Realization document

Mats Fockaert
R0695565
3 ITF BI
Thomas More, Geel
2020

1 INDEX

Contents

2 ACKNOWLEDGEMENTS

Finally, getting a project to prove myself after three years behind the school benches. Therefore, I would like to thank Ordina VisionWorks that provided me the opportunity.

The internship would not have been the same without the guidance of Jochen Van de Voorde and Aron Geerts. Who took time out their busy schedule to hear, guide and keep me on the correct path.
I also want to congratulate Jochen and his wife to becoming parents!

I would also like to thank Dirk Mervis for monitoring my journey and giving feedback when and where necessary.

A big thanks to all the teachers in my life that got me the skillset needed to start this project.

Thank you!

3 INTRODUCTION

Social media is booming, data is being generated at an astronomical rate and climate change exists.
What do these three topics have in common? That is what this project will help figuring out.

Company 'Maat van het Klimaat' wishes to get a more in-depth view around the perception of climate change. By analyzing the correlation between comments on social media and weather history/updates. This project will try to help in achieving those wishes.
They also want a way to analyze what people are saying about their services and/or products. An overview of the analysis of the comments will be made and displayed in a dashboard-like manner.

The project will be divided in four main components. To start any data driven project, we need data. Hence, phase one: collecting and researching data. During this phase, a plan of attack is made and research into data sources is made. The collected data is stored locally, and a very elementary analysis will be performed.

The next phase consists of analyzing the data and researching ways to save it correct and efficiently. In this phase, machine learning models will be created to help in detecting patterns and creating sentiment predictions for stored data.

Phase three: AWS. This phase will be all about uploading the data, models, analysis and other necessary functions to an AWS product.

The final phase is creating a RESTful API that can interact with requests from the client and will send interesting information and data. To make an easy-to-use endpoint, an extra page to the client's site will be added where all results and queries can be found on.

4 INTRODUCTION TO ORDINA VISIONWORKS

“Ordina’s BI & Analytics unit, VisionWorks, works for companies that focus on data driven decision-making. More and more data is being collected and recorded digitally. It is becoming increasingly more complex to get the right information, efficiently merged and made available.

Efficiently merging data streams and making them available. That is our specialization. In addition to expertise in ETL-tools, our team is familiar with traditional business intelligence (BI) solutions such as Microsoft BI, SAP BW, Cognos, Qlik, Tableau, Informatica, etc. With business analytics and data science we go one step further. We translate quantitative data into valuable insights and forecasts. We increase our customers’ Return on Data by taking them on a journey to a modern and innovative data culture.”[1]

The competence centers are:

- Microsoft Data Platform & BI
  Striving to create a modern and innovative data culture while serving customers to their needs in all BI related tasks.
- DataTalks, (big) data science
  Configuring cloud and big data architectures and managing their flows and predictive modeling with advanced analysis. Both require the necessary visualizations that are also build here.

[1] (Amazon Web Services, 2020)

5 COLLECTING DATA

When the plan of action was made and the Trello-board schedule was finished, I created a GitHub account to keep all my code accessible.

Python 3.7 was the backbone of the project. All initial data science operations are performed using Anaconda's Jupyter environment. Later on, this will be mostly done in Visual Studio Code. Setting up Python and the Anaconda environment where the first step in the preparation of the project.

After this, I scoured the web to find multiple sources of historical weather reports and services that could offer helpful services. WeatherUnderground, and OpenWearherMap were high ranking services unfortunately not free. I decided to create a service that would collect daily weather data with a free service from OWM and thus creating my own data source in time. WU did have useful and free reports between the years 2015 and 2017.

The function created with OWM will need to be invoked every three days since it will generate that amount of days in data. For now, it will be invoked by hand. This will be automated with a Lambda function.

Having trouble finding free historical data sources the search was changed to finding social media data. Staring out with collecting Twitter tweets and Reddit posts, and performing simple analysis on the collected data to get a picture of what will be important.
In the beginning I used TweePy to collect tweets. TweePy is an easy-to-use Python library for accessing the Twitter API. Later this turns out to be not the best choice.
For reddit I used a tool called Praw.
Tweets where from 2006 until now and where all about climate, climate change or environment that were important and/or trending. In total, in the end, nine thousand six hundred posts were collected.

After the collection of the social media posts, new resources for weather data were found. Storm data from 2017 and historical data from 1997 to 2017.

6 ANALYZING DATA AND SETTING UP MODELS

6.1 Data analysis

6.1.1 Weather analysis

First, we need a way to structure the different data sets. For this we perform simple analytical functions that will display the importance of certain components of the data. E.g. making a correlation matrix:

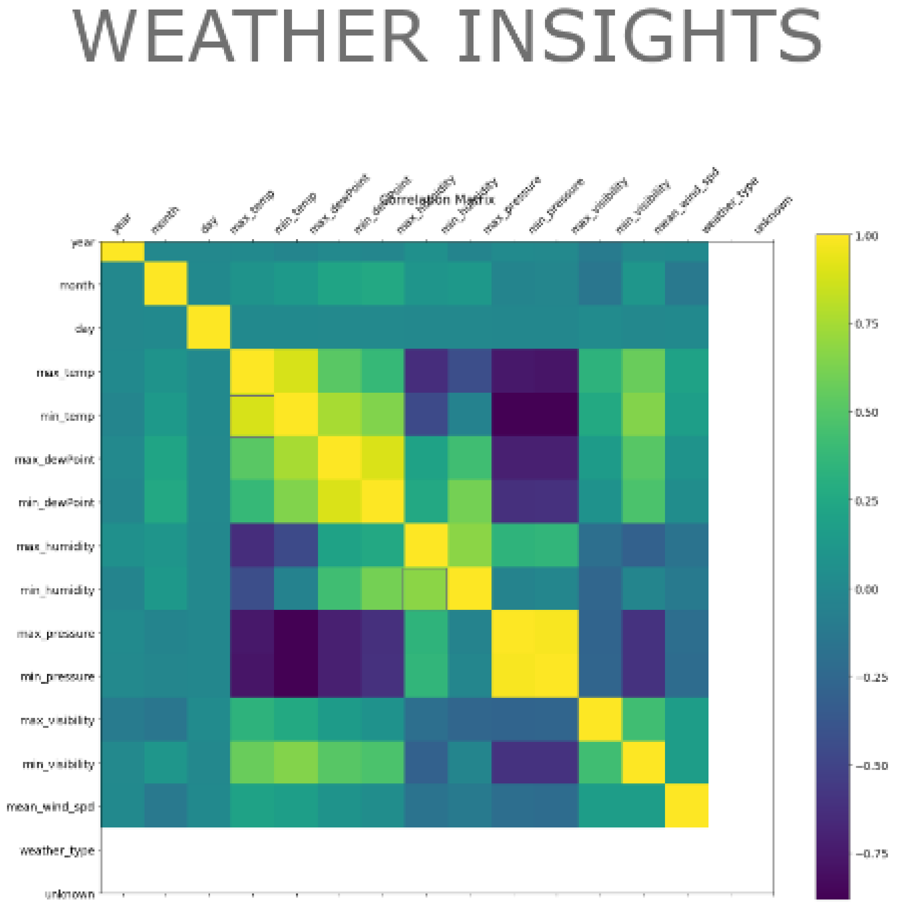


Figure 1: Correlation Matrix displaying the different columns in a weather dataset and showing what impacts what the most.

6.1.2 Sentimental analysis

To get a bigger picture of the social media data, a Machine Learning algorithm was created (more about this later) that will estimate if a post has a negative or a positive sentiment. With this trained and over 89% accurate, the rest of the analysis of the data can be done.

A very helpful tool is to generate word clouds. This then shows what words are most popular according to their sentiment.

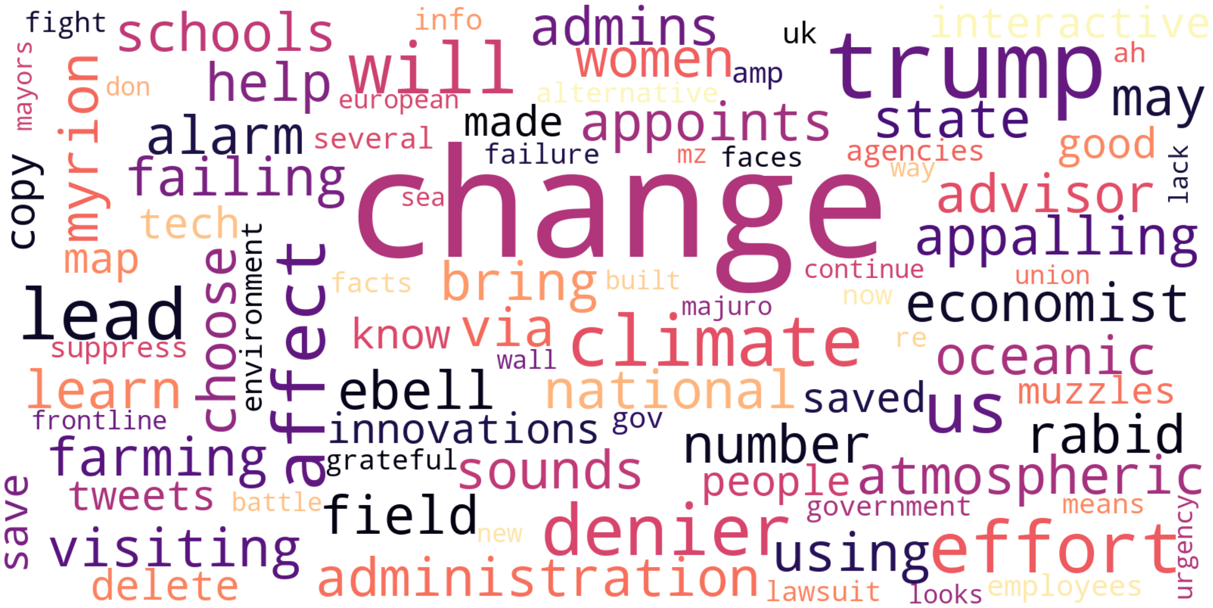


Figure 2: Word cloud showing negative social media posts regarding climate change

6.2 Algorithms and models

6.2.1 Sentiment analysis

6.2.1.1 How does sentiment analysis work?

The main model of how a computer knows what a phrase is trying to convey, has following steps as backbone:

- Word tokenization:
  It is the process of dividing the input text into a set of pieces like words or sentences. These pieces are called tokens.

- Stemming:
  It is basically a process that cuts off the ends of words to extract their base forms.

- Text lemmatization:
  Lemmatization is another way of reducing words to their base form. The lemmatization process uses a vocabulary and morphological analysis of words. It obtains the base forms by removing word endings such as -ing or -ed. This base form of a word is known as a lemma. If you lemmatize the word "calves", you should get "calf" as the output. One thing to note is that the output depends on whether the word is a verb or a noun.

- POS tagging:
  The target of Part-of-Speech (POS) Tagging is to identify the grammatical group of a given word, whether it is a noun, pronoun, adjective, verb, adverb, etc. based on the context. POS Tagging looks for relationships within the sentence and assigns a corresponding tag to the word.

- Named entity recognition:
  refers to the identification of words in a sentence as an entity e.g. the name of a person, place, organization, etc.

6.2.1.2 Tfidf (Term Frequency – Inverse document frequency)

The features will be extracted and simplified by using TfidfVectorizer that converts textual data to numeric form. It will be trigram (three consecutive words in a sentence) extraction.
Example: 'Trump denies the effect of oil on climate change':

bigram: 'Trump denies', 'denies the','the effect','effect of','of oil','oil on','on climate','climate change' (= 8 long)
trigram: 'Trump denies the','denies the effect','the effect of','effect of oil','of oil on','oil on climate','on climate change' (= 7 long)

It works using a simple formula:
TF = (Frequency of a term in the document)/(Total number of terms in documents)
IDF = log((total number of documents)/(number of documents with term t))
=> TFIDF = TF * IDF

6.2.1.3 Lexicon and rule-based sentiment analysis tools

For starters I used TextBlob and VADER to get an easy analysis of collected tweets
There are existing methods to do sentiment analysis. One of them is called 'TextBlob'. It uses a lexicon to calculate if a word is positive or negative. It uses three structures for each word: polarity (negative, positive or neutral), subjectivity and intensity (does it modify next word?). When the word 'very' is analyzed, polarity and subjectivity will be ignored because it is an intensifying word used to modify the following word.

Complete calculations for a phrase with negations:
Phrase: 'not very great':

- 'very': intensity = 1.3
- 'great': polarity = 0.8, subjectivity = 0.75

The rule for negation: Negation multiplies the polarity by -0.5 and doesn't affect subjectivity.
Polarity: -0.307 (- 0.5 * 1/1.3 * 0.8)
Subjectivity: 0.577 (1/1.3 * 0.75)

This is a perfect example of how a computer knows what sentiment a message will try to convey.

Another library like this is VADER. VADER is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media.
Advantages of VADER:

- It works exceedingly well on social media type text, yet readily generalizes to multiple domains
- It doesn't require any training data but is constructed from a generalizable, valence-based, human-curated gold standard sentiment lexicon
- It is fast enough to be used online with streaming data, and
- It does not severely suffer from a speed-performance tradeoff.

6.2.1.4 Working with algorithms

Another option would be by training an algorithm that finds these lexicons itself. It will try to find those values and calculations itself in the 'personal' data found on Reddit and Twitter. This way there is a better view of what the algorithm finds and allows for improvement of the model.

Meet the algorithms:

6.2.1.4.1 linearSVC (Support Vector Classification):

SVC is an approach that is systematic, reproducible and properly motivated by statistical learning theory. Training involves optimization of a convex cost function: there are no false local minima to complicate the learning process.

LinearSVC is a type of SVC that is faster to converge the larger the number of samples is. This is since the linear kernel is a special case.

The linear model implements "one-vs-the-rest" multi-class strategy, thus training n_class models. If there are only two classes, only one model is trained. This means that it will create multiple instances in which it will train on the data.

6.2.1.4.2 AdaBoostClassifier:

A meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Meaning that it will start by using educated guesses, when they are wrong it will try again with more weight on the incorrectly classified instances.

6.2.1.4.3 naive_bayes:

These algorithms are based on applying Bayes' theorem with strong (naïve) independence assumptions between the features.

- MultinomialNB: classifier that works great with TFIDF counts (see above). gives the probability of any combination of numbers of successes for the various categories
- When k (number of categories) is 2 and n (number of events) is 1, the multinomial distribution is the Bernoulli distribution.

6.2.1.4.4 RidgeClassifier:

This classifier first converts the target values into {-1, 1} and then treats the problem as a regression task.

6.2.1.4.5 Perceptron:

Works like an SGD (stochastic gradient descent) learning routine which supports different loss functions and penalties for classification.

It does not require a learning rate; it is not regularized (penalized) and it updates its model only on mistakes.

6.2.1.4.6 PassiveAggressiveClassifier:

They are like the Perceptron in that they do not require a learning rate. However, contrary to the Perceptron, they include a regularization parameter. Works with the hinge lost function found in SVC algorithms. It is passive on a correct classification. On a misclassification the update rule becomes very aggressive due to the fact that the algorithm looks for a near-perfect answer.

6.2.1.4.7 LogisticRegression:

The probabilities describing the possible outcomes of a single trial are modeled using a logistic function. It can fit binary, One-vs-Rest, or multinomial logistic regression or Elastic-Net (=penalized logistic regression and regularized logistic regression) regularization.

6.2.1.4.8 NearestCentroid:

Simply put, each class is represented by its centroid, with test samples classified to the class with the nearest centroid. Because we are using TFIDF vectors, it is more aptly named 'Rocchio classifier'.

6.2.1.5 Pipeline of chaos

We try to make the most efficient and fast model. We try to attain this by using vectorizers, pipelines and, most importantly, by comparing different algorithms and not just using one.

Now I created pipeline that checked all the above algorithms and kept tabs on their performance. The result is:

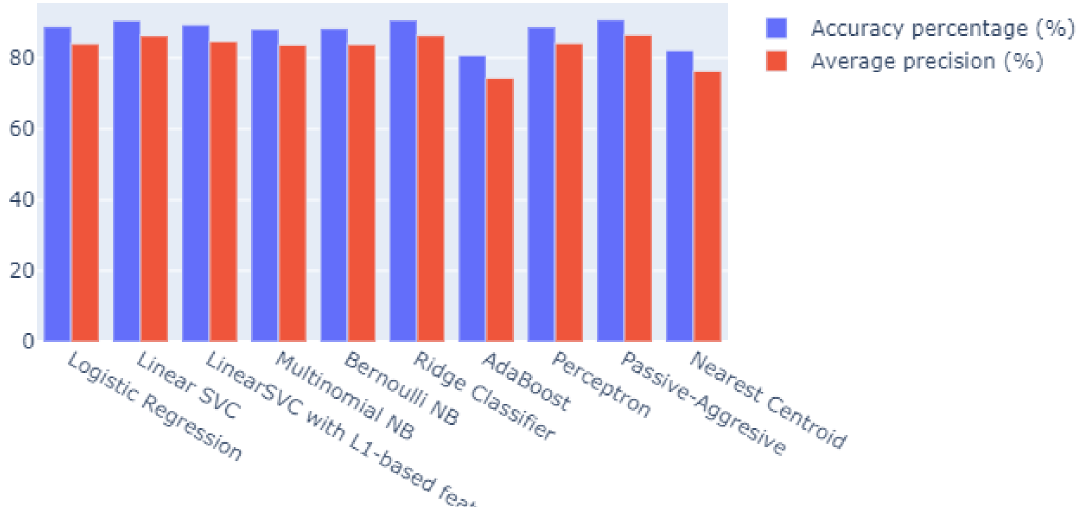


Figure 3: Accuracy vs Precision bar chart

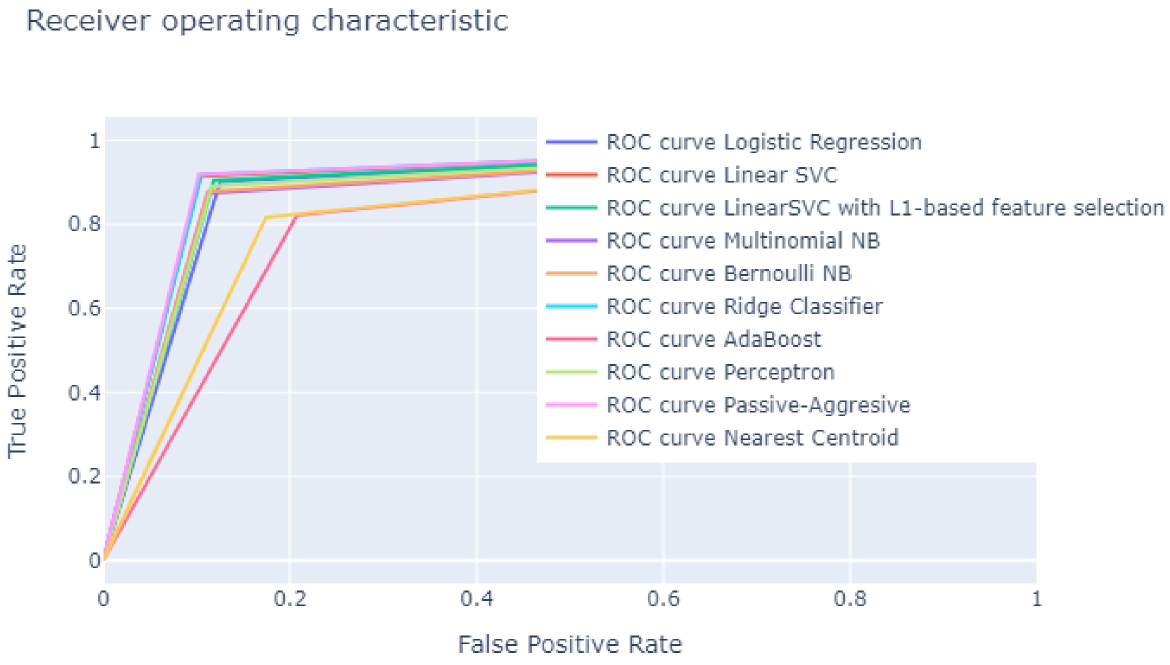


Figure 4: ROC all algorithms

The first chart is self-explanatory. The difference between precision and accuracy is precision is the degree to which repeated measurements will show the same results while accuracy is the measurement to the true (correct) value.

The second chart displays the receiver operating characteristic (ROC); illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.
The true positive rate is the recall or probability of detection of rightfully categorized positive results
The false positive rate is the ratio between the number of negative events wrongly categorized as positive (false positives) and the total number of actual negative events.

We now know that the SVC algorithm and the Ridge classifier are the best performing algorithms on this data (early high peak and then stagnation).

Now we train an ensemble classifier of the top four accurate models, and see if there is any improvement
A voting classifier that combines conceptually different machine learning classifiers and uses a majority vote or the average predicted probabilities (soft vote) to predict the class labels.
Afterwards another ROC plot about the four models is shown.

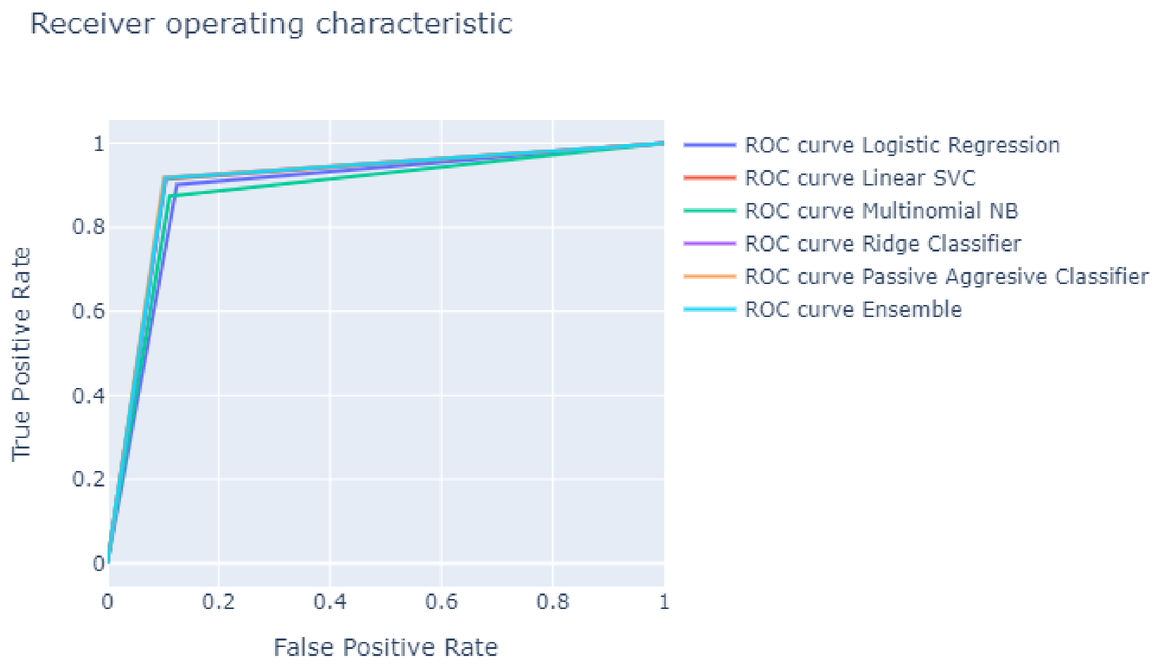


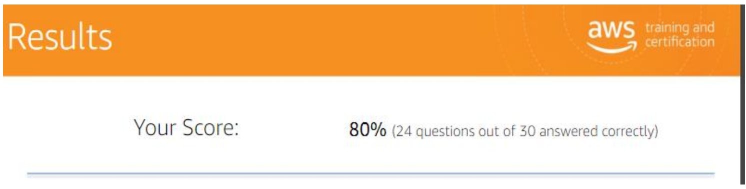
Figure 5: ROC-ensemble

Although it is hard to see, the highest ranking is the ensemble classifier that combines the best results.
This is the last step (for now) in the sentimental analysis.

6.3 Saving the data

6.3.1 Where?

After following an online course in AWS, the data storage platform of choice was S3. Not only will it be beneficial for myself because it is easier to link different AWS services, it will also be faster and more reliable than different services due to the structure of AWS servers and the very fact that all the data will now run in one giant family swarm of servers.



Results

Your Score: **80%** (24 questions out of 30 answered correctly)

Figure 6: AWS Certification Exam Results

6.3.2 What?

In the beginning, everything was going to be saved as JSON-files since MongoDB was still an option to use as database server.
When S3 was locked in, research into fast and reliable data types was committed. The two most prominent were, Feather and Parquet. The main difference is that Parquet is more expensive, in time and processing power, to write since it uses more layers of encoding making the file smaller.
Since we want to write smaller files and need to read and write on the fly, Feather was a more suitable option.

6.3.2.1 Social media data

A general file that contains all the data from Twitter and one from Reddit is created. Creating this file, the realization of why TweePy was not the right option for this project happened. TweePy will not show an error but it cannot look further back into the future than a couple of weeks. The Twitter data was insufficient to perform any sort of analysis.
Looking into other technologies to collect historic tweets, 'GetOldTweets3' was found. This tool bypasses the Twitter API limitation of time constraints. After adjusting the original code and letting it purge Twitter for climate related tweets, a new feather file was made.

The data can now be separated by sort (Twitter or Reddit), year and month.
The S3 of a feather file: /Raw/Twitter/2006/twitter_climate_2006_1.feather

6.3.2.2 Weather data

Like the social media data, a general file is being created that contains all the different files. This file will only contain the most important features of all files. This will be called the 'clean' file.

To keep the original division between the data sets, per category there is a general raw file.

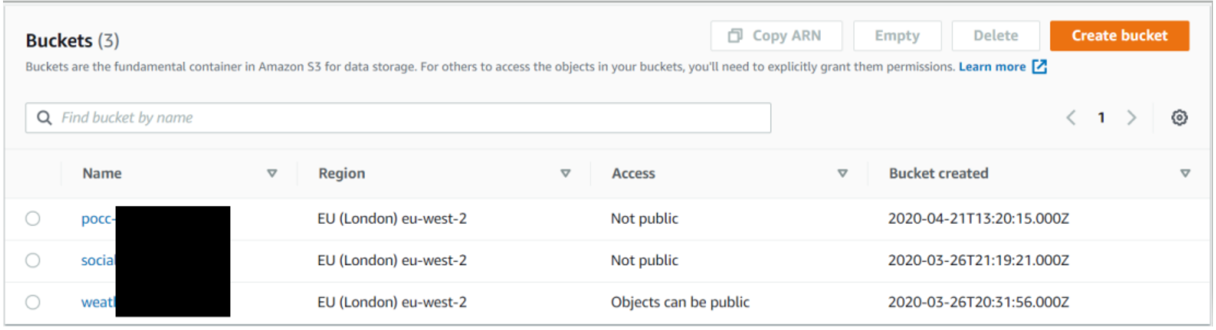
For quick and easy access of the data, it will be separated into their own category, year, month and the file will be data from that day.
Due to difficulties with packages on a Lambda service, the feather files were no longer an option. All the OWM files were changed to JSON files since this will be how the new data will be saved.

7 AWS

7.1 S3

"Amazon Simple Storage Service is a service offered by Amazon Web Services (AWS) that provides object storage through a web service interface."[2]

As said in the previous section, this is used to (mainly) store weather data files and social media data files. The database exists out of buckets, these are the main containers used to separate file structures. After this, in each bucket, there are keys. Keys are simple folder structures on S3.



Buckets (3)

Buckets are the fundamental container in Amazon S3 for data storage. For others to access the objects in your buckets, you'll need to explicitly grant them permissions. Learn more

	Name	Region	Access	Bucket created
○	[redacted]	EU (London) eu-west-2	Not public	2020-04-21T13:20:15.000Z
○	[redacted]	EU (London) eu-west-2	Not public	2020-03-26T21:19:21.000Z
○	[redacted]	EU (London) eu-west-2	Objects can be public	2020-03-26T20:31:56.000Z

Figure 7: Buckets on S3

7.2 IAM

"AWS Identity and Access Management (IAM) enables you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources."[3]

Due to conflicts with my account, I could not make any changes to the access management system and needed to adapt the other services to the already made IAM roles.

7.3 Lambda

Lambda is a way to connect different applications in the AWS environment with ease.
To get daily weather updates, this function was used to gather OWM data of that day and stored on S3; raw and cleaned.

7.4 EC2

To put simply, EC2 are AWS' virtual computers. It allows you to build your own virtual computers, choosing the amount of CPU, RAM, storage, and what type of OS will be running on it. Creating two different EC2 instances for this project. More in-depth explanation in the next section.

[2] (AWS, Amazon S3, 2020)

[3] (Amazon Web Services, 2020)

8 API AND CLIENT SITE

All the previous steps are now fitted in a RESTful API. This makes it easier for the client to access our services.

The API is running on an AWS EC2 Linux t2.medium server that contains 2CPUs, 8GiBs of RAM, 30GiBs of storage. The main data will come from the S3 file server but files that need to be saved during the API-request process will be stored on the EC2 server itself. These files can contain crucial images, files send by the client and others.

The files that are being hold on the EC2 instance are models and word clouds.

Flask is the foundation of the API. The API holds pickled models that the client can use for researching their own sentimental or weather data.

The client' site is running on an AWS EC2 Linux t2.micro server that contains 1CPU, 4GiBs of RAM, 30Gibs of storage. For this project the client's site would already exist and the sole purpose of creating the site was only for seeing the API in action. This is why a low-cost, low-performance server was chosen.

8.1 Requests a customer can make:

8.1.1 Sentiment:

- Get sentiment of a self-typed query and a simple analysis
- Get sentiment of a csv file with a large analysis
- Get sentiment of a certain date with a large analysis

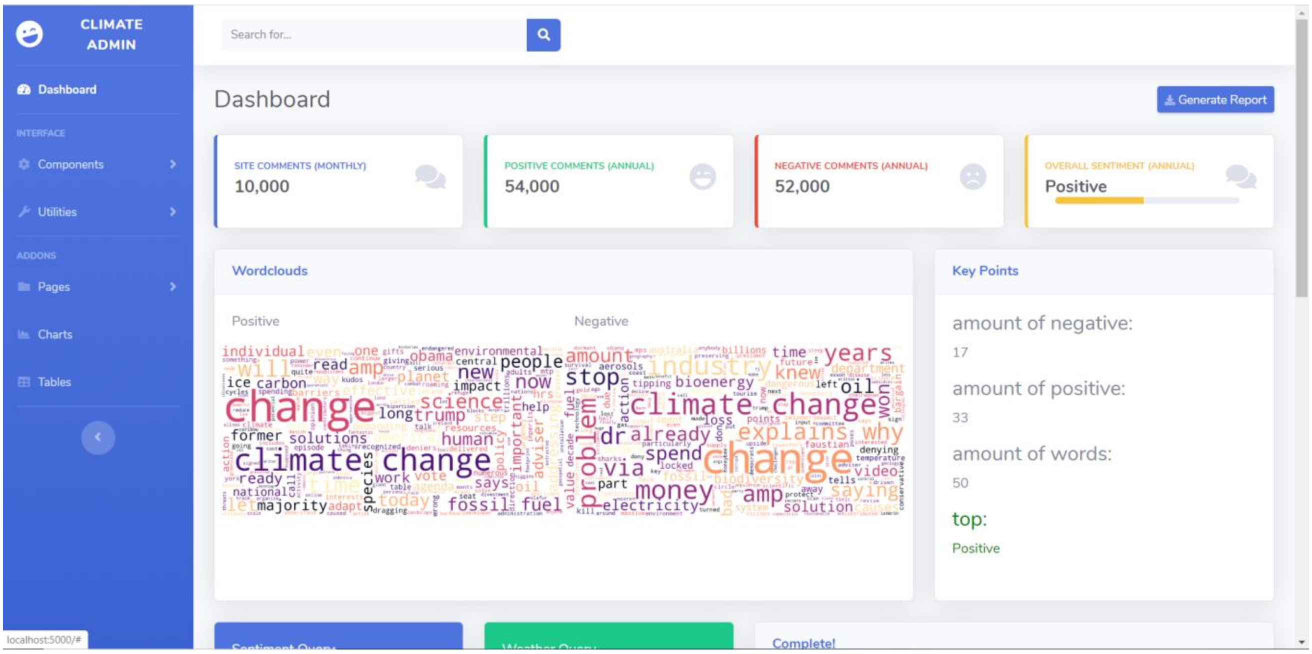
8.1.2 Weather:

- Get weather analysis of a self-typed query
- Get weather analysis of a certain date
- Get an analysis of all the weather data in the API's database

8.1.3 Combined (in progress):

- Get sentiment analysis of all dates that contain a certain given weather type

8.2 Client site



Dashboard

SITE COMMENTS (MONTHLY) 10,000

POSITIVE COMMENTS (ANNUAL) 54,000

NEGATIVE COMMENTS (ANNUAL) 52,000

OVERALL SENTIMENT (ANNUAL) Positive

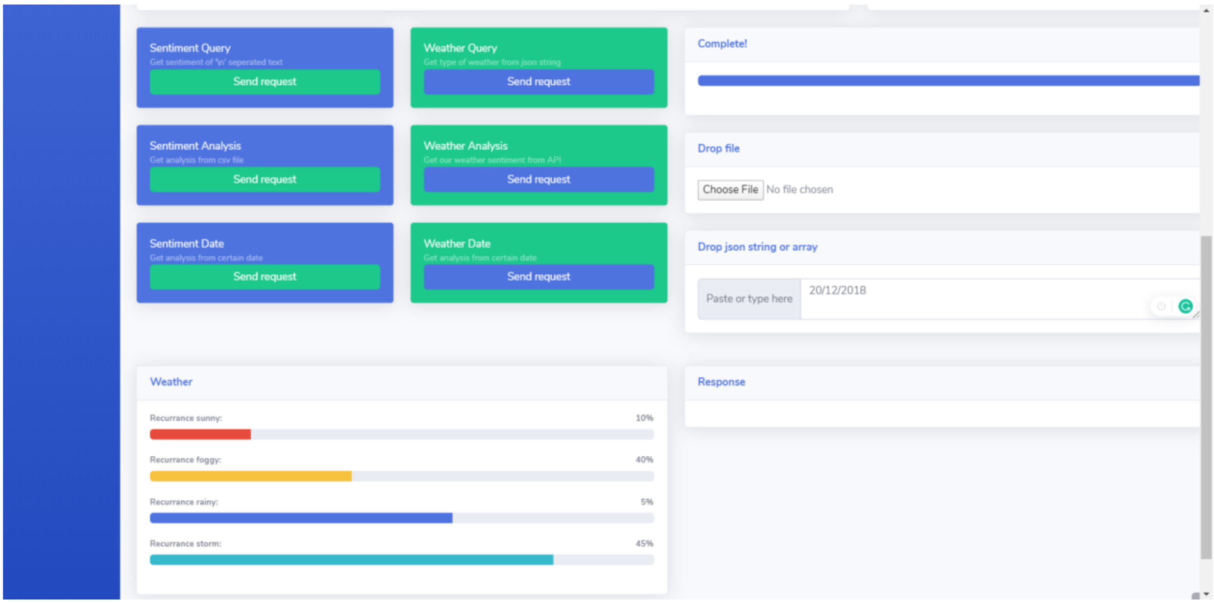
amount of negative: 17

amount of positive: 33

amount of words: 50

top: Positive

Figure 8: Client Dashboard pt.1



Sentiment Query: Get sentiment of '\n' seperated text

Weather Query: Get type of weather from json string

Sentiment Analysis: Get analysis from csv file

Weather Analysis: Get our weather sentiment from API

Sentiment Date: Get analysis from certain date

Weather Date: Get analysis from certain date

Complete!

Drop file: No file chosen

Drop json string or array: Paste or type here 20/12/2018

Recurrance sunny: 10%

Recurrance foggy: 40%

Recurrance rainy: 5%

Recurrance storm: 45%

Figure 9: Client Dashboard pt. 2

The first thing we see are four KPI messages regarding data of the client itself. This has nothing to do with our API but with the program of the client.
Then we see a “Wordclouds” and a “Key Points” section. The word clouds are self-explanatory. The key points differ on what API call is chosen.

Looking at the second screen, we see the six possible calls to the API.
The “Weather” tab shows what percentage of the data contains what type of weather.
The response tab displays more in-depth information in deeper analysis.

See the demo for more information.

8.3 API

8.3.1 Sentiment:

- Get sentiment of a self-typed query and a simple analysis
    - It runs every line through the Pickled model that has been trained to predict sentiment.
    - When all the lines have been labeled, an accuracy is added.
    - All of this is sent back to the client's back end.
- Get sentiment of a csv file with a large analysis
    - The csv file is adapted to fit in the model and all sentiments are predicted.
    - A bigger analysis is performed that contains word clouds.
    - All is sent back.
- Get sentiment of a certain date with a large analysis
    - First the date is checked and formatted. If the date exists in the data, it will be pulled up.
    - All files that contain the data, will be analyzed for sentiment.
    - Again, a bigger analysis is performed that contains word clouds.
    - All is sent back.

8.3.2 Weather:

- Get weather analysis of a self-typed query
    - The weather query is pulled and formatted.
    - This will then be ran through the weather model that will attach a type to it.
    - This type will be returned.
- Get weather analysis of a certain date
    - Like sentiment analysis only for weather.
- Get an analysis of all the weather data in the API's database
    - This gathers all the weather data containing weather types.
    - An analysis of the types is performed.
    - The percentage of each type is returned.

8.3.3 Combined (in progress):

- Get sentiment analysis of all dates that contain a certain given weather type
    - Based on a given weather type, all dates containing these types will be gathered.
    - These dates are then used to get all the social media data.

- A sentiment analysis is performed (with word clouds)
- All is returned.

9 CONCLUSION

The goal of the project was to help the client to find correlations between weather, sentiment of the public and the sentiment of their customers during certain periods.

There is now a solution that offers:

- a sentiment analysis tool,
  The goal, set by the customer, was to have a model that can predict the sentiment with an accuracy of at least eighty-five percent. The actual model has an accuracy of ninety-three percent during the making of this document. When the company gains more ratings on their site concerning climate change, it can use that data to adjust or even retrain the model to get even more fit predictions for their needs.
- a weather analysis tool,
  It can display the different weather types, what the values were and can moderately predict what weather it will be on a new set of data.
- a combined analysis tool,
  This will help the client in making analysis upon the data and their data. It may provide new insights and different approaches the client may take.

One of the conclusions made was that, when there is something bad happening to the climate in the most recent years, Twitter is flooded with negative tweets that contain ‘Trump’. Obama and Bill Nye will come along on the positive side of this flood.
We can also see that there are a lot of more positive tweets regarding climate change during the pandemic.

10 BIBLIOGRAPHY

Amazon Web Services, I. (2020, 05 22). Retrieved from aws: https://aws.amazon.com/

AWS. (2020, 04 22). *Amazon S3*. Retrieved from Wikipedia: https://en.wikipedia.org/wiki/Amazon_S3

AWS. (2020). *AWS Identity and Access Management (IAM)*. Retrieved from AWS: https://aws.amazon.com/iam/

BI & ANALYTICS. (n.d.). Retrieved 05 19, 2020, from https://www.ordina.be/en/competences/bi-analytics/

Boto3 Documentation. (2020, 05 01). Retrieved from Boto3: https://boto3.amazonaws.com/v1/documentation/api/latest/index.html

Cielen, D., Meysman, A. D., & Ali, M. (2016). *Introducing Data Science.* New York: Manning Publications.

Joshi, P. (2017). *Artificial Intelligence with Python.* Mumbai: Packt Publishing.

Understanding the Process of Collecting, Cleaning, Analyzing, Modeling and Visualizing Data . (2020). Retrieved from data science graduate programs: https://www.datasciencegraduateprograms.com/the-data-science-process/