of the outputs are cleaned dataframes. The overall accuracy of 77% is relatively high though there are still some limitations in our tabular extraction pipeline which is explained further in the following subsection.

8.4.1 Limitations & Methods to Overcome

Even though the recall and precision are relatively high, a common tabular structure not well extracted in our tabular pipeline are tables with merged cells and corresponding middle to bottom text alignment. These tables require a series of different table cleaning steps to cater for the mismatched row alignments between the content of different columns, and thus they are extracted as noisy dataframes in the pipeline. However, as the initial detected table is indeed in a well-structured tabular format, these noisy dataframes extracted are often not sieved out in the tables cleaning phase that focuses on identifying and discarding non-tabular formats. As a result, these kept dataframes often have incomplete headers or a couple of missing columns and rows when displayed on the Dashboard. Another limitation is also the incomplete extraction of table headers displayed above tabular structures in the reports as these headers are often not identified by the initial table detection pipeline as part of the table. A sample of such tabular structure and its corresponding extracted dataframe is reflected in Fig 15. To prevent the possible loss of key decarbonisation-related information, the pipeline stores the corresponding image of the table extracted as well, and can be viewed on the Dashboard.

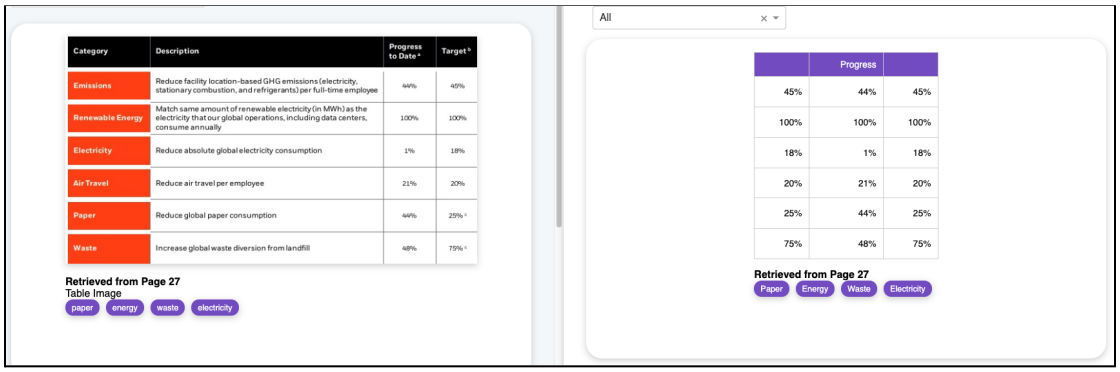


Fig 15. Sample output of the original table image and corresponding dataframe as displayed in Dashboard

9 Dashboard & Use Cases

Our end product is an interactive dashboard that captures all the decarbonisation information and metrics extracted from reports, after running it through our pipeline. It seamlessly integrates the output from text, charts and table extraction pipelines into an dashboard application to allow users to analyse decarbonisation related information in sustainability reports in a more structured and comprehensive way. The following subsections provide a brief overview of the functionalities in the individual pages. A detailed user guide can be found in **Appendix Q**.

9.1 *Library* Page

The 'Library' page, as shown in Fig 16, will be the landing page where users can access extracted information from existing reports that are stored in our database. The directory is sorted by firms and for each firm, sorted by year. A search bar can also be used to search for a

company and check if the desired report exists in our database. If it does not and users wish to analyse a new report, they can simply input the required fields: PDF URL, company name and year of report and submit it which will trigger our end-to-end pipeline to extract and generate relevant metrics and information from the new report. The information extracted can then be displayed on the dashboard and stored in the database for future reference. Users will not be limited to the existing companies in our database, as our dashboard is designed for long-term adoption with a growing database. On top of basic field input validation, additional validation checks are also put in place to ensure that only valid URLs (PDFs in English that can be parsed) as well as URLs that do not exist in our database can be inputted.

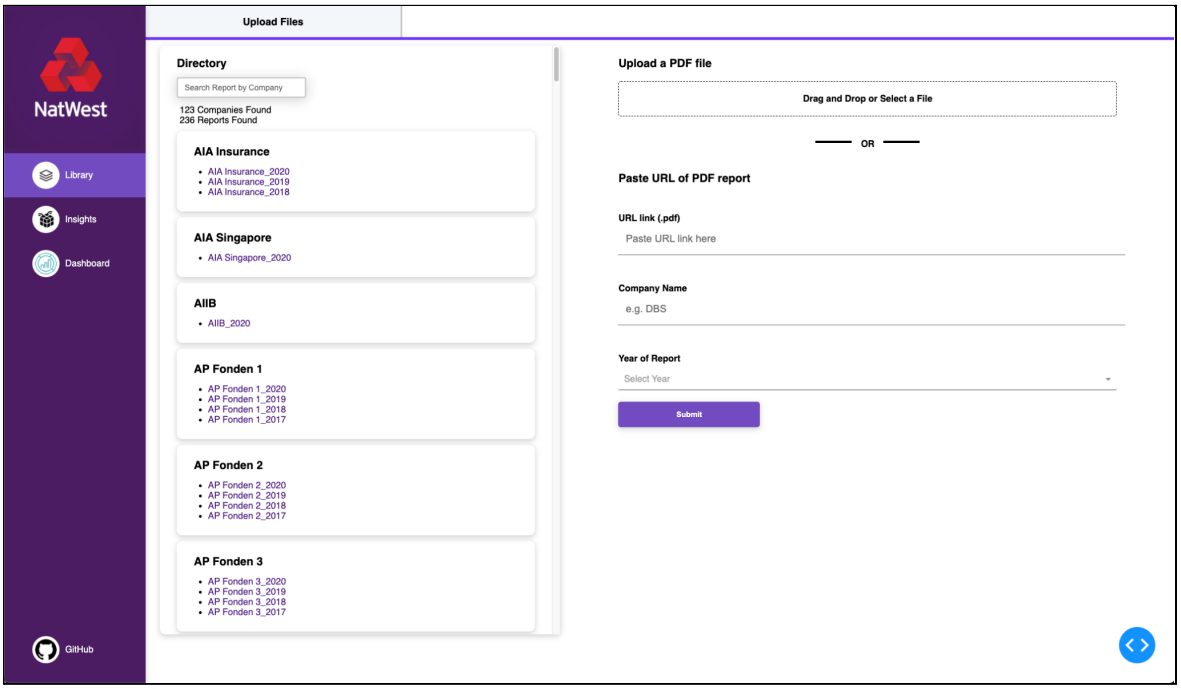


Fig 16. Library Page of our Dashboard

9.2 *Insights* Page

Once users select a report under the directory from the 'Library' page, they will be directed to the 'Insights' page as shown in Fig 17. On the left, they can view decarbonisation-related charts and table images that were detected and extracted by the image processing pipelines discussed in Sections 6 & 7. On the right, they can view the neatly cleaned tables, in the form of dataframes, of relevant metrics that are extracted from the tabular data extraction model. These tables and charts are extracted from a huge amount of unstructured data with different reporting scopes and formats by different FIs. Our pipelines convert these raw, non-standardized corporate disclosures into a standardised form displayed on the dashboard for all reports, allowing NatWest Markets to perform deeper analyses based on the company's use cases and business needs. This page allows users to examine all relevant information at a glance, without having to manually read through sustainability reports. They can also potentially supplement any ESG data, by third-party providers, used by NatWest Markets when making decisions. Thus, we believe that this component greatly adds value to existing internal tools used by NatWest Markets to provide more meaningful recommendations with regards to sustainable investing.

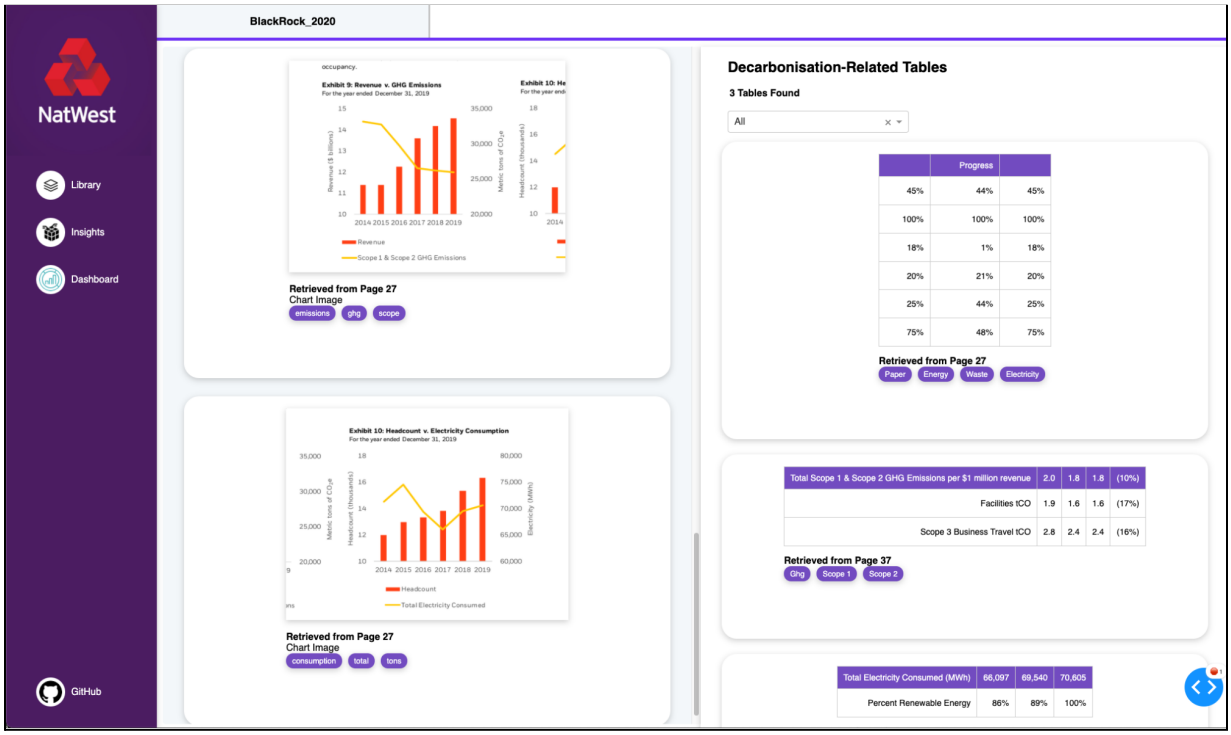


Fig 17. Insights Page of our Dashboard

9.3 *Dashboard* Page

The 'Dashboard' page as shown in Fig 18 presents different types of visualisations, giving a holistic view of key information automatically extracted from the entire financial report using text mining. The first visualisation displayed on the main dashboard page is word cloud for 4 of the 5 distinct carbon classes - Carbon Emissions, Energy, Waste and Sustainable Investing. We excluded the "Others" class for word cloud visualisation as we observed that it contains more noise and would not provide much value to the user. At first glance, users will be able to view a summary of important keywords that a company is focusing on in different environmental related areas. If users would like to deep dive into the actual textual data used to create these word clouds, a neatly collated table with relevant sentences extracted from the report is presented below the word clouds. This table offers a summarised view of specific emissions reduction goals and/or progress in the form of sentences, as well as key initiatives rolled out by the company, as mentioned in their report disclosures. Users can sort the textual information according to its relevance (i.e. Low, Medium & High) which is the discretized version of the relevance probability scores predicted by the relevance model discussed in Section 5.1. They can also filter the table by the 5 Carbon Classes predicted by the text classification model in Section 5.2. All these can be done with a search function across all fields to filter records according to users' inputs.

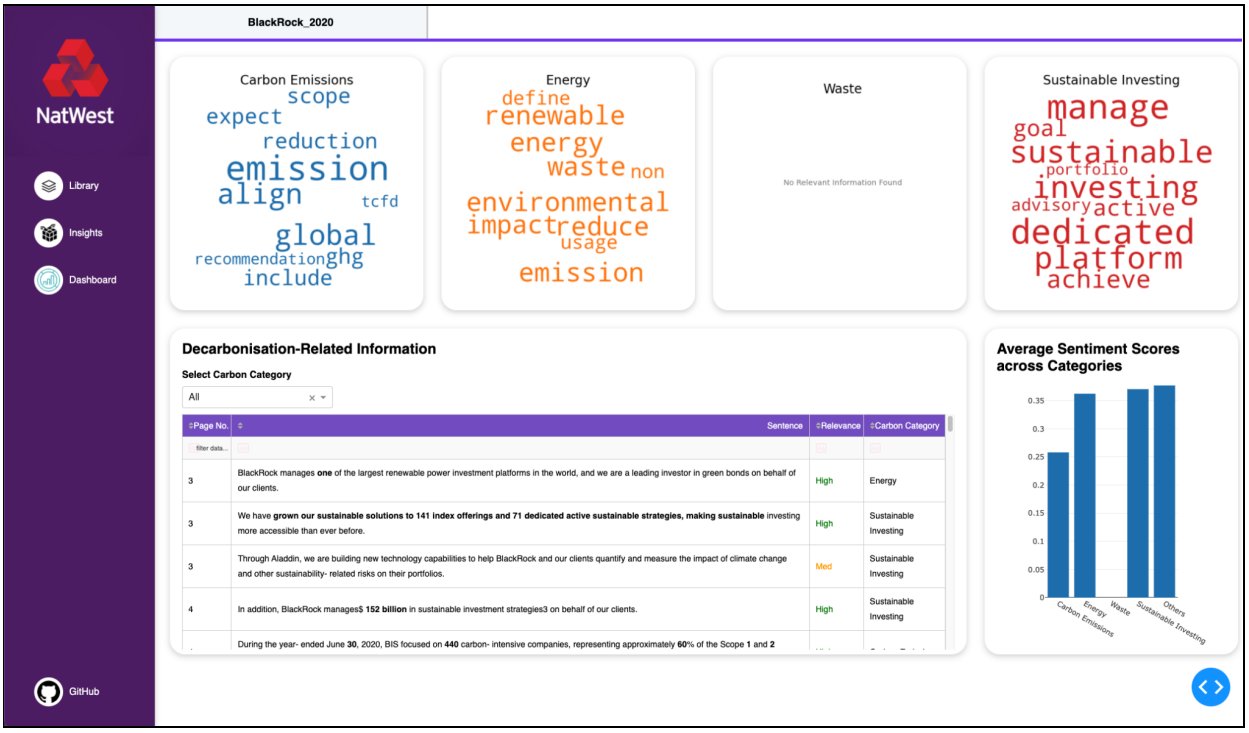


Fig 18. Main Dashboard Page

The last visualisation that we have included on this page is a bar chart of sentiment scores across all carbon classes, giving users a gauge of the company's overall sentiments about their achievements in the ESG sphere, especially in the environmental aspect. ESG sentiment analysis has been increasingly used by analysts as a quantifiable measure to guide investment decisions (alva, 2021). It can serve as a proxy for the actual performance of a company in these aspects. For example, if a company has a relatively higher sentiment score in the class *Energy*, it can potentially reflect the company's active transition to renewable energy investment in the report year while reducing its energy consumption.

9.4 Use Cases

With a holistic view of key decarbonisation related information, stakeholders of NatWest Markets can utilize the dashboard for different use cases to suit their needs, such as using it as a basis to assess climate-related risks for a specific firm, to make better-informed investment decisions. In summary, we believe that our solution offers several functionalities that can benefit NatWest Markets:

1. Ability to upload, analyse and store new PDF reports to add to our existing database.
2. Data extraction capabilities to supplement existing ESG data used by NatWest Markets.
3. Integration of an end-to-end text, chart and table extraction process to provide a complete picture of the decarbonisation progress and sustainable investing efforts by FIs.
4. Breakdown of 4 subcategories of the 'E' factor in ESG to capture companies' varying exposures to different environmental topics.

Even though the amount of ESG data by third-party providers has increased over the years, the lack of comprehensive ESG data extracted from corporate disclosures is still a problem firms face today due non-standardized reporting (Probert, 2021). Therefore, we hope that our solution has showcased the value in using a range of data driven techniques to extract decarbonisation information and sentiments, enabling NatWest Markets to be uniquely positioned to support its customers and investors in developing holistic sustainability strategies.

10 Functional Requirements

The step-by-step instructions to run the code and set up the codes can be found in the README.md on our github link.

10.1 Operating System

Our codes currently work for any MacOS machines that are running on intel processor chips. MacOS machines with Apple M1 processor chips are currently not supported due to bert as a service requiring tensorflow dependency versions of earlier than 1.15.0, which at present cannot be run on machines with the newer M1 chip.

10.2 Python Version

The unique tensorflow version dependency brought about by our use of the bert as a service package also means that we need to run our code in a Python 3.7.6 environment. This is because the installation of the tensorflow package via pip requires a pip version that is only compatible with Python versions lesser than 3.7.6.

10.3 Virtual Environment

Given the above Python and differing package requirements, we have generated a requirements.txt to ensure a smooth installation and handover process for Natwest. We believe that a conda environment is better suited for this project as compared to using a virtualenv. This is because for the former, we can easily specify the python version required when creating a conda environment but for the latter, we will need to point the virtualenv to the correct python compiler required. For the requirements.txt, we generated it from the conda environment we used and have converted it to a hybrid conda and pip compatible txt file as some of our required package binaries are unavailable on anaconda channels (e.g. bert-serving-server, bert-serving-client, camelot-py 0.10.1, etc). In order to make the handover process smooth, detailed instructions and terminal commands are presented in the Github ReadMe. Further details of the functional setups can be found in our Github ReadMe as well.

10.4 Disabling GPU compilation requirement for Detectron2

Released by Facebook AI Research, Detectron2 is a library that offers state-of-the-art detection and segmentation algorithms. As part of the tabular pipeline, Detectron2 was used and is embedded to enhance the accuracy of table detection. Due to the heavy computations involved for such CV tasks, Detectron2 requires a graphics processing unit (GPU) for higher efficiency. To disable the GPU requirement for users without GPU, users should navigate to the build.py file

and indicate “cpu” as the default settings instead. More details can be found in our GitHub ReadMe.

11 Future Extensions

Our project extensively explored approaches to extract required information to a satisfactorily extent, for visualisation on a developed dashboard. However, we acknowledge that there are still limitations to the different approaches used and they can be further extended in the future to obtain even more satisfactory results. The three main extensions we identify are mainly in the areas of Modeling, Dashboard, and Technology Stack.

11.1 Modeling

In terms of chart extraction, we believe that the 15 image properties as mentioned in Section 6.2 can definitely serve as features for machine learning models to predict whether the images obtained after the first round of filtering are charts that have been extracted correctly. This is because the bounding coordinates derived from *OpenCV* are not perfect, which causes some of our filtered charts to be surrounded by a significant amount of noise.

Secondly, in terms of text, we believe a reasonable extension for rule mining is to be able to summarise the main gist of the entire report in a paragraph. This is because currently, we are only working on sentence level extractions and substantial value can be brought to stakeholders if a high level summary is constructed using summarisers such as bert-extractive-summariser.[9]

11.2 Dashboard

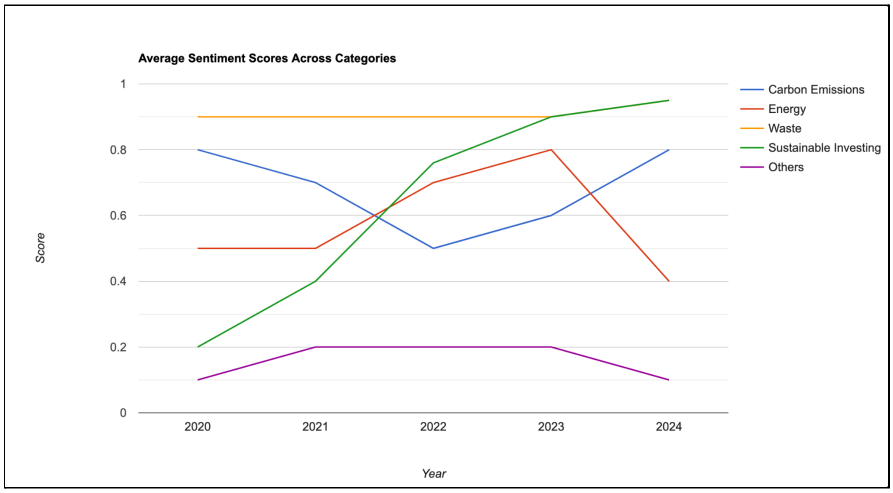


Fig 19. A sample visualisation of Sentiments Analysis Trendline we proposed

Currently, the Dashboard displays a bar chart of sentiment scores across all carbon classes at one time point according to the reporting year of the selected report. As advised by Dr. Shalinda, a useful visualisation to incorporate in the Dashboard view would be a trendline that displays the

[9] https://pypi.org/project/bert-extractive-summarizer/

sentiment scores of these carbon classes over time. One possible visualisation is shown in Fig 19. Such time series analysis is useful to see how the ESG performance of a firm has changed over time across the different carbon classes, and can further add value to our client's workflows. However, due to sustainability disclosures being a relatively new phenomenon and the lack of a global reporting standard, very limited number of companies have comprehensive disclosures across the years among our collated URLs. As a result, this feature was not implemented due to limited utility. However, as ESG reporting gains greater transparency in the coming years, we propose introducing this feature in the future to further value-add to the Dashboard view.

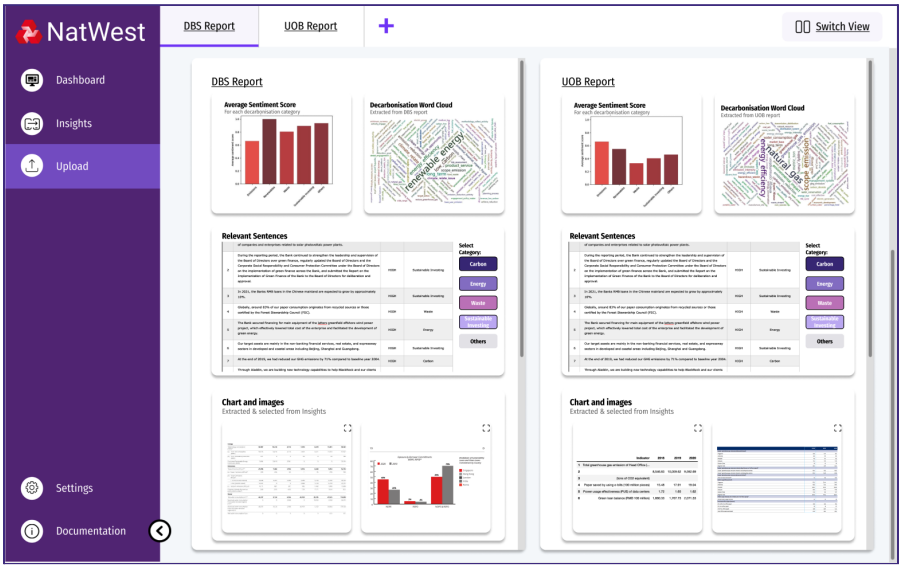


Fig 20. A sample visualisation of the multi-tab feature we proposed

Another extension we are proposing is the introduction of a multi-tab feature. Currently, the Dashboard interface allows users to only view insights of a single report at any one time. In view of this, a multi-tab user interface that makes it possible for users to open multiple views belonging to different sustainability reports concurrently as shown in Fig 20 can be introduced. With the multi-tab feature, users can easily toggle between different Dashboard views and carry out direct metric comparisons across firms to make a more informed decision, further easing the ESG-related workflows for stakeholders in NatWest Markets. However, with the non-uniformity in current ESG reporting standards across firms, it is challenging to find a standardised set of decarbonisation-related metrics for comparisons across multiple firms. Therefore, in view of this limitation, we have opted to delay the development of the multi-tab feature due to low utility.

11.3 Tech Stack

Currently, both the combined pipeline and dashboard are hosted on localhost servers due to the lack of access to hosted resources. For future extension, we believe it would be beneficial to host our end-to-end solution on cloud hosting services and make the application accessible to different users via cloud resources. This offers much more possibilities in terms of real-time updates across the different users when a new PDF URL is uploaded and appended to the shared database. Though we see high utility in cloud hosting, due to the lack of access to a hosted database and time, we were unfortunately unable to host our solution on cloud.

12 Key Takeaways

12.1 Functional Knowledge

Through extensive research on ESG frameworks, this project has deepened our understanding of the latest ESG investing trends in the financial sector and the importance of ESG reporting. With a growing interest in sustainable investing across the globe, our group is very fortunate to be able to work on a decarbonisation-related finance project with NatWest Markets to gain first-hand knowledge of how the organization helps clients to manage climate-related risks. During our data collection phase, we read through hundreds of financial reports across the 4 types of FIs and learnt about how companies embed ESG issues into their risk management framework, with the primary focus on the 'E' factor. From our consolidated findings, we have gained valuable insights into 3 main aspects - ESG quantification, measurement, and reporting, to allow investors to make better-informed investment decisions.

At present, five global organisations including CDP and SASB have set the frameworks for sustainability disclosures. However, the general observation from the reports we have seen is that there still exists non-uniformity in reporting. Therefore, we also recognize the need for a global set of reporting standards to achieve greater transparency and more structured data points to enable comparison and we look forward to these exciting developments.

12.2 Technical Knowledge

In terms of technical skills, one key takeaway would be the exposure to advanced CV libraries for the extraction of decarbonisation-related charts and tables from PDF reports. As none of us had any prior experience working with CV and PDF extractions, incorporating CV in our data extraction pipeline was extremely challenging but rewarding. Through the learning process, we get to research and experiment with state-of-the-art machine learning implementations such as *OpenCV*, *Detectron2* and *Camelot* to develop our pipeline that is capable of extracting the unstructured information pertaining to decarbonisation. With the completion of the pipeline, we learnt how to select and apply appropriate advanced libraries based on our use cases and the knowledge gained from the Business Analytics curriculum, and not limit ourselves to the traditional algorithms.

Another key takeaway would be learning how to tailor the general text preprocessing and natural language processing skills we have learnt to fit the requirements of the project. Common text preprocessing steps such as removal of numbers were not included as these numbers may possibly be ESG-related numerical metrics that our stakeholders are interested in. Different from other general-purpose text preprocessing pipelines, we also included a preliminary filter with a curated ESG keywords list in order to sieve out and preserve the pages with relevant decarbonisation-related content for subsequent charts and tables extraction.

Pertaining to the Dashboard deliverable, one advice given by Mr Gupta was that it is key to think and imagine a story on how the Dashboard would be used by the end users in order to create a dashboard that is truly useful for the stakeholders. With this advice in mind, we went through several revisions of the Dashboard layout as we strive to maximise the utility to NatWest Market. In particular, we focus on having a user-friendly interface as well as useful visualisations like the

word cloud, sentiment scoring to allow the users to gain a deeper insight into the ESG performance of companies. In this way, our dashboard allows the users to monitor and analyse ESG performance of a company in an interactive manner.

12.3 Professional Skills

Last but not least, the project has allowed us to gain experience in a professional setting where we get to develop our communications and interpersonal skills. Through our weekly catch-ups with Mr Gupta, we learnt to exhibit and maintain professionalism in our communications to better understand the needs and expectations of NatWest Markets for the project. Internally, we deepened our interpersonal skills for effective collaboration to ensure project deliverables are submitted timely. Moreover, with the worsening Covid-19 situation, we learnt to utilise digital meeting and collaboration tools to stay connected and keep one another updated on the current progress, which would be essential in the post-pandemic workplace. Externally, we learnt to work together with other teams working on NatWest's projects in order to speed up the data collection progress which would otherwise delay the timeline of the project as well as share advice with one another pertaining to the projects' deliverables. Finally, our weekly catch-ups with Dr. Shalinda and Dennis as well as the log submissions has also definitely brushed up on our oral and written communication skills as we learnt to summarise our individual and the team's progress concisely.

13 Conclusion

In conclusion, we believe that the work done in this project represents a crucial first step in allowing NatWest Markets to derive decarbonisation related information from unstructured PDF reports. With ESG becoming an increasing area of focus for companies, this efficient method leveraging machine learning and domain specific heuristics will definitely help NatWest save time and effort spent on looking through the ever growing number of ESG disclosure reports. Specifically, our project was able to explore and apply different state of the art NLP and Computer Vision approaches such as *Multi-Type-TSR*, *BERT-as-service* and *Detectron2* to accurately extract decarbonisation data from unstructured sources such as text, charts and tables. These approaches represent the latest advancements in the industry and we are happy to say that we have successfully applied them to a decarbonisation context and intuitively visualised these data on our developed Dashboard application. Furthermore, we took steps to deliver an end-to-end interface that is able to take in new PDF links, run them through our extraction pipeline, and store the extracted information in our database for future viewing.

Through interacting with Mr Gupta, we know that this project represents the initial stages of an ambitious ESG project for NatWest. Therefore, we are extremely humbled to be able to hand over our decarbonisation tailored models and approaches to contribute to their overall goals in informing their clientele about decarbonisation progress. With NatWest's keen focus in the area of decarbonisation, we are excited to see how NatWest's will further develop in the area of ESG machine learning applications.

Appendix

Appendix A: Database Schema

JSON structure for 1 sample FI


```
1:
  company:                "CMBC Capital"
  year:                   "2017"
  url:                    "http://en.cmbc.com.cn/up…/2017%20ESG%20REPORT.pdf"
  text_output:
    page:                 […]
    sentence:             […]
    relevance_prob:       […]
    carbon_class:         […]
    mined_text:           […]
  wordcloud_img_path:
    0:                    "data/dashboard_data/word…17_Carbon Emissions.png"
    1:                    "data/dashboard_data/word…Capital_2017_Energy.png"
    2:                    "data/dashboard_data/word…mages/NO_DATA_Waste.png"
    3:                    "data/dashboard_data/word…stainable Investing.png"
  sentiment_score:
      0:                  0.2933
      1:                  0.3715
      2:                  null
      3:                  0.8481
      4:                  0.4814
  table_keywords:
    46:                   […]
    47:                   […]
    48:                   […]
      49:                 []
  table_image_keywords:
    46:                   […]
    47:                   […]
    48:                   […]
      49:                 []
  table_images:
    46:                   […]
    47:                   […]
    48:                   […]
      49:                 []
  chart_images:
      46:                 []
      47:                 []
      49:                 []
  chart_images_keywords:
      46:                 []
      47:                 []
      49:                 []
```

JSON Fields Description

Key	Description	Comments
company	Name of company	
year	Year of report	
url	URL to PDF report	Primary Key
text_output[“page”]	Pages of relevance sentences	
text_output[“sentence”]	Text of relevant sentences	
text_output[“relevance_prob”]	Predicted probability scores of relevant sentences	
text_output[“carbon_class”]	Predicted carbon categories	
text_output[“mined_text”]	Text mined from relevant sentences using rule mining	
wordcloud_image_path	Path to wordcloud images generated for each carbon class that exists in relevant sentences	wordcloud_image_path[0] returns wordcloud path for wordcloud of carbon class = 0
sentiment_score	Sentiment scores generated for each carbon class that exists in relevant sentences	sentiment_score[0] returns sentiment score for carbon class = 0
table_keywords	Decarbonisation-related keywords extracted from the cleaned dataframe of metrics using tabular data extraction	[‘table_keywords’][‘page’] returns list of keywords corresponding to the cleaned table dataframe in the page of the pdf report
table_image_keywords	Decarbonisation-related keywords extracted from each table image extracted	[‘table_image_keywords’][‘page’] returns list of keywords corresponding to the table image in the page of the pdf report
table_images	Path to table images extracted from table detection	[‘table_images’][‘page’] returns a list of table imagepath for the page of the pdf report
chart_images	Path to chart images extracted from chart detection	[‘chart_images’][‘page’] returns a list of chart imagepath for the page of the pdf report

chart_images_keywords	Decarbonisation-related keywords extracted from each chart image extracted	['chart_images_keywords']['page'] returns list of keywords corresponding to the chart image in the page of the pdf report

Appendix B: List of Words and Criteria for Page Filter

relevant_terms_directFilter = ['metrics and targets', 'climate value-at-risk','Greenhouse Gas','decarbon', 'energy consumption','decarbonization','green','decarbonisation','carbon emissions','co2','fuel','power','waste management','environment','Carbon Intensity','green energy','emission','waste output','cdp','GHG emissions','Scope 2','energy','gas','emissions','electricity consumption','sustainable','business travel','cvar','carbon','sustainability','sustainability goals','global warming','water consumption','Scope 1','GHG','renewable','climate','TCFD','climate solutions','WACI','paper consumption','carbon intensity','environmental','carbon footprint','carbon pricing','net-zero','Scope 3']

relevant_terms_combinationA = ["emissions","exposure","carbon related","esg","sustainable","green","climate sensitive","impact investing", "investment framework", 'msci', 'ftse', 'responsible investing', 'responsible investment','transition']

relevant_terms_combinationB = ["portfolio","assets","AUM","investment","financing","ratings","revenue","bond","goal","insurance", "equity", "swap", "option", "portfolio holdings", "risk management",'financial products']

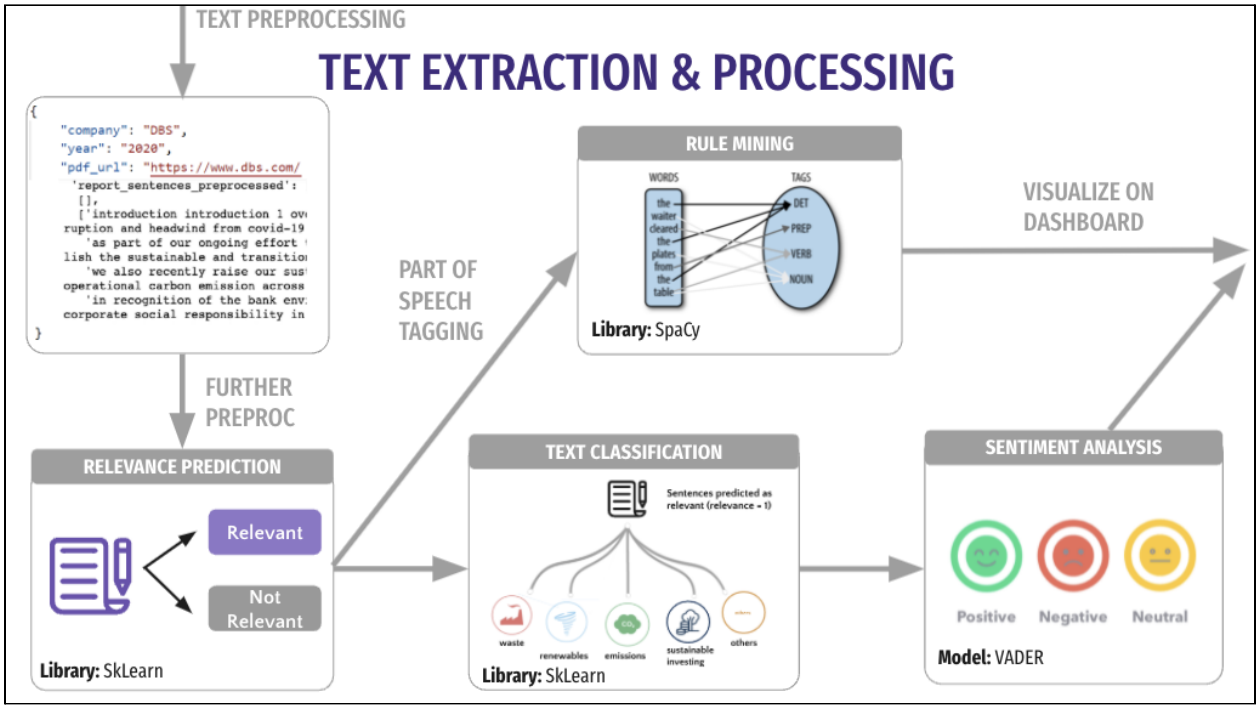
relevant_terms_combinationC = ["net zero","carbon footprint","CO2","carbon","oil","coal", "gas", "fossil fuel","green"]

<u>Criteria</u>

The above lists are all lemmatized and lowercase before using them to filter the report.

1. If the page contains at least 3 words from ***relevant_terms_directFilter,*** keep the page, else filter it out.
2. If the page contains any word from (***relevant_terms_combinationC*** *AND* ***relevant_terms_combinationA)*** *OR* ***(relevant_terms_combinationC*** *AND* ***relevant_terms_combinationB)****,* keep the page, else filter it out.

Appendix C: Text Extraction Pipeline



Appendix D: List of Relevant Sentences Collated for BERT Filtering

relevant_sentences = [

'In 2019, Citi financed $74 million of subordinate lien bonds that were certified green, given the projects environmental aspects.',

'In addition, our cogeneration plant, fueled by natural gas, will produce heat and electricity on-site, reducing the building's carbon footprint by 34 percent.',

'These efforts reduced energy consumption by more than 2,100 metric tons (mt) of carbon dioxide equivalents (CO2e) during the one-year challenge.',

'The companies in our equity portfolio emitted around 133 tonnes of CO2 -equivalents for every million US dollars of revenue.',

'The equity portfolio's carbon intensity was 9 percent below that of the benchmark index.',

'A total of 106 companies that produce certain types of weapon, tobacco or coal, or use coal for power production, are currently excluded from the fund',

'For public and private assets, excluding cash and non-equity derivatives as they were not reported in 2019, our year-overyear portfolio weighted average carbon intensity was reduced by approximately 23%.',

'Having met these targets, we have set new, more ambitious ones: to reduce the Fund's emissions intensity by 40% and fossil fuel reserves by 80% by 2025.',

'The carbon footprint of the non-listed companies was 0.6 tCO_2e per million SEK invested',
'Energy consumption and carbon emissions per unit area were 149 kWh/ m² and 0.037 tCO_2e/m^2, which means a reduction of 9 per cent and 12 per cent, respectively.',
'The carbon intensity (CO2 equivalent tons per million yen of sales) of GPIF's equity and corporate bond portfolio decreased by 15.3%, from 2.29 tons to 1.94 tons, in the space of a year.'
'Based on our percentage holdings in each company, the total emissions of the equity portfolio were 108 million tonnes of CO2 - equivalents in 2019.',
'The carbon footprint of the companies in our equity portfolio',
'The companies in our equity portfolio emitted around 156 tonnes of CO2 -equivalents for every million US dollars (USD) of revenue.'
'The carbon intensity of the companies in the equity portfolio and the benchmark index decreased by 16 and 17 percent respectively from 2018 to 2019.',
'We are focused on supporting the goal of net zero greenhouse gas emissions by 2050, in line with global efforts to limit warming to 1.5°C. ',
'Quantitative target for ESG-themed investments and finance of ¥700 billion ',
'Commit to reduce investment carbon footprint by',
'esg investing', 'green bonds', 'Green Investment target', 'Achieve 100% renewable electricity by 2025'
]

Appendix E: Relevance Model Class Distributions

Class distributions before oversampling was conducted.

Training Data

Class	Size	%
0	2867	91.2
1	276	**8.8**

Validation Data

Class	Size	%
0	965	92.1
1	83	**7.9**

Test Data

Class	Size	%
0	943	90.0
1	105	**10.0**

Appendix F: Relevance Model Results

Results reported are on the validation set. Best base models are highlighted in lilac while the overall best model we selected is highlighted in purple.

Model	Minority Class Method	Vectorizer / Combination	Clean / Raw Text	Weighted F1	Accuracy	Weighted F1 (Class 1)	Weighted F1 (Class 0)
Logistic Regression	SMOTE	BoW	Raw	0.87	0.86	0.32	0.92
		TFIDF	Clean	0.91	0.91	0.50	0.95
		BERT Word Embeddings	-	0.90	0.87	0.48	0.93
	ROS	BoW	Raw	0.91	0.91	0.46	0.95
		TFIDF	Clean	0.90	0.90	0.48	0.94
		BERT Word Embeddings	-	0.89	0.85	0.46	0.93
Naive Bayes	SMOTE	BoW	Raw	0.90	0.89	0.43	0.94
		TFIDF	Clean	0.89	0.88	0.42	0.93
	ROS	BoW	Clean	0.89	0.88	0.43	0.93
		TFIDF	Raw	0.88	0.86	0.41	0.92
	-	BERT Word Embeddings	-	-	-	-	-
Support Vector Machine	SMOTE	BoW	Raw	0.89	0.90	0.22	0.95
		TFIDF	Clean	0.91	0.91	0.45	0.95
		BERT Word Embeddings	-	0.91	0.90	0.52	0.93
	ROS	BoW	Clean	0.90	0.88	0.45	0.94
		TFIDF	Raw	0.92	0.92	0.48	0.95
		BERT Word Embeddings	-	0.91	0.91	0.50	0.91
Random Forest	SMOTE	BoW	Raw	0.87	0.85	0.28	0.92
		TFIDF	Clean	0.92	0.92	0.49	0.96
	ROS	BoW	Raw	0.92	0.92	0.47	0.95
		TFIDF	Raw	0.92	0.92	0.48	0.96
	-	BERT Word Embeddings	-	-	-	-	-
Voting Ensemble	SMOTE	Soft	-	0.91	0.91	0.48	0.95
		Hard	-	0.91	0.91	0.50	0.95
Stacking Model 1	SMOTE	LR,SVM,NB,RF, LR_BERT,SVM_BERT, SVM Meta Model	-	0.91	0.91	0.51	0.93
Stacking Model 2	**SMOTE**	**LR,SVM,NB,RF, LR_BERT, SVM Meta Model**	**-**	**0.92**	**0.92**	**0.53**	**0.92**

Appendix G: Text Classification Results

Results reported are on the validation set. Best base models are highlighted in lilac while the overall best model we selected is highlighted in purple.

Model	Vectorization	Clean / Raw Text	Weighted F1	Accuracy	Weighted F1 (Class 0)	Weighted F1 (Class 1)	Weighted F1 (Class 2)	Weighted F1 (Class 3)	Weighted F1 (Class 4)
Log Reg	BoW	Clean	0.627	0.641	0.761	0.788	0.400	0.462	0.516
		Raw	0.636	0.641	0.765	0.824	0.286	0.486	0.516
	TFIDF	Clean	0.660	0.685	0.822	0.824	0.500	0.524	0.444
		Raw	0.630	0.652	0.812	0.800	0.500	0.512	0.345
NB	BoW	Clean	0.597	0.609	0.754	0.667	0.444	0.465	0.467
		Raw	0.620	0.641	0.836	0.778	0.444	0.476	0.333
	TFIDF	Clean	0.621	0.641	0.806	0.765	0.333	0.474	0.429
		Raw	0.608	0.641	0.789	0.769	0.444	0.526	0.296
SVM	BoW	Clean	0.638	0.641	0.800	0.800	0.167	0.486	0.514
		Raw	0.606	0.620	0.758	0.857	0.364	0.410	0.424
	TFIDF	Clean	0.652	0.663	0.789	0.759	0.400	0.524	0.541
		Raw	0.666	0.674	0.824	0.759	0.333	0.537	0.550
RF	BoW	Clean	0.703	0.707	0.806	0.788	0.400	0.622	0.621
		Raw	0.701	0.696	0.746	0.824	0.308	0.667	0.645
	TFIDF	Clean	0.700	0.717	0.817	0.824	0.571	0.636	0.500
		Raw	0.681	0.696	0.806	0.811	0.667	0.605	0.452
Catboost	BoW	Clean	0.648	0.652	0.733	0.778	0.667	0.514	0.553
		Raw	0.652	0.652	0.719	0.778	0.333	0.600	0.563
	TFIDF	Clean	0.674	0.674	0.781	0.848	0.667	0.564	0.476
		Raw	0.666	0.674	0.781	0.778	0.571	0.565	0.516
Word Filter	-	Clean	0.610	0.610	0.710	0.750	0.460	0.580	0.390
		Raw	0.610	0.610	0.700	0.760	0.480	0.560	0.430
Voting Ensemble	**Modified Soft**	-	**0.701**	**0.710**	**0.820**	**0.890**	**0.400**	**0.590**	**0.490**
	Hard		0.690	0.710	0.830	0.820	0.500	0.530	0.560

Appendix H: Rule Mining Results

Results reported on our labelled ground zero truth dataset. The best selection of parameters is highlighted in lilac.

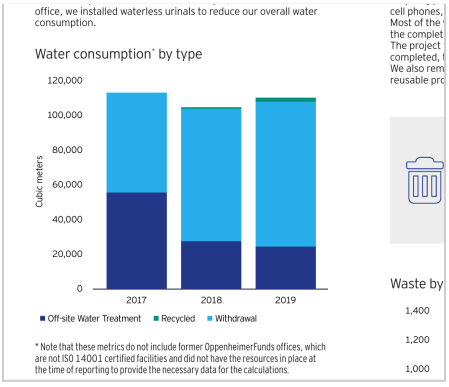
Excluded Fine Grain Verb POS	Lemma Match Accuracy	Non-Lemma Match Accuracy
None	72.7%	72.9%
VBP	77.1%	74.3%
VB	65.4%	73.0%
VBP, VB	67.1%	74.5%

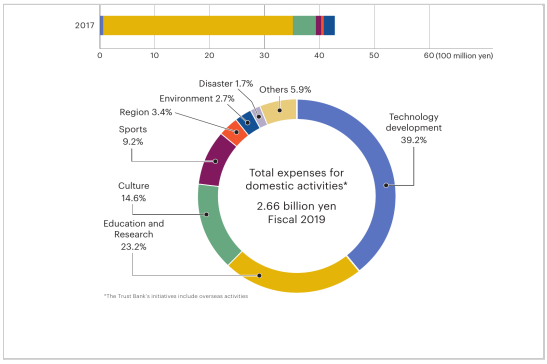
Appendix I: Custom VADER Dictionary

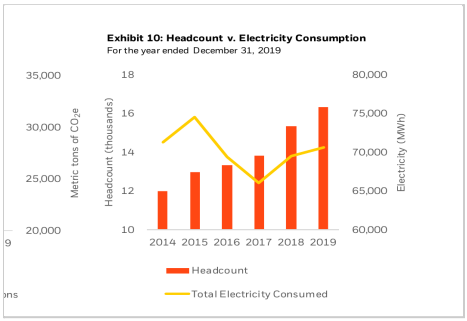
Domain specific words used to update the VADER dictionary

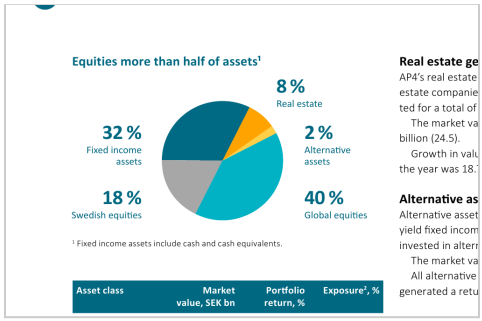
```
update_lexicon_esg =
{'initiate': 1, 'initiatives':1, 'energy-saving': 1, 'solar': 1, 'transition': 1, 'low-carbon': 1, 'reduction': 1,
 'reducing': 1, 'achieve': 1, 'achieved': 1, 'achievement':1, 'new': 1, 'reduce':1, 'waste':0, 'energy':0, 'wind':1,
 'decrease':1, 'increase':0, 'asset':0, 'assets':0, 'green':1, 'recycle':1, 'recycled':1, 'recycling':1 }
```

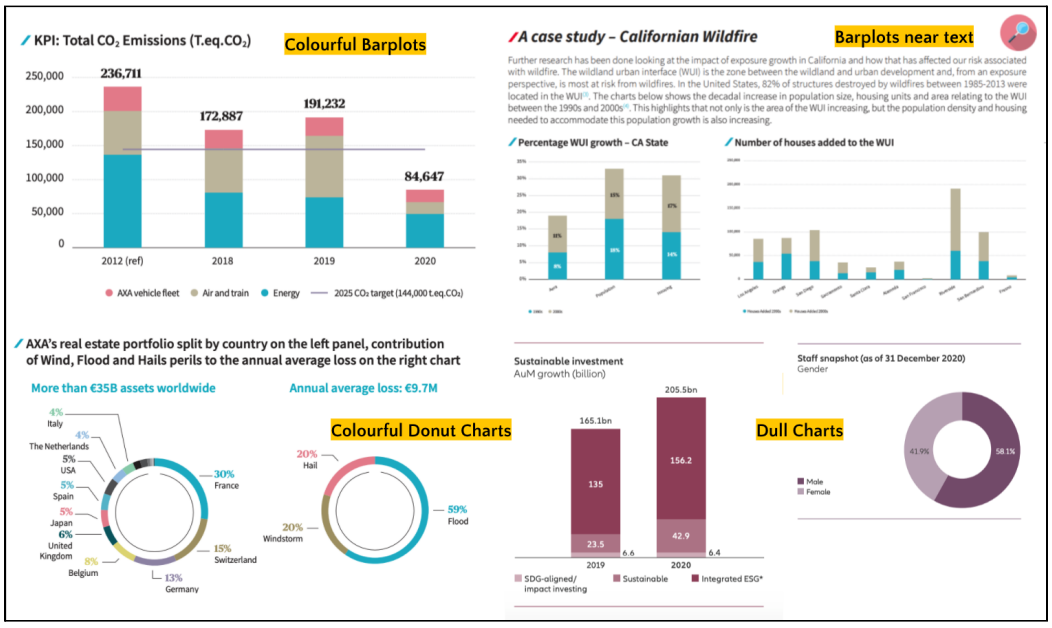
Appendix J: Examples of Chart and Graph Images











Appendix K: Features Generated for Chart Detection

Key Metrics		Description	Filter Rationale
Color Property of an Image			
1	Color Dilation (dr)	Total number of colored pixels in an image calculated from using adaptive image thresholding to detect colors and tabulate dilation results. Used adaptive threshold gaussian as optimal threshold compared to about 5 other combinations of image thresholding in OpenCv package.	Filter images with little colors as most graphs usually have a lot of colors.
2	White pix	Total number of white pixels in an image	Filter away images with too much white space indicating that image is a possible white space image.
3	Black pix	Total number of black pixels in an image	
4	Black to White pixels Ratio (BW ratio)	Black pixels compared to white pixels in an image	
Text Content of an Image			
5	Keywords	Total count of unique keywords.	Filter only relevant charts with decarbonisation-related text.
6	Total length of string (total_len)	Total occurrence of string value present in the image.	Filter away images with too much text
7	Total Text count (textonly_len)	Total occurrence of a text present in the image (exclude punctuations)	
8	Text to String Ratio (tt_ratio)	Total occurrence of text divided by total occurrence of string in the image (exclude punctuations)	
8	ta_ratio	Total occurrence of text divided by total area of image (exclude punctuations)	
Numerical Content of an Image			
10	Total Number count (numonly_len)	Total occurrence of numbers present in the image.	Filter away images with too little numbers.
11	Number to Area Ratio (na_ratio)	Total occurrence of number divided by total area of image (exclude punctuations)	
12	Number to Text Ratio (nt_ ratio)	Total occurrence of numbers divided by total occurrence of string in the image (exclude	

		punctuations)	
Dimensions of an Image			
13	Height	Height of image	Filter away images with very small height value
14	Width	Width of image	Filter away images with very small width value
15	Area	Area of image	Filter away images with very small area value

Appendix L: Snapshot of Features Generated in Excel

Column1	source	actual	dr	total_len	keywords	textonly_len	numonly_len	tt_ratio	nt_ratio	height	width	channels	area	ta_ratio	na_ratio	white_pix	black_pix	wb_ratio	bw_ratio
40	ChartExtracti	WRONG	14	4	0	4	0	1	0	467	1537	3	717779	0.56	0	42188	675591	0.06	16.01
156	ChartExtracti	WRONG	15	25	0	25	1	1.04	0.04	467	1534	3	716378	3.63	0.14	43233	673145	0.06	15.57
114	ChartExtracti	WRONG	10	33	0	25	3	0.91	0.15	467	1546	3	721982	4.16	0.69	29847	692135	0.04	23.19
116	ChartExtracti	WRONG	13	13	0	12	2	1	0.15	467	1546	3	721982	1.8	0.28	36923	685059	0.05	18.55
172	ChartExtracti	WRONG	9	25	0	22	0	1	0	468	1407	3	658476	3.8	0	23979	634497	0.04	26.46
138	ChartExtracti	WRONG	24	116	2	82	3	0.97	0.07	667	1654	3	1103218	10.15	0.73	107662	995556	0.11	9.25
179	ChartExtracti	WRONG	12	13	0	13	1	1	0.08	468	1546	3	723528	1.8	0.14	35689	687839	0.05	19.27
149	ChartExtracti	WRONG	21	92	4	39	33	0.58	0.52	759	1500	3	1138500	4.66	4.22	94423	1044077	0.09	11.06
162	ChartExtracti	WRONG	10	24	0	21	2	0.96	0.17	468	1597	3	747396	3.08	0.54	31937	715459	0.04	22.4
63	ChartExtracti	WRONG	14	14	0	12	2	0.86	0.14	471	1537	3	723927	1.66	0.28	40971	682956	0.06	16.67
158	ChartExtracti	WRONG	16	16	0	15	0	1	0	512	1654	3	846848	1.89	0	53499	793349	0.07	14.83
188	ChartExtracti	WRONG	19	10	0	9	1	0.9	0.1	518	1654	3	856772	1.05	0.12	66195	790577	0.08	11.94
67	ChartExtracti	WRONG	11	25	0	21	5	0.88	0.24	521	1654	3	861734	2.55	0.7	38152	823582	0.05	21.59
95	ChartExtracti	WRONG	20	168	7	85	35	0.89	0.27	964	1654	3	1594456	9.34	2.88	129378	1465078	0.09	11.32
164	ChartExtracti	WRONG	9	29	0	23	3	0.9	0.1	547	1342	3	734074	3.54	0.41	28486	705588	0.04	24.77
45	ChartExtracti	WRONG	13	5	0	5	0	1	0	555	1042	3	578310	0.86	0	31208	547102	0.06	17.53
30	ChartExtracti	WRONG	5	4	0	3	1	0.75	0.25	555	1573	3	873015	0.34	0.11	20393	852622	0.02	41.81
33	ChartExtracti	WRONG	5	4	0	4	0	1	0	560	1420	3	795200	0.5	0	18207	776993	0.02	42.68
46	ChartExtracti	WRONG	13	9	0	8	1	0.89	0.11	572	1654	3	946088	0.85	0.11	51263	894825	0.06	17.46
65	ChartExtracti	WRONG	17	20	0	18	1	0.95	0.05	580	959	3	556220	3.42	0.18	37383	518837	0.07	13.88
111	ChartExtracti	WRONG	13	37	0	22	7	0.81	0.24	590	1654	3	975860	3.07	0.92	52027	923833	0.06	17.76
147	ChartExtracti	WRONG	11	65	0	52	1	0.98	0.05	595	1510	3	898450	7.12	0.33	38785	859665	0.05	22.16
135	ChartExtracti	WRONG	10	16	0	13	2	0.81	0.19	595	972	3	578340	2.25	0.52	24887	553453	0.04	22.24
154	ChartExtracti	WRONG	6	5	0	5	0	1	0	596	934	3	556664	0.9	0	13604	543060	0.03	39.92
160	ChartExtracti	WRONG	6	9	0	9	0	1	0	596	1680	3	1001280	0.9	0	25990	975290	0.03	37.53
72	ChartExtracti	WRONG	14	46	0	40	1	1	0.04	598	1147	3	685906	6.71	0.29	38736	647170	0.06	16.71
58	ChartExtracti	WRONG	15	38	0	32	2	0.92	0.08	598	1207	3	721786	4.85	0.42	44499	677287	0.07	15.22
94	ChartExtracti	WRONG	13	28	0	25	1	1	0.04	600	1013	3	607800	4.61	0.16	32883	574917	0.06	17.48
132	ChartExtracti	WRONG	12	27	0	22	3	0.89	0.11	601	1017	3	611217	3.93	0.49	28914	582303	0.05	20.14
66	ChartExtracti	WRONG	12	32	0	24	3	0.88	0.13	601	1871	3	1124471	2.49	0.36	56045	1068426	0.05	19.06
181	ChartExtracti	WRONG	15	51	0	38	7	0.84	0.16	601	1027	3	617227	6.97	1.3	37617	579610	0.06	15.41
157	ChartExtracti	WRONG	4	9	0	9	0	1	0	602	1489	3	896378	1	0	17325	879053	0.02	50.74
192	ChartExtracti	WRONG	20	71	3	51	8	1.04	0.15	678	1500	3	1017000	7.28	1.08	81804	935196	0.09	11.43
55	ChartExtracti	WRONG	4	5	0	5	0	1	0	608	933	3	567264	0.88	0	10755	556509	0.02	51.74
68	ChartExtracti	WRONG	6	9	0	9	0	1	0	608	1681	3	1022048	0.88	0	25644	996404	0.03	38.86

Appendix M: List of Relevant Decarbonisation Keywords for Charts Pipeline

keywords_dictionary = {'carbon', 'ghg', 'emission',
'emissions', "scope", "WACI", "net-zero",
'energy', 'water', 'waste', 'coal', 'power', 'green', 'paper', 'consumption', 'renewable',
'breakdown', 'loans', 'tonnes', 'tons', 'kWh', 'kg', 'kilogram', 'kilowatt hour',
'gigajoules', 'GJ', 'litre', 'liter', 'CO2e', 'tCO', 't CO', 'MWh', 'megawatt hour',
'%', 'cubic metres', 'per employee', 'm3', 'co2', 'o2', 'million', 'total', 'trillion', 'set' }

Appendix N: List of Relevant Decarbonisation Keywords & Units for Table Pipeline

The list of common keywords and their units of measurement collated through reading 50 reports across all FI types.

ESG_DICTIONARY = ['ghg', 'scope 1',
'scope 2', 'scope 3', 'energy',
'paper', 'green bonds', 'renewable energy',
'water', 'carbon intensity',
'carbon emissions', 'waste',
'electricity',
'weighted average carbon intensity', 'WACI']

ESG_UNITS = ['tonnes', 'tons', 'kWh', 'kilogram', 'kilowatt hour',
'gigajoules', 'GJ', 'litre', 'liter', 'CO2e', 'tCO', 't CO', 'MWh',
'megawatt hour', 'GWh', 'gigawatt hour',
'cubic metres', 'cm3', 'm3', 'per employee', 'ream', 'quire', 'sheet', 'bundle', 'bale']

Appendix O: Testing Set for Chart and Tabular Pipeline Evaluation

Company	Year	Tables	Charts
UBS Asset Management (Asset Manager)	2019	Semi-bordered tables Single-row tables	Single-coloured donut charts Multi-colored bar charts
UOB Asset Management (Asset Manager)	2020	Borderless tables Single-row tables	Multi-coloured figures
Allianz Global Investors (Asset Manager)	2020	Borderless tables Far apart columns	False positive sample
China Merchants Bank (Asian Bank)	2020	Bordered tables Single-rows tables Multi-rows tables	False positive sample
OCBC (Asian Bank)	2020	Borderless tables Near columns	Multi-coloured pie charts
Asahi Life (Insurance)	2020	False positive sample	False positive sample
AXA Insurance (Insurance)	2021	Single-row tables Multi-row tables Borderless tables	Single-coloured bar charts Single-coloured line charts Multi-coloured bar charts Multi-coloured line charts

			Multi-coloured donut charts
Government Pension Fund (Pension Fund)	2020	Semi-bordered tables Far apart columns	Single-coloured bar charts Multi-coloured bar charts
AP Fonden 1 (Pension Fund)	2018	Multi-lines headers Bordered tables	Single-coloured bar charts Multi-coloured bar charts Multi-coloured donut charts

Appendix P: Definition and Formula of Precision and Recall

Chart Evaluation Metrics and Definition

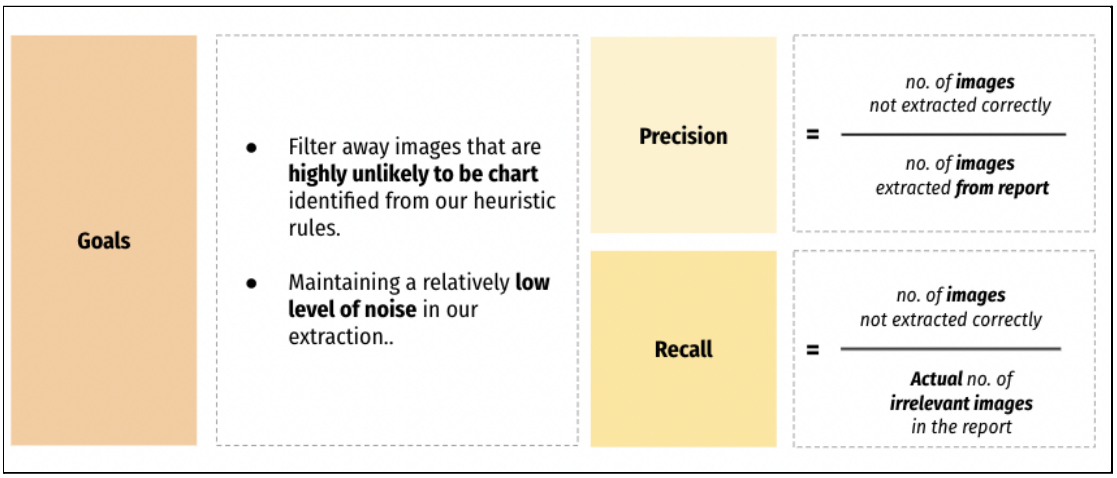
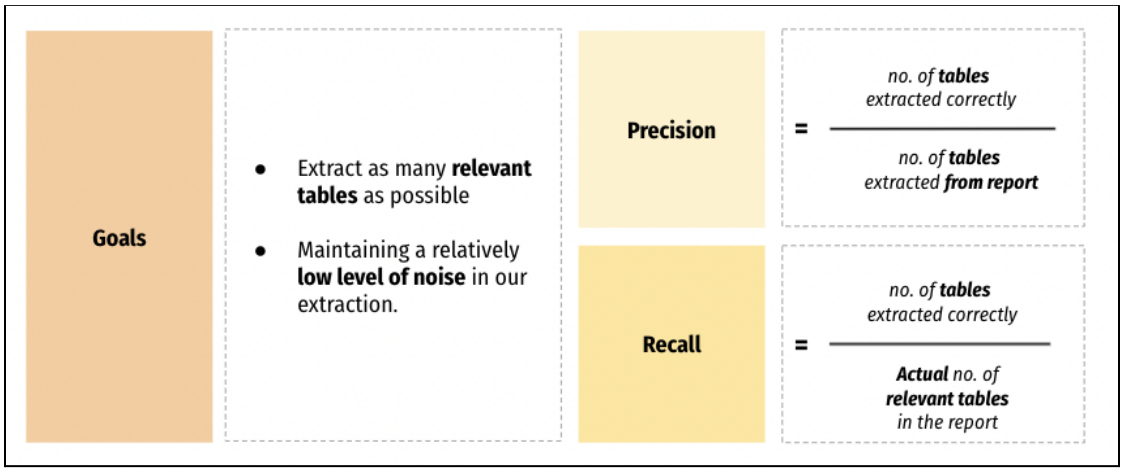


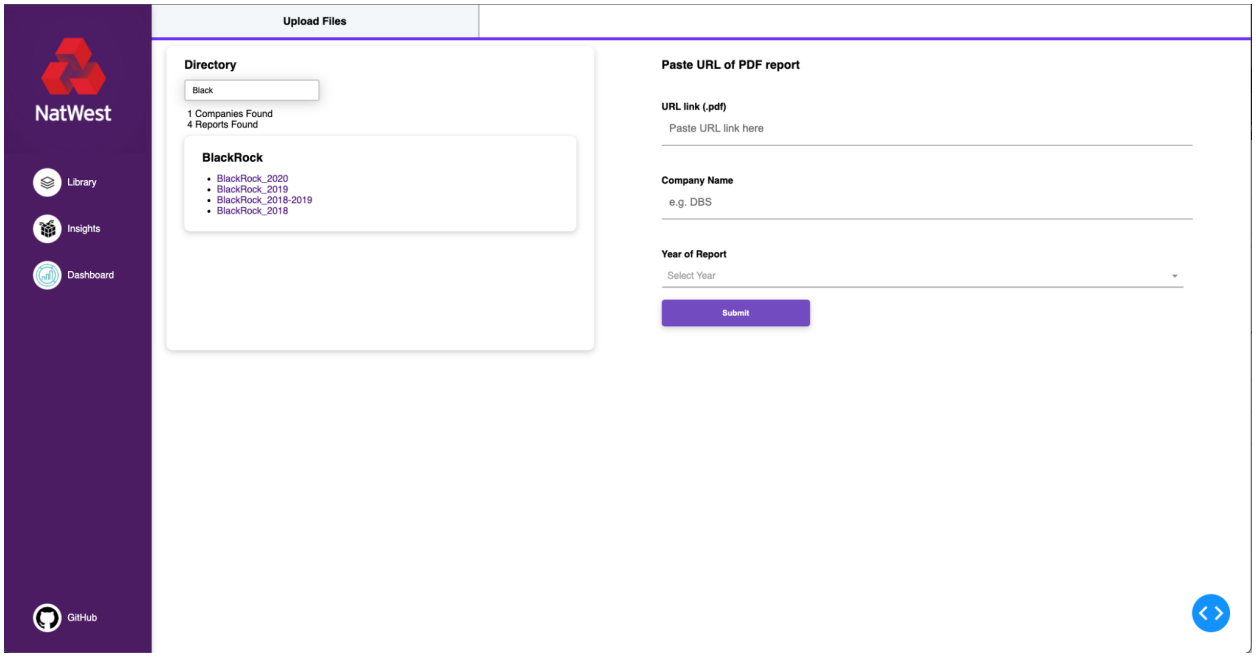
Table Evaluation Metrics and Definition



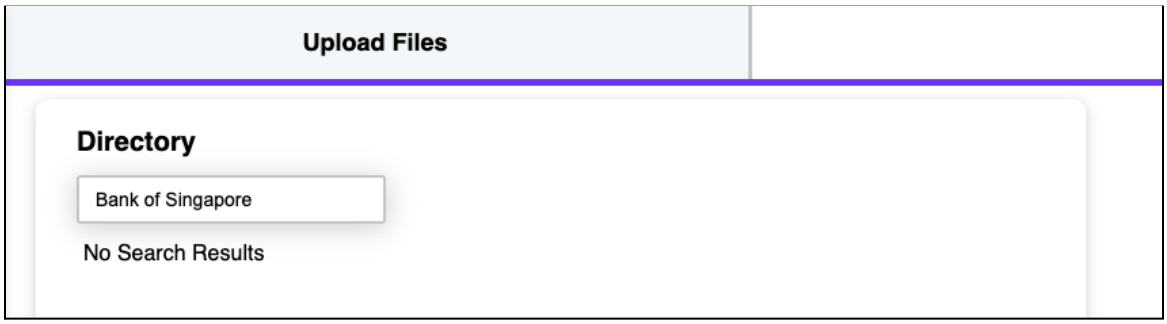
Appendix Q: Dashboard User Guide

Library Page:

1. At the landing page, users can use the search bar in the directory (below the "**Directory**" header) to do a company name search on the reports in our database. Our current database comprises a sizable number of processed sustainability reports thus for users to easily find reports for a company, we have grouped the reports on company level. To view the extracted data for a company and year, they can simply click on the desired report and it will bring them to the "Insights" page for that selected report.



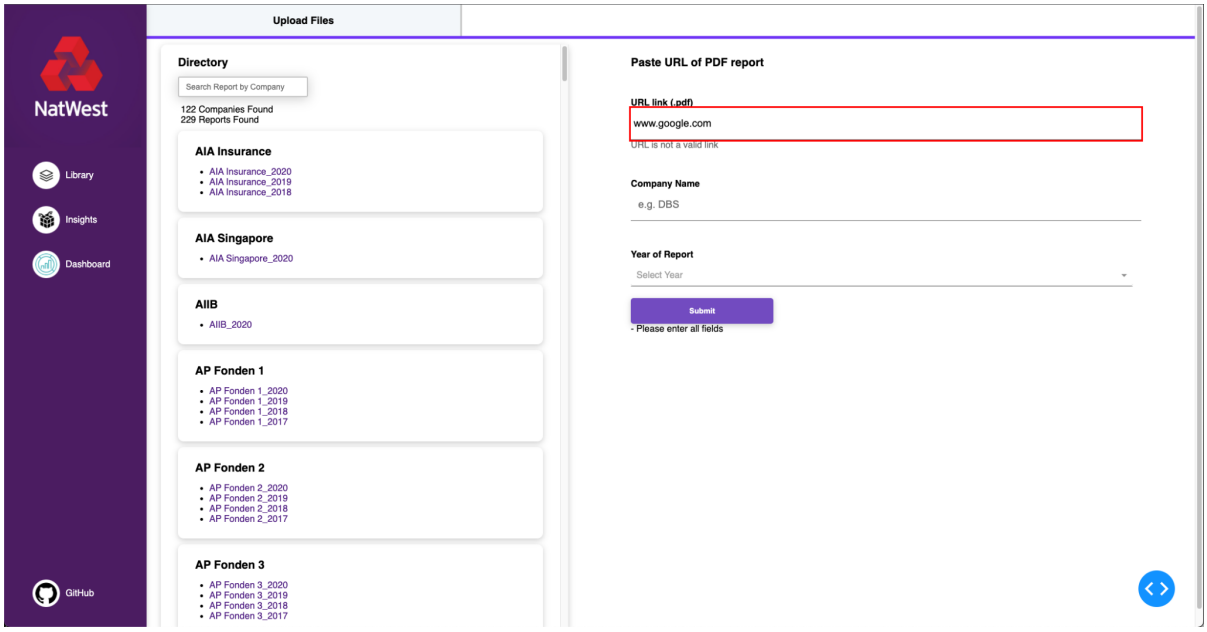
In a scenario whereby a user searches for a company that does not exist in our database, the following output will be shown in the directory.



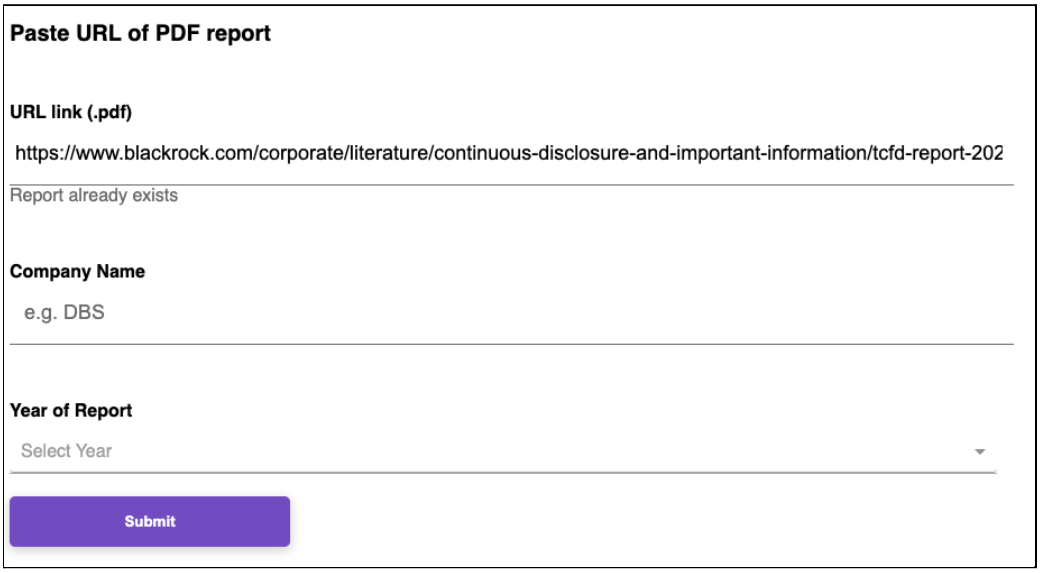
2. Users can also input a new URL for a PDF report that does not exist in our database. Validation checks will be conducted on the input fields to check if (1) the report URL is a

valid URL to a PDF report, (2) the report URL exists in our database and (3) if all required fields, company name and year of report are filled. Should any of these 3 criterias not be satisfied, the user will not be able to click submit to trigger the data extraction pipeline to generate insights from the required URL.

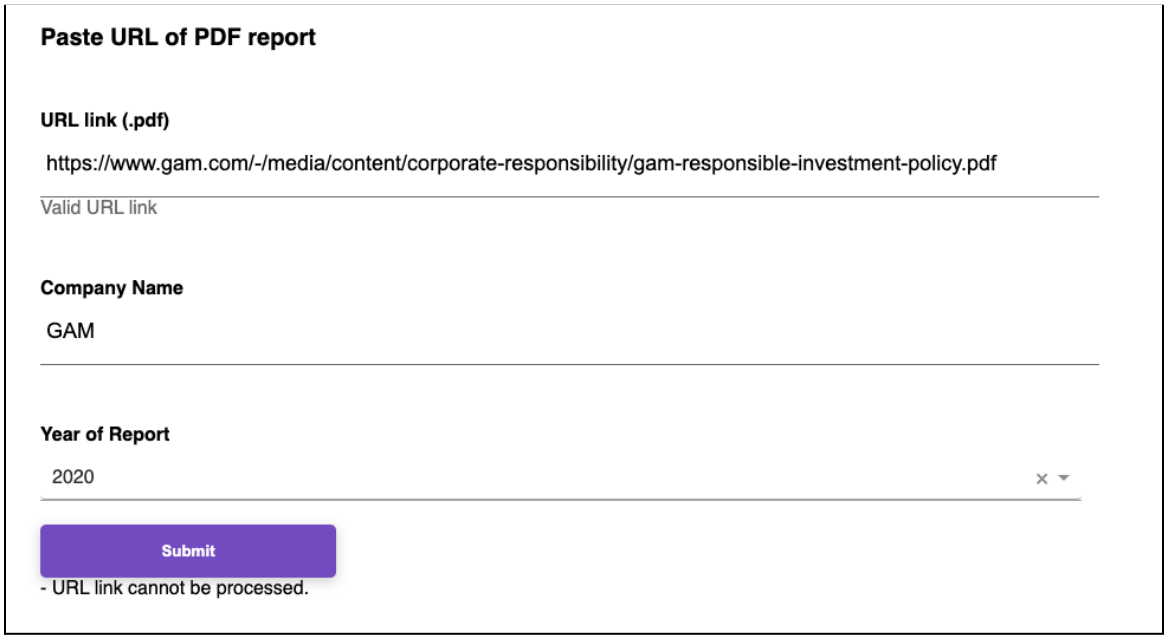
(1 & 3) When the report URL is not valid and not all fields are filled.



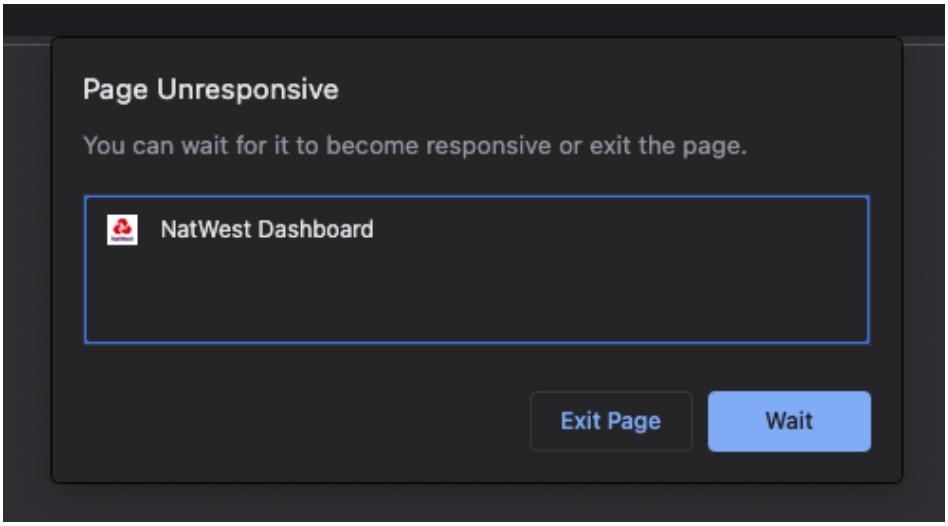
(2) The report URL exists in our database. This is to ensure that duplicated report URLs already processed and stored in our database cannot be submitted, saving time and computational resources. Users should expect a “Report already exists” message on the landing page, and will not be allowed to submit and trigger the pipeline.



3. For a PDF URL link that is valid and satisfies all the validation checks in 2, there may also be occasions where we don't store and visualise the new PDF URL link. This is because apart from checking whether input fields are valid, we also check whether the report will return any decarbonisation-related textual information. We coded our pipeline in such a way that exceptions will be thrown if we are unable to parse the PDF or no relevant text is extracted by our pipeline. This is because without relevant text information, there will not be any visualisations to be displayed on the dashboard and thus we see no value in processing and adding this PDF into our database. Our dashboard responds to the exception thrown and alerts users of this by displaying a "URL link cannot be processed" message.

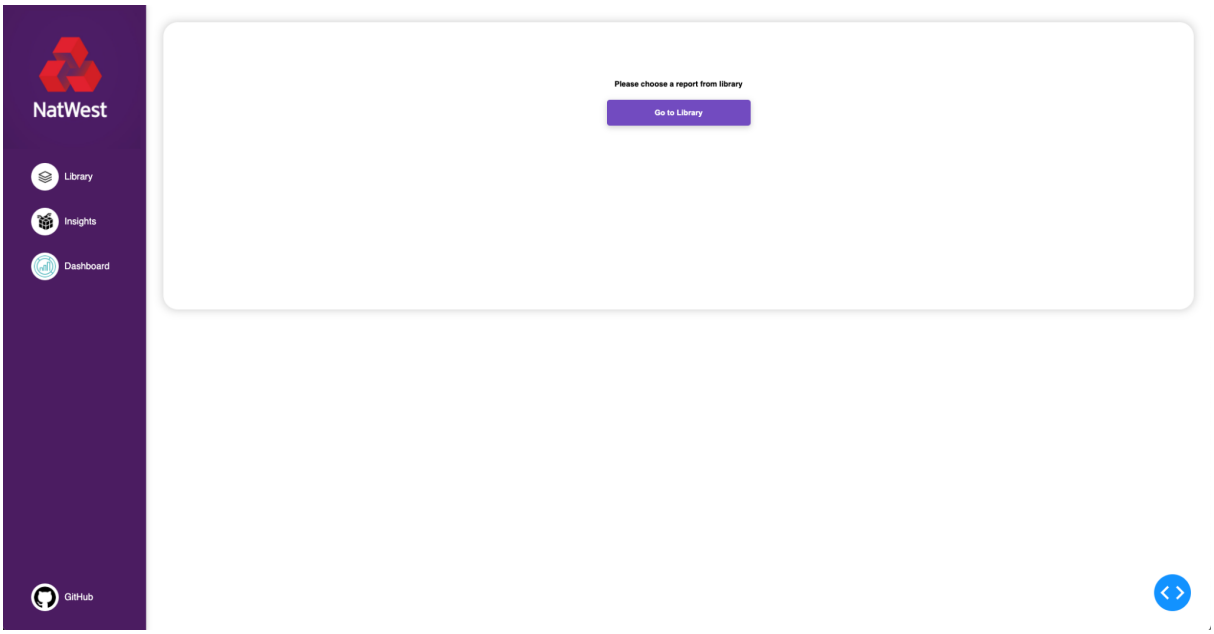


4. Upon successful submission of a new PDF URL link, the link will be passed into our combined pipeline for respective text, chart and table information extraction. The process takes about 8-15 mins depending on the size of the report (number of pages) and the number of tables/charts available for extraction. Upon successfully running the pipeline and saving the results in our database, the user will be brought directly to the "Insights" Page where they can view the extracted tables/charts. They can also view the word clouds, relevant sentences and sentiment scores in the "Dashboard" Page.
5. There might be occasions when after the combined pipeline has successfully added the new extracted information for the new report into the database, the application will crash due to the page becoming unresponsive. In such situations, the user will have to re-run the dash app in order to view the new report outputs.

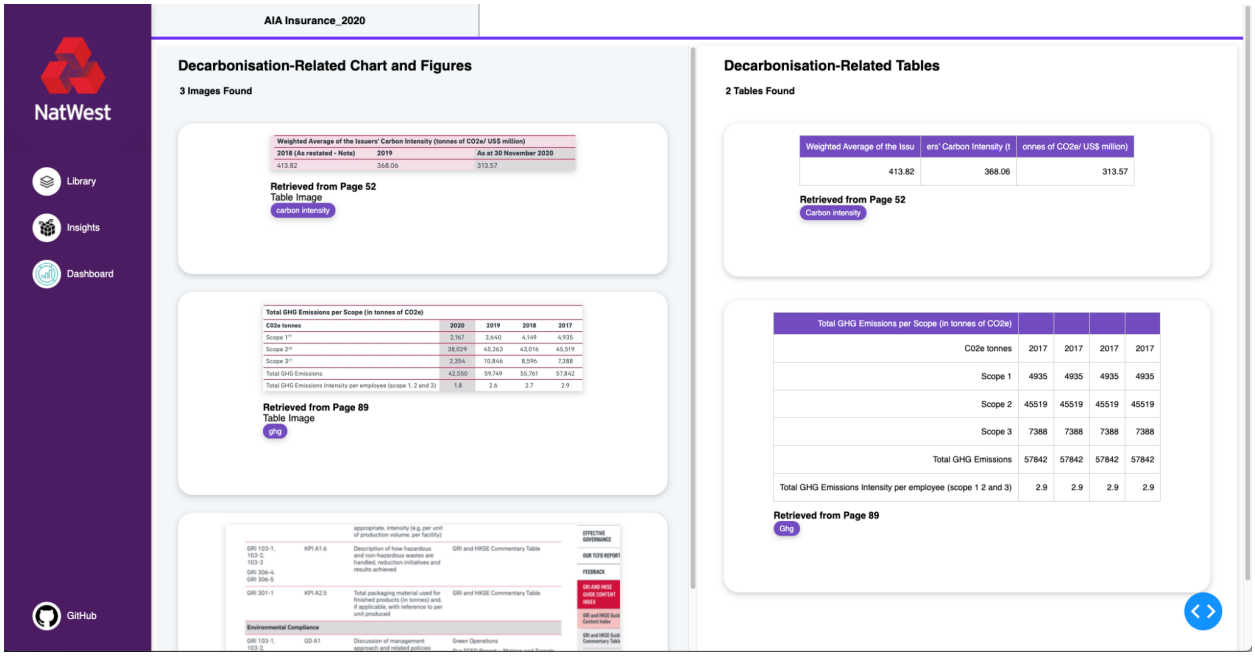


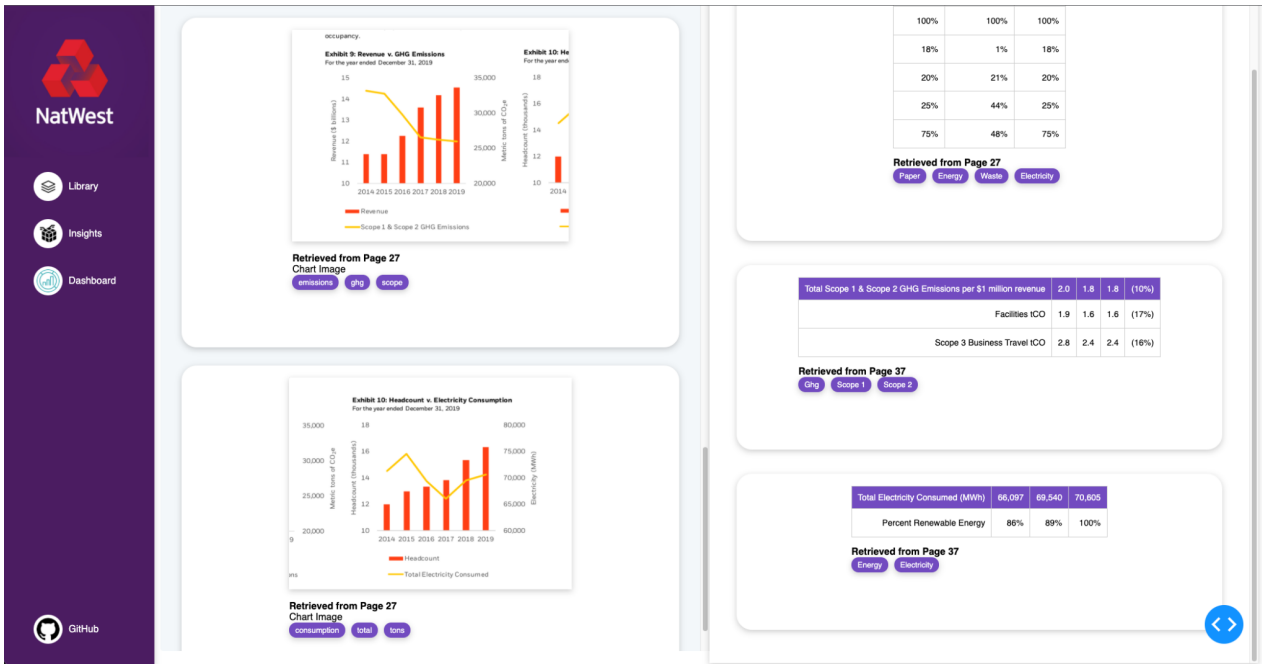
Insights Page:

1. To load the insights page, a report has to be first chosen from the "Library". If a report has yet to be loaded, the page below will be shown and users will be directed to the "Library" page.

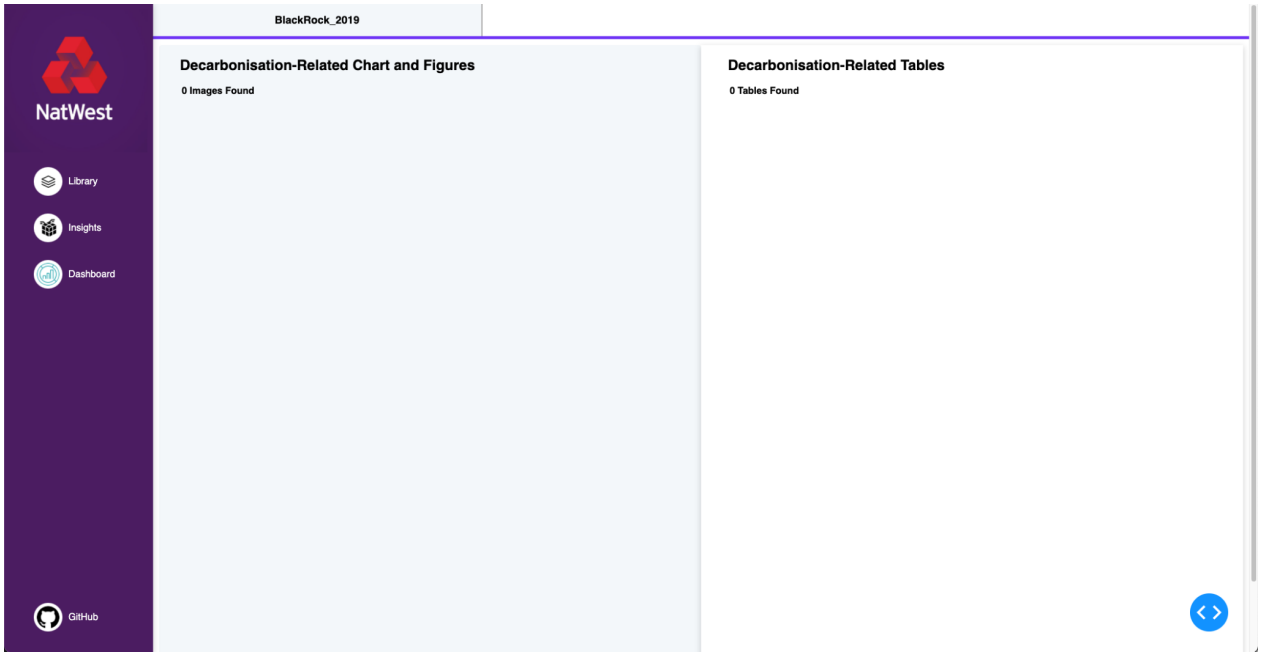


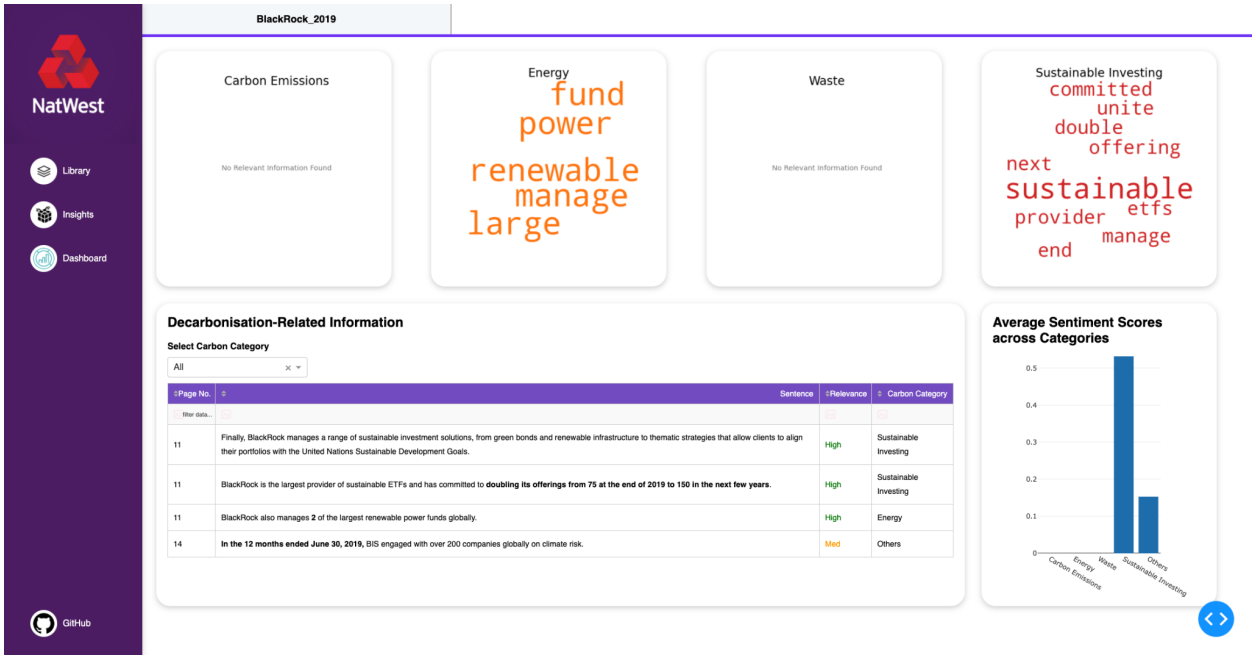
2. After a report has been chosen and loaded, users can view the relevant information on the insights page as shown. The keywords under each image and dataframe are also shown to allow users to easily filter for relevant information for their differing use cases.





3. However, in cases where no relevant charts or tables are extracted, the page below will be displayed. Even though it is not ideal to show an empty page, we decided to still process the PDF as there will be text information that is valuable and can still be displayed and analysed in the “Dashboard” page.

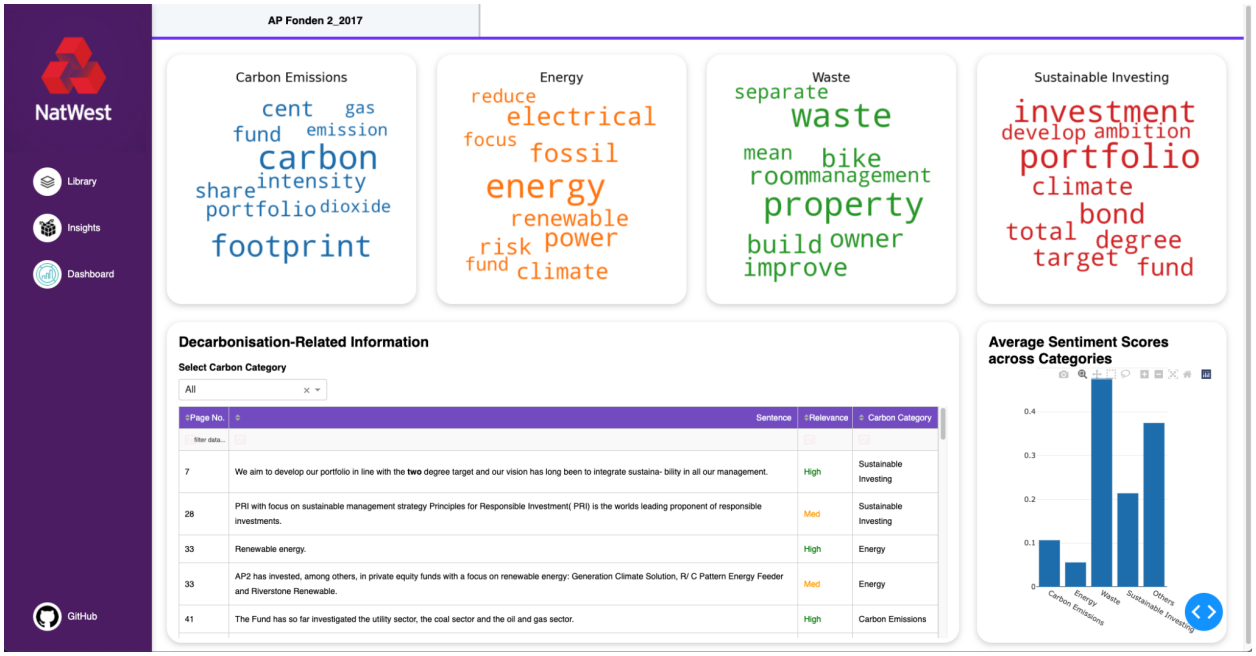




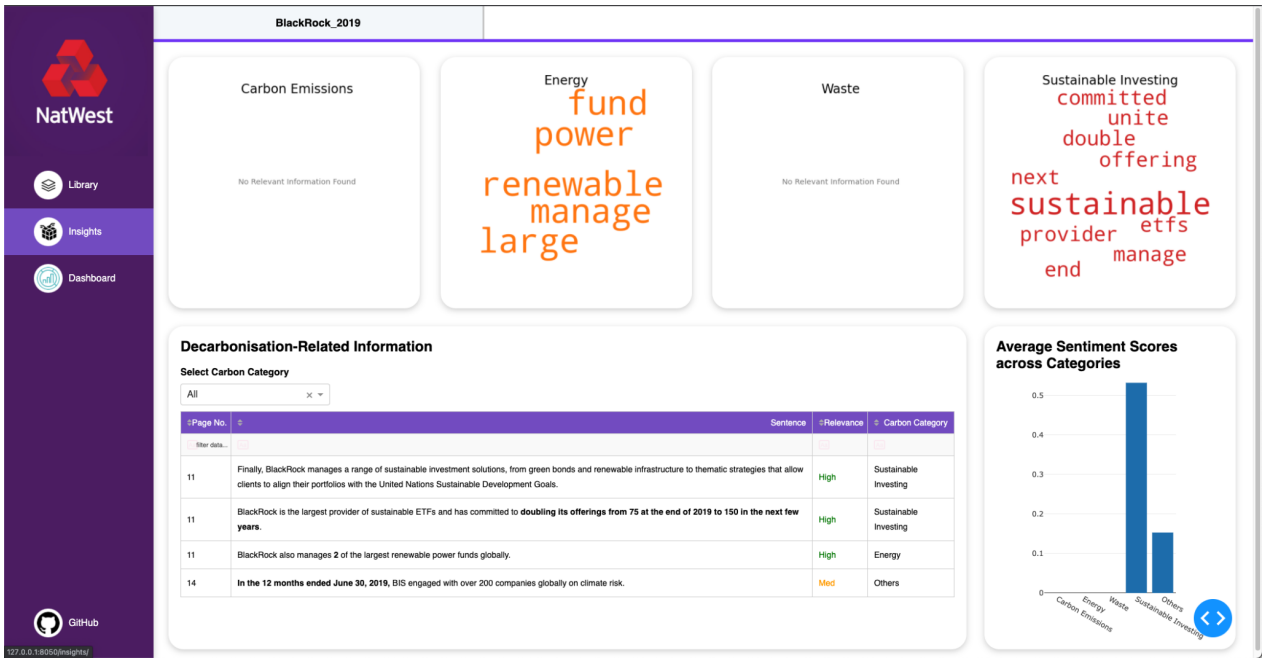
Dashboard Page:

Overview

1. Similar to the “Insights” page, to load the “Dashboard” page, a report has to be first chosen and loaded from the “Library” page. This page consists of several visualisations as shown below.

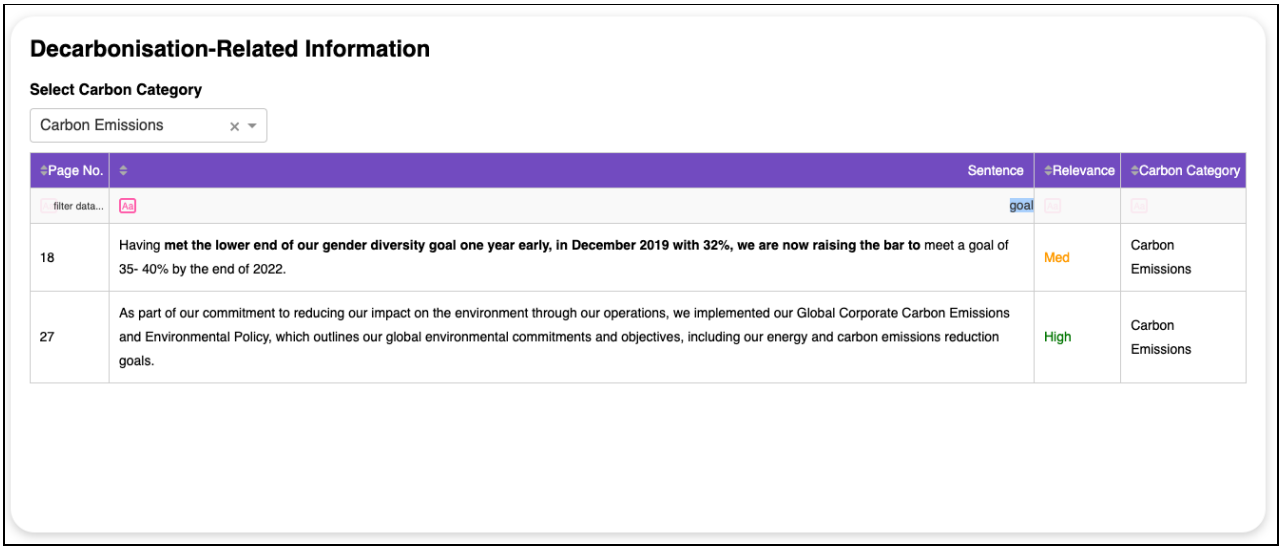


2. For a chosen report, not all visualisations might be shown. This depends on the availability of information extracted by our pipeline, as there may be carbon classes that are not mentioned in the report. Thus, there might be empty visualisations in this page, as seen below for the word cloud and the bar chart. This informs the user that that class has no extracted sentences at one glance.

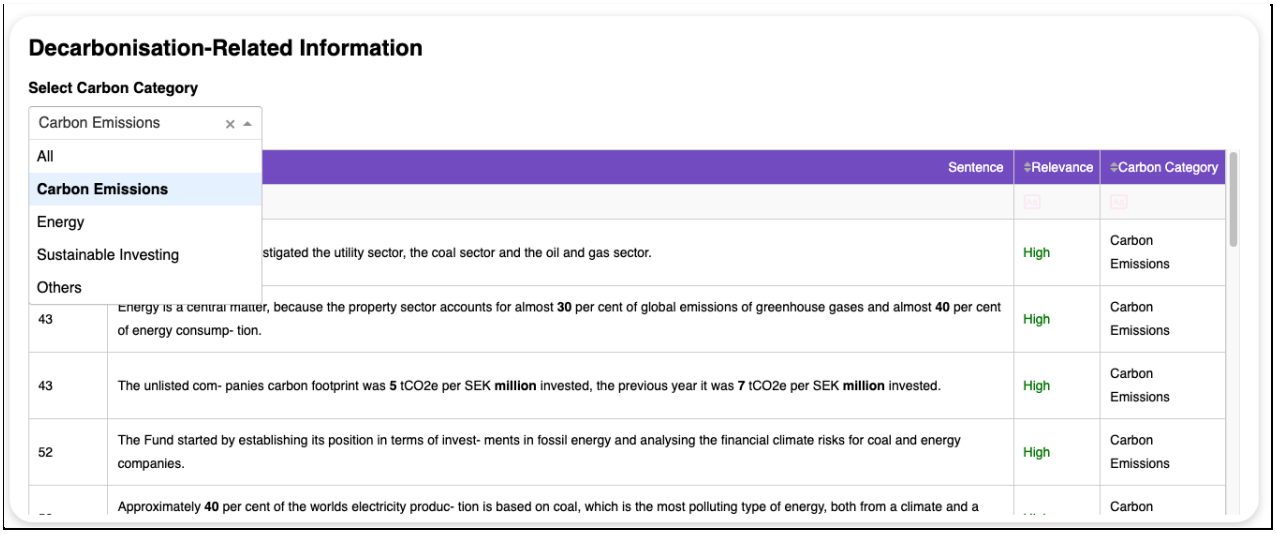


Functionalities under "Decarbonisation-Related Information"

1. Within the "Decarbonisation-Related Information" card, users can also do a keyword search to display sentences that contain specific keywords. For example in this case the word used was "goal". This allows users to query the entire relevant corpus for a topic that they may be more interested in.

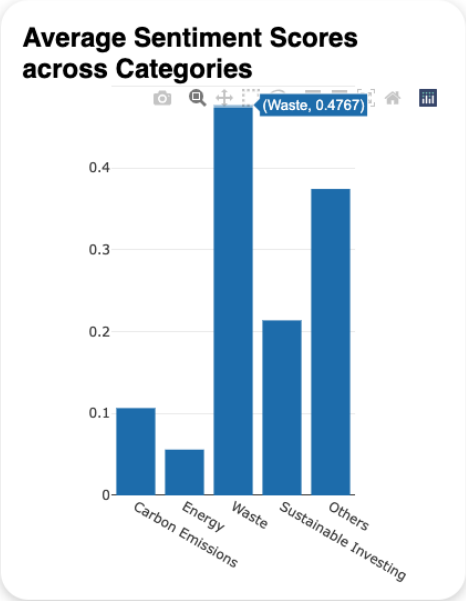


2. There is also a drop down to allow users to filter the sentences by carbon categories as shown below. Users can also sort the sentences by relevance score by clicking on the "Relevance" column header. This allows users to easily focus their attention on the most relevant sentences.



Functionalities under Sentiment Bar Chart

1. For deeper exploration, users can hover over the barchart to identify the actual magnitude of sentiment scores



References

Alva. (2021, June 21). *The need for speed: sentiment analysis in ESG measurement*. https://www.alva-group.com/blog/the-need-for-speed-sentiment-analysis-in-esg-measurement/#:%7E:text=Sentiment%20analysis%20will%20become%20an,do%20it%20in%20real%20time.

Bloomberg Intelligence. (2021, February 23). *ESG assets may hit $53 trillion by 2025, a third of global AUM*. Bloomberg. https://www.bloomberg.com/tosv2.html?vid=&uuid=93f2b942-4386-11ec-9b19-6d7656784c4f&url=L3Byb2Zlc3Npb25hbC9ibG9nL2VzZy1hc3NldHMtbWF5LWhpdC01My10cmlsbGlvbi1ieS0yMDI1LWEtdGhpcmQtb2YtZ2xvYmFsLWF1bS8=

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/jair.953

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Fischer, P., Smajic, A., Abrami, G., & Mehler, A. (2021, September). Multi-Type-TD-TSR–Extracting Tables from Document Images Using a Multi-stage Pipeline for Table Detection and Table Structure Recognition: From OCR to Structured Table Representations. In *German Conference on Artificial Intelligence (Künstliche Intelligenz)* (pp. 95-108). Springer, Cham.

Huynh-Van, T., Nguyen-An, K., Khanh, T. L. B., Yang, H. J., Tran, T. A., & Kim, S. H. (2018, February). Learning to detect tables in document images using line and text information. In *Proceedings of the 2nd International Conference on Machine Learning and Soft Computing* (pp. 151-155).

Nason, D. (2020, December 21). *"Sustainable investing" is surging, accounting for 33% of total U.S. assets under management*. CNBC. https://www.cnbc.com/2020/12/21/sustainable-investing-accounts-for-33percent-of-total-us-assets-under-management.html

Paliwal, S. S., Vishwanath, D., Rahul, R., Sharma, M., & Vig, L. (2019, September). Tablenet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images. In *2019 International Conference on Document Analysis and Recognition (ICDAR)* (pp. 128-133). IEEE.

Prasad, D., Gadpal, A., Kapadni, K., Visave, M., & Sultanpure, K. (2020). CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops* (pp. 572-573).

Probert, A. (2021, July 13). *Lack of Standardized ESG Reporting System Biggest Threat to Effective ESG Disclosures*. Duff & Phelps. https://www.duffandphelps.com/about-us/news/esg-reporting-system-effective-esg-disclosures

Rosén, G. (n.d.). Analysis of Tabula, a PDF-Table extraction tool. http://uu.diva-portal.org/smash/get/diva2:1363917/FULLTEXT01.pdf