# SwatCloud Job Recommendation System

**Introduction:**

SwatCloud (d.b.a SiliconValley4U) is a small privately owned company in San Ramon, CA which provides educational services and resources to high school and college students with a particular focus on preparing its students for a future career in the technology industry.

The company wanted to implement its own recommendation system that would match its students with active job listings that best fit each person's individual background and experience. Because each student actively posts projects & blog posts to the Swatcloud website, the company wanted to incorporate these accomplishments through an in-house model instead of just relying on a third-party job board that usually focuses on just a resume (e.g. LinkedIn & Indeed)

**Data Wrangling - HTML Scraping:**

The first task at hand is to construct a job listing board. Initially, we attempted to scrape the job listings from LinkedIn and Indeed but these websites block attempts to use HTML scraping on their websites and explicitly prohibit such action in their terms of agreement. The workaround solution to this was to go to tech company career websites and scrape jobs directly from the source.

The scraping process is done by using BeautifulSoup and Selenium Webdriver. The URL for each company's careers page is fed into the soup and then the job titles and job descriptions are extracted by identifying all relevant text underneath the key tags/classes that these features are stored under. Note that the specific syntax for each company differs depending on the webpage's HTML layout, but the overall process flow remains the same. However, this means that each time a new company is added to the database it will require manual effort to repurpose the scraping code to specifically handle the respective career page. Additionally, should a company's webpage change anything about its HTML layout, it would also require manual effort to refresh its respective extraction function before it can work again. See below for an example using Google:

Google

```python
def google_job_description(title, link):
    qualifications = []
    description = []
    jobtitle = []
    joblink = []

    driver=webdriver.Chrome('chromedriver',options=chrome_options)
    for i in range(len(link)):
        s = ''
        d = ''

        soup = get_html(driver, link[i])
        try:
          tag = soup.find("h3", text='Minimum qualifications:').parent.find("ul")
          if tag:
            s = s + tag.text
        except: pass

        try:
          tag = soup.find('h3', text='Preferred qualifications:').parent.find("ul").findNextSibling('ul')
          if tag:
            s = s + tag.text
        except: pass

        try:
          tag = soup.find("div", {'id': 'accordion-responsibilities'})
          if tag:
            d = tag.text
        except: pass

        qualifications.append(s)
        description.append(d)
        jobtitle.append(title[i])
        joblink.append(link[i])
    driver.quit()

    return jobtitle, link, qualifications, description
```

```python
def scrape_jobs_google():
    # retrieve job titles and job links
    df_title_link = get_titles_links_byUrl('h2', 'class', 'gc-card__title gc-heading gc-heading--beta',
                                           'a', 'class', 'gc-card', 'https://careers.google.com',
                                           google_url1, google_url2)
    print(df_title_link.shape[0])

    # retrieve the qualification and descriptions for each job.
    title, link, qual, descrp = google_job_description(df_title_link['JOB_TITLE'].values[:test], df_title_link['JOB_LINK'].values[:test])

    post_process_and_ouput('Google', title, link, qual, descrp)
```

Once this process is completed for each company, the outputs are merged into a single dataframe and saved as a csv file to use in our following steps (each company's output is also saved individually as separate csv files for convenience). See below for the result:

| | Company | Job Title | Job Link | Job Description |
|---|---|---|---|---|
| 0 | Microsoft | Lead Technical Animator – Gears of War – The C... | https://careers.microsoft.com/us/en/job/150444... | Must have 5+ years of experience in AAA video... |
| 1 | Microsoft | Software Engineer - CTJ | https://careers.microsoft.com/us/en/job/151261... | Bachelor's Degree in Computer Science, or rel... |
| 2 | Microsoft | ESO Datacenter Operations Specialist | https://careers.microsoft.com/us/en/job/151199... | High School Diploma or equivalent AND 1+ year... |
| 3 | Microsoft | ESO Datacenter Program Manager | https://careers.microsoft.com/us/en/job/151198... | High School Diploma or equivalent AND 1+ year... |
| 4 | Microsoft | Senior Software Engineer | https://careers.microsoft.com/us/en/job/148279... | 4+ years of experience in software developmen... |
| ... | ... | ... | ... | ... |
| 10782 | IBM | Treasury Front Office Professional | https://careers.ibm.com/job/17070819/treasury-... | 5+ years managing external relationships with... |
| 10783 | IBM | Certified Special Security Officer (CSSO) | https://careers.ibm.com/job/17024978/certified... | 5+ years of experience in Industrial Security... |
| 10784 | IBM | 2023 Financial Analyst Co-op | https://careers.ibm.com/job/17072357/2023-fina... | Country/Region:US State:NEW YORK City:Armonk ... |
| 10785 | IBM | Digital Sales Specialist - Storage | https://careers.ibm.com/job/17155837/digital-s... | A provable track record of 3 years consultati... |
| 10786 | IBM | IBM Public Cloud HPC Architect | https://careers.ibm.com/job/17170704/ibm-publi... | 5+ years deep demonstrated experience as a cl... |

10787 rows × 4 columns

The main shortfall to our workaround approach is that it will leave out several job listings from smaller companies and/or non-tech companies with technology roles, but nevertheless this process provides a solid foundation upon which we build the basic version of our model. Future iterations of this project can easily improve by adding more companies and/or increasing the scope of our available job listings.

**Exploratory Data Analysis & Data Preprocessing - Industry Labeling:**

Once we have our job descriptions ready, our next step is to preprocess our job descriptions so that it's ready to feed into our recommendation models. We also add an industry label to each job listing in this step which will be a key feature in our NLP model.

We remove blank spaces and special characters from the job descriptions and then use WordNetLemmatizer and NLTK's stopwords package to polish the data so that only the keywords of interest in each description are being used in our model.

Because there are a variety of different jobs for each company, we use CountVectorizer to identify what the most common words are in our job titles. The results are shown below:

```
#Counts of different words in the job titles
vectorizer = CountVectorizer(ngram_range = (1,4),stop_words = 'english')
cv = vectorizer.fit_transform(agg)
counts = pd.DataFrame(vectorizer.fit_transform(agg).sum(axis=0),
                      columns=vectorizer.get_feature_names())
counts = counts.T
common_words = counts[0].sort_values(ascending=False)[:50]
print(common_words)
```

```
manager             2676
engineer            1768
senior              1762
associate           1093
software             936
tax                  719
management           679
business             662
data                 630
cloud                578
services             552
intern               547
software engineer    542
engineering          541
specialist           536
consultant           506
lead                 501
program              486
```

This gives us a high level overview of what types of industries our job board is composed of. We can then start creating different industry labels to encompass these job titles and assigning specific keywords to each of these industries. To label all of our jobs with an industry category, there are two steps:

1. Identify if a job title has any of the specific keywords used in our industry labels. If so, then assign that industry to that job.
2. If there are no keywords identified in the title, then the code will loop through the job description and assign an industry label to the job based on which industry's keywords showed up the most in the description.

Our final industry counts and the resulting data frame looks as follows:

```
Software Engineering    4458
Operations & Finance    2935
Data Analysis           1298
Sales                    996
Cybersecurity & IT       478
Marketing                338
Research Scientist       284
```

| | Industry | Company | Job Title | Job Description |
|---|---|---|---|---|
| 0 | Software Engineering | Amazon | Senior Software Development Engineer | 4 year professional software development expe... |
| 1 | Software Engineering | Amazon | Software Development Engineer - Payments | programming experience least one modern langu... |
| 2 | Software Engineering | Amazon | Software Development Engineer - Fintech | bachelor degree computer science related field... |
| 3 | Software Engineering | Amazon | Software Development Engineer | 1 year experience contributing system design a... |
| 4 | Software Engineering | Amazon | Embedded Software Development Engineer, Satell... | 1 year experience contributing system design a... |

**Recommendation System Part 1 - Direct Matching to Job Listings**

Our recommendation system is actually comprised of two different types of models. The first model matches users to their most suitable job listings based on the cosine similarity between their input (e.g. the user's resume, blog post content & tags, and overall profile on SwatCloud's website) and the job database.

See below for an example of the output - this is a test user who is a current software engineer at Amazon but has an extensive background working in and studying data science

Candidate #2: Software Engineer @ Amazon with academic background in computer science and data science

```
top_x_recommendations(10,df,'-Worked on the backend team to develop the Edtera web application, a learning engagement platform, using the Java Spring
```

```
                                      Job Title    Company  Similarity  \
0                  Federal - Senior Data Architect  Accenture    0.417321
1       Federal - Data Strategy & Management Manager  Accenture    0.380363
2              Data Engineering Solution Associate   Deloitte    0.373577
3                     SAP Data Management Consultant        IBM    0.372678
4                        Data Engineer Summer Intern        JnJ    0.360744
5                     Senior Cloud Big Data Engineer        JPM    0.355947
6         Data Analytics Practitioner with Polygraph   Deloitte    0.355252
7   Senior Data Scientist - Machine Learning Opera...        JPM    0.351399
8             Federal - Data Strategy Senior Manager  Accenture    0.342962
9                 Senior Manager - Data Engineering         HP    0.342095

               Match
0  Very Strong Match
1       Strong Match
2       Strong Match
3       Strong Match
4       Strong Match
5       Strong Match
6       Strong Match
7       Strong Match
8         Good Match
9         Good Match
```

```
Applicant's qualifications:  worked backend team develop edtera web application learning engagement platform using java spring mvc framework implement
ed data access layer using spring data jpa allow various crud service edtera postgresql database developed performance tracker using spring resttempla
te retrieve student enrollment grade data third party learning management system configured resttemplate interceptor reduce redundancy code built rest
ful service publish data creating rest controller grade course enrollment information etc developed high performance laser health monitoring program p
ython highly recognized course instructor project sponsor selected exhibition department senior design day implemented random forest regression using
scikit learn library predict laser survival rate achieving mape 12 created interactive data visualization web application python dash framework explor
ative analysis designed feature engineering procedure sum time series data convert supervised learning problem

Job Descriptions for the Recommended jobs:
0 .  Job Description:  Support medium-to-large size data engineering projects Architect scalable data environments, integrating cloud technologies par
ticularly Azure, to bridge data across on-premise and cloud environments Work with various data source owners and consuming application stakeholders t
o understand their streaming, batch, unstructured, and structured data requirements Build an understanding of the variety of stakeholder domains requi
red to build analytic insights, and provide clear frameworks for connecting each stakeholder domain's data into a complete picture Translate data doma
in requirements with new or emerging sources to hydrate data sets Evaluate analytic efforts and production application architectures to support data i
ntegration activities 5+ years of experience with two or more of the following: Integrating various Data ingress and Data ingestion and Data Quality p
oints, Creating Data Dependency map, Data Structures and Data Access controls, ETL and/or Data pipelines and/or data process tools such as Databricks,
spark and Data analysis/visualization tools Experience with tools like Azure Databricks, Spark, Synapses, Snowflake, Cloudera, etc. Experience is usin
g the Azure and/or AWS native cloud services for data. Experience in Data Backup and Data migration strategy

1 .  Job Description:  Bachelor's Degree or equivalent experience 5-6 years of experience in defining enterprise data strategy in the areas such as A
I, data management, data governance, data supply chain, data security, data archival, data quality, master data management, data architecture, and/or
cloud data migration for medium to large scale corporations Proven ability to demonstrate how the outputs of data strategy lead to more effective data
management and innovation capabilities Entrepreneurial mindset and ability to identify sources of value from data analytics across client's business u
nits and to define paths toward realizing value that lead and/or contribute to building future product/data strategy approaches in a collaborative man
ner Lead and/or contribute to building future product/data strategy approaches in a collaborative manner Proven experience working with cross-function
al teams (i.e., org change enablement, digital, cloud, design-thinking, cybersecurity) Exhibit ability to use diagnostics and strategy methods, as wel
l as, own and ensure delivery of high-quality analysis and recommendations to clients Expert ability to utilize various research methodologies, media
research, and external data sources Experience in establishing and maintaining data governance processes, including data quality and master data manag
ement (MDM) Build client relationships and senior internal relationships, highlighting the role of data strategy and management Drive efficiency in wo
rk, defining smarter processes and innovation in new approaches Manage and mentor team members to develop technical skills and knowledge Maintain know
ledge of relevant case studies, data, tools and approaches using it to trial new analytical techniques, progressing craft skills of the team Experienc
e and knowledge of cloud data platforms (such as AWS, Azure, Google Cloud Platform and Snowflake) and other data technologies (such as Informatica, Co
llibra, etc.) Experience in business development activities such as responding to request for proposals (RFP), structuring solutions, client orals/pre
sentations, drafting statement of work and managing sales pipeline. Effectively delivering multiple projects simultaneously Data architecture and/or e
ngineering or AI experience is desired Knowledge of statistical techniques and programming languages is desired Certifications in data science or tech
nologies Publications on topics related to innovation - data or otherwise C-suite client experiences is desired US Citizenship required
```

**Recommendation System Part 2 - NLP Model**

The second model in our system is an NLP model which predicts which industries are the most suitable for SwatCloud's users. We implement a basic Keras sequential model with 4 layers as follows:

```
input_dim = 10000
output_dim = 448
filters = 416
kernel_size = 7
num_layers = 1
units_0 = 96
activation = 'softmax'
#dropout = False
lr = 0.0016963
units_1 = 160
units_2 = 224

model = tf.keras.models.Sequential([
tf.keras.layers.Embedding(input_dim,output_dim,input_length=max_length),
tf.keras.layers.Conv1D(filters,kernel_size,activation='relu'),
tf.keras.layers.GlobalMaxPooling1D(),
tf.keras.layers.Dense(96, activation='softmax'),
])


model.compile(
     optimizer=keras.optimizers.Adam(learning_rate=lr),
     loss="sparse_categorical_crossentropy",
     metrics=['accuracy']
)
```

To prepare our input for this model, we take our job dataset and tokenize the job descriptions (using a vocabulary size of 10,000) and then split it into training, validation, and test sets.

Before we begin training the model, we perform cross validation to find the optimal hyperparameters for the following:

- Embedding Layer:
    - Input dimension ranging from 10,000 to 100,000 vocabulary size
    - Output dimension ranging from 32 to 512 vector length
- Convolutional 1D Layer:
    - Number of output filters ranging from 32 to 512
    - Kernel size ranging from 1 to 10
    - We use the ReLu activation function here which is standard for intermediate layers in neural network models.
- Dense Layer(s):
    - Number of dense layers to add ranging from 1 to 3
    - Number of output units in each layer ranging from 32 to 512
    - Activation function (one of ReLU, Tanh, or Softmax)

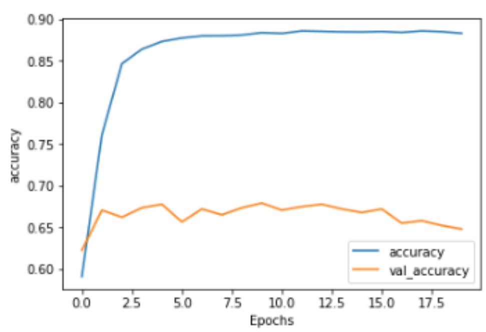
Lastly, we tune whether or not to add a dropout layer and then the learning rate of the model as final hyperparameters. The model is compiled with the Adam optimizer and our model then performs the cross validation.

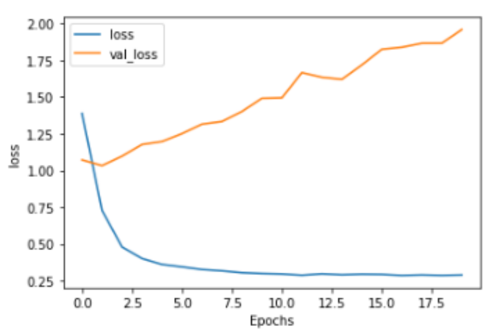
We run 20 epochs through the training & validation set and then evaluate on our test set. The results are shown below:

```
Epoch 1/20
177/177 - 66s - loss: 1.3885 - accuracy: 0.5907 - val_loss: 1.0718 - val_accuracy: 0.6223 - 66s/epoch - 374ms/step
Epoch 2/20
177/177 - 40s - loss: 0.7275 - accuracy: 0.7606 - val_loss: 1.0328 - val_accuracy: 0.6704 - 40s/epoch - 228ms/step
Epoch 3/20
177/177 - 39s - loss: 0.4786 - accuracy: 0.8461 - val_loss: 1.0972 - val_accuracy: 0.6620 - 39s/epoch - 218ms/step
Epoch 4/20
177/177 - 46s - loss: 0.3999 - accuracy: 0.8636 - val_loss: 1.1769 - val_accuracy: 0.6733 - 46s/epoch - 259ms/step
Epoch 5/20
177/177 - 39s - loss: 0.3595 - accuracy: 0.8728 - val_loss: 1.1969 - val_accuracy: 0.6775 - 39s/epoch - 223ms/step
Epoch 6/20
177/177 - 38s - loss: 0.3430 - accuracy: 0.8770 - val_loss: 1.2508 - val_accuracy: 0.6563 - 38s/epoch - 213ms/step
Epoch 7/20
177/177 - 37s - loss: 0.3250 - accuracy: 0.8795 - val_loss: 1.3152 - val_accuracy: 0.6719 - 37s/epoch - 211ms/step
Epoch 8/20
177/177 - 38s - loss: 0.3158 - accuracy: 0.8797 - val_loss: 1.3334 - val_accuracy: 0.6648 - 38s/epoch - 216ms/step
Epoch 9/20
177/177 - 41s - loss: 0.3031 - accuracy: 0.8805 - val_loss: 1.4016 - val_accuracy: 0.6733 - 41s/epoch - 233ms/step
Epoch 10/20
177/177 - 39s - loss: 0.2977 - accuracy: 0.8834 - val_loss: 1.4917 - val_accuracy: 0.6789 - 39s/epoch - 218ms/step
Epoch 11/20
177/177 - 38s - loss: 0.2945 - accuracy: 0.8825 - val_loss: 1.4945 - val_accuracy: 0.6704 - 38s/epoch - 214ms/step
Epoch 12/20
177/177 - 38s - loss: 0.2863 - accuracy: 0.8855 - val_loss: 1.6665 - val_accuracy: 0.6747 - 38s/epoch - 215ms/step
Epoch 13/20
177/177 - 38s - loss: 0.2954 - accuracy: 0.8850 - val_loss: 1.6346 - val_accuracy: 0.6775 - 38s/epoch - 215ms/step
Epoch 14/20
177/177 - 38s - loss: 0.2894 - accuracy: 0.8844 - val_loss: 1.6207 - val_accuracy: 0.6719 - 38s/epoch - 213ms/step
Epoch 15/20
177/177 - 38s - loss: 0.2931 - accuracy: 0.8843 - val_loss: 1.7189 - val_accuracy: 0.6676 - 38s/epoch - 213ms/step
Epoch 16/20
177/177 - 39s - loss: 0.2916 - accuracy: 0.8848 - val_loss: 1.8257 - val_accuracy: 0.6719 - 39s/epoch - 219ms/step
Epoch 17/20
177/177 - 37s - loss: 0.2847 - accuracy: 0.8837 - val_loss: 1.8395 - val_accuracy: 0.6549 - 37s/epoch - 212ms/step
Epoch 18/20
177/177 - 37s - loss: 0.2883 - accuracy: 0.8855 - val_loss: 1.8664 - val_accuracy: 0.6577 - 37s/epoch - 210ms/step
Epoch 19/20
177/177 - 38s - loss: 0.2850 - accuracy: 0.8846 - val_loss: 1.8668 - val_accuracy: 0.6521 - 38s/epoch - 212ms/step
Epoch 20/20
177/177 - 38s - loss: 0.2880 - accuracy: 0.8827 - val_loss: 1.9598 - val_accuracy: 0.6478 - 38s/epoch - 214ms/step
```

```
model.evaluate(test_padded,test_labels)
```

```
23/23 [==============================] - 1s 30ms/step - loss: 1.8302 - accuracy: 0.6455
[1.830186367034912, 0.645480215549469]
```




Although the training accuracy was able to reach ~88%, the accuracy on the test set was only ~65% suggesting that the model is likely overfitting and further improvements should focus on the data quality rather than the model itself. One area of improvement that immediately stands out is the method in how we delineate and select our industry labels. These categories were manually defined based on human assessment of the most common words in the job titles (as demonstrated in the Exploratory Data Analysis & Data Preprocessing section) combined with intuition on how these words are related to the most common industries in today's job marketplace.

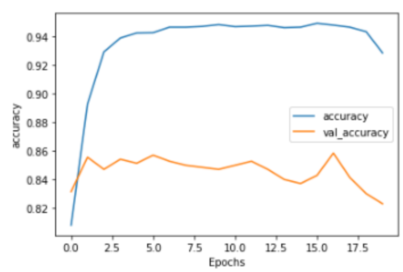
What exact industries we include and/or how we define them are both significant features we can revisit, but even something more derivative like the number of industries we select also matters. For example, we rerun this model on the same dataset but instead of using the seven industry labels as before, this time we only define each job as being a “Technology” job or a “Non-Technology” job.

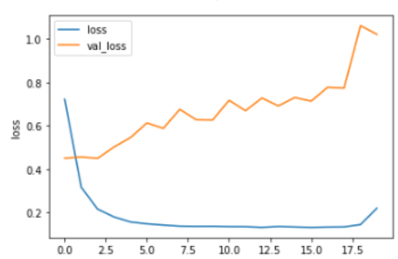
```
model2 = tf.keras.models.clone_model(
    model, input_tensors=None, clone_function=None
)
model2.compile(
     optimizer=keras.optimizers.Adam(learning_rate=lr),
     loss="sparse_categorical_crossentropy",
     metrics=['accuracy']
)
num_epochs = 20
history2 = model2.fit(train_padded2,train_labels2,epochs=num_epochs,validation_data=(val_padded2,val_labels2),verbos
```

```
Epoch 1/20
177/177 - 41s - loss: 0.7221 - accuracy: 0.8083 - val_loss: 0.4499 - val_accuracy: 0.8317 - 41s/epoch - 234ms/step
Epoch 2/20
177/177 - 41s - loss: 0.3164 - accuracy: 0.8929 - val_loss: 0.4557 - val_accuracy: 0.8557 - 41s/epoch - 230ms/step
Epoch 3/20
177/177 - 42s - loss: 0.2148 - accuracy: 0.9293 - val_loss: 0.4492 - val_accuracy: 0.8472 - 42s/epoch - 237ms/step
Epoch 4/20
177/177 - 42s - loss: 0.1786 - accuracy: 0.9390 - val_loss: 0.5018 - val_accuracy: 0.8543 - 42s/epoch - 237ms/step
Epoch 5/20
177/177 - 40s - loss: 0.1556 - accuracy: 0.9426 - val_loss: 0.5451 - val_accuracy: 0.8515 - 40s/epoch - 229ms/step
Epoch 6/20
177/177 - 40s - loss: 0.1469 - accuracy: 0.9427 - val_loss: 0.6129 - val_accuracy: 0.8571 - 40s/epoch - 228ms/step
Epoch 7/20
177/177 - 40s - loss: 0.1414 - accuracy: 0.9466 - val_loss: 0.5877 - val_accuracy: 0.8529 - 40s/epoch - 228ms/step
Epoch 8/20
177/177 - 41s - loss: 0.1364 - accuracy: 0.9466 - val_loss: 0.6757 - val_accuracy: 0.8501 - 41s/epoch - 232ms/step
Epoch 9/20
177/177 - 41s - loss: 0.1354 - accuracy: 0.9472 - val_loss: 0.6275 - val_accuracy: 0.8487 - 41s/epoch - 233ms/step
Epoch 10/20
177/177 - 41s - loss: 0.1357 - accuracy: 0.9484 - val_loss: 0.6264 - val_accuracy: 0.8472 - 41s/epoch - 231ms/step
Epoch 11/20
177/177 - 41s - loss: 0.1342 - accuracy: 0.9470 - val_loss: 0.7173 - val_accuracy: 0.8501 - 41s/epoch - 232ms/step
Epoch 12/20
177/177 - 40s - loss: 0.1339 - accuracy: 0.9473 - val_loss: 0.6693 - val_accuracy: 0.8529 - 40s/epoch - 228ms/step
Epoch 13/20
177/177 - 40s - loss: 0.1307 - accuracy: 0.9479 - val_loss: 0.7279 - val_accuracy: 0.8472 - 40s/epoch - 225ms/step
Epoch 14/20
177/177 - 40s - loss: 0.1348 - accuracy: 0.9463 - val_loss: 0.6914 - val_accuracy: 0.8402 - 40s/epoch - 227ms/step
Epoch 15/20
177/177 - 40s - loss: 0.1326 - accuracy: 0.9466 - val_loss: 0.7303 - val_accuracy: 0.8373 - 40s/epoch - 228ms/step
Epoch 16/20
177/177 - 40s - loss: 0.1304 - accuracy: 0.9493 - val_loss: 0.7136 - val_accuracy: 0.8430 - 40s/epoch - 226ms/step
Epoch 17/20
177/177 - 40s - loss: 0.1322 - accuracy: 0.9480 - val_loss: 0.7774 - val_accuracy: 0.8586 - 40s/epoch - 227ms/step
Epoch 18/20
177/177 - 41s - loss: 0.1329 - accuracy: 0.9466 - val_loss: 0.7735 - val_accuracy: 0.8416 - 41s/epoch - 231ms/step
Epoch 19/20
177/177 - 41s - loss: 0.1436 - accuracy: 0.9435 - val_loss: 1.0620 - val_accuracy: 0.8303 - 41s/epoch - 230ms/step
Epoch 20/20
177/177 - 41s - loss: 0.2191 - accuracy: 0.9286 - val_loss: 1.0219 - val_accuracy: 0.8232 - 41s/epoch - 231ms/step
```

```
model2.evaluate(test_padded2,test_labels2)
```

```
23/23 [==============================] - 1s 30ms/step - loss: 1.0670 - accuracy: 0.8008
[1.0670350790023804, 0.8008474707603455]
```




Immediately we can see that the model has a higher accuracy on the test set compared to our original dataset. Of course, the practicality of this second dataset is not nearly at the same level (i.e. classifying a user into one of several specific industries like “Software Engineering” or “Marketing” is far more useful than simply recommending whether a user fits a “tech” job or a “non-tech” job), but this example demonstrates the potential that future iterations of this project can improve upon.

We now test this model with our sample users. Here is the same software engineer from Amazon with the data science background which we used earlier in our cosine similarity model.

Candidate #2: Software Engineer @ Amazon with academic background in computer science and data science

```
classify('-Worked on the backend team to develop the Edtera web application, a learning engagement platform, using t
```

```
1/1 [==============================] - 0s 33ms/step
                       % Match
Industry
Data Analysis        55.016786
Marketing            36.207151
Software Engineering   3.333176
Sales                 3.207008
Operations & Finance   1.994777
Cybersecurity & IT     0.232973
Research Scientist     0.00813
```

The model correctly identifies that his strongest fit is in the Data Analysis sector and also suggests that he is a surprisingly good match for marketing as well. Intuitively we could reason that this is sensible for most professionals working in data who would have sharp business acumen to solve similar problems that marketing professionals would also address.

Here is another test user whose background is a Senior Marketing Analytics Manager at a tech start up company:

Candidate #3: Senior Marketing Analytics Manager @ Rippling with extensive work history as a marketing data analyst

```
classify('1. Create measurement framework across different funnel stages (TOF, MOF, and BOF) and marketing channels
```

```
1/1 [==============================] - 0s 25ms/step
                       % Match
Industry
Marketing            84.602392
Software Engineering   7.485791
Data Analysis         6.054982
Cybersecurity & IT     1.055249
Sales                 0.408382
Operations & Finance   0.239769
Research Scientist     0.153403
```

And below is a fringe case where we input an investment banker's resume - this individual has no background whatsoever in tech and his experience is very dissimilar to any jobs from the companies that our job database covers (even the non-technology roles in these companies [e.g. Marketing, Sales, etc.] would be very different than this banker's background):

Candidate #5: Investment Banker @ Brookwood Associates. No background in tech - purely a finance person.

```
classify('- Represented CAIRE on its second U.S. acquisition of MGC Diagnostics, a manufacturer of cardiorespiratory
```

```
1/1 [==============================] - 0s 24ms/step
                       % Match
Industry
Software Engineering  48.778135
Operations & Finance  36.889637
Research Scientist     5.123736
Data Analysis          4.115829
Sales                  2.025266
Cybersecurity & IT     1.993226
Marketing              1.074169
```

The model does recognize that Operations & Finance is a decent fit, but ultimately recommends Software Engineering as the "best match". This is likely because Software Engineering roles are the most abundant in our database so in an extreme case that the model hasn't learned very well, it would be biased towards classifying into Software Engineering. This is another area of improvement - the need to gather a wider breadth of data so that the model gains a deeper understanding of the different types of professionals in these industries.

**Final Thoughts & Wrap Up:**
After completing our modeling stage and discussing the results with SwatCloud management, the notebooks were deployed onto the company's website through their Flask production environment.

The company plans to run the HTML scraping script each weekend to refresh the job listings database on a weekly basis and then implement the recommendation models to automatically be an available feature on each of its users' profile pages. The NLP model will actually target SwatCloud's younger users (1st & 2nd year in college/university) because its output for an "industry match" is more useful to clients that are still in the exploration stage of their career. On the other hand, the cosine similarity model will focus on SwatCloud's older users (3rd & 4th year & post-graduate) who are more interested in this model's direct recommendations for active job listings they can immediately prepare and apply for.