CP468- Artificial Intelligence

Final Report

Adam Joe- 170801790 Shane Riley-170696320 Deepinder Sidhu- 170274550 Abraham Banka-150411560

Introduction

For this project, we created a web crawler to search through articles and scrap for vector words that we predetermined. We then compiled these results into a Document-Term matrix and created calculations to check for frequency of each vector per field and used random articles to test for accuracy which will be explained in further detail below, along with the results of our tests.

Project Description

(Phase 1) Web Crawler

During phase 1 we developed a web crawling program to scrape through the html documents collected by the group and to search for words that could be recognized among the html characters. The process of developing the web crawler first involved deciding on the feature vectors we would use. Our group had decided to use the topics "Technology", "Cars", and "Sports" so to help decide on vectors we first created lists of words we thought had a very strong relevance to these topics. We then collected 10 documents for each topic by searching for these terms online and choosing appropriate documents that featured many relevant words. Collecting documents then involved retrieving an html copy of the website (ctrl+u) and saving it to our files.

Name	Туре	Compressed size	Password pr	Size	Ratio	Date modified
Ce cars1	Microsoft Edge HTML Docu	43 KB	No	204 KB	80%	2021-07-27 2:08 PM
C cars2	Microsoft Edge HTML Docu	81 KB	No	412 KB	81%	2021-07-27 2:08 PM
C Sport 1	Microsoft Edge HTML Docu	49 KB	No	249 KB	81%	2021-07-27 2:08 PM
C Sport 2	Microsoft Edge HTML Docu	58 KB	No	282 KB	80%	2021-07-27 2:08 PM
C Sport 3	Microsoft Edge HTML Docu	50 KB	No	252 KB	81%	2021-07-27 2:08 PM
C Sport 4	Microsoft Edge HTML Docu	52 KB	No	254 KB	80%	2021-07-27 2:08 PM
C Sport 5	Microsoft Edge HTML Docu	50 KB	No	249 KB	81%	2021-07-27 2:08 PM
C Sport 6	Microsoft Edge HTML Docu	52 KB	No	255 KB	80%	2021-07-27 2:08 PM
C Sport 7	Microsoft Edge HTML Docu	53 KB	No	257 KB	80%	2021-07-27 2:08 PM
C Sport 8	Microsoft Edge HTML Docu	56 KB	No	276 KB	80%	2021-07-27 2:08 PM
C Sport 9	Microsoft Edge HTML Docu	51 KB	No	254 K8	81%	2021-07-27 2:08 PM
C Sport 10	Microsoft Edge HTML Docu	51 KB	No	251 KB	80%	2021-07-27 2:08 PM
👽 sports1	Microsoft Edge HTML Docu	56 KB	No	254 K8	79%	2021-07-27 2:08 PM
👽 sports2	Microsoft Edge HTML Docu	134 KB	No	657 KB	80%	2021-07-27 2:08 PM
C tech1	Microsoft Edge HTML Docu	91 KB	No	471 K8	81%	2021-07-27 2:08 PM
C tech2	Microsoft Edge HTML Docu	65 KB	No	413 KB	85%	2021-07-27 2:08 PM
C tech3	Microsoft Edge HTML Docu	57 KB	No	261 KB	79%	2021-07-27 2:08 PM
Ce tech4	Microsoft Edge HTML Docu	55 KB	No	242 KB	78%	2021-07-27 2:08 PM
C tech5	Microsoft Edge HTML Docu	57 KB	No	249 KB	78%	2021-07-27 2:08 PM
💽 tech6	Microsoft Edge HTML Docu	350 KB	No	1,705 KB	80%	2021-07-27 2:08 PM
C tech7	Microsoft Edge HTML Docu	204 KB	No	1,591 KB	88%	2021-07-27 2:08 PM
💽 tech8	Microsoft Edge HTML Docu	203 KB	No	1,590 KB	88%	2021-07-27 2:08 PM
Ce tech9	Microsoft Edge HTML Docu	209 KB	No	1,674 KB	88%	2021-07-27 2:08 PM

Next, the documents needed to be parsed to traverse them easier and perform the equivalence checks between words and our vectors. To do this we implemented the "re" library to split the lines of the html document at common html symbols or numeric values.

Once the data was cleaned, we created a function to search for a list of keywords in the given

auto

0 0

9

88

Tech1

Tech2 Tech3 Tech4

Tech5

Tech6 Tech7 Tech8

Tech9

Cars3 Cars4 Cars5

Cars6 Cars7 Cars8 Cars9

Cars10 Sports1 Sports2

Sports3 Sports4 Sports5

Sports6 Sports7 Sports8 Sports9

Sports10

Tech10 Cars1 Cars2

documents. By searching through each of the parsed documents, we collected the occurrence count of each feature vector within every document collected. We then improved upon our vector lists by looking through our occurrence tables and removing vectors that seemed to be somewhat ineffective compared to others, this caused my group to rework the vectors we used to ensure that we chose effective key terms. As can be seen in the table on the right, this issue of vectors that relate to multiple types makes it difficult to accurately classify documents. When the term "auto" is used for a car vector we see it's highest count comes from the car-type documents; however, there are also many occurrences in tech documents that could cause these documents to be classified as cars as well. Because of this, the final vector list went through multiple revisions to increase its usefulness in our classification and decrease the

Initial Vectors:

```
vector0 = ["iphone", "coding", "internet", "software", "network", "samsung", "computer", "device"]
vector1 = ["engine", "gas", "vehicle", "drive", "motor", "car", "truck", "road"]
vector2 = ["hockey", "football", "golf", "soccer", "basketball", "baseball", "play", "sport"]
```

applicability of vectors to multiple classification types.

Final Vectors:

```
vector0 = ["iphone", "technology", "microsoft", "phone", "network", "samsung", "computer", "device", "program", "digital"]
vector1 = ["engine", "gas", "vehicle", "automobile", "drive", "car", "transportation", "road", "wheel", "tire"]
vector2 = ["hockey", "football", "golf", "soccer", "basketball", "baseball", "play", "sport", "run", "kick"]
```

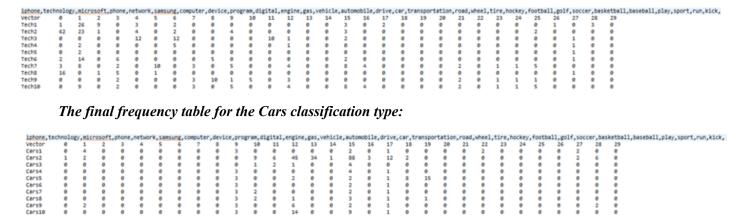
(Phase 2) Feature Extraction

The initial frequency table for the Sports classification type:

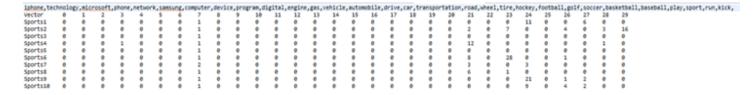
Using some previous vectors

hockey,	foot	tball	,go	lf,so	occer	,bas	sketl	ball,	baseball,play,sport
Vector	0	1	2	3	4	5	6	7	
Doc1	0	0	0	0	11	0	0	6	
Doc2	0	2	0	7	0	0	4	0	
Doc3	0	0	0	0	0	0	0	0	
Doc4	a	12	a	a	a	a	a	a	

The final frequency table for the Tech classification type:

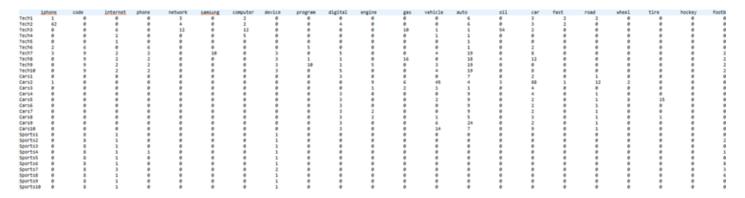


The final frequency table for the Sports classification type:



After this revision we integrated the individual occurrence tables and feature vectors to gather the occurrences of all vectors within our documents. By combining the rows in each of these tables we have the total occurrences of each feature vector and the label corresponding to the document it is found in.

Document-Term Matrix:



Tech Feature Vector

Tech Vector Group											
iphone	code	internet	phone	network	Samsung computer device	program	digital				

The tech feature vector comprises ten common terms that are common in articles of that nature, and limited in articles that are not related to tech. These words were selected in order to maximize the accuracy of our limited training set.

As for occurrences across our training sets the following snippets of the DTM show occurrences relative to each of the possible classifications.

	iphone	code	internet	phone	network	Samsung	computer	device	program	digital
Vector#	0	1	2	3	4	5	6	7	8	9
Tech 1	1	0	0	0	3	0	2	0	0	0
Tech 2	62	0	0	0	4	0	2	0	0	4
Tech 3	0	0	6	0	12	0	12	0	0	0
Tech 4	0	0	1	0	0	0	5	0	0	0
Tech 5	0	0	1	0	0	0	0	0	0	0
Tech 6	2	6	0	6	0	0	0	0	5	0
Tech 7	3	9	2	2	0	10	0	3	0	5
Tech 8	16	0	0	5	0	1	0	3	0	0
Tech 9	0	9	2	2	0	0	0	3	10	1
Tech 10	0	9	2	2	0	0	0	0	0	5

	iphone	code	internet	phone	network	Samsung	computer	device	program	digital
Cars 1	0	0	0	0	0	0	0	0	0	3
Cars 2	1	0	0	0	0	0	0	0	0	0
Cars 3	0	0	0	0	0	0	0	0	0	0
Cars 4	0	0	0	0	0	0	0	0	0	3
Cars 5	0	0	0	0	0	0	0	0	0	3
Cars 6	0	0	0	0	0	0	0	0	0	3
Cars 7	0	0	0	0	0	0	0	0	0	3
Cars 8	0	0	0	0	0	0	0	0	0	3
Cars 9	0	0	0	0	0	0	0	0	0	3
Cars 10	0	0	0	0	0	0	0	0	0	3

	iphone	code	internet	phone	network	Samsung	computer	device	program	digital
Sports 1	0	8	1	0	0	0	0	0	0	0
Sports 2	0	8	0	0	0	0	0	0	0	0
Sports 3	0	8	0	0	0	0	0	0	0	0
Sports 4	0	8	0	1	0	0	0	0	0	0
Sports 5	0	8	0	0	0	0	0	0	0	0
Sports 6	0	8	0	0	0	0	0	0	0	0
Sports 7	0	8	3	0	0	0	0	0	0	0
Sports 8	0	8	0	0	0	0	0	0	0	0
Sports 9	0	8	0	0	0	0	0	0	0	0
Sports 10	0	8	0	0	0	0	0	0	0	0

Cars Feature Vector

The car feature vectors were chosen based on common car words we were seeing. There was slightly more overlap with the sports and tech vectors as words like fast and auto are used there too.

	Car Vector Group													
en	gine		gas		vehicle	auto		oil	car	fast	road	wheel	tire	
		10		11	1	2	13	14	15	16	17	18	19	

For the sports vectors there was again some overlap but we mostly found that they accurately classified the articles.

	Sports Vector Group												
hockey football golf soccer basketball baseball play sport run kick													
	20	21	22	23	24	25	26	27	28	29			

Data Classification

Likelihood Class Prior Probability
$$P(c \mid x) = \frac{P(x \mid c)P(c)}{P(x)}$$
Posterior Probability Predictor Prior Probability

$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

:https://uc-r.github.io/naive bayes (Image)

Using the Naive Bayes classifier method we were able to augment our scraper to define the articles it read as one of our target classification topics, technology, automotive, or sports.

Figure 1

	iphone	code	internet	phone	network	Samsung	computer	device
Vector	0	1	2	3	4	5	6	7
Tech 1	1	0	0	0	3	0	2	0
Tech 2	62	0	0	0	4	0	2	0
Tech 3	0	0	6	0	12	0	12	0
Tech 4	0	0	1	0	0	0	5	0
Tech 5	0	0	1	0	0	0	0	0
Tech 6	2	6	0	6	0	0	0	0
Tech 7	3	9	2	2	0	10	0	3
Tech 8	16	0	0	5	0	1	0	3
Tech 9	0	9	2	2	0	0	0	3
Tech 10	0	9	2	2	0	0	0	0
Cars 1	0	0	0	0	0	0	0	0
Cars 2	1	0	0	0	0	0	0	0

By selecting a set of words that would be common to articles of each perspective typing we were able to form the feature vectors mentioned previously and form a DTM.

(Above a subset of the data is shown, however some columns are left out due to space constraints, Fig 1.)

Table 2. Data Set of word frequency in sports articles as vectors

	Hockey	Golf	Football	Soccer	Basketball	Baseball	Play	Sports	Run	Kick
sports 1	0	0	0	0	4	0	6	2	1	0
sports 2	0	0	0	0	0	0	12	0	4	11
sports 3	0	0	0	0	0	0	4	1	0	0
sports 4	0	0	3	0	0	0	0	0	1	0
sports 5	0	0	0	0	0	0	6	0	0	0
sports 6	0	0	3	28	0	0	11	1	1	1
sports 7	0	0	3	0	3	0	0	10	0	0
sports 8	0	0	3	0	0	0	0	0	0	0
sports 9	0	0	0	0	14	0	3	2	0	0
sports 10	0	1	0	0	16	0	13	3	1	0
total	0	1	12	28	40	0	55	18	8	12

As seen in the graph each column represents a word and each row represents the article and the word frequency seen through numbers.

This table and the due to the word hockey not being mentioned throughout the 10 articles view. If we were to implement an algorithm to calculate the word frequency of these target vectors, we would see an absence of them as they would be skipped over due to the lack of impact. However, when we see the word "basketball" or "play" have a greater impact when searching those words due to the frequency.

Using this DTM we were able to apply our test training set in order to extract data used in the Naive Bayes Classifier method

(Below a subset of the data is shown, however some columns are left out due to space constraints, Fig. 3)

Figure 3

	iphone	code	internet	phone	network	Samsung	computer	device	program	digital	engine	gas
Tech	84	33	14	17	19	11	21	9	15	15	5	10
Cars	1	0	0	0	0	0	0	0	0	24	14	8
Sports	0	80	4	1	0	0	0	0	0	0	0	0
Total	85	113	18	18	19	11	21	9	15	39	19	18

These numbers assist us in learning the Class prior probability, likelihood and predictor probability which will be used to classify additional articles.

Something interesting to note is that there are no occurrences of the target word "hockey" that is specified in the target vector. Hence, this term shall be ignored from any calculations.

When adding the First Unknown article to be classified, the first real classification of the scraping tool, an article was selected and the only word it contained from the targets was the term "basketball", which it included four times. This leaves our Naive Bayes equation as follows:

$$P\left(\frac{Cars}{Basketball}\right) = \frac{P\left(\frac{Basketball}{Cars}\right) \cdot P(Cars)}{P(Basketball)} = 0$$

$$P\left(\frac{\textit{Tech}}{\textit{Basketball}}\right) = \frac{P\left(\frac{\textit{Basketball}}{\textit{Tech}}\right) \cdot P(\textit{Tech})}{P(\textit{Basketball})} = .058$$

$$P\left(\frac{Sports}{Basketball}\right) = \frac{P\left(\frac{Basketball}{Sports}\right) \cdot P(Sports)}{P(Basketball)} = .863$$

Now that the weight for each article has been calculated based upon the addition to the dataset (the unknown article), we can calculate the probability score, whichever it is closest to is going to be our classification.

$$P(Unkown) = \frac{P(\frac{Basketball}{Unkown}) \cdot P(Unkown)}{P(Basketball)} = 1$$

As "basketball" is the only term to appear in the unknown article the program defines the probability score as 1, which is closest to the probability score of Sports, from our judgement with the data alone this seems to be a fitting score. All the data and calculations used in the above are referenced from the excel appendix sheet which is zipped with this file.

Conclusion

In conclusion, we found that our web crawler accurately scrapped words from articles to create a document-term matrix. The use of the Naive Bayes calculation was successful and all 6 tests we did with new articles were shown to be accurate due to the minimal overlap of key vector words between the three groups. Overall, this experiment was a success and showed how classification algorithms can be used in AI and machine learning to learn from articles and return relevant data.

References

Naïve Bayes Classifier · UC Business Analytics R Programming Guide . (2021). Retrieved 30 July 2021, from https://uc-r.github.io/naive_bayes