Comparing LinkedIn Job Ads for Data Scientists and Data Analysts

Group 2 : Clayton Chan, Curtis Pan, Yamini Sharma

STA 141B Winter 2023

Project Outline

Motivation

Our project comapres Linkedin job advertisements for data analysts and data scientists. We hope to gain insights into what kind of companies are looking for data analysts/scientists and what they are looking for in a candidate. This can be helpful to anyone looking to break into the data analytics industry (a majority of our class) and help them understand better what kind of qualifications they should have and potentially help them with the application process. We'll also attempt to find where the line separates data analysts from data scientists as these terms haven't been well defined in the job market lately.

Research Questions

We use our analysis to answer the following questions:

1. What are some key differences between data analysts and data scientists?
2. How generic are the ads to each other? Within data analysts and data scientists respectively.
3. How similar are data analyst and data scientist ads to each other?

Methods

1. Web Scraping
2. Data Collection
    A. Data Description
    B. Data Cleaning

3. Data Visualizations
4. Natural Language Processing (NLP)

Analysis

1. Web Scraping

In order to form our dataset from scratch, we used LinkedIn (www.linkedin.com) to manually collect 100 job advertisements for Data Analysts and 100 job advertisement for Data Scientists.

Our web scraper is built upon the requests package to retrieve the html of the webpage for the Linkedin job ads. Then we converted the html into a BeautifulSoup object and found the tag that corresponded to the actual ad body. Once we extracted the actual ad body, we used a lot of regular expressions to retrieve the data.

We ran these url's through the web scrapper in order to collect features like the salary, benefits, languages, education, and machine learning requirements from each job posting. By having equal amounts of data analysts and scientists ads, we can properly compare the differences between the two roles and highlight meaningful differences in terms of salary and education requirements, all to emphasize the expectations for new college graduates considering applying to these positions in industry.

```
url = 'https://www.linkedin.com/jobs/view/3471853855/?alternateChannel=search&refId=pQHPaG22
collect_data(url)
```

```
{'Languages': 4,
 'Salary': '$300,240',
 'Machine Learning': 'Yes',
 'Education': 'PhD',
 'Benefits': 'Yes',
 'Experience': '8',
 'url': 'https://www.linkedin.com/jobs/view/3471853855/?alternateChannel=search&refId=pQHPa
G22V6rYs24Zr4j02Q%3D%3D&trackingId=Nzgf8dJsYLMC9wFU2oGd%2BQ%3D%3D'}
```

2. Data Collection

A. Data Description

Each of our features that we have extracted are as follows.

- Languages - The number of programming languages mentioned in the Linkedin Ad
- Salary - The maximum salary in the job ad
- Machine Learning - Is machine learning required for the position? With Levels Yes/No
- Education - The highest degree of education preferred with levels Bachelors/Masters/PhD/Unspecified.
- Benefits - Are benefits such as 401K or healthcare included? With Levels Yes/No
- Experience - The years of experience required for the role
- Type - What position was the job advertising? Levels Data Analyst/Data Scientist

	Languages	Salary	Machine Learning	Education	Benefits	Experience	url	Type
0	2	96955.0	No	Master	Yes	1.0	https://www.linkedin.com/jobs/view/3249211510/...	Analyst
1	6	96955.0	No	Master	No	1.0	https://www.linkedin.com/jobs/view/3487242175/...	Analyst
2	4	96955.0	No	Bachelor	Yes	1.0	https://www.linkedin.com/jobs/view/3505418682/...	Analyst
3	0	96955.0	No	Unspecified	No	2.0	https://www.linkedin.com/jobs/view/3498974602/...	Analyst
4	6	96955.0	No	Unspecified	No	5.0	https://www.linkedin.com/jobs/view/3508607089/...	Analyst
...	...	...	...	...	...	...	...	...
193	6	144500.0	Yes	Unspecified	Yes	0.0	https://www.linkedin.com/jobs/view/3483125003/...	Scientist
194	5	144500.0	Yes	Master	Yes	0.0	https://www.linkedin.com/jobs/view/3485548898/...	Scientist
195	6	162000.0	No	Bachelor	No	0.0	https://www.linkedin.com/jobs/view/3507782958/...	Scientist
196	4	144500.0	Yes	Master	Yes	3.0	https://www.linkedin.com/jobs/view/3494351211/...	Scientist
197	3	144500.0	Yes	Unspecified	Yes	0.0	https://www.linkedin.com/jobs/view/3496401704/...	Scientist

B. Data Cleaning

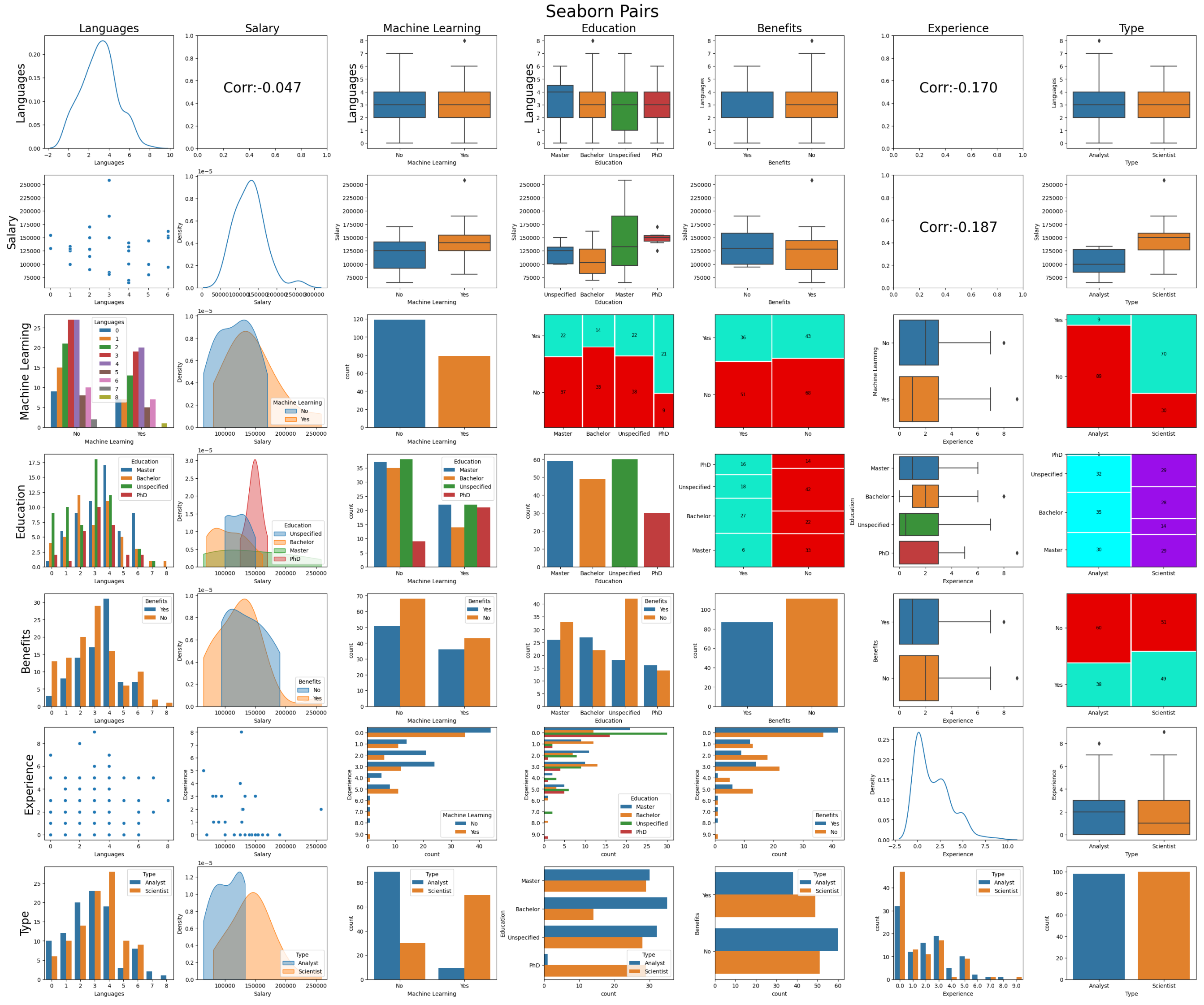
After compiling all 200 urls between the two roles, we collected the features of interest as stated before, using the web scrapper function "collect_data." First , we first dropped any duplicate ads. Then many of these postings however omitted various aspects of the role, such as salary,years of experience, or preferred degree, which we remedied by imputation. We replaced the NA values with the average salary of a data analyst/scientist (depending on the ad) from the subset of ads with salaries listed, creating an "Unspecified" label for ads with no education level mentioned, and years of experience as zero for ads with no experience mentioned (as not mentioning experience may imply that experience is not required). With these problems alleviated, we created pandas dataframes from the csv files of the exported data and concatenated the two separate roles into one dataframe in order to create visualizations and perform other statistical analyses in our comparison.

Furthermore, the web scrapper is not 100% accurate as regular expressions do not always return a unique value. We have ran into some incorrect values like the years of experience being 10,000 years which is obviously not correct and had to double check the ad and manually replace it with the right value.

3. Data Visualizations

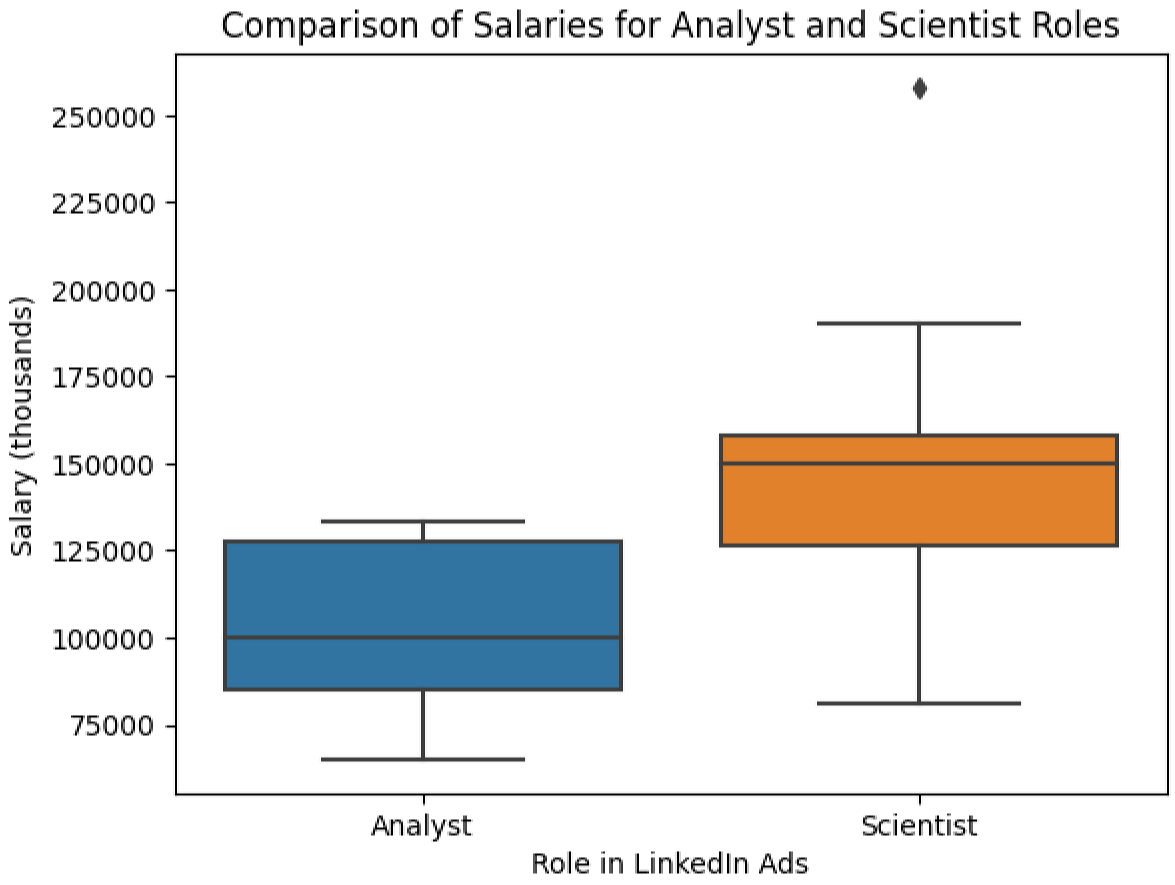
Using seaborn we created a pair plot matrix for all of our qualitative and quantitative variables.

Figure 1



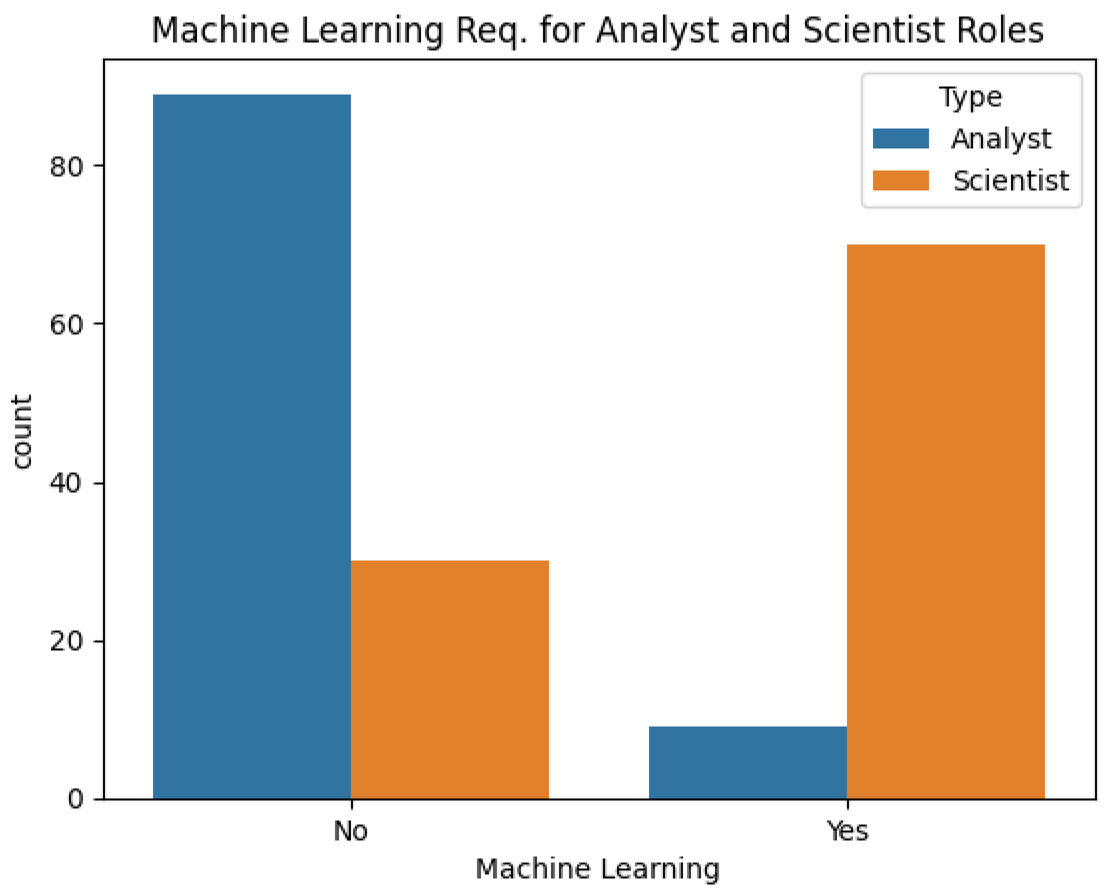
The first main difference we wanted to highlight is the salary disparity between the data analysts and data scientists ads. As illustrated by the above box-plot, scientist ads offer a higher average pay along with a wider salary range overall from eighty thousand to two-hundred thousand compared the analyst ads ranging from sixty-five thousand to one-hundred thirty thousand.

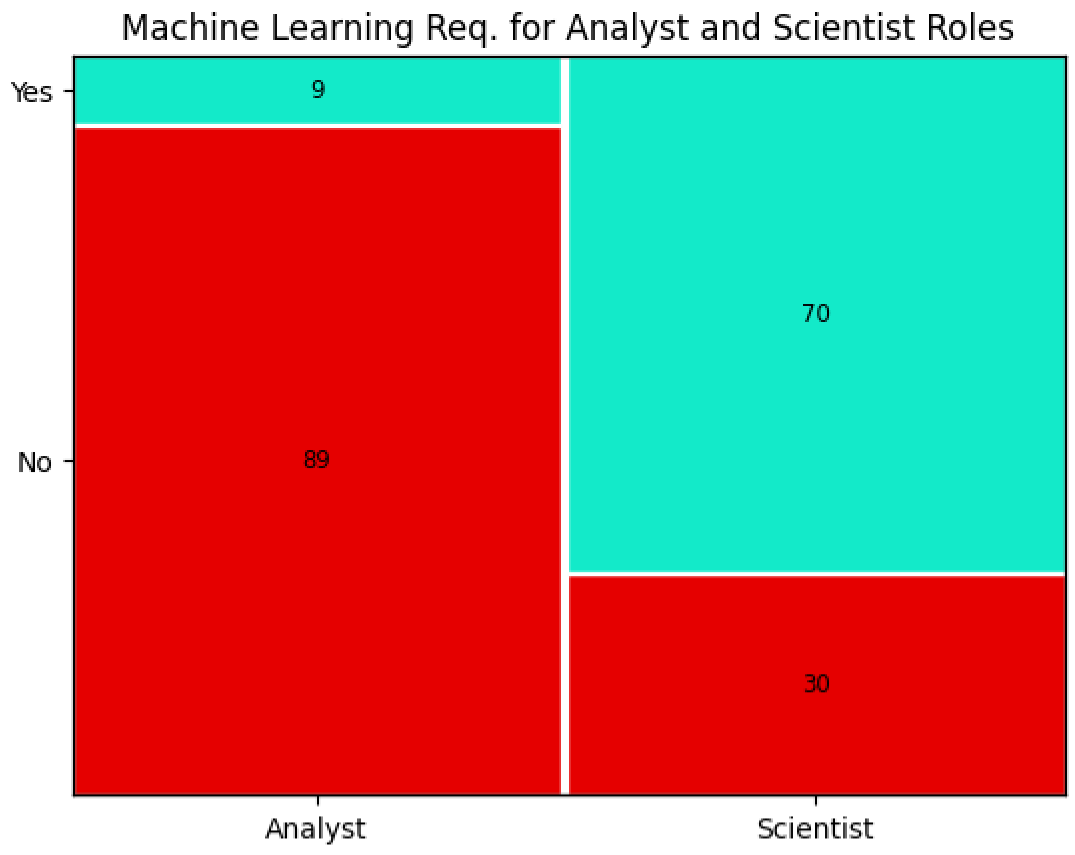
Figure 3



The machine learning requirements for the ads differed great between data analyst and data scientist roles. Almost all postings for analyst positions did not require machine learning, with only nine analyst job ads mentioning machine learning qualifications. On the other hand, machine learning is a requirement for a majority of data scientist positions, with seventy ads requiring machine learning qualifications.

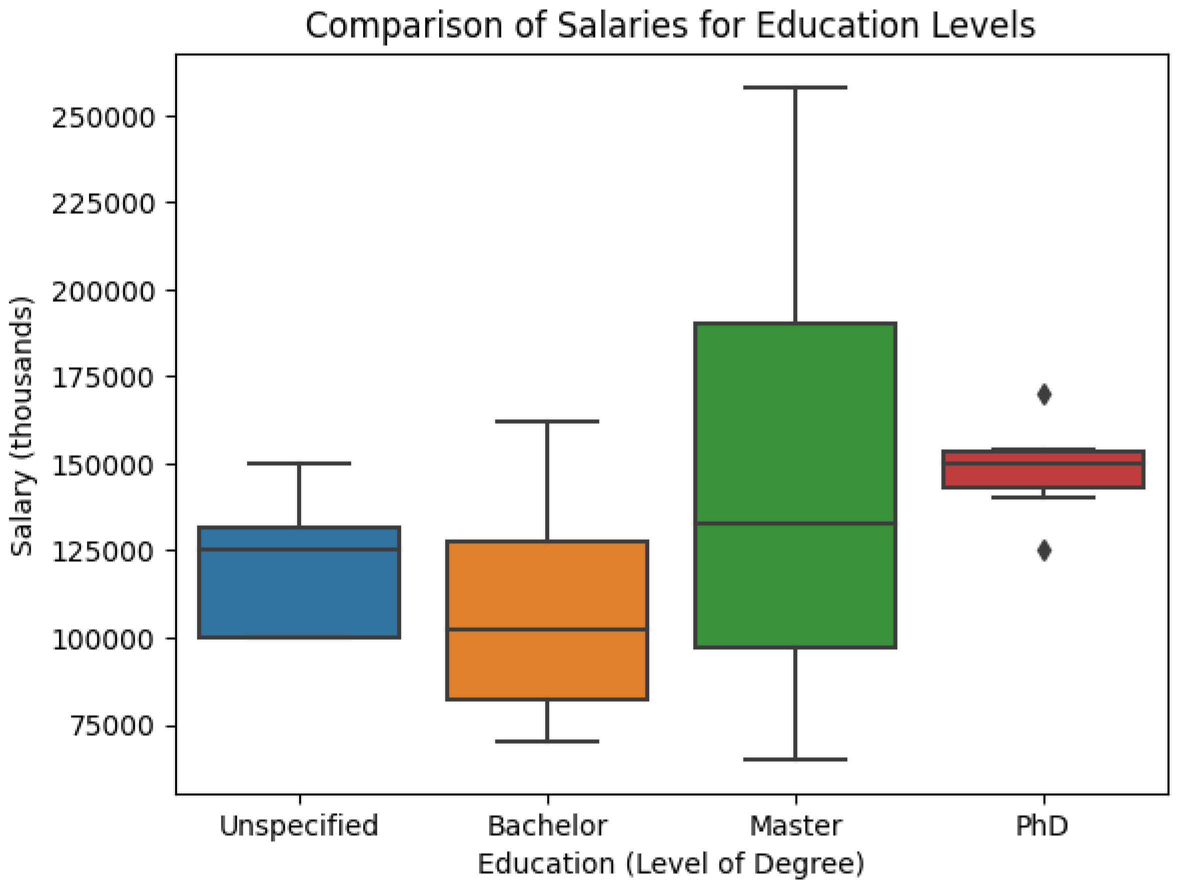
Figure 4





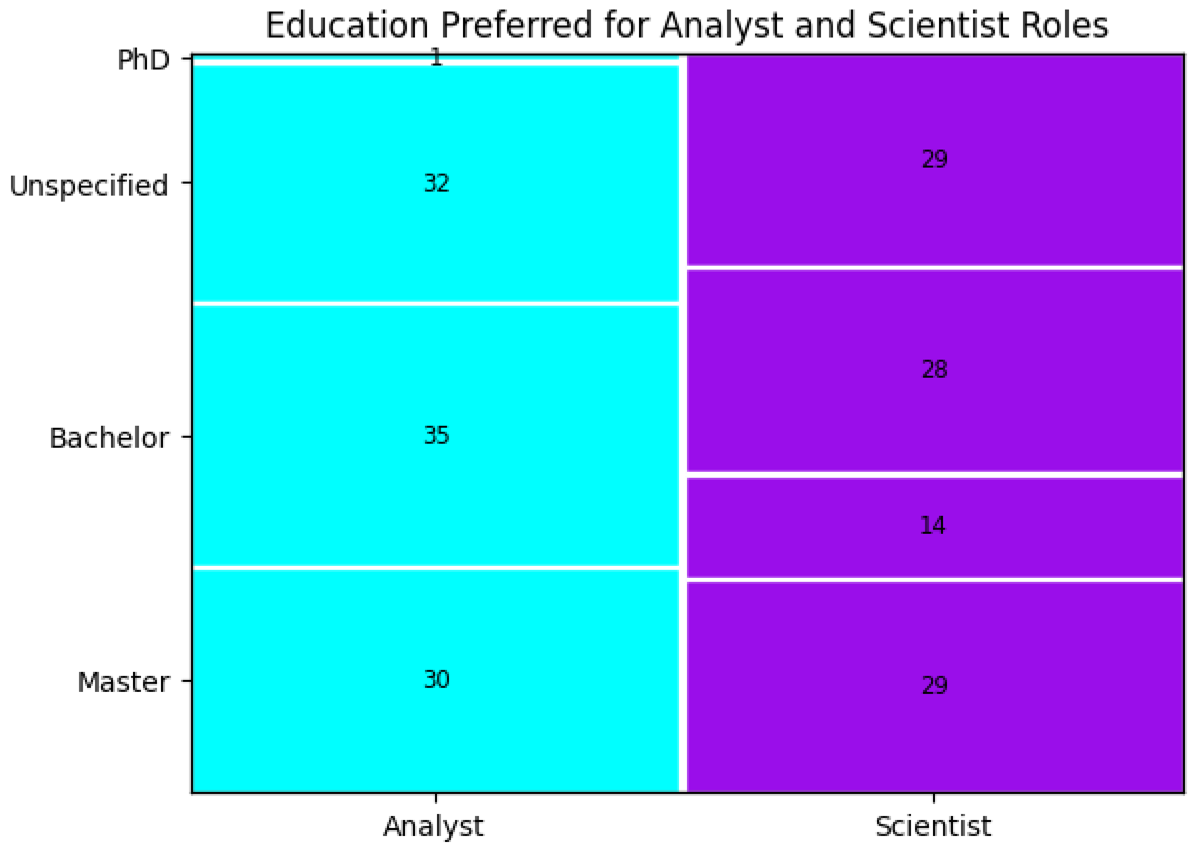
The salary disparity is also emphasized in the difference between education levels required. As expected, positions requiring at most a Bachelors degree has the lowest salary range compared to ads with the PhD or Masters requirement. From the box-plot, a Masters degree has a wide salary range but still a lower median salary compared to the highest education level, PhD.

Figure 5



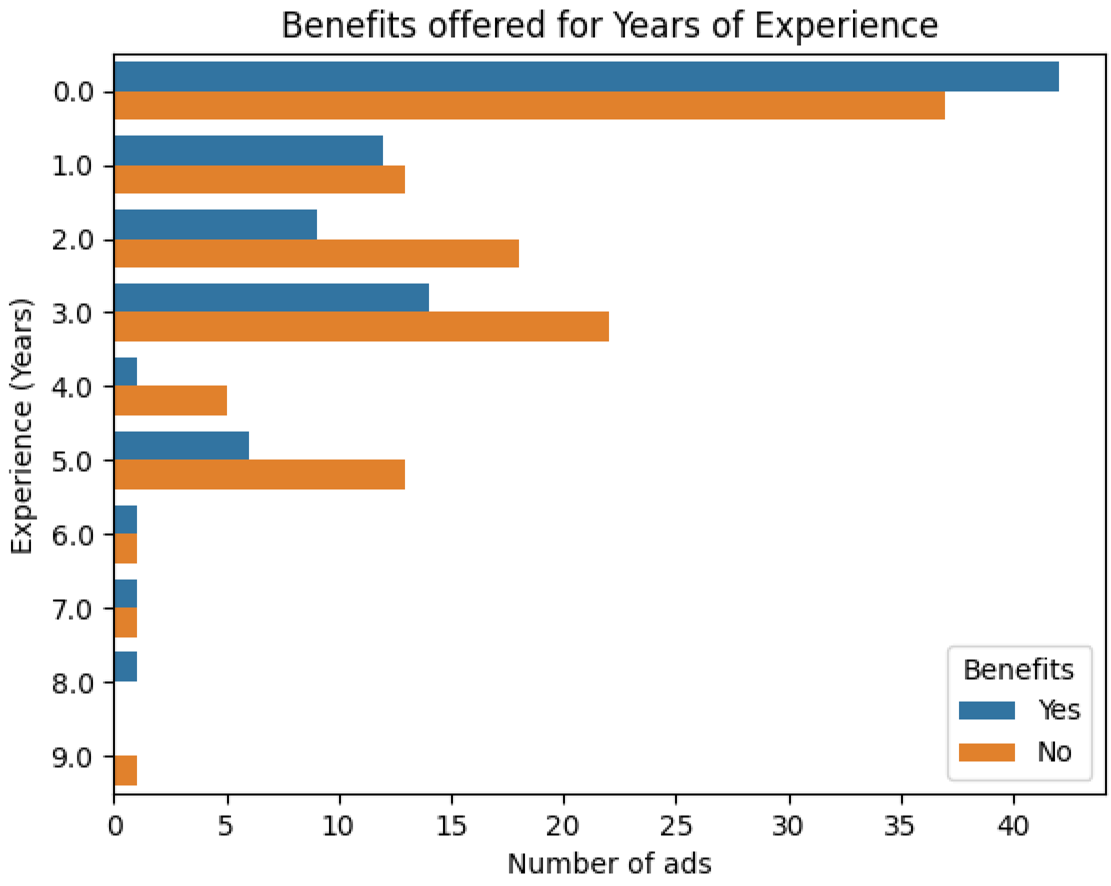
Looking at the education level breakdown between the two roles, the majority of data analyst postings require at most a Bacherlors or Masters; however, data scientist postings mostly require a degree level above a Bachelors degree. Only fourteen ads for data scientist had Bachelors degrees as the highest level of education mentioned, but twenty-nine ads desired a PhD level of education, compared to only one ad wanting a PhD for a data analyst. This suggests there is a high chance the employer will want a PhD for a Data Scientist position while a Bachelor's degree will most likely land you the role of an Analyst.

Figure 6



Surprisingly as the years of experience increases from these various job postings, the amount of benefits offered actually decreases.

Figure 7



4. Natural Language Processing (NLP)

To answer our question regarding how generic Linkedin Ads are, we will use NLP to determine how similar one ad is to another. Since we are purely interested in the vocabulary of each Linkedin Ad, we will employ a bag of words model where order does not matter. Furthermore, since we only care about vocabulary, frequency doesn't matter, only prescence does which is why will make the counts binary. Finally, when it comes to cleaning the text, we will standardize each text by lowercasing and lemmatizing each word. We will then remove common stop words and only include alphabetic characters with the exception of # and + due the importance of not dropping programming languages like C# or C++.

Based on the cosine similarities for both data scientists and analysts, a lot of the documents have a cosine similarity somewhere between .1 and .3. This shows us that the 100 data analyst ads tend to be fairly unique from each other as these values are very close to 0 (least similar) but that they certainly share a few keywords/vocabularly. The same conclusion applies to the data scientist ads also as the results were similar to the data analyst ad comparisons.

Figure 8

Similarity across Data Analyst jobs

	0	1	2	3	4	5	6	7	8	9	...	90	91	92	93	94
0	1.000000	0.244096	0.294795	0.220416	0.227980	0.313014	0.296946	0.212776	0.271024	0.244492	...	0.307538	0.066227	0.279553	0.103905	0.274226
1	0.244096	1.000000	0.273442	0.204450	0.239200	0.318666	0.271386	0.298146	0.215985	0.289584	...	0.211487	0.092144	0.280912	0.168662	0.264022
2	0.294795	0.273442	1.000000	0.231778	0.277188	0.345041	0.330621	0.276305	0.262467	0.299312	...	0.303136	0.089538	0.308533	0.130444	0.348627
3	0.220416	0.204450	0.231778	1.000000	0.172169	0.239793	0.280415	0.271749	0.223807	0.189029	...	0.155440	0.069338	0.209061	0.145048	0.188982
4	0.227980	0.239200	0.277188	0.172169	1.000000	0.302507	0.253035	0.266181	0.169143	0.285262	...	0.234950	0.112867	0.187168	0.121742	0.254380
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	0.225493	0.229565	0.212449	0.153551	0.215580	0.277626	0.246414	0.221398	0.210613	0.188667	...	0.272607	0.166091	0.225352	0.097719	0.274224
96	0.127192	0.149440	0.124468	0.094696	0.122834	0.169738	0.185735	0.148791	0.125459	0.138168	...	0.164018	0.192055	0.154418	0.125550	0.140930
97	0.351159	0.284579	0.331264	0.227043	0.258283	0.341012	0.327547	0.282340	0.267019	0.275795	...	0.303874	0.083722	0.277674	0.131354	0.330579
98	0.351159	0.284579	0.331264	0.227043	0.258283	0.341012	0.327547	0.282340	0.267019	0.275795	...	0.303874	0.083722	0.277674	0.131354	0.330579
99	0.201316	0.186733	0.170581	0.181309	0.138343	0.248687	0.187507	0.167578	0.209123	0.189357	...	0.149168	0.196116	0.187249	0.076923	0.154189

Similarity across Data Scientist jobs

	0	1	2	3	4	5	6	7	8	9	...	90	91	92	93	94
0	1.000000	0.256697	0.290933	0.220796	0.235323	0.338832	0.218250	0.245554	0.339119	0.211435	...	0.267817	0.250670	0.245486	0.216112	0.202579
1	0.256697	1.000000	0.157413	0.176184	0.149467	0.280844	0.161121	0.163702	0.281434	0.159011	...	0.181530	0.135232	0.154662	0.131204	0.157644
2	0.290933	0.157413	1.000000	0.202669	0.190465	0.240778	0.263380	0.268892	0.267111	0.281591	...	0.263191	0.240882	0.243457	0.240954	0.248152
3	0.220796	0.176184	0.202669	1.000000	0.225075	0.250573	0.215562	0.229576	0.295785	0.221248	...	0.284002	0.247582	0.220088	0.252400	0.296087
4	0.235323	0.149467	0.190465	0.225075	1.000000	0.264854	0.345798	0.283688	0.280434	0.261436	...	0.333209	0.347518	0.218326	0.245134	0.228486
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	0.250430	0.266762	0.175680	0.216919	0.173593	0.256099	0.207820	0.206142	0.287055	0.139373	...	0.192972	0.168168	0.206289	0.200002	0.222100
96	0.342643	0.233686	0.228068	0.212826	0.293398	0.305176	0.267617	0.256141	0.357464	0.208088	...	0.278192	0.246827	0.244563	0.239664	0.221928
97	0.219583	0.130931	0.217552	0.174817	0.231939	0.265569	0.283981	0.253683	0.257096	0.170025	...	0.233508	0.275427	0.242812	0.217119	0.209184
98	0.225460	0.174268	0.246888	0.251294	0.319514	0.318123	0.302381	0.333406	0.331198	0.287081	...	0.340324	0.298676	0.308158	0.261420	0.261070
99	0.184356	0.197867	0.253789	0.259554	0.219071	0.236686	0.241602	0.240979	0.280253	0.179456	...	0.279368	0.240979	0.297524	0.241181	0.274467

Based on the cosine similarity, data analysts and scientist ads tend to use pretty similar words/vocabulary as the cosine similarity is roughly .6 which is closer to 1 meaning they're quite similar. However they do have some differences as it's not super close to 1.

Figure 9

	Analyst	Scientist
Analyst	1.000000	0.603135
Scientist	0.603135	1.000000

Below, we can see the top 20 most frequent words for data analyst and scientist ads. They seem to share a lot of frequent words like data (which might not be a surprise), experience, work, skill and etc. While we will not use the frequencies in our analysis for similarity, it's still insightful to look at frequent words these ads like to use.

Frequency of Words - Data Analysts

Figure 10

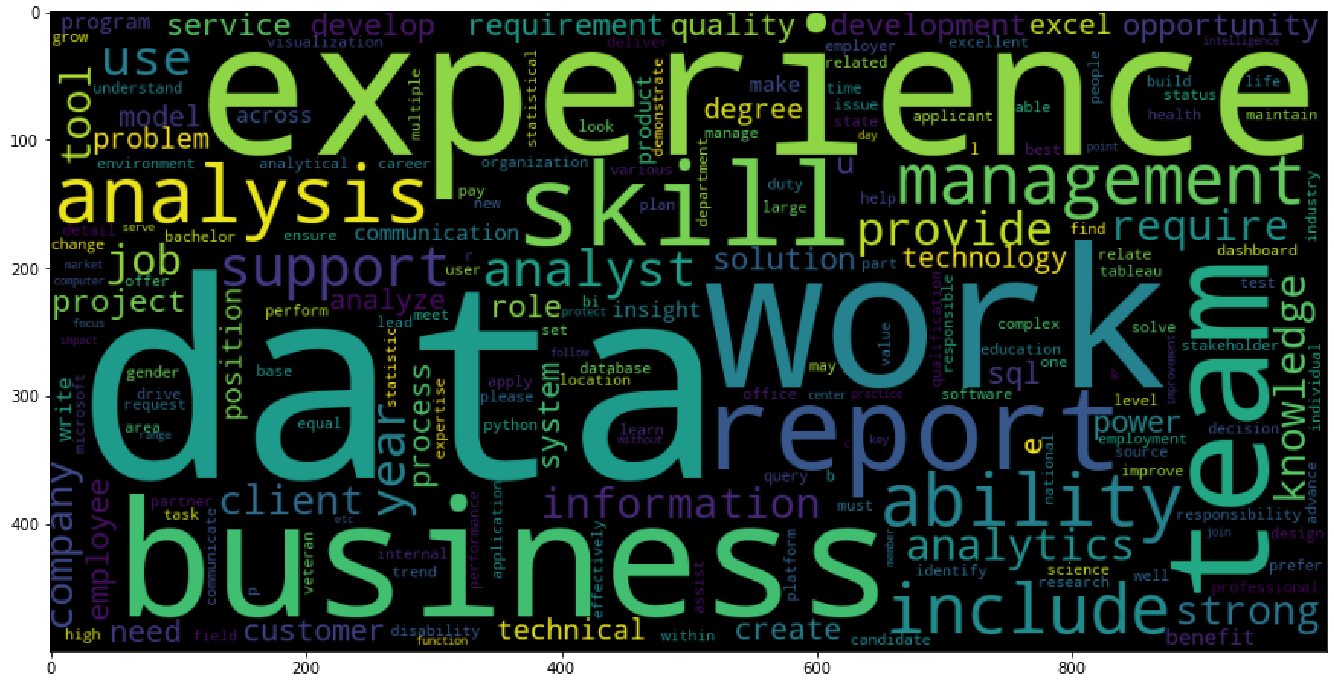
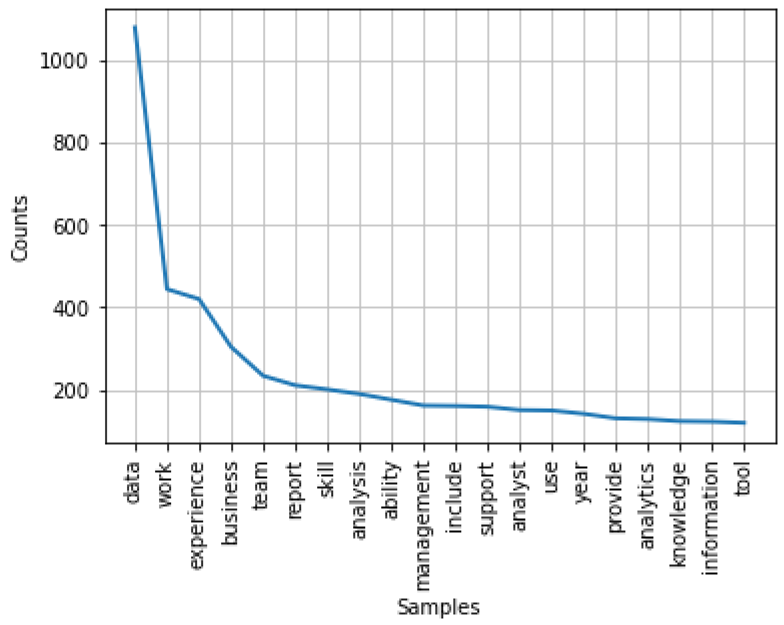


Figure 11



Frequency of Words - Data Scientists

Figure 12

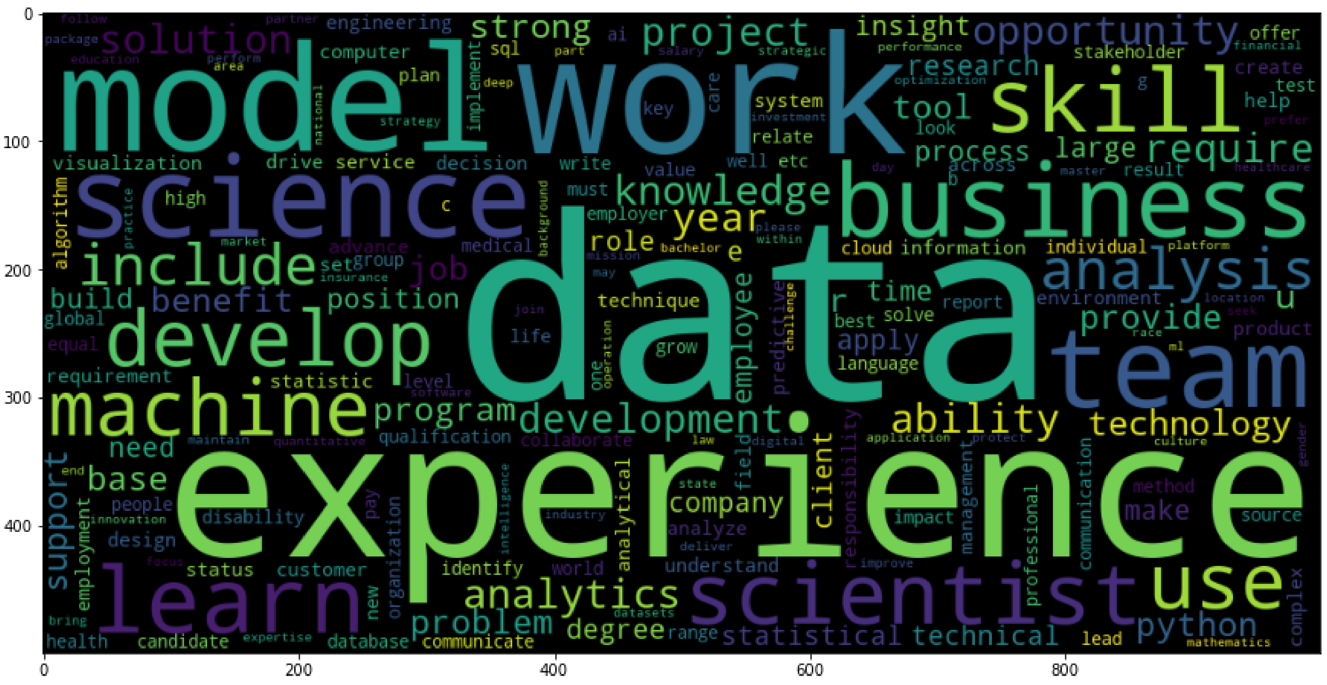
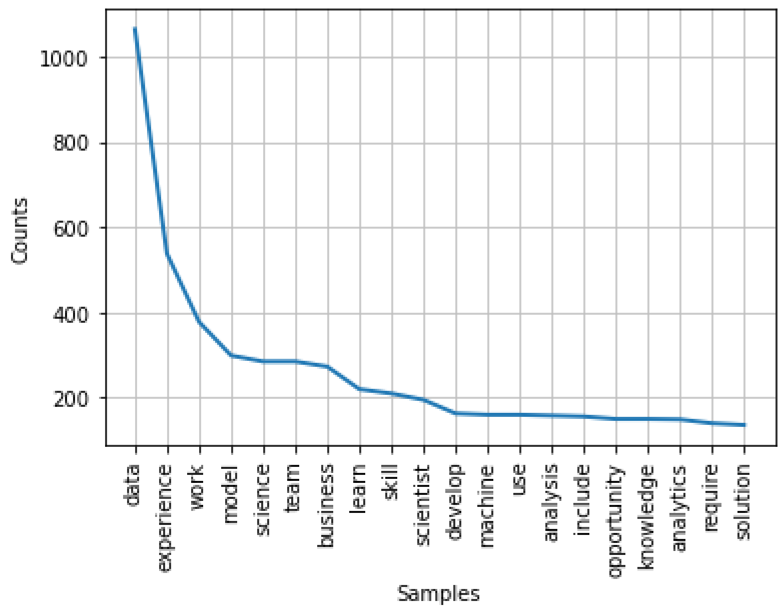


Figure 13



Challenges

We manually collected 200 job advertisements but majority of these ads had missing information. For instance, only 32 ads, out of 200, contained information on salary. We decided to take an average of the salary column and use that figure for the missing salary values. Morever, some advertisements did not specify education requirements, which seems to be a requirement for any job posting. We modified the missing values here with "Unspecified".

Our web scraper was also challenging to write. We had to constantly refine the regular expressions since we kept getting the incorrect values or some errors. We ended up making our regular expressions more specific to fine tune the accuracy but even then, our web scraper is not perfect because regular expressions do not return unique answers. For example, we used a regular expression to return 5/6 digit numbers but there were cases where we did not get the number we wanted which was the salary and we ended up manually changing some of the incorrect values.

In our proposal, we mentioned using machine learning models, mainly clustering. Although we tried clustering and visualizing through k-means and dendograms, we did not see a defining pattern and our data seemed skewed. Due to limited time, we decided to scrap clustering. An improvement for the project could have been spending more time on trying various machine learning models to decipher defining patterns.

Conclusion

From our analysis, we were able to make observations on the differences and similarities between the job ads for Data Analysts and Data Scientists. We learned that most Data Scientist jobs required a PhD as well as machine learning experience whereas Data Analysts are more flexible with a Bachelors or Masters degree. Consequently, one would expect the median salary to be higher for Data Scientists and this also confirmed by our ggpairs graph. A job encompassing Machine Learning leads to a higher median salary since it's a more challenging and demanding architecture to build.

We learnt that Data Analyst and Data Scientist job descriptions are quite similar within their own groups giving us a cosine similarity ranging from 0.1 and 0.3, which is quite close to 1. Both descriptions shared a high frequency of words like "data", "experience", "skills", "team", "technology", helping paint a picture of the expectation for someone thinking of venturing into this field. Our graphs show that both seem to have a similarity in number of languages required. However, the roles might differ in the architecture and complexity of these languages.On comparing Data Analysts with Data Scientist ads, we learned that they are similar but still have key differences, with a cosine similarity of 0.6.

Overall, both roles are highly technical with a promising salary range which is bound to increase with a higher education level. A Data Scientist role will most likely require a PhD and machine learning experience. For someone graduating with a Bachelors degree, it might be more lucrative to apply for Data Analyst roles and eventually transition into Data Science with more experience and a stronger track record with necessary skills.

Code Appendix

In [ ]:

```
## Web Scraping + Data Visualization

#Data collection code
import requests
import numpy as np
import pandas as pd
from bs4 import BeautifulSoup
import nltk
import numpy as np
import re
from nltk.corpus import wordnet
from selenium import webdriver

def raw_text(url):
    """Takes a URL as input and performs web scrapping to retrieve the body of the
    webpage (in this case a Linkedin ad)"""
    ad = requests.get(url) #Retrieve webpage
    Html = BeautifulSoup(ad.text, 'html.parser') #Convert html into a nicer format
    text_body = Html.find_all('div',
                              {'class':"show-more-less-html__markup show-more-less-html_
    text_body = text_body[0].text
    return text_body
def clean_text(doc):
    """Take an unstructured document and tokenize it into a list of words.
    Then standardize it by lowercasing and lemmatizing each word"""
    words = re.findall(r'(?:[a-zA-Z]|#|"+")+',doc) #Find all alphabetical words (Preserv
    clean = [i for i in words if i.isupper() or i.islower()] #Retrieve all words that ar
    dirty = [i for i in words if not i.islower() and not i.isupper()] #Retrieve words st
    dirty = [re.findall('[a-zA-Z][^A-Z]*',i) for i in dirty] #Split all the tangled word
    clean2 = [j for i in dirty for j in i] #Unlist the list of lists
    words = clean + clean2 #Combine all the words together
    stopwords = nltk.corpus.stopwords.words("english")
    words = [i.lower() for i in words] #Lowercase all words
    words = [i for i in words if i not in stopwords] #Filter out stopwords
    tag_words = nltk.pos_tag(words) #Begin lemmatizing by tagging each word
    tag_words = [(i, wordnet_pos(j)) for (i, j) in tag_words] #Convert the tags into som
    lemmatizer = nltk.WordNetLemmatizer()
    clean_words = [lemmatizer.lemmatize(i, j) for i, j in tag_words] #Lemmatize the word
    #Document should be cleaned up
    return clean_words
def wordnet_pos(tag):
    """Map a Brown POS tag to a WordNet POS tag."""

    table = {"N": wordnet.NOUN, "V": wordnet.VERB, "R": wordnet.ADV, "J": wordnet.ADJ}

    # Default to a noun.
    return table.get(tag[0], wordnet.NOUN) #Function created by Bo Ning in Week 6-2
def lang_count(TXT):
    """ Take a body of clean text and count the number of programming languages present"
    languages = ['python','r','sql','sa','c',
                 'c++','c#','java','javascript',
                 'julia','matlab','swift','tableau',
                'microsoft','github','excel'] #SAS turns into sa after lemmatization
    #ADD MORE LANGUAGES IF NECESSARY
    count = sum([i in TXT for i in languages]) #Check if each language is in the ad
    #And sum the number of programming languages present
    return count
def get_salary(TXT):
    """From a body of raw text, retrieve the salary"""
    salaries = re.findall(r"(\$\d+\,\d+\.\d{1,2})",TXT) #Find all numbers with $ , and .
    if salaries != []:
        return salaries[-1] #Let's work with the maximum salary
    else:
        salaries = re.findall(r"(\$\d+\,\d+)",TXT)  #Account for no decimals too!
        if salaries != []:
            return salaries[-1]
        else: #If list is empty no salary is present and return NA
            return None
def ML_skill(TXT):
    """Using a body of clean text, check whether the words machine learning is present
    to see if it is a required skill"""
    if ('machine' in TXT and 'learn' in TXT) or ('ml' in TXT): #Check for the words rela
        return 'Yes' #If it's present return yes as in machine learning is required
    else:
        return 'No'
def get_edu(TXT):
    """Using a body of raw text, retrieve the highest education level"""
    if "PhD" in TXT or 'Ph.D' in TXT: #Start looking for PhD to see if it's the highest
        return "PhD"
    elif "Master" in TXT or 'MS' in TXT or 'MA' in TXT:#If PhD is absent do the same thi
        return "Master"
    elif "Bachelor" in TXT or 'BS' in TXT or 'BA' in TXT:
        return "Bachelor"
    else:
        return None #No education specified
def benefits(TXT):
    """Using a body of raw text, check if benefits are included"""
    if 'Benefit' in TXT or 'benefit' in TXT: #Check if benefit is in the ad to determine
        return 'Yes'
    else:
        return 'No'
def exp(TXT):
    """Using a body of raw text, check if experience is required/preferred"""
    sentences = nltk.sent_tokenize(TXT) #Split text into sentences
    years = [re.findall(r"\d+.*year", i) for i in sentences] #Find sentences with years
    for items in years:
        if items != []:
            years = [i for i in years if i != []][0][0] #Get rid of empty values and tur
            year = re.findall(r'\d+',years)[0]
            return year
    return None #If we make it through the loop years of experience is absent and return
def collect_data(url):
    """Input a URL for a Linkedin Ad and retrieve all relevant data"""
    raw = raw_text(url)
    clean = clean_text(raw)
    return {'Languages':lang_count(clean),
            'Salary':get_salary(raw),
            'Machine Learning':ML_skill(clean),
            'Education':get_edu(raw),
            'Benefits':benefits(raw),
            'Experience':exp(raw),
            'url':url}
```

```
url = [
     'https://www.linkedin.com/jobs/view/3249211510/?alternateChannel=search&refId=VD198Z
     'https://www.linkedin.com/jobs/view/3487242175/?alternateChannel=search&refId=JXABqO
     'https://www.linkedin.com/jobs/view/3505418682/?alternateChannel=search&refId=%2BpaS
     'https://www.linkedin.com/jobs/view/3498974602/?alternateChannel=search&refId=UgS%2B
     'https://www.linkedin.com/jobs/view/3508607089/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3456862041/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3507843178/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3507141700/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3492916139/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3499269839/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3499255853/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3511632332/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3505403106/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3512272186/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3499126864/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3505977879/?alternateChannel=search&refId=mjF%2B
     'https://www.linkedin.com/jobs/view/3509655528/?alternateChannel=search&refId=gPWJCL
     'https://www.linkedin.com/jobs/view/3505977879/?alternateChannel=search&refId=gPWJCL
     'https://www.linkedin.com/jobs/view/3487142091/?alternateChannel=search&refId=eUf97B
     'https://www.linkedin.com/jobs/view/3488691439/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3491778866/?alternateChannel=search&eBP=CwEAAAGG
     'https://www.linkedin.com/jobs/view/3496731411/?alternateChannel=search&refId=eUf97B
     'https://www.linkedin.com/jobs/view/3488630612/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3496102239/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3495674661/?alternateChannel=search&refId=NWABB4
     'https://www.linkedin.com/jobs/view/3491959780/?alternateChannel=search&refId=UgS%2B
     'https://www.linkedin.com/jobs/view/3482557663/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3497655155/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3496092247/?alternateChannel=search&refId=tD86Es
     'https://www.linkedin.com/jobs/view/3493897268/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3496092247/?alternateChannel=search&refId=VHK7B1
     'https://www.linkedin.com/jobs/view/3495686468/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3485531526/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3503792094/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3495654543/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3495654842/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3491485328/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3491485328/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3480285201/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3488092382/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3500262772/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3489487289/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3495634532/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3494529354/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3494564440/?alternateChannel=search&refId=bUuSO6
     'https://www.linkedin.com/jobs/view/3495693106/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3487777637/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3487709897/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3512466597/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3495654381/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3497733873/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3499554593/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3494451117/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3512451344/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3494238784/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3500266283/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3495578342/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3490804946/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3512520788/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3490303130/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3493909748/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3485549081/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3496797402/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3507194381/?alternateChannel=search&refId=dkG0Og
     'https://www.linkedin.com/jobs/view/3488658887/?alternateChannel=search&refId=dkG0Og
```

```
    'https://www.linkedin.com/jobs/view/3505492538/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3505496763/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3496431666/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3511827145/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3512429439/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3512515787/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3505448647/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3507896566/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3505424217/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3505905528/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3500269812/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3511369169/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3495610932/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3507844077/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3507860531/?alternateChannel=search&refId=tjE8t1
    'https://www.linkedin.com/jobs/view/3490606955/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3493466568/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3495647309/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3494073924/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3485575875/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3498811906/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3494592372/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3486412422/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3498130371/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3501081381/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3495815900/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3489481743/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3495476456/?alternateChannel=search&refId=Poisoz
    'https://www.linkedin.com/jobs/view/3489526424/?alternateChannel=search&refId=UgS%2B
    'https://www.linkedin.com/jobs/view/3488623693/?alternateChannel=search&refId=UgS%2B
    'https://www.linkedin.com/jobs/view/3492602300/?alternateChannel=search&refId=UgS%2B
    'https://www.linkedin.com/jobs/view/3503741339/?alternateChannel=search&refId=UgS%2B
    'https://www.linkedin.com/jobs/view/3497727530/?alternateChannel=search&refId=UgS%2B
    'https://www.linkedin.com/jobs/view/3497727530/?alternateChannel=search&refId=UgS%2B
    'https://www.linkedin.com/jobs/view/3488239050/?alternateChannel=search&refId=UgS%2B
]
data_analyst1= [collect_data(i) for i in url[0:10]]
data_analyst2 = [collect_data(i) for i in url[10:20]]
data_analyst3 = [collect_data(i) for i in url[20:30]]
data_analyst4 = [collect_data(i) for i in url[30:40]]
data_analyst5 = [collect_data(i) for i in url[40:50]]
data_analyst6 = [collect_data(i) for i in url[50:60]]
data_analyst7 = [collect_data(i) for i in url[60:70]]
data_analyst8 = [collect_data(i) for i in url[70:80]]
data_analyst9 = [collect_data(i) for i in url[80:90]]
data_analyst10 = [collect_data(i) for i in url[90:100]]
#create one list
data_analyst_final =  (
    data_analyst1 +
    data_analyst2 +
    data_analyst3 +
    data_analyst4 +
    data_analyst5 +
    data_analyst6 +
    data_analyst7 +
    data_analyst8 +
    data_analyst9 +
    data_analyst10) #merge the data_analyst folds
len(data_analyst_final)
#check length of each sublist
print(len(data_analyst1))
print(len(data_analyst2))
print(len(data_analyst3))
print(len(data_analyst4))
print(len(data_analyst5))
print(len(data_analyst6))
print(len(data_analyst7))
print(len(data_analyst8))
print(len(data_analyst9))
print(len(data_analyst10))
url = [
'https://www.linkedin.com/jobs/view/3480836400/?alternateChannel=search&refId=hzJz0m5p3L
'https://www.linkedin.com/jobs/view/3482317582/?alternateChannel=search&refId=hzJz0m5p3L
'https://www.linkedin.com/jobs/view/3485598972/?alternateChannel=search&refId=hzJz0m5p3L
'https://www.linkedin.com/jobs/view/3474104997/?alternateChannel=search&refId=hzJz0m5p3L
'https://www.linkedin.com/jobs/view/3499243413/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3497809413/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3485589284/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3499231522/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3484736484/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3482943863/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3485092940/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3498209260/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3497410604/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3482024773/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3497412440/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3477186036/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3492905077/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3495650895/?alternateChannel=search&refId=%2BF%2BFXj
'https://www.linkedin.com/jobs/view/3477765885/?alternateChannel=search&refId=B74ZU7%2BK
'https://www.linkedin.com/jobs/view/3507858351/?alternateChannel=search&refId=B74ZU7%2BK
'https://www.linkedin.com/jobs/view/3486730652/?alternateChannel=search&refId=B74ZU7%2BK
'https://www.linkedin.com/jobs/view/3510798014/?alternateChannel=search&refId=B74ZU7%2BK
'https://www.linkedin.com/jobs/view/3483520818/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3502126460/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3475904812/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3497488989/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3493573277/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3487708421/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3490335163/?alternateChannel=search&refId=lCMJLZ7NgJ
'https://www.linkedin.com/jobs/view/3500300099/?alternateChannel=search&refId=tx24jCRLjQ
'https://www.linkedin.com/jobs/view/3493932962/?alternateChannel=search&refId=tx24jCRLjQ
'https://www.linkedin.com/jobs/view/3478642744/?alternateChannel=search&refId=tx24jCRLjQ
'https://www.linkedin.com/jobs/view/3478646349/?alternateChannel=search&refId=tx24jCRLjQ
'https://www.linkedin.com/jobs/view/3490315684/?alternateChannel=search&refId=tx24jCRLjQ
'https://www.linkedin.com/jobs/view/3495947949/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3499198986/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3483125003/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3497996600/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3485531966/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3483162019/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3491751485/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3503247533/?alternateChannel=search&refId=J44dYJy3n8
'https://www.linkedin.com/jobs/view/3474104997/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3491480925/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3490390307/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3491783046/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3482039211/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3494351211/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3479875781/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3487876716/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3494440488/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3492155495/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3490320222/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3497740104/?alternateChannel=search&refId=GnHhQQXnnW
'https://www.linkedin.com/jobs/view/3478070149/?alternateChannel=search&refId=4A%2FyoMyb
'https://www.linkedin.com/jobs/view/3500236943/?alternateChannel=search&refId=4A%2FyoMyb
'https://www.linkedin.com/jobs/view/3487708185/?alternateChannel=search&refId=4A%2FyoMyb
'https://www.linkedin.com/jobs/view/3503257941/?alternateChannel=search&refId=4A%2FyoMyb
'https://www.linkedin.com/jobs/view/3494092583/?alternateChannel=search&refId=giEumJfy6z
'https://www.linkedin.com/jobs/view/3497865734/?alternateChannel=search&refId=giEumJfy6z
'https://www.linkedin.com/jobs/view/3483751776/?alternateChannel=search&refId=giEumJfy6z
```

```
'https://www.linkedin.com/jobs/view/3482049975/?alternateChannel=search&refId=giEumJfy6z
'https://www.linkedin.com/jobs/view/3490315892/?alternateChannel=search&refId=giEumJfy6z
'https://www.linkedin.com/jobs/view/3494496847/?alternateChannel=search&refId=giEumJfy6z
'https://www.linkedin.com/jobs/view/3490818587/?alternateChannel=search&refId=%2B2Fnmr3Z
'https://www.linkedin.com/jobs/view/3495956692/?alternateChannel=search&refId=%2B2Fnmr3Z
'https://www.linkedin.com/jobs/view/3498714814/?alternateChannel=search&refId=%2B2Fnmr3Z
'https://www.linkedin.com/jobs/view/3475943161/?alternateChannel=search&refId=15cszUJB8%
'https://www.linkedin.com/jobs/view/3507105466/?alternateChannel=search&refId=15cszUJB8%
'https://www.linkedin.com/jobs/view/3441702869/?alternateChannel=search&refId=gGykCqqW7P
'https://www.linkedin.com/jobs/view/3474339133/?alternateChannel=search&refId=gGykCqqW7P
'https://www.linkedin.com/jobs/view/3296336105/?alternateChannel=search&refId=quOfBYU%2F
'https://www.linkedin.com/jobs/view/3305892542/?alternateChannel=search&refId=quOfBYU%2F
'https://www.linkedin.com/jobs/view/3491905657/?alternateChannel=search&refId=AcP99l%2BE
'https://www.linkedin.com/jobs/view/3443791832/?alternateChannel=search&refId=QT%2FkAD%2
'https://www.linkedin.com/jobs/view/3485843952/?alternateChannel=search&refId=QT%2FkAD%2
'https://www.linkedin.com/jobs/view/3501408792/?alternateChannel=search&refId=THGmPp5pcL
'https://www.linkedin.com/jobs/view/3476789804/?alternateChannel=search&refId=6%2BGdZbNL
'https://www.linkedin.com/jobs/view/3492929121/?alternateChannel=search&refId=6%2BGdZbNL
'https://www.linkedin.com/jobs/view/2914502/?alternateChannel=search&refId=DOcFFWyaGQugL
'https://www.linkedin.com/jobs/view/3480674131/?alternateChannel=search&refId=2qEPhgj%2F
'https://www.linkedin.com/jobs/view/3487706839/?alternateChannel=search&refId=2qEPhgj%2F
'https://www.linkedin.com/jobs/view/3457192628/?alternateChannel=search&refId=2qEPhgj%2F
'https://www.linkedin.com/jobs/view/3507857368/?alternateChannel=search&refId=IpiB3zTYY9
'https://www.linkedin.com/jobs/view/3479875781/?alternateChannel=search&refId=IpiB3zTYY9
'https://www.linkedin.com/jobs/view/3483757696/?alternateChannel=search&refId=IpiB3zTYY9
'https://www.linkedin.com/jobs/view/3358202763/?alternateChannel=search&refId=SfcYnvpclb
'https://www.linkedin.com/jobs/view/3464653380/?alternateChannel=search&refId=SfcYnvpclb
'https://www.linkedin.com/jobs/view/3500478205/?alternateChannel=search&refId=PUPGvLaaEs
'https://www.linkedin.com/jobs/view/3485589284/?alternateChannel=search&refId=pbtG%2BpKc
'https://www.linkedin.com/jobs/view/3507569128/?alternateChannel=search&refId=IN0DvsJIuj
'https://www.linkedin.com/jobs/view/3483520818/?alternateChannel=search&refId=pGBsiuo3aY
'https://www.linkedin.com/jobs/view/3467081732/?alternateChannel=search&refId=P1YoVgcPvP
'https://www.linkedin.com/jobs/view/3249197629/?alternateChannel=search&refId=P1YoVgcPvP
'https://www.linkedin.com/jobs/view/3458734550/?alternateChannel=search&refId=P1YoVgcPvP
'https://www.linkedin.com/jobs/view/3482023892/?alternateChannel=search&refId=Xo0kXhfdA3
'https://www.linkedin.com/jobs/view/3483125003/?alternateChannel=search&refId=BHDOKV6ZbY
'https://www.linkedin.com/jobs/view/3485548898/?alternateChannel=search&refId=NsgoInM8U3
'https://www.linkedin.com/jobs/view/3507782958/?alternateChannel=search&refId=rxVUjOGLI0
'https://www.linkedin.com/jobs/view/3494351211/?alternateChannel=search&refId=rxVUjOGLI0
'https://www.linkedin.com/jobs/view/3496401704/?alternateChannel=search&refId=ozUPHUao4H
data_scientist1= [collect_data(i) for i in url[0:10]]
data_scientist2 = [collect_data(i) for i in url[10:20]]
data_scientist3 = [collect_data(i) for i in url[20:30]]
data_scientist4 = [collect_data(i) for i in url[30:40]]
data_scientist5 = [collect_data(i) for i in url[40:50]]
data_scientist6 = [collect_data(i) for i in url[50:60]]
data_scientist7 = [collect_data(i) for i in url[60:70]]
data_scientist8 = [collect_data(i) for i in url[70:80]]
data_scientist9 = [collect_data(i) for i in url[80:90]]
data_scientist10 = [collect_data(i) for i in url[90:100]]
#create one list
data_scientist_final =  (
    data_scientist1 +
    data_scientist2 +
    data_scientist3 +
    data_scientist4 +
    data_scientist5 +
    data_scientist6 +
    data_scientist7 +
    data_scientist8 +
    data_scientist9 +
    data_scientist10) #merge the data_analyst folds
#creating csv file for 100 data anlyst postings
import csv

keys =data_scientist_final[0].keys()
```

```
with open('data_scientist.csv', 'w', newline='') as output_file:
    dict_writer = csv.DictWriter(output_file, keys)
    dict_writer.writeheader()
    dict_writer.writerows(data_scientist_final)
def f(x):
    x= x.strip("$")
    x= x.replace(',','')
    return x
df.loc[~df ["Salary"].isna(),"Salary"]=df.loc[~df['Salary'].isna(),'Salary'].map(f)
df['Salary']= df['Salary'].astype(float)
#creates csv file
df.to_csv('data_analyst.csv')
#pairs plots
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
import matplotlib.gridspec as gridspec
#read in the csv's
df_analyst=pd.read_csv('data_analyst.csv')
df_scientist = pd.read_csv('data_scientist.csv')
#strips dollar sign/ cleans the salary column
def f(x):
    x= x.strip("$")
    x= x.replace(',','')
    return x
df_scientist.loc[~df_scientist["Salary"].isna(),"Salary"]=df_scientist.loc[~df_scientist
df_scientist['Salary']= df_scientist['Salary'].astype(float)
df_scientist.head()
#combine the two pd dataframes
df = pd.concat([df_analyst.reset_index(drop=True), df_scientist.reset_index(drop=True)],

del df_plot[df_plot.columns[0]] #delete index column
df = df.drop_duplicates()

df_plot=df #just in case to compare with original combined df
df_plot.head()

#Only for the salary pairs
df_na = df.loc[df['Salary'] != 96955.0]
df_na = df_na.loc[df_na['Salary'] != 144500.0]
df_na.to_csv('with_category.csv')
df_na.head() #unfactorized categories dataset

#for everything that isn't salary use df_plot
df_na.to_csv('with_category.csv')
len(df_na)
#figure, axes and axis
fig, axes = plt.subplots(nrows=7, ncols = 7,figsize=(20, 20), layout = 'constrained')

#Language on itself
sns.kdeplot(df_plot, x="Languages",ax=axes[0][0])
ax=axes[0][0].set_title('Languages')
ax=axes[0][0].set_ylabel('Languages')

#Lang-Salary coef
sal_corr=df_na['Languages'].corr(df_na['Salary'])
ax=axes[0][1].text(0.1, 0.5,'Corr:-0.047', fontsize=15)
ax=axes[0][1].set_title('Salary')


#Language-Machine Learning (box)
sns.boxplot(data=df_plot, x="Machine Learning",y="Languages",ax=axes[0][2])
ax=axes[0][2].set_xlabel('Machine Learning')
ax=axes[0][2].set_ylabel('Languages')
```

```
ax=axes[0][2].set_title('Machine Learning')

#Language-Education (box)
sns.boxplot(data=df_plot, x="Education",y="Languages",ax=axes[0][3])
ax=axes[0][3].set_xlabel('Education')
ax=axes[0][3].set_ylabel('Languages')
ax=axes[0][3].set_title('Education')

#Language-Benefits (box)
sns.boxplot(data=df_plot, x="Benefits",y="Languages",ax=axes[0][4])


ax=axes[0][4].set_title('Benefits')

#Language-Experience (correlation coef)
ax=axes[0][5].text(0.1, 0.5,'Corr:-0.170', fontsize=15)


ax=axes[0][5].set_title('Experience')

#Language-Type (box)
sns.boxplot(data=df_plot, x="Type",y="Languages",ax=axes[0][6])
ax=axes[0][6].set_xlabel('Type')
ax=axes[0][6].set_ylabel('Languages')
ax=axes[0][6].set_title('Type')

#Columns
#Language-Salary(scatterplot)
sns.scatterplot(df_na, x="Languages", y="Salary",ax=axes[1][0])
ax=axes[1][0].set_xlabel('Languages')
ax=axes[1][0].set_ylabel('Salary')

#Language-Machine Learning (bar)
sns.countplot(data=df_plot, x="Machine Learning",hue="Languages",ax=axes[2][0])

ax=axes[2][0].set_ylabel('Machine Learning')

#Language-Education (bar)
sns.countplot(data=df_plot, x="Languages",hue="Education",ax=axes[3][0])
ax=axes[3][0].set_ylabel('Education')

#Language-Benefits (bar)
sns.countplot(data=df_plot, x="Languages",hue="Benefits",ax=axes[4][0])
ax=axes[4][0].set_ylabel('Benefits')

#Language-Experience (scatterplot)
sns.scatterplot(df_plot, x="Languages", y="Experience",ax=axes[5][0])
ax=axes[5][0].set_ylabel('Experience')

#Language-Type (bar)
sns.countplot(data=df_plot, x="Languages",hue="Type",ax=axes[6][0])
ax=axes[6][0].set_xlabel('Type')
ax=axes[6][0].set_ylabel('Type')

##Salary
#rows
##Salary itself
sns.kdeplot(df_na, x="Salary",ax=axes[1][1])
##Salary-Machine Learning (box)
sns.boxplot(data=df_na, x="Machine Learning",y="Salary" ,ax=axes[1][2])
##Salary-Education (box)
sns.boxplot(data=df_na, x="Education",y="Salary",ax=axes[1][3])
##Salary-Benefits (box)
sns.boxplot(data=df_na, x="Benefits",y="Salary",ax=axes[1][4])
##Experience (correlation coef)
ax=axes[1][5].text(0.1, 0.5,'Corr:-0.187', fontsize=15)
```

```
##Type (box)
sns.boxplot(data=df_na, x="Type",y="Salary", ax=axes[1][6])

#column
##Sal-Machine Learning (bar) (subfigures)
sns.kdeplot(data=df_na, x="Salary", hue="Machine Learning", cut=0, fill=True, common_nor
#sns.barplot(data=df_na, x="Machine Learning",y="Salary", ax=axes[2][1])
###wei says to just adjust transparency or colors rather than try to make subfigures and

##Sal-Education (bar)
##sns.barplot(data=df_na, y="Salary",x="Education", ax=axes[3][1])
sns.kdeplot(data=df_na, x="Salary", hue="Education", cut=0, fill=True, common_norm=False
##Sal-Benefits (bar)
##sns.barplot(data=df_na, y="Salary",x="Benefits", ax=axes[4][1])
sns.kdeplot(data=df_na, x="Salary", hue="Benefits", cut=0, fill=True, common_norm=False,
##Sal-Experience (scatterplot)
sns.scatterplot(df_na, x="Salary", y="Experience",ax=axes[5][1])
##Sal-Type (stacked bar)
##sns.barplot(data=df_na, x="Type",y="Salary", ax=axes[6][1])
sns.kdeplot(data=df_na, x="Salary", hue="Type", cut=0, fill=True, common_norm=False, alp

#Machine Learning
#Row
##Machine Learning (bar on itself)
sns.countplot(df_plot,x='Machine Learning',ax=axes[2][2])
##Education (mosaic)
from statsmodels.graphics.mosaicplot import mosaic

props={}
props[('Machine Learning','Yes')]={'facecolor':'red', 'edgecolor':'white'}
props[('Machine Learning','No')]={'facecolor':'red', 'edgecolor':'white'}
props[('Bachelor','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Bachelor','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Unspecified','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Unspecified','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Master','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Master','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('PhD','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('PhD','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
labelizer=lambda k:{('Bachelor','Yes'):14,('Master','Yes'):22,('PhD','Yes'):21,('Unspeci
                    ('Bachelor','No'):35,('Master','No'):37,('PhD','No'):9,('Unspecified
mosaic(df_plot,['Education','Machine Learning'],labelizer=labelizer,properties=props,ax=

##Benefits (mosaic)
props={}
props[('Machine Learning','Yes')]={'facecolor':'red', 'edgecolor':'white'}
props[('Machine Learning','No')]={'facecolor':'red', 'edgecolor':'white'}
props[('Yes','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Yes','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('No','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('No','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
labelizer=lambda k:{('Yes','Yes'):36,('No','Yes'):43,
                    ('Yes','No'):51,('No','No'):68}[k]
mosaic(df_plot,['Benefits','Machine Learning'],labelizer=labelizer,properties=props,ax=a
##Experience (box)
sns.boxplot(data=df_plot, x="Experience",y="Machine Learning", ax=axes[2][5])
## Type (mosaic)
props={}
props[('Machine Learning','Yes')]={'facecolor':'red', 'edgecolor':'white'}
props[('Machine Learning','No')]={'facecolor':'red', 'edgecolor':'white'}
props[('Analyst','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Analyst','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Scientist','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Scientist','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
labelizer=lambda k:{('Analyst','Yes'):9,('Scientist','Yes'):70,
                    ('Analyst','No'):89,('Scientist','No'):30}[k]
```

```
mosaic(df_plot,['Type','Machine Learning'],labelizer=labelizer,properties=props,ax=axes[

#ML-Columns
##ML-Education (bars 2*4) stacked bar plot maybe
sns.countplot(data=df_plot, x="Machine Learning",hue="Education", ax=axes[3][2])
##Benefits (stacked bars)
sns.countplot(data=df_plot, x="Machine Learning",hue="Benefits", ax=axes[4][2])
##Experience (horizontal bars)
sns.countplot(data=df_plot, hue="Machine Learning", y="Experience", ax=axes[5][2])
##Type (stacked bars)
sns.countplot(data=df_plot, x="Machine Learning", hue="Type", ax=axes[6][2])

#Education
#row
##Education (itself)
sns.countplot(df_plot,x='Education',ax=axes[3][3])
##Educ-Benefits (mosaic)
props={}
props[('Yes','Benefits')]={'facecolor':'aqua', 'edgecolor':'white'}
props[('No','Benefits')]={'facecolor':'red', 'edgecolor':'white'}
props[('Yes','Bachelor')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('No','Bachelor')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Yes','Unspecified')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('No','Unspecified')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Yes','Master')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('No','Master')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Yes','PhD')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('No','PhD')]= {'facecolor':'xkcd:red','edgecolor':'white'}
labelizer=lambda k:{('Yes','Bachelor'):27,('Yes','Master'):6,('Yes','PhD'):16,('Yes','Un
                    ('No','Bachelor'):22,('No','Master'):33,('No','PhD'):14,('No','Unspe
mosaic(df_plot,['Benefits','Education'],labelizer=labelizer,properties=props,ax=axes[3][
##Educ-Experience (box)
sns.boxplot(data=df_plot, x="Experience",y="Education", ax=axes[3][5])
##Educ-Type(mosaic)
props={}
props[('Analyst','Bachelor')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','Bachelor')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
props[('Analyst','Master')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','Master')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
props[('Analyst','PhD')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','PhD')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
props[('Analyst','Unspecified')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','Unspecified')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
labelizer=lambda k:{('Analyst','Bachelor'):35,('Analyst','Master'):30,('Analyst','PhD'):
                    ('Scientist','Bachelor'):14,('Scientist','Master'):29,('Scientist','
mosaic(df_plot,['Type','Education'],labelizer=labelizer,properties=props,ax=axes[3][6])
#columns
##Benefits (2 bars)
sns.countplot(df_plot,x='Education',hue="Benefits",ax=axes[4][3])
##Experience (4 horizontal bars)
sns.countplot(df_plot,hue='Education',y='Experience',ax=axes[5][3])
##Type (2 rows of 4 columns)
sns.countplot(data=df_plot, y="Education",hue="Type",ax=axes[6][3])

#Benefits
#Rows
##itself
sns.countplot(data=df_plot, x="Benefits",ax=axes[4][4])
##Experience
sns.boxplot(data=df_plot, x="Experience",y="Benefits", ax=axes[4][5])
##Type (mosaic)
props={}
props[('Analyst','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Analyst','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Scientist','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Scientist','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
```

```
labelizer=lambda k:{('Analyst','Yes'):38,('Scientist','Yes'):49,
                    ('Analyst','No'):60,('Scientist','No'):51}[k]
mosaic(df_plot,['Type','Benefits'],labelizer=labelizer,properties=props,ax=axes[4][6])
#Column
##Experience (horizontal bars)
sns.countplot(data=df_plot, y='Experience',hue="Benefits",ax=axes[5][4])
##Type
sns.countplot(data=df_plot, y='Benefits',hue="Type",ax=axes[6][4])

#Experience
#rows
##itself (line)
sns.kdeplot(df_plot, x="Experience",ax=axes[5][5])
##Type
sns.boxplot(data=df_plot, x="Type",y="Experience", ax=axes[5][6])

#columns
##Type
sns.countplot(df_plot, x="Experience",hue="Type",ax=axes[6][5])
##Type on itself
sns.countplot(data=df_plot, x="Type",ax=axes[6][6])

fig.suptitle(t = 'Seaborn Pairs', fontsize = 30)

#chosen plots/visuals most important differences
sns.boxplot(data=df_na, x="Type",y="Salary")
#for slides box plot of salary-type
#specify axis labels
plt.xlabel('Role in LinkedIn Ads')
plt.ylabel('Salary (thousands)')
plt.title('Comparison of Salaries for Analyst and Scientist Roles')
plt.show()

#For slides Machine Learning for the two roles
sns.countplot(data=df_plot, x="Machine Learning", hue="Type").set(title="Machine Learnin

## ML-Type (mosaic)
props={}
props[('Machine Learning','Yes')]={'facecolor':'red', 'edgecolor':'white'}
props[('Machine Learning','No')]={'facecolor':'red', 'edgecolor':'white'}
props[('Analyst','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Analyst','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
props[('Scientist','Yes')]={'facecolor':'xkcd:aqua','edgecolor':'white'}
props[('Scientist','No')]= {'facecolor':'xkcd:red','edgecolor':'white'}
labelizer=lambda k:{('Analyst','Yes'):9,('Scientist','Yes'):70,
                    ('Analyst','No'):89,('Scientist','No'):30}[k]
mosaic(df_plot,['Type','Machine Learning'],labelizer=labelizer,properties=props)

plt.ylabel('Machine Learning')
plt.title('Machine Learning Req. for Analyst and Scientist Roles')
plt.show()


#box for education and salary
sns.boxplot(data=df_na, x="Education",y="Salary")
plt.xlabel('Education (Level of Degree)')
plt.ylabel('Salary (thousands)')
plt.title('Comparison of Salaries for Education Levels')
plt.show()

##Educ-Type(mosaic)
props={}
props[('Analyst','Bachelor')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','Bachelor')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
props[('Anal
       yst','Master')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
```

```
props[('Scientist','Master')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
props[('Analyst','PhD')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','PhD')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
props[('Analyst','Unspecified')]={'facecolor':'xkcd:cyan','edgecolor':'white'}
props[('Scientist','Unspecified')]= {'facecolor':'xkcd:violet','edgecolor':'white'}
labelizer=lambda k:{('Analyst','Bachelor'):35,('Analyst','Master'):30,('Analyst','PhD'):
                    ('Scientist','Bachelor'):14,('Scientist','Master'):29,('Scientist','
mosaic(df_plot,['Type','Education'],labelizer=labelizer,properties=props)
plt.ylabel('Salary (thousands)')
plt.title('Education Preferred for Analyst and Scientist Roles')
plt.show()

#bar/countplot for years of experience and benefits offered
sns.countplot(data=df_plot, y='Experience',hue="Benefits")
plt.xlabel('Number of ads')
plt.ylabel('Experience (Years)')
plt.title('Benefits offered for Years of Experience')
plt.show()

#crosstable for benefits-ML (for mosaic plots)
crosstable=pd.crosstab(df_plot['Benefits'],df_plot['Machine Learning'])
crosstable
crosstable=pd.crosstab(df_plot['Benefits'],df_plot['Education'])
crosstable
crosstable=pd.crosstab(df_plot['Type'],df_plot['Machine Learning'])
crosstable
crosstable=pd.crosstab(df_plot['Type'],df_plot['Education'])
crosstable
crosstable=pd.crosstab(df_plot['Type'],df_plot['Benefits'])
crosstable

#Pearson correlations
sal_corr=df_na['Languages'].corr(df_plot['Salary'])
sal_corr
#Pearson correlation for benefits
exp_corr=df_plot['Languages'].corr(df_plot['Experience'])
exp_corr
#Pearson correlation for salary and experience
sal_exp_corr=df_na['Salary'].corr(df_na['Experience'])
sal_exp_corr
df_na.groupby('Education')['Salary'].describe()
df_na.groupby('Type')['Salary'].describe() #box plot specifics for salary-type
```

In [ ]:

```
## NLP

import pandas as pd
import requests
import numpy as np
import pandas as pd
from bs4 import BeautifulSoup
import nltk
import numpy as np
import re
from nltk.corpus import wordnet
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.preprocessing import normalize
import matplotlib.pyplot as plt
from wordcloud import WordCloud
def raw_text(url):
    """Takes a URL as input and performs web scrapping to retrieve the body of the
    webpage (in this case a Linkedin ad)"""
    ad = requests.get(url) #Retrieve webpage
    Html = BeautifulSoup(ad.text, 'html.parser') #Convert html into a nicer format
    text_body = Html.find_all('div',
                              {'class':"show-more-less-html__markup show-more-less-html_
    text_body = text_body[0].text
    return text_body
def clean_text(doc):
    """Take an unstructured document and tokenize it into a list of words.
    Then standardize it by lowercasing and lemmatizing each word"""
    words = re.findall(r'(?:[a-zA-Z]|#|"+")+',doc) #Find all alphabetical words (Preserv
    clean = [i for i in words if i.isupper() or i.islower()] #Retrieve all words that ar
    dirty = [i for i in words if not i.islower() and not i.isupper()] #Retrieve words st
    dirty = [re.findall('[a-zA-Z][^A-Z]*',i) for i in dirty] #Split all the tangled word
    clean2 = [j for i in dirty for j in i] #Unlist the list of lists
    words = clean + clean2 #Combine all the words together
    stopwords = nltk.corpus.stopwords.words("english")
    words = [i.lower() for i in words] #Lowercase all words
    words = [i for i in words if i not in stopwords] #Filter out stopwords
    tag_words = nltk.pos_tag(words) #Begin lemmatizing by tagging each word
    tag_words = [(i, wordnet_pos(j)) for (i, j) in tag_words] #Convert the tags into som
    lemmatizer = nltk.WordNetLemmatizer()
    clean_words = [lemmatizer.lemmatize(i, j) for i, j in tag_words] #Lemmatize the word
    #Document should be cleaned up
    return clean_words
def wordnet_pos(tag):
    """Map a Brown POS tag to a WordNet POS tag."""

    table = {"N": wordnet.NOUN, "V": wordnet.VERB, "R": wordnet.ADV, "J": wordnet.ADJ}

    # Default to a noun.
    return table.get(tag[0], wordnet.NOUN) #Function created by Bo Ning in Week 6-2
def unlist(LIST):
    """Take a list and concatenate all objects inside"""
    STRING = ''
    for i in LIST:
        STRING = STRING +' '+ i
    STRING = STRING.strip() #Get rid of white space
    return STRING
analyst = pd.read_csv('data_analyst.csv')
science = pd.read_csv('data_scientist.csv')
def f(x):
    x = x.strip('$')
    x = x.replace(',','')
    return x
science.loc[~science['Salary'].isna(),'Salary'] = science.loc[~science['Salary'].isna(),
science['Salary'] = science['Salary'].astype(float)
science
df = pd.concat([analyst,science])
df = df.iloc[:,1:]
df = df.drop_duplicates() #Drop duplicate ads
df = df.reset_index(drop = True)
analysts = df.loc[df['Type'] == 'Analyst','url']
analysts = analysts.tolist() #Extract all the urls for Data Analyst Ads
scientists = df.loc[df['Type'] == 'Scientist','url']
scientists = scientists.tolist() #Extract all the urls for Data Scientist Ads
anl_text1 = [raw_text(i) for i in analysts[:10]] #Retrieve the body of the first 10 Link
#Ads for Data Analysts. Do this to the rest of the ads for Analysts and Scientists.
anl_text2 = [raw_text(i) for i in analysts[10:20]]
anl_text3 = [raw_text(i) for i in analysts[20:30]]
anl_text4 = [raw_text(i) for i in analysts[30:40]]
anl_text5 = [raw_text(i) for i in analysts[40:50]]
anl_text6 = [raw_text(i) for i in analysts[50:60]]
anl_text7 = [raw_text(i) for i in analysts[60:70]]
anl_text8 = [raw_text(i) for i in analysts[70:80]]
anl_text9 = [raw_text(i) for i in analysts[80:90]]
anl_text10 = [raw_text(i) for i in analysts[90:]]
sci_text1 = [raw_text(i) for i in scientists[:10]]
sci_text2 = [raw_text(i) for i in scientists[10:20]]
sci_text3 = [raw_text(i) for i in scientists[20:30]]
sci_text4 = [raw_text(i) for i in scientists[30:40]]
```

```
sci_text5 = [raw_text(i) for i in scientists[40:50]]
sci_text6 = [raw_text(i) for i in scientists[50:60]]
sci_text7 = [raw_text(i) for i in scientists[60:70]]
sci_text8 = [raw_text(i) for i in scientists[70:80]]
sci_text9 = [raw_text(i) for i in scientists[80:90]]
sci_text10 = [raw_text(i) for i in scientists[90:]]
anl_text =  (anl_text1 + anl_text2 + anl_text3 + anl_text4 + anl_text5 +
            anl_text6 +anl_text7 + anl_text8 + anl_text9 + anl_text10) #Combine
#all the bodies of the text into one list
sci_text = (sci_text1 + sci_text2 + sci_text3 + sci_text4 + sci_text5 + sci_text6 +
           sci_text7 + sci_text8 + sci_text9 + sci_text10) #Do the same for scientists
clean_anl = [clean_text(i) for i in anl_text] #Clean the text by lowercasing lemmatizing
#And removing stopwords with our function clean_text
clean_sci = [clean_text(i) for i in sci_text]
anl_par = [unlist(i) for i in clean_anl] #Put the words back into sentences so we can
#Use our model
sci_par = [unlist(i) for i in clean_sci]
vec = CountVectorizer(tokenizer = nltk.word_tokenize, binary = True)
#Fit a bag of words model where order doesn't matter and the "frequency" is binary
#for data analyst ads
freq = vec.fit_transform(anl_par)
sim = normalize(freq.todense(), axis = 1, norm = 'l2') #Normalize vectors
similarity = pd.DataFrame(sim @ sim.T) #Calculate cosine similarity
((similarity <.3) & (similarity>.1)).values.sum()
vec = CountVectorizer(tokenizer = nltk.word_tokenize, binary = True) #Repeat this proces
freq = vec.fit_transform(sci_par)
sim = normalize(freq.todense(), axis = 1, norm = 'l2')
similarity = pd.DataFrame(sim @ sim.T)
((similarity <.3) & (similarity>.1)).values.sum()
anly = unlist(anl_par) #Pool 100 data analyst ads into one document
sci = unlist(sci_par) #Pool 100 data scientist ads into one document
vec = CountVectorizer(tokenizer = nltk.word_tokenize, binary = True) #Fit a bag of words
#and make the "frequency" binary
freq = vec.fit_transform([anly,sci])
sim = normalize(freq.todense(), axis = 1, norm = 'l2') #Normalize the vectors
similarity = pd.DataFrame(sim @ sim.T, index = ['Analyst','Scientist'],
                          columns = ['Analyst','Scientist']) #Compute cosine similarity
words = clean_text(anly) #Split all the words in the 100 data analyst ads into words
fq = nltk.FreqDist(w for w in words if w.isalnum()) #Count the frequencies of the words
wordcloud = WordCloud(width = 1000, height = 500).generate_from_frequencies(fq)
#Create a word cloud for most frequent words in data analyst ads
plt.figure(figsize=(15,8))
plt.imshow(wordcloud)
fq.plot(20) #Create a frequency plot
words = clean_text(sci) #Repeat this process for data scientists
fq = nltk.FreqDist(w for w in words if w.isalnum())
wordcloud = WordCloud(width = 1000, height = 500).generate_from_frequencies(fq)
plt.figure(figsize=(15,8))
plt.imshow(wordcloud)
fq.plot(20)
```

```
In [ ]:
```