BUILDING MATCH UP

The Story of Combining Doc2Vec and Similarity to Create a Recruiting Engine

RAFFA NIMIR
School of Engineering and Applied Science
George Washington University

AGENDA

What Is Match Up

The Dataset

Learnings from the data

Doc2vec and Similarity

Building the engine

Evaluating the engine

Conclusion

WHAT IS MATCHUP

- A recommender system for matching job seekers with suitable jobs
- Uses natural language processing techniques, word embeddings, and similarity.
- Focused on the narrated job description field to capture the most from a job posting
- Data sources contain data science job openings from different companies in the United States, covering all levels (junior-mid and senior).

THE DATASET

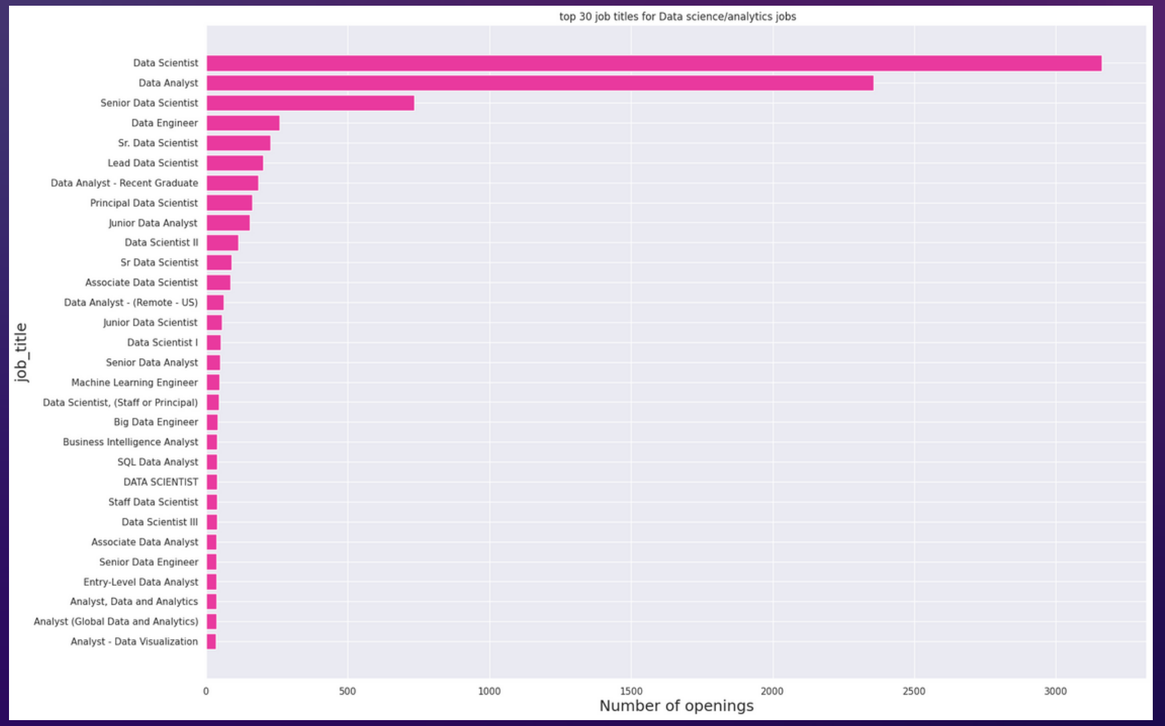
- THE DATA USED FOR BUILDING THE ENGINE CONSISTS OF:
  - DATA SCIENCE JOB OPENINGS FROM KAGGLE AND DATA.WORLD
- The job_description field describes roles' requirements and responsibilities and is used entirely for building the engine

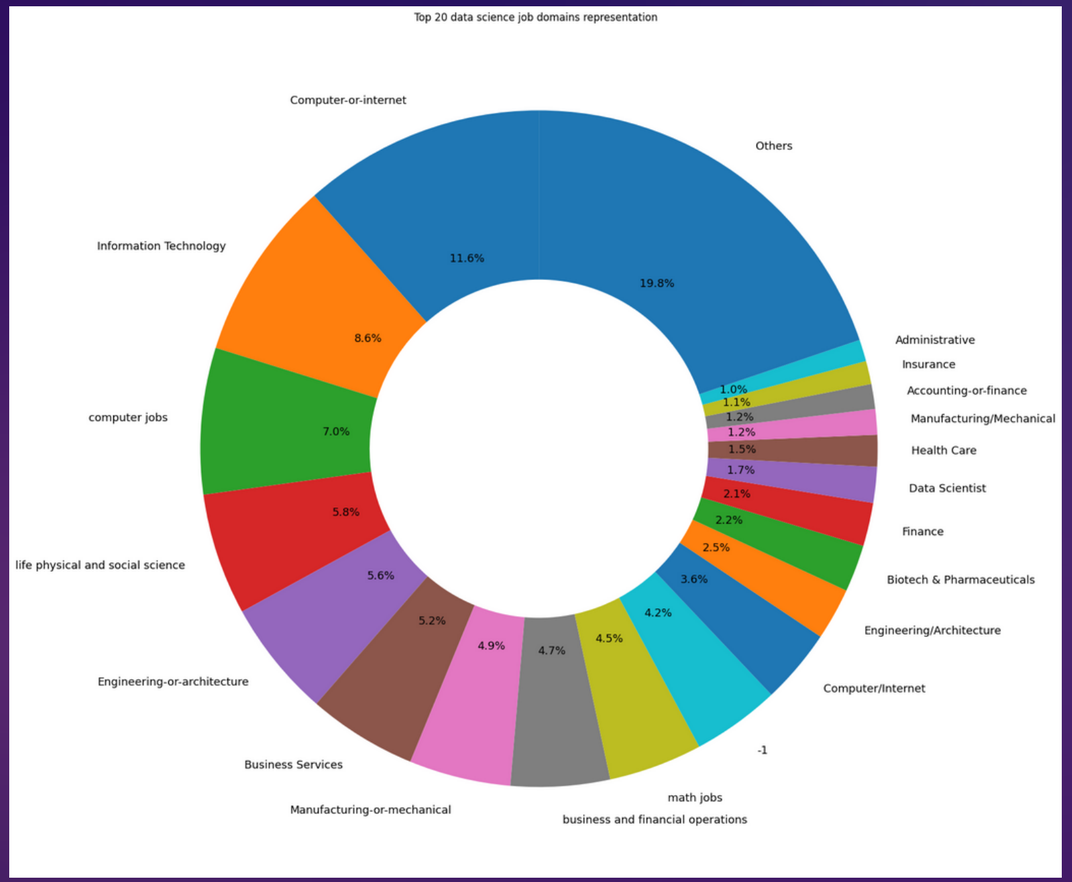
```
jobs.sample(10)
```

	job_ID	job_title	domain	salary	job_description	location	company_name
6292	6301	Data Scientist, Healthcare Methodology	Engineering-or-architecture	0	This is a unique opportunity to join a team at...	Boston	UnitedHealth Group 15,558 reviews
20999	21008	ASSISTANT PLAN EXAMINER (BLDGS	Engineering, Architecture, & Planning	64135.5	The NYC Department of Buildings is responsible...	NaN	NaN
6839	6848	Intern Data Scientist	Engineering-or-architecture	0	GroundTruth is the leading global location pla...	Mountain View	GroundTruth
18518	18527	CLERICAL ASSOCIATE	Public Safety, Inspections, & Enforcement	47671.5	*** ONLY CANDIDATES that are currently permane...	NaN	NaN
19703	19712	ARCHITECT	Engineering, Architecture, & Planning Building...	114000	NYC Parks is an award-winning city agency that...	Citywide Services, Randallâ□□s Island	NaN
23030	23039	INVESTMENT MANAGER (COMPTROLLE	Finance, Accounting, & Procurement	87500	The Bureau of Asset Management (BAM) is respon...	NaN	NaN
8854	8863	Senior Data Scientist in Green Bay, WI	Senior Data Scientist	80K - 100K	Are you interested in designing and implementi...	Green Bay	Schneider
16344	16353	Associate Data Analyst	NaN	40.00\r\n -\r\n 60.00	Currently seeking an Associate of Data Analyti...	New York City Metropolitan Area	Synergy Interactive
21251	21260	COMMUNITY COORDINATOR	Administration & Human Resources Communication...	69040.5	â□¢ Under general direction of a Borough Chief...	Citywide	NaN
14763	14772	Data Analyst - Recent Graduate	NaN	0	At PayPal (NASDAQ: PYPL), we believe that ever...	Scottsdale, AZ	PayPal

ABOUT THE DATA

- Conducted exploratory data analysis on job market data
- Data Scientists are the most common job titles being posted in the analytics market, followed by data Analyst, and others including Data Visualization, Big Data Engineer, and SQL Data Scientis

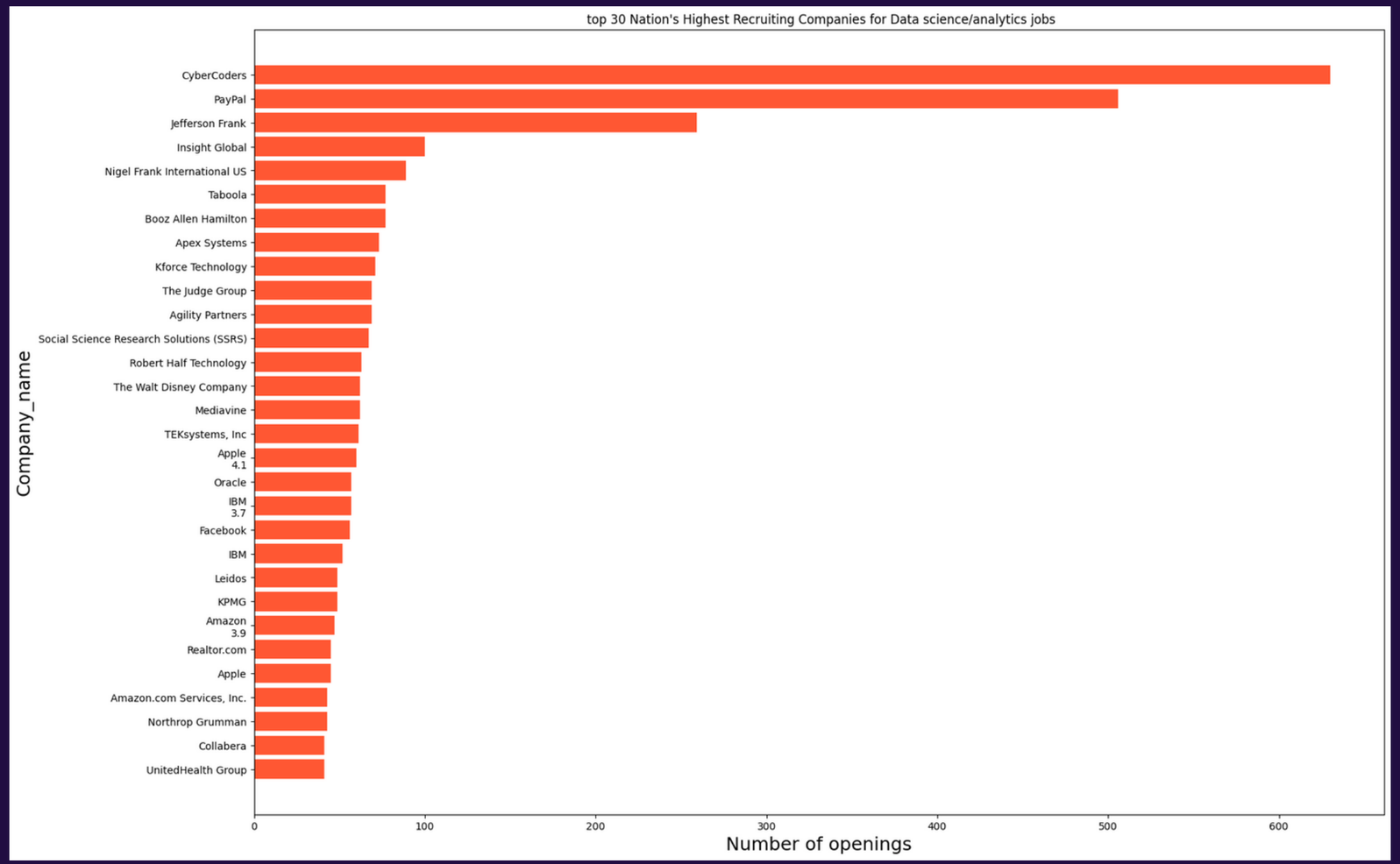


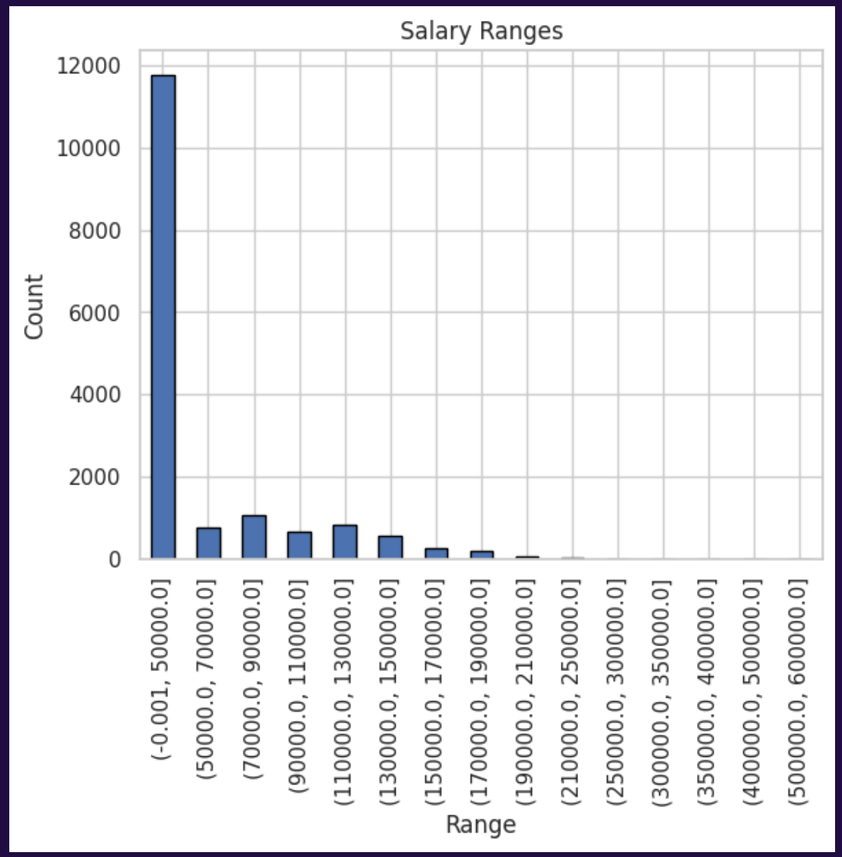


- Investigated the top sectors in the analytics job market
- Found that 7% of openings in the computer/internet sector, 8.6% in the information technology sector
- Physical and social sciences accounted for 5.8% of job openings, followed by engineering or architecture at 5.6%, and business services at 5.2%

ABOUT THE DATA

- Top companies offering data science/analytics jobs were analyzed
- Cyber Coders had the most job openings, followed by Paypal and Jefferson Frank
- Booz Allen ranked sixth in terms of job openings within the dataset





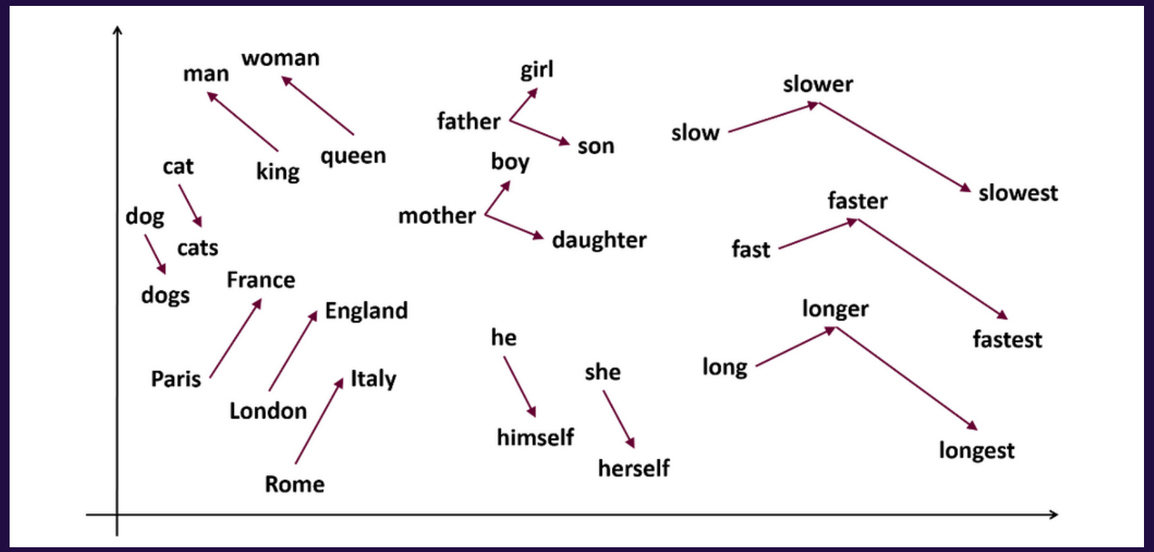
- To calculate salary data reported in ranges, mean salary was used
- Salary bins were created, starting from less than $50,000 up to $600,000
- Salaries less than $50,000 constituted 70.24% of the data


gender identity
communication skill
veteran status
year experience
skill ability
job summary
fast pace
big data
decision making
business problem
statistical analysis
protect veteran
sexual orientation
color religion
experience work
data analytic
work environment
deep learning
race color
bachelor degree
religion sex
team member
applicant receive
regard race
data scientist
united state
visualization tool
natural language
learning model
large data
master degree
computer science
equal opportunity
data set
large scale
data analyst
programming language
predictive modeling
problem solve
work experience
opportunity employer
datum mining
machine learning
data drive
big datum
degree computer
senior data
marital status
ability work
business intelligence
receive consideration
artificial intelligence
datum analysis
data science
minimum year
work closely
cross functional
related field
job description
advanced analytic
national origin
datum scientist
machine learn
internal external
experience datum
datum visualization
qualified applicant
orientation gender
datum source

THE RECOMMENDER ENGINE

Math Behind the Engine

- Word2Vec is a neural network model that represents words as vectors in a high-dimensional space.
- The model places semantically similar words closer together in the vector space.
- Word embeddings are vectors of real numbers used to represent words.
- The embeddings are based on the idea that similar words have similar vector representations.
- Word2Vec can be trained using either the skip-gram or the continuous bag-of-words (CBOW) algorithm.

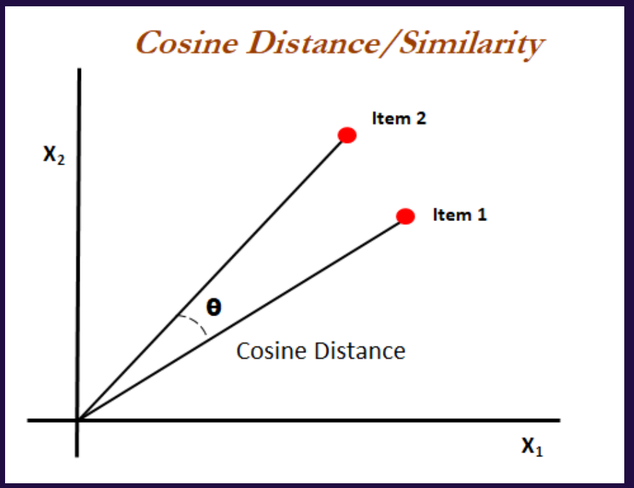


WORD EMBEDDINGS OR WORD2VEC

DOC2VEC

- Doc2Vec is an extension of Word2Vec that adds a unique document ID or tag to the input of the model in addition to the words, allowing it to learn document-level embeddings.
- The trained Doc2Vec model can be used to compute document similarity, find documents similar to a query document, or even generate new documents by sampling from the vector space.

COSINE SIMILARITY



- Cosine similarity measures similarity between two non-zero vectors in a space.
- The measure is computed by finding the cosine of the angle between the vectors.
- The values range from -1 to 1, with -1 indicating completely opposite vectors and 1 indicating identical vectors.
- Cosine similarity is widely used in information retrieval and recommendation systems, as it is invariant to vector length and sensitive to the angle between vectors, and is efficient to calculate and can be used on sparse data.

$$\text{cosine similarity} = S_C(A, B) := \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}},$$

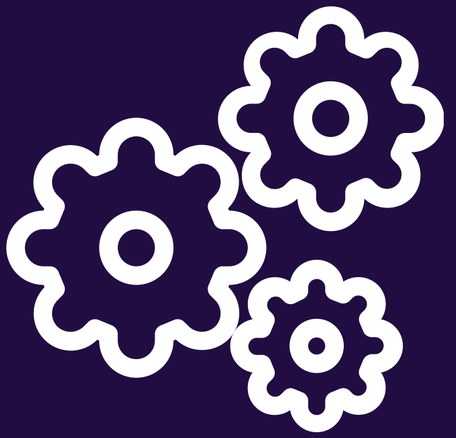
- Euclidean similarity measures the similarity between two vectors using distance.
- It uses the Pythagorean theorem to calculate the distance between the vectors as points in a multi-dimensional space.

$$E(x, y) = \sqrt{\sum_{i=0}^{n} (x_i - y_i)^2}$$

- Manhattan similarity is a distance-based measure of similarity between two vectors in a given space.
- It calculates the distance between the two vectors as the sum of the absolute differences of their coordinates along each dimension, similar to how one would measure the distance between two points on a grid-like map.

$$distance = \sum_{1}^{n} |p_i - q_i|$$

THE LOGIC



1. Input: PDF-formatted resume
2. Text Transformation: PyPDF2 package is used to convert the resume into a textual field
3. NLP Processing: Basic NLP operations like stopword removal and punctuation are applied to enhance semantic capture
4. Vectorization: The tokens extracted from the text are transformed into vectors using a pre-trained Doc2Vec model
5. Similarity measures: Various similarity measures, such as cosine similarity, are computed between the model vectors and the recently vectorized field
6. Algorithm: Based on the similarity scores, the algorithm selects the top five ranked jobs and displays them in a data frame

DOC2VEC MODEL

1. Preprocessing the job_description field

```python
# text preprocessing function
def preprocess_text(text):
    # convert text to lowercase
    text = text.lower()
    # tokenize the text into words
    tokens = word_tokenize(text)
    # remove stopwords
    tokens = [token for token in tokens if token not in nltk_stopwords]
    # remove punctuation
    tokens = [token for token in tokens if token not in string.punctuation]
    # join the tokens back into text
    text = ' '.join(tokens)
    return text
```

2. Fitting the doc2vec anc creating job['vector'[field

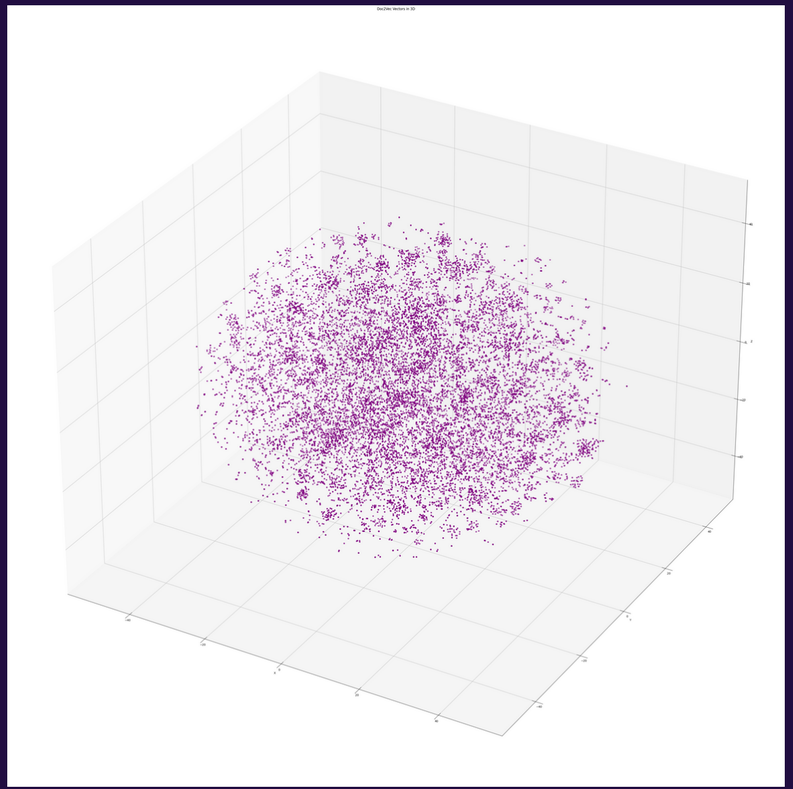
```
[55] jobs['processed_text'] = jobs['job_description'].apply(preprocess_text)
```

```
[56] # Convert preprocessed job postings text into TaggedDocument objects
     tagged_docs = [TaggedDocument(words=text.split(), tags=[i]) for i, text in enumerate(jobs['processed_text'])]
```

```
[57] # Train a Doc2Vec model using the TaggedDocument objects
     model = Doc2Vec(tagged_docs, vector_size=30, window=5, min_count=7, workers=3)
```

```
[58] model.save("doc2vec.model")
```

```
[86] # Infer vector representations for preprocessed job postings text
     jobs['vector'] = jobs['job_description'].apply(lambda x: model.infer_vector(x.split()))
```



Model 3-D Visulaization

FITTING THE SIMILARITY

```
# Calculate the cosine similarity between my text vector and job vectors
cosine_similarities = cosine_similarity(text_vector.reshape(1, -1), jobs['vector'].tolist())[0]

# I will normalize the cosine similarity scores between 0 and 1 using min-max normalization
normalized_similarities = (cosine_similarities - np.min(cosine_similarities)) / (np.max(cosine_similarities) - np.min(cosine_similarities))

# I will retrieve the indices of the top 5 most similar jobs
top_job_indices = np.argsort(cosine_similarities)[-5:]

# I will create a DataFrame of the top 5 most similar jobs for my resume
top_jobs_df = pd.DataFrame({
    'job_ID': jobs.iloc[top_job_indices]['job_ID'].tolist(),
    'job_title': jobs.iloc[top_job_indices]['job_title'].tolist(),
    'job_description': jobs.iloc[top_job_indices]['job_description'].tolist(),
    'similarity_rank': [5, 4, 3, 2, 1],
    'similarity_score': cosine_similarities[top_job_indices],
    'normalized_score': normalized_similarities[top_job_indices]
})
```

```
#show results
top_jobs_df
```

	job_ID	job_title	job_description	similarity_rank	similarity_score	normalized_score
0	5914	Data Scientist Sr	Job Description\n\nDescription\n\nSAIC is seek...	5	0.476867	0.963816
1	11090	Research Scientist - Data Science (Earth & Atm...	Thank you for your interest in a faculty posit...	4	0.483092	0.970895
2	8259	Data Scientist	In collaboration with campus stakeholders and ...	3	0.491282	0.980209
3	10537	Instructor, Data Scientist	Data Scientist Instructor\n\nThis Jobot Job is...	2	0.492699	0.981819
4	10364	Data Scientist	About IpsosIpsos is the world's third largest ...	1	0.508687	1.000000

EVALUATION OF RECOMMENDER ENGINE PERFORMANCE

1. The concept was confirmed to work and extended to a larger dataset of 30 resumes stored in a CSV file.
2. PyPDF2 package was used to create the file, and the same code used to process the initial resume was applied.
3. Basic natural language processing (NLP) cleansing was performed on the text in the CSV file.
4. The CSV file was then vectorized using the model_infer parameter with the Doc2Vec model.

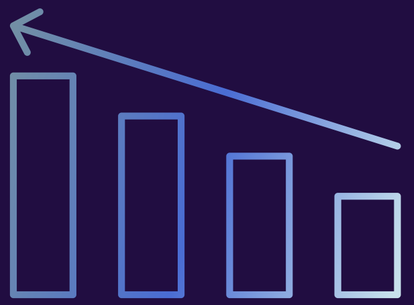
```
resumes
```

	resume_text	resume_title	processed	resume_vector
1	First Last\nData Scientist\nData Scientist wi...	Data Scientist	[first, last, data, scientist, data, scientist...	[-0.036104918, -0.40184838, -1.0239247, 1.3042...
2	FIRST LAST\nBay Area, California • +1-234-456-...	SQL Developer	[first, last, bay, area, california, professio...	[-0.373463, 0.88739663, -0.2061928, 0.7619801,...
3	FIRST LAST\nBay Area, California • +1-234-456-...	SQL Developer	[first, last, bay, area, california, professio...	[-0.80879974, 0.47542474, -0.6333653, 0.928722...
4	FIRST LAST\nBay Area, California • +1-234-456-...	PL/SQL Developer	[first, last, bay, area, california, professio...	[-0.9101211, 0.35757476, -1.1577423, -0.563272...
5	FIRST LAST\nNew York, NY | P: +44 123456789 | ...	Data Analyst	[first, last, new, york, ny, p, first, last, r...	[-0.49276206, 0.40492558, -0.49274924, 1.40053...
6	First Last\nNew York, NY 10001 · (212) 123-456...	Sr. Data Scientist	[first, last, new, york, ny, first, last, resu...	[-0.5769407, -0.4954766, -0.66477805, 0.954171...
7	FIRST LAST\nData Science Manager\nNew York Cit...	Data Science Manager	[first, last, data, science, manager, new, yor...	[-0.85066956, 0.10897474, -0.36648703, 1.38414...
8	FIRST LAST\nSQL Analyst\nNew York City, NY 100...	SQL Analyst	[first, last, sql, analyst, new, york, city, n...	[0.17975643, 0.5067796, -0.63754976, 0.3661177...
9	FIRST LAST\nBay Area, California • +1-234-456-...	Data Scientist	[first, last, bay, area, california, professio...	[-0.58855367, -0.02412484, -0.10636644, 1.2737...
10	KINDERGARTEN TEACHER\nProfessional Summary\nEn...	Kindergarten teacher	[kindergarten, teacher, professional, summary,...	[-0.89593464, 0.43843645, 0.8866533, 1.72294, ...
11	HEALTH CARE ADMINISTRATOR\nInterests\nAs a hob...	Health Care Administrator	[health, care, administrator, interests, hobby...	[-2.800843, 1.7891511, 0.023786955, 1.6658515,...
12	PASTRY SOUS CHEF\nSummary\nService oriented pr...	Pastry Sous Chef	[pastry, sous, chef, summary, service, oriente...	[-2.067439, 0.5706448, 0.115838945, 0.95145226...
13	EXECUTIVE CHEF\nSummary\nDedicated, hardworkin...	Executive Chef	[executive, chef, summary, dedicated, hardwork...	[-1.37801, 0.57578206, 1.2676702, -0.81290084,...
14	CHEF\nSummary\nQuality-focused and efficient C...	CHEF	[chef, summary, quality, focused, efficient, c...	[-1.5489218, 0.92002624, 0.63087434, 0.6227245...
15	CHEF\nSummary\nFocused Operations Manager succ...	CHEF	[chef, summary, focused, operations, manager, ...	[-0.5906737, 0.57365125, 0.36680394, 0.2468395...
16	SENIOR BANQUET CHEF\nProfessional Summary\nDyn...	SENIOR BANQUET CHEF	[senior, banquet, chef, professional, summary,...	[-0.8539177, 1.6659745, 0.21618335, -0.9895264...
17	BISTRO CHEF\nCareer Overview\nA result-oriente...	BISTRO CHEF	[bistro, chef, career, overview, result, orien...	[-2.0540333, 1.8157413, 0.8547894, 1.8529499, ...
18	CHEF\nSummary\nCustomer-oriented fast food wor...	CHEF	[chef, summary, customer, oriented, fast, food...	[-0.34084883, 1.198757, -0.9228574, -0.8391118...
19	CHEF\nCareer Overview\nDedicated Customer Serv...	CHEF	[chef, career, overview, dedicated, customer, ...	[-1.0626658, 0.96112996, 0.08792493, -0.246966...
20	First Last\nData Scientist\nEx-Business Develo...	Data Scientist	[first, last, data, scientist, ex, business, d...	[0.15258029, 0.050364755, -1.0997026, 0.557756...
21	First Last\nData Scientist\nData Scientist wit...	Data Scientist	[first, last, data, scientist, data, scientist...	[-0.19117843, -0.63382614, -1.0637611, 0.53466...
22	First Last\nData Entry Operator\nSt. Paul, Min...	Data Entry Operator	[first, last, data, entry, operator, st, paul,...	[-0.6135706, 1.1239768, -0.57313484, 0.8313160...
23	First Last\nData Science Manager\nWORK EXPERIE...	Data Science Manager	[first, last, data, science, manager, work, ex...	[-1.1694077, 0.32522842, -0.58669674, 1.130024...
24	First Last\nData Scientist\nWORK EXPERIENCE\n_...	Data Scientist	[first, last, data, scientist, work, experienc...	[-1.40143, 0.65588444, -0.98869735, 0.74130076...
25	IBM Data Science\nCoursera\nMicrosof Professio...	IBM Data Science	[ibm, data, science, coursera, microsof, profe...	[-0.16303256, -0.0050656134, -0.47811285, 0.49...
26	First Last\nJunior SQL Developer\nBurlington, ...	Junior SQL Developer	[first, last, junior, sql, developer, burlingt...	[-0.9638015, 0.35972193, -1.4489869, 0.4675447...

Executing (8m 53s) <cell line: 14> > fit_transform() > _fit() > _train() > gradient_descent() > _kl_divergence_bh()

CATEGORIZATION FOR EVALUATION OF RECOMMENDER ENGINE PERFORMANCE

- Categorized 30 resumes into three groups: analytics jobs (resumes 0-10 and 20-30) and intentionally biased chef jobs (resumes 11-20) for evaluation of precision, recall, and F1 scores.
- Assigned a value of 1 to the "matched" column if the normalized similarities exceeded 0.8, otherwise assigned 0.
- Any resumes between 11 and 20 that were matched were considered irrelevant and classified as such in the "status" column.
- For resumes 0-10 and 20-30, a value of 0 in the matched column indicated irrelevance, and a value of 1 indicated relevance.
- Computed the precision, recall, and F1 score of the engine using these criteria.

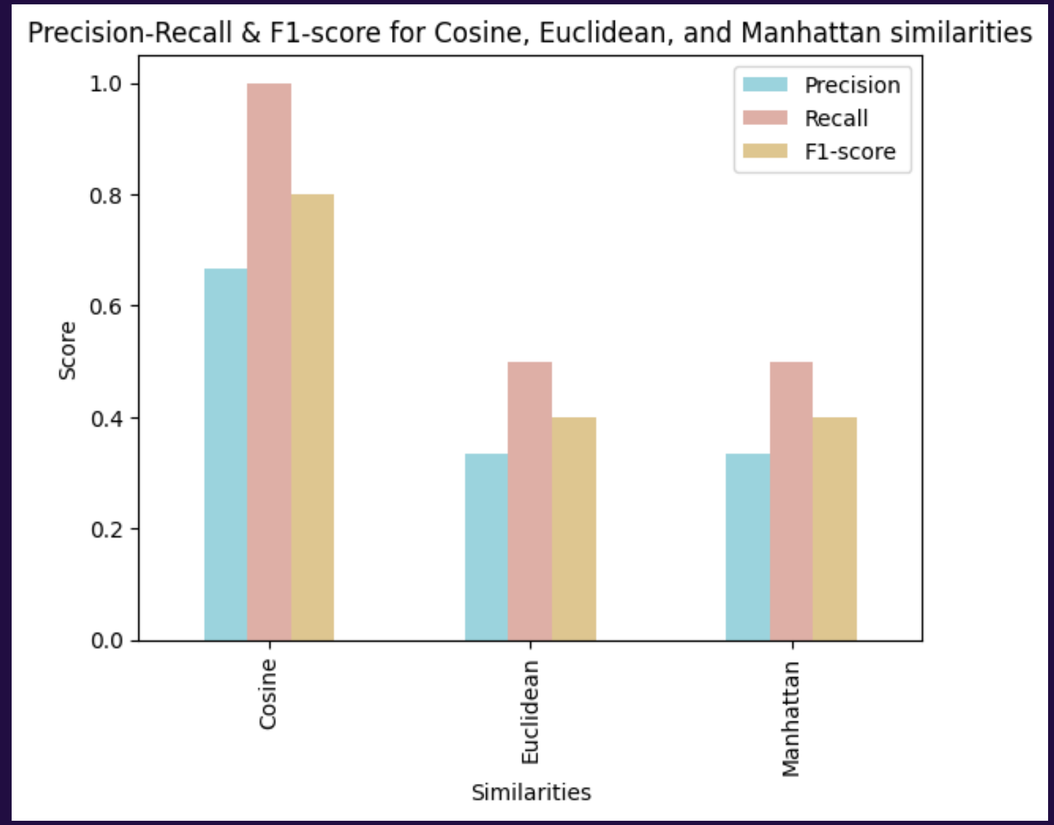


CATEGORIZATION AND FITTING CODE

```python
# Step 1: Create an empty dataframe to store the results
result_df = pd.DataFrame(columns=['resume_index', 'resume_title', 'resume_text', 'job_id', 'job_title', 'job_description', 'similarity', 'normalized_similarity', 'similarity_rank', 'match', 'status'])

# Step 2: Iterate over each row in the resumes dataframe
for resume_index in range(len(resumes)):

    # Get the current resume's vector representation
    resume_vector = resumes['resume_vector'][resume_index]

    # Calculate the cosine similarity between the current resume and all jobs
    similarities = cosine_similarity(resume_vector.reshape(1, -1), jobs['vector'].tolist())[0]

    # Get the indices of the top 5 highest similarity scores
    top_indices = similarities.argsort()[-5:][::-1]

    # Step 3: Iterate over the top 5 similar jobs and add them to the result dataframe
    for rank, job_index in enumerate(top_indices):
        similarity = similarities[job_index]
        normalized_similarity = (similarity - np.min(similarities)) / (np.max(similarities) - np.min(similarities))

        if (normalized_similarity > 0.80).any():
            match = 1
        else:
            match = 0

        if resume_index < 10:
            if match == 1:
                status = 'relevant'
            else:
                status = 'irrelevant'
        elif resume_index < 20:
            if match == 1:
                status = 'irrelevant'
            else:
                status = 'relevant'
        else:
            if match == 1:
                status = 'relevant'
            else:
                status = 'irrelevant'

        job = jobs.iloc[job_index]
        result_df = result_df.append({
            'resume_index': resume_index,
            'resume_title': resumes['resume_title '][resume_index],
            'resume_text': resumes['resume_text'][resume_index],
            'job_id': job['job_ID'],
            'job_title': job['job_title'],
            'job_description': job['job_description'],
            'similarity': similarity,
            'normalized_similarity': normalized_similarity,
            'similarity_rank': rank+1,
            'match': match,
            'status': status
        }, ignore_index=True)

# Step 6: Print the result dataframe
result = result_df.groupby('resume_index').apply(lambda x: x[['resume_title', 'resume_text', 'job_id', 'job_title', 'job_description', 'similarity', 'normalized_similarity', 'similarity_rank', 'match','status']].sort_values('similarity_rank', ascending=False))
```

RESULTS & DISCUSSION

- Cosine similarity showed better performance than Euclidean and Manhattan similarity in terms of the F1-score.
- However, measuring model performance without ground truth was a significant obstacle.
- Cosine similarity achieved a precision of 67%, a recall of 1, and an F1 score of 0.8.
- Euclidean and Manhattan similarity had a precision and recall of 0.33 and 0.5, respectively, and an F1-score of 0.5.




```
[47] grouped_df = result_df.groupby("resume_index")
     group_0 = grouped_df.get_group(0)
```

group_0

	resume_index	resume_title	resume_text	job_id	job_title	job_description	similarity	normalized_similarity	similarity_rank	match	status
0	0	Data Scientist	First Last\nData Scientist\nData Scientist wi...	383	Data - Data Scientist BHJOB11946_357478	Position: Data Scientist Location: New York, N...	0.410456	1.000000	1	1	relevant
1	0	Data Scientist	First Last\nData Scientist\nData Scientist wi...	183	Lead Data Scientist	Hobsons connects learning to life by matching ...	0.398609	0.985126	2	1	relevant
2	0	Data Scientist	First Last\nData Scientist\nData Scientist wi...	5858	Lead Data Scientist	Since 2002, Quantium have combined the best of...	0.396960	0.983055	3	1	relevant
3	0	Data Scientist	First Last\nData Scientist\nData Scientist wi...	7034	Data Scientist	To support our project teams, we are looking f...	0.387075	0.970645	4	1	relevant
4	0	Data Scientist	First Last\nData Scientist\nData Scientist wi...	9986	Data Scientist	EOG Resources, Inc. is recruiting for a Data S...	0.386663	0.970127	5	1	relevant

CONCLUSION

1. Without the ground truth measurement, gaining a true measurement of model performance was a major obstacle.
2. The algorithm may lead to errors by force-fitting data from the feed.
3. The overly interconnected and clustered vector space representation of the data may result in overfitting and potential errors.
4. The volume of data was small compared to models trained on textual data.
5. One way to address these limitations and potential errors is to collect a broader range of data that is more diverse and representative.
6. Implementing a human-in-the-loop approach could help reduce the risks of false positives or false negatives.
7. Despite the limitations, the recommendation engine demonstrated promising results and served as a proof of concept.

THANK YOU