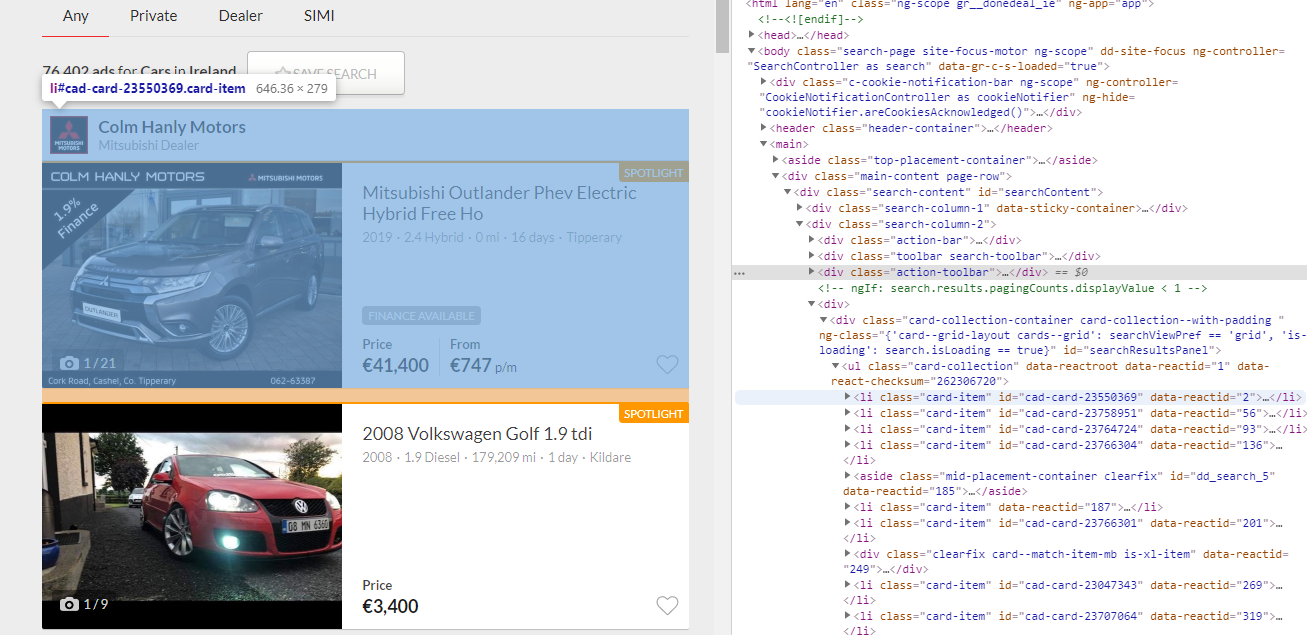
Week of Monday 22nd of March 2021 – 28th March 2021

Ideas for C.A.

1. Scrape data from done deal used car sales and apply some analysis to the resulting dataset. Some analytical ideas:
   1. Predict the depreciation value year on year.
      1. Dovetails nicely into the sort of things we are doing in class
   2. Predict the price
      1. Again, dovetails nicely into the sort of things we are doing in class
   3. Use CNN image classification to try and predict the accuracy of an advertisement i.e., is a car in the condition it is advertised to be.
      1. Seems a bit complex, but very interesting.
      2. Wouldn’t require as much scraping.
2. Try and predict the sentiment of some large text dataset or scrape my own i.e. twitter
   1. Get to use NLP an area I have a lot of interest in
   2. Lots of large readily available datasets, may require no scraping
   3. NLP is not an area that has been covered so far/ at all
   4. Not a lot of features in a data set so might not offer the range of experiences as the scraping approach.
   5. If no scraping is needed I Don’t get to lean into my programming, an area I am strong in

Just wanted to see if the scraping was viable, it seems simple enough to scrape



* <div class="main-content page-row"> contains the listings on the given page and the empty spaces around it
* <div class="search-content"> contains the listings on the given page
* <div class="search-column-2"> contains listings and action bar, not search options
* <div class="card-collection-container ..."> contains the list of car and general ads
* <ul data-reactroot class="card-collection"> unordered list tab

Next challenge is to get to other pages from this first page.

After I have parsed a page's contents and you want to move on to the next page, I will need to extract the link to the next page.

The next page is formatted as a button.

The card id can be gotten from the links also.

When you press the next button, the url changes from

* [*https://www.donedeal.ie/cars?source=private*](https://www.donedeal.ie/cars?source=private) : page 1
* to [*https://www.donedeal.ie/cars?source=private&start=28*](https://www.donedeal.ie/cars?source=private&start=28) : page 2
* and then to [*https://www.donedeal.ie/cars?source=private&start=56*](https://www.donedeal.ie/cars?source=private&start=56) : page 3

The final page tested has the url

* [*https://www.donedeal.ie/cars?source=private&start=23968*](https://www.donedeal.ie/cars?source=private&start=23968) : page 857

Thus, the number appended to the url is (page number - 1) \* 28

Then I need to extract info from the adds.

All the relevant data seems to be stored in a json object located at window.adDetails.

So, the scraping process would be:

1. Start with page ones url.
   1. Use a for loop to access each ad on the page.
      1. With the add extract the json object
         1. Append it to a list of dictionaries.
   2. Repeat for the next ad
   3. When there is no adds left exit the loop
2. Apply the (page number - 1) \* 28 formula to the url
3. Repeat steps a b and c until all cars are extracted.

Week of Monday 29th of March 2021 – 4th April 2021

So, after scraping all the data I was only left with a csv that was about 50mb in size. So, I put it to one side and looked a bit deeper into the NLP and Sentiment analysis

Dataset Candidates

1. Amazon product data

* 1. <http://jmcauley.ucsd.edu/data/amazon/>
     1. 42.8 million reviews spanning May 1996 - July 2014.
     2. Size depends on the subcategory used
     3. JSON – never used in this context before, difficult to use?
     4. Popular Subcategories
        1. Skills learned using this may be useful down to line if I come across these types of datasets again.

1. Multi-Domain Sentiment Dataset
   1. <http://www.cs.jhu.edu/~mdredze/datasets/sentiment/>
      1. Again based on amazon reviews
      2. Dataset seems pre-processed for the purpose of sentiment analysis
         1. Might not leave much room for programming, a strength of miine
      3. 1.5g archived
      4. Data looks a bit more complex including XML files etc.
         1. Could end up biting off more than I can chew.
2. Paper Reviews Data Set
   1. <https://archive.ics.uci.edu/ml/datasets/Paper+Reviews>
      1. reviews from English and Spanish languages on computing and informatics conferences
      2. again in json format, implications of this?
      3. Only 500kb, extremely small
      4. I like the idea of attempting to use sentiment analysis across two different languages using the google translate api and seeing how of effects the accuracy of some model

I downloaded the 3 datasets and played around with them a bit. It seems that the Amazon product data from <http://jmcauley.ucsd.edu/data/amazon/> seems to be the best fit. Specifically, the Video Game Reviews. I like the idea of training some model on the amazon dataset and then down the line maybe trying it on steam reviews? To see how accurate the model is. The data is large enough, but not too large to cause my system any problems. It is also unprocessed, leaving me a good chunk of work to do. I am going to go ahead with this for the assignment knowing I have the used car dataset as back up should things go wrong.

Week of Monday 5th of April 2021 – 13th April 2021

The first aim is to get the data into the jupyter notebook environment. I have not really used json files in this context before, but I have used them in web development, so I understand the over all structure. Luckily Python dictionaries are almost identical to the json objects, so it was really just a case of iterating through each line, appending it to a list and storing it as a list of dictionaries.

The second step was to prepare the data for sentiment analysis. I created a new reduced dataframe, containing the sample dataset. The goal here is to balance the sample set so that it is evenly distributed across negative, positive and neutral user scores and in turn user reviews. So, anything with a type 1 and 2 is negative, anything 3 is neutral and anything 4 or 5 is positive. That equals 60000 records for each category, for a total of 180000 records in the sample set.

A note on how I arrived at 180000 records totals. This took a lot of trial and error in terms of the length of time the sample dataset took to train and test later on during the machine learning section. I originally wanted the largest possible sample set that was also evenly balanced, which was 637038 records. The training and testing on this sample sets sat at about 9 hours on my desktop before I cancelled it. Due to how resource heavy the training and testing was, it was not viable to leave the process unattended. I arrived at 180000 by staring small and working up, which was a nice, sweet spot of total records in the sample set and processing resources needed.

The data pre-processing is straight forward in the sense of the understanding required behind why its implemented but a sticking point for me is the stop wards. Stop words are words such “as” “no” “a “and “the” that for the most part are not useful in sentiment analysis but removing them is a bit of a double edge sword. Sometimes some of these words can be useful. I try and think what kind of stop words might be used in a review from which sentiment could be inferred but are words that are already in the stop words nltk library, thus would be a mistake to be removed.

For example, “I will never play this game”. Or “I will not buy the sequel” or “This is not only the best game this year it’s the best game ever!”. I created a short set of words I felt would be important enough to not include in the stop words. I did not want to add too many just in case it effected the sentiment analysis in some way.

Training and Testing the models is a pain. It can take up to a few hours at a time while there is no feedback from the jupyter notebook that the training or testing has crashed. I tried making the sample datasets smaller just to test some things out but as I kept adding more and more data and the model building took longer a longer I would get wildly differing results in the classification reports so I didn’t see much use in that approach and had to just stick it out.