ST. CLOUD STATE UNIVERSITY

INTERNSHIP PORTFOLIO

by

Harihara Varma Aketi (15598866)

A Portfolio

Submitted to the Graduate faculty of

St. Cloud State University

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Science

March 2024

Portfolio Committee:

Dr. Mark Petzold (Advisor)

Dr. Mark B. Schmidt

Dr. Akalanka Bandara Mailewa

ABSTRACT

Handling extensive datasets for training machine learning (ML) models presents significant challenges due to their complexity and the potential presence of numerous highly correlated features. The intricacies escalate when datasets consist of a vast array of crucial features exhibiting substantial collinearity. In response, a prevalent recommendation is to streamline the dataset, condensing its dimensions while retaining essential information. Our study focuses on conducting an in-depth exploratory data analysis (EDA) to unveil underlying trends within the dataset, particularly investigating the covariance between its various attributes. Through rigorous examination, I endeavor to gain comprehensive insights into the dataset's structure and characteristics.

Subsequently, our research delves into the application of various dimensionality reduction techniques, aiming to assess their efficacy across different ML models. By systematically exploring these methodologies, I aim to elucidate their impact on model performance and discern the most suitable approaches for optimizing model training efficiency while preserving data integrity. This multifaceted investigation promises to contribute valuable insights to the field of ML, offering practical strategies for managing large-scale datasets and enhancing model interpretability and generalization capabilities.

ACKNOWLEDGEMENTS

I wish to express my heartfelt appreciation to the members of my committee for their unwavering support, guidance, and encouragement throughout the duration of my internship.

I deeply express my sincere thanks to our Department Chair, my Advisor, and committee Chairperson, Dr. Mark Petzold, for his consistent encouragement and for granting me the opportunity to present my "Project/Internship Portfolio" at our department's facilities as part of the fulfillment requirements for my master's degree. I am equally grateful to the Project Committee members, Dr. Mark B. Schmidt, and Dr. Akalanka Bandara Mailewa.

I also seize this moment to express my gratitude to all the professors who have played a direct or indirect role in supporting my internship. Special thanks are extended to my family and friends, whose unwavering support has been a constant source of encouragement throughout my pursuit of a master's degree.

TABLE OF CONTENTS

TABLE OF FIGURES & TABLES

1. INTRODUCTION

1.1 Background

I am a graduate student in the Department of Computer Science at St. Cloud State University, scheduled to complete my studies this Spring. My academic journey began with a bachelor's degree in Information Technology from Jawaharlal Nehru Technological University Hyderabad in 2021.

I made the decision to further my education by enrolling in the master's program at St. Cloud State University, majoring in Computer Science, in the Spring of 2022. I am currently in my final semester of this program. As part of my master's course structure, I secured a Curricular Practical Training (CPT) opportunity, enabling me to undertake an internship with Kivyo Inc, which started in December 2023.

1.2 About the Company

I am doing my internship with Kivyo Inc, situated in Texas, a leading IT staffing solution provider with longstanding history and relationships with Fortune 1000 and mid-market companies across US and India. Kivyo is powered by Harvard Faculty, IT Professionals, technology and domain experts with strong knowledge in providing effective Custom Software Solutions, ensuring your application developed is well responsive. The company boasts a robust team comprised of application developers, data engineers, and dedicated business & IT consultants, ensuring a comprehensive approach to problem-solving.

1.3 About my Team

During our internship collaboration, I had the privilege of working alongside another intern who shared my enthusiasm and dedication to our project. Together, we formed a dynamic team committed to achieving excellence in all aspects of our work. Under the guidance of our experienced manager, we embarked on a journey of learning and growth, diving deep into the intricacies of our project.

Throughout the internship, our manager played a pivotal role, offering invaluable insights and guidance as we navigated through our tasks. Beyond just supervision, our manager actively engaged with us on a weekly basis, providing constructive feedback and refining our understanding of project requirements.

As a team, we were united in our commitment to leverage our collective skills and knowledge to not only meet but exceed expectations. Through open communication, mutual support, and a shared vision for success, we worked together efficiently and precisely to achieve our goals. Our collaborative efforts created a supportive environment where each team member could thrive and contribute their best, ensuring that we made the most of our internship opportunity.

1.4 About my Project

Since the advent of the field of machine learning, we have been able to develop a wide range of intelligent models. There are many powerful machine learning models out there which can be used for different tasks. However, as powerful as these models are. It is still vital to preprocess the data very well to ensure the best possible outcome for these models. The preprocess technique we employ will totally depend on the kind of data. Data usually comes with many issues, like missing data values, dimensionality issues, and the covariance between variables, all of which may impact the performance of the models in many ways.

It's not only an issue with data, but can be an issue with the model as well. Different models may require being fed with a different method of preprocessing. Overall the best performance is got with the right preprocessing for the right model for a particular dataset. In this project, we want to address how the issue of the high dimensionality of data can impact a model's performance. We would like to preprocess the data with different dimensionality reduction techniques, specifically, Principal Component Analysis(PCA) as studied by the authors of [1], Linear Discriminant Analysis(LDA) as demonstrated in [2], Kernal PCA as analysed by authors in [3], and the newly suggested Uniform Manifold Approximation and Projection(UMAP) which highlights being very beneficial for clustering in [4].

We would like to access the performance of some widely used ML models and study their performance after employing these different dimensionality reduction techniques using some standard performance metrics. We are using the 20 news groups dataset, for this purpose.

2. LITERATURE REVIEW

In this literature review, I explore prominent dimensionality reduction techniques, encompassing both linear and non-linear methodologies tailored to address the challenge of high-dimensional data.

Linear methods, traditionally prevalent, include innovative approaches such as the technique proposed by the authors of [5], which optimally select functions based on CTG's pathological categorization. Additionally, Principal Component Analysis (PCA) stands out for its role in enhancing computational efficiency and accuracy, although limitations like feature interpretability and reliance on linear relationships between variables warrant consideration.

On the contrary, Linear Discriminant Analysis (LDA) operates within a supervised learning framework, prioritizing robust feature representation through maximized group separation—an advantage particularly valuable in scenarios necessitating clear category delineation.

In the realm of non-linear techniques, Kernel PCA and Uniform Manifold Approximation and Projection (UMAP) offer alternative paths for dimensionality reduction. A comprehensive study outlined in [4] meticulously evaluates UMAP's impact on clustering algorithms, revealing substantial enhancements in both accuracy and runtime efficiency. This is exemplified by significant improvements in clustering accuracy and runtime reduction over the MNIST dataset using HDBSCAN, underscoring UMAP's efficacy in expediting and enhancing clustering processes.

3. METHODOLOGY

In this section, I delve into a range of dimensionality reduction techniques encompassing both linear methods such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), as well as non-linear approaches including Kernel PCA and Uniform Manifold Approximation and Projection (UMAP). Additionally, I provide insights into the experimental setup and the dataset utilized for this project. This comprehensive exploration allows for a thorough understanding of the methodologies employed and the context in which they are applied.

3.1 Principal Component Analysis (PCA)

Principal Component Analysis (PCA)[1], a widely employed dimensionality-reduction method, proves invaluable in streamlining large datasets by transforming numerous variables into a more manageable set while retaining crucial information.

This reduction facilitates swifter analysis and visualization, expediting machine learning (ML) algorithms' processing times. Despite a marginal sacrifice in accuracy, the trade-off becomes advantageous due to the substantial time savings in dealing with extensive datasets. PCA's role as a powerful tool in balancing computational efficiency and information retention underscores its significance in enhancing the practicality of ML applications, particularly in scenarios where time constraints are paramount. This methodological approach stands as a testament to the nuanced interplay between accuracy and efficiency in the realm of dimensionality reduction, marking PCA as a valuable asset in optimizing the performance of ML models on large-scale datasets.

Algorithm:

The Principal Component Analysis (PCA) algorithm involves several key steps for efficient dimensionality reduction and enhanced data evaluation:

1. **Data Standardization:** Begin by standardizing the dataset, ensuring uniformity in variable contributions. This step is crucial for the equitable analysis of each variable. Transform the variables using a prescribed formula, optimizing them for subsequent processing.

$$Z = \frac{(value - mean)}{standard\ deviation}$$

2. **Covariance Matrix Computation:** Calculate the covariance matrix to discern the correlation between variable pairs. This matrix provides insights into the interdependence of variables, simplifying their categorization. Understanding these correlations lays the foundation for subsequent steps.

$$covariance\ matrix = [cov(x,x)\ cov(x,y)\ cov(x,y)\ cov(y,y)]$$

3. **Eigenvalue and Eigenvector Computation:** Determine the eigenvector components and their corresponding eigenvalues. Organize these eigenvalues in descending order, yielding a list of principal components. These components essentially signify the directions of the data. The result is a set of axes that form a new coordinate system. This transformation facilitates enhanced data evaluation, offering clear insights into variations between observations.

4. **Data Recasting along Principal Component Axes:** Act on the newly derived principal components to reorient the data. This step involves shifting the data from its original axes to those computed from the principal components. The recalibrated axes provide a more insightful perspective, aiding in the identification and interpretation of patterns within the dataset.

In essence, PCA serves as a multifaceted process, encompassing standardization, covariance analysis, and eigenvalue-eigenvector computations. These meticulous steps collectively enable a profound restructuring of the data, ultimately enhancing the interpretability and utility of the dataset in subsequent analyses.

3.2 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA)[2] serves as both a preprocessing technique in data mining and machine learning, as well as a robust classification method. In LDA, the original feature set is condensed to C-1 features, where C denotes the number of classes, achieved through variance minimization and maximizing inter-class mean distances. It facilitates enhanced classification accuracy by extracting discriminative features that effectively separate different classes. By reducing dimensionality while preserving class-discriminatory information, LDA offers a streamlined approach to data representation, particularly in scenarios with a limited number of classes.

Algorithm:

1. **Mean Vector Initialization:** Initial creation of a d-dimensional mean vector for each class in the dataset, where d represents the number of features.
2. **Scatter Matrix Computation:** Computation of scatter matrices to capture the spread of data within each class, followed by the calculation of eigenvectors and eigenvalues for each scatter matrix.
3. **Eigenvector Sorting:** Sorting of eigenvectors in descending order based on their associated eigenvalues to identify the most significant directions of variance within the data.
4. **Selection of Top Eigenvectors:** Selection of a subset of k eigenvectors with the largest eigenvalues to form a transformation matrix, facilitating the projection of data into a lower-dimensional space while preserving discriminative information.

These four steps outline the core process of the LDA algorithm, which aims to reduce dimensionality while maximizing class separability for improved classification performance.

3.3 Kernel PCA

Kernel PCA [3] is an advancement of Principal Component Analysis (PCA) tailored for nonlinear data separation through kernel functions. The fundamental objective is to transform non-separable data into a linearly separable format by projecting it onto a higher-dimensional space. This transformation is achieved through the construction of a kernel matrix based on the dataset. By deriving the eigenvectors of this matrix, I extract the principal components of the data points and compute their projections onto these eigenvectors. This process enables the delineation of intricate nonlinear patterns, facilitating enhanced data representation and analysis in higher-dimensional spaces.

Algorithm:

1. **Selecting a Suitable Kernel Function:** Choose an appropriate kernel function, denoted as $k(x, y)$, to map the data points into a higher-dimensional space.
2. **Computing the Kernel Matrix:** Compute the Kernel matrix, K, where each element K_{ij} represents the similarity between data points x_i and x_j based on the chosen Kernel function.
3. **Centering the Kernel Matrix:** Center the Kernel matrix to ensure symmetry and zero mean by subtracting the mean across rows and columns:

$$K_{centered} = (I - 1_n)\, K(I - 1_n)$$

   where I is the identity matrix and n is the number of data points.
4. **Computing Eigenvectors and Eigenvalues:** Compute the eigenvectors and eigenvalues of the centered Kernel matrix. These eigenvectors represent the directions of maximum variance in high-dimensional space.
5. **Scaling Eigenvectors:** Scale each eigenvector by the square root of its corresponding eigenvalue. These scaled eigenvectors serve as the principal components of the data.
6. **Projecting the Data:** Project the data onto the principal components to obtain the reduced-dimensional representation of the original dataset.

The resulting reduced dataset captures the essential structure and patterns of the original data in a lower-dimensional space, facilitating easier visualization and analysis, particularly for datasets exhibiting nonlinear relationships.

3.4 Uniform Manifold Approximation and Projection (UMAP)

Uniform Manifold Approximation and Projection (UMAP), as elucidated in [4], represents a versatile non-linear dimensionality reduction technique. UMAP operates by leveraging the underlying Riemannian manifold structure within the data, seeking to identify a low-dimensional embedding that preserves the fuzzy topological relationships present in the original high-dimensional space. By tapping into the manifold's intrinsic properties, UMAP facilitates the creation of compact yet informative representations of complex datasets, thereby enabling enhanced data visualization, clustering, and classification tasks. This comprehensive understanding of UMAP's underlying principles underscores its significance as a powerful tool in the realm of non-linear dimensionality reduction.

Algorithm:

To present the UMAP algorithm in a structured and refined manner, let's break down the process into key steps for clarity:

1. **Construction of a Fuzzy Topological Structure:** UMAP begins by representing the dataset within a high-dimensional space, forming a weighted graph. Edges in this graph signify the likelihood of connectivity between points, establishing a basis for exploring data relationships.

2. **Computation of Similarity Between Points:** The algorithm employs an exponential probability distribution to quantify the similarity between any two points,

$$p(j) = exp\,(-d(x_i, x_j) - p_i$$

where $d(x_i, x_j)$ denotes the distance between points i and j, and ρ_i represents the minimum distance from i^{th} data point to its nearest neighbor, emphasizing the local data structure.

3. **Symmetrization of Probabilities:** In instances where the edge weights are asymmetric, meaning the connection from i to j differs from j to i , UMAP symmetrizes the probabilities to ensure a consistent representation:

$$p_{ij} = p(i \mid j) + p(j \mid i) = p(i \mid j)p(j \mid i)$$

4. **Specifying the Number of Nearest Neighbors:** UMAP requires the specification of k, the number of nearest neighbors, to refine the graph structure, calculated as:

$$k = 2 * \sum_{i} \quad p_{ij}$$

5. **Low-Dimensional Layout Optimization:** The algorithm endeavors to create a low-dimensional representation of the data that mirrors the high-dimensional fuzzy topological structure as closely as possible, using a probability measure akin to the student t-distribution for modeling distances in the reduced space:

$$q_{ij} = \frac{1}{1 + a(y_i - y_j)^2}$$

   with parameters a and $b$tuned to optimize layout fidelity.

6. **Employment of Binary Cross-Entropy as Cost Function:** To capture the global structure of the data, UMAP utilizes binary cross-entropy (CE) as the cost function:

$$CE(P, Q) = \sum[p_{ij}\ log\ log\left(\frac{p_{ij}}{q_{ij}}\right) + (1 - p_{ij})\ log\ log\left(\frac{1 - p_{ij}}{1 - q_{ij}}\right)]$$

   where P and Q denote the probabilistic similarities in the high and low-dimensional spaces, respectively.

This stepwise breakdown elucidates UMAP's approach to dimensionality reduction, highlighting its focus on preserving local neighborhood structures while facilitating a comprehensive understanding of the dataset's intrinsic geometric properties.

4. DATASET

The 20 Newsgroups dataset is a substantial compilation of approximately 18,000 newsgroup documents, meticulously categorized across 20 distinct categories or newsgroups. This dataset, originating from the early days of the internet, serves as a benchmark for text analysis and machine learning research, providing a rich landscape for exploring a variety of classification, clustering, and information retrieval tasks.

Figures 1 and 2 highlight how the words of each class are overall well related using word clouds for the *sci.med* and *talk.politics.guns* classes. Word clouds as we know emphasize words in a given text based on their respective frequency.

Figure 1: Word cloud for sci.med class. As per the word cloud health care terms like patient, drug, treatment, etc… are more frequent

Structured meticulously, the dataset organizes the documents into 20 separate text files, with each file dedicated to a unique newsgroup category. These files collectively encompass all messages pertinent to their respective newsgroup, with each message uniquely identified by an ID. A specialized list.csv file is included, offering a comprehensive mapping of each message ID to its corresponding newsgroup category, thereby simplifying the task of data organization and retrieval for analysis purposes.

Moreover, the 20 Newsgroups dataset's widespread use in the machine learning community has led to the development of numerous preprocessing techniques and methodologies tailored specifically for text data. These include methods for text normalization, feature extraction, and the handling of sparse datasets, which are critical for improving model performance. The dataset's inherent challenges, such as the presence of cross-posted messages across multiple newsgroups and the variability in the length and content of the posts, provide a realistic setting for testing and refining these techniques.

Figure 2: Word cloud for talk.politics.guns class. As per the word cloud political and crime terms like gun, fire, government, etc… are more frequent

5. EXPERIMENTAL SETUP

The experimental setup of this project is meticulously designed to investigate the impact of various dimensionality reduction techniques on the performance of machine learning (ML) models when applied to the 20 newsgroups dataset. This investigation is structured into a comprehensive multi-step process:

1. **Feature Engineering:** Initially, the dataset undergoes a series of feature engineering steps to prepare the data for analysis. This includes normalization processes, where the min-max standard scaler method is employed to ensure all data points are scaled uniformly across the dataset. Following normalization, categorical data elements are converted into a numerical format to facilitate processing by ML algorithms. Additionally, a vectorization process is applied where a frequency threshold of 0.1% is implemented, ensuring only words that appear with high frequency are retained for analysis, thereby enhancing the focus on relevant features.

Figure 3 highlights that all the text follows a method for pre-processing. First the words are tokenized and stop words are removed. Then the remaining words are lemmatized and vectorized. Using NLTK library

```
word_lemmatizer = WordNetLemmatizer()
stopwords_set = set(stopwords.words("english"))

for i in range(len(title_tr)):
  msg_words = word_tokenize(title_tr[i])
  lemmatized_msg = ""
  for word in msg_words:
    if word not in stopwords_set:
      lemmatized_msg += word_lemmatizer.lemmatize(word) + " "
  title_tr[i] = lemmatized_msg

for i in range(len(title_te)):
  msg_words = word_tokenize(title_te[i])
  lemmatized_msg = ""
  for word in msg_words:
    if word not in stopwords_set:
      lemmatized_msg += word_lemmatizer.lemmatize(word) + " "
  title_te[i] = lemmatized_msg

stop_words = nltk.corpus.stopwords.words("english")
vectorizer = TfidfVectorizer(stop_words=stop_words, min_df=0.001)
vectorizer.fit(title_tr)

Xtr = vectorizer.transform(title_tr)
Xte = vectorizer.transform(title_te)
```

Figure 3: Sample code for feature engineering

2. **Baseline Model Experimentation:** With the dataset normalized and prepared, experimentation begins with a selection of machine learning algorithms, including Support Vector Machines (SVM), KMeans Clustering, and Random Forest. These models are applied to the dataset to establish baseline performances, providing a comparative foundation for assessing the impact of dimensionality reduction techniques.

3. **Linear Dimensionality Reduction Application:** The project proceeds by applying linear dimensionality reduction methods, specifically Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA), to the prepared dataset. Following the application of these techniques, the machine

learning models are re-evaluated to determine the effect of linear dimensionality reduction on their performance.

Figure 4 is a sample code for PCA dimension reduction using the scikit-learn library. The features are normalized and sent to PCA. Finally, the code plots these 2-dimensional features.

```
start_time = time()
pca = PCA(n_components=2048)
pca.fit(scaled_data_Xtr.toarray())
reduced_pca_Xtr = pca.transform(scaled_data_Xtr.toarray())
print("Time taken to perform dimensionality reduction using PCA" ,time() -
start_time)
print("Number of features after reduction : ", reduced_pca_Xtr.shape[1])

# Plotting the dataset in 2 dimensional space
pca_df = pd.DataFrame()
pca_df["y"] = [df.target_names[lab] for lab in category_tr]
pca_df["comp-1"] = reduced_pca_Xtr[:,0]
pca_df["comp-2"] = reduced_pca_Xtr[:,1]
sns.set(rc={'figure.figsize':(20,10)})
sns.scatterplot(x="comp-1", y="comp-2", hue=pca_df.y.tolist(),
data=pca_df).set(title="PCA projection with features=2")
```

Figure 4: Sample code for dimensionality reduction

4. **Non-linear Dimensionality Reduction Application:** To explore the effects of non-linear dimensionality reduction, techniques such as Kernel PCA and Uniform Manifold Approximation and Projection (UMAP) are applied to the dataset. The machine learning algorithms are then tested against these transformed datasets to observe variations in model performance resulting from non-linear dimensionality reduction.

5. **Comparative Analysis:** An in-depth analysis is conducted to compare the performance metrics of the ML algorithms before and after the application of dimensionality reduction techniques. This comparison is critical for understanding how each method of dimensionality reduction—linear and non-linear—affects the accuracy, precision, recall, and overall efficacy of the models.

6. **Performance Evaluation:** Finally, the performance of the machine learning models, post-dimensionality reduction, is evaluated using a set of metrics including Precision, Recall, Accuracy, Purity Score, Homogeneity Score, and

Completeness Score. This comprehensive evaluation provides insights into which dimensionality reduction techniques most effectively enhance model performance, thereby guiding future applications and research in machine learning and data preprocessing.

This structured approach allows for a detailed examination of the interplay between dimensionality reduction techniques and machine learning model performance, offering valuable insights into optimal preprocessing strategies for complex datasets like the 20 newsgroups.

6. RESULTS AND DISCUSSIONS

The results obtained from different dimensionality reduction techniques have been meticulously analyzed to understand the tradeoff between accuracy and computation time, a crucial consideration in real-world machine learning applications. Notably, these analyses were conducted using the 20 newsgroups dataset, which comprises over 100,000 features, posing significant computational challenges. To mitigate this complexity, the dataset was initially reduced based on a frequency threshold, resulting in a more manageable feature set of 9,000 features.

To comprehensively evaluate the performance of PCA and LDA in conjunction with different machine learning algorithms, the average results were computed over 5 trial runs. These results were subsequently tested across a spectrum of classifiers, including Random Forest, Support Vector Machines (SVM), and K-Means. The outcomes of these experiments are meticulously documented and presented in Tables 1, 2, and 3 for PCA and Tables 4, 5, and 6 for LDA, providing a detailed overview of the performance metrics obtained across various dimensionality reduction settings and classification models.

Following the initial preprocessing steps, the dataset underwent examination with linear dimensionality reduction techniques, specifically Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). These techniques are instrumental in reducing the dimensionality of the dataset while preserving the most relevant information for subsequent analysis.

Table 1 presents the results of a classification task using the Random Forest algorithm with different feature counts reduced by Principal Component Analysis (PCA).

Table 1: PCA-Random Forest

Features	Accuracy	Precision	Recall	Computation Time
9k	68.6%	0.682	0.686	102.42s
2	12.2% ±0.277%	0.120	0.122	**3.46s**
512	**68.8% ±0.245%**	**0.688**	**0.688**	43.68s
1024	67.7%±0.135%	0.680	0.677	62.92s
2048	68.0%±0.732%	0.670	0.680	92.81s

Table 2 presents the results of a classification task using the Support Vector Machine (SVM) algorithm with different feature counts reduced by Principal Component Analysis (PCA).

Table 2: PCA – SVM

Features	Accuracy	Precision	Recall	Computation Time
9k	72.6%	0.747	0.726	225.67s
2	15.2%±0.044%	0.134	0.152	**10.86s**
512	74.3%±0.162%	0.749	0.743	49.90s
1024	75.1%±0.073%	0.759	0.751	217.51s
2048	**75.6% ±0.196%**	**0.766**	**0.756**	349.28s

Table 3 presents the results of a classification task using the KMeans algorithm with different feature counts reduced by Principal Component Analysis (PCA).

Table 3: PCA – KMeans

Features	Purity Score	Homogeneity Score	Completeness Score	Computation Time
9K	**0.775**	0.188	0.502	94.92s
2	0.154±0.001	0.148	0.227	**1.76s**
512	0.152±0.014	0.218	0.499	13.16s
1024	0.174±0.008	**0.227**	**0.510**	27.26s
2048	0.215±0.039	0.106	0.446	50.37s

For PCA, a range of reduced components of 2, 512, 1024, and 2048 components were explored. Each of these dimensions had the explained variance ratios of 0.4%, 21%, 34%, and 54.6% respectively. As expected, the more the number of dimensions more the explained ratios.

Table 4 presents the results of a classification task using the Random Forest algorithm with different feature counts reduced by Linear Discriminant Analysis (LDA).

Table 4: LDA - Random Forest

Features	Accuracy	Precision	Recall	Computation Time
9K	**68.6%**	**0.682**	**0.686**	102.42s
2	17.8% ±0.5%	0.195	0.180	**2.74s**
15	57.0% ±0.0%	0.552	0.558	5.05s
17	57.6% ±0.9%	0.564	0.564	7.47s
19	57.2% ±0.45%	0.558	0.558	5.83s

Table 5 presents the results of a classification task using the Support Vector Machine (SVM) algorithm with different feature counts reduced by Linear Discriminant Analysis (LDA).

Table 5: LDA - SVM

Features	Accuracy	Precision	Recall	Computation Time
9K	**72.6%**	**0.747**	**0.726**	225.67s
2	16.8%	0.198	0.173	3.50s
15	60.0%	0.590	0.594	0.54s
17	61.6%	0.606	0.606	0.52s
19	64.0%	0.630	0.630	**0.46s**

Table 6 presents the results of a classification task using the KMeans algorithm with different feature counts reduced by Linear Discriminant Analysis (LDA).

Table 6: LDA - KMeans

Features	Purity Score	Homogeneity Score	Completeness Score	Computation Time
9K	**0.775**	0.188	0.502	94.92s
2	0.199	0.165	0.258	0.96s
15	0.320	**0.282**	0.484	0.98s
17	0.319	0.279	0.496	1.13s
19	0.312	0.282	**0.510**	**0.79s**

Conversely, Linear Discriminant Analysis (LDA) operates under the constraint that it can only reduce the number of features to at most n − 1 features, where n represents the number of classes in the dataset. This limitation stems from LDA's objective of maximizing class separability, necessitating a reduction in the feature space while preserving discriminatory information. Consequently, feature dimensions 2, 15, 17, and 19 were considered for LDA, each tailored to the specific class structure inherent in the dataset.

The visualization of the reduced feature sets using Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), Kernel PCA, and Uniform Manifold Approximation and Projection (UMAP) with 2 components provides valuable insights into the distribution and clustering of data points in reduced-dimensional space. Figures 5, 6, 7 and 8 illustrate the distinct representations generated by each dimensionality reduction technique, offering a visual depiction of the dataset's underlying structure and patterns.

Figure 5 is the 2-dimensional reduction using PCA, which doesn't show a proper distinction between classes as most of them are clustered together

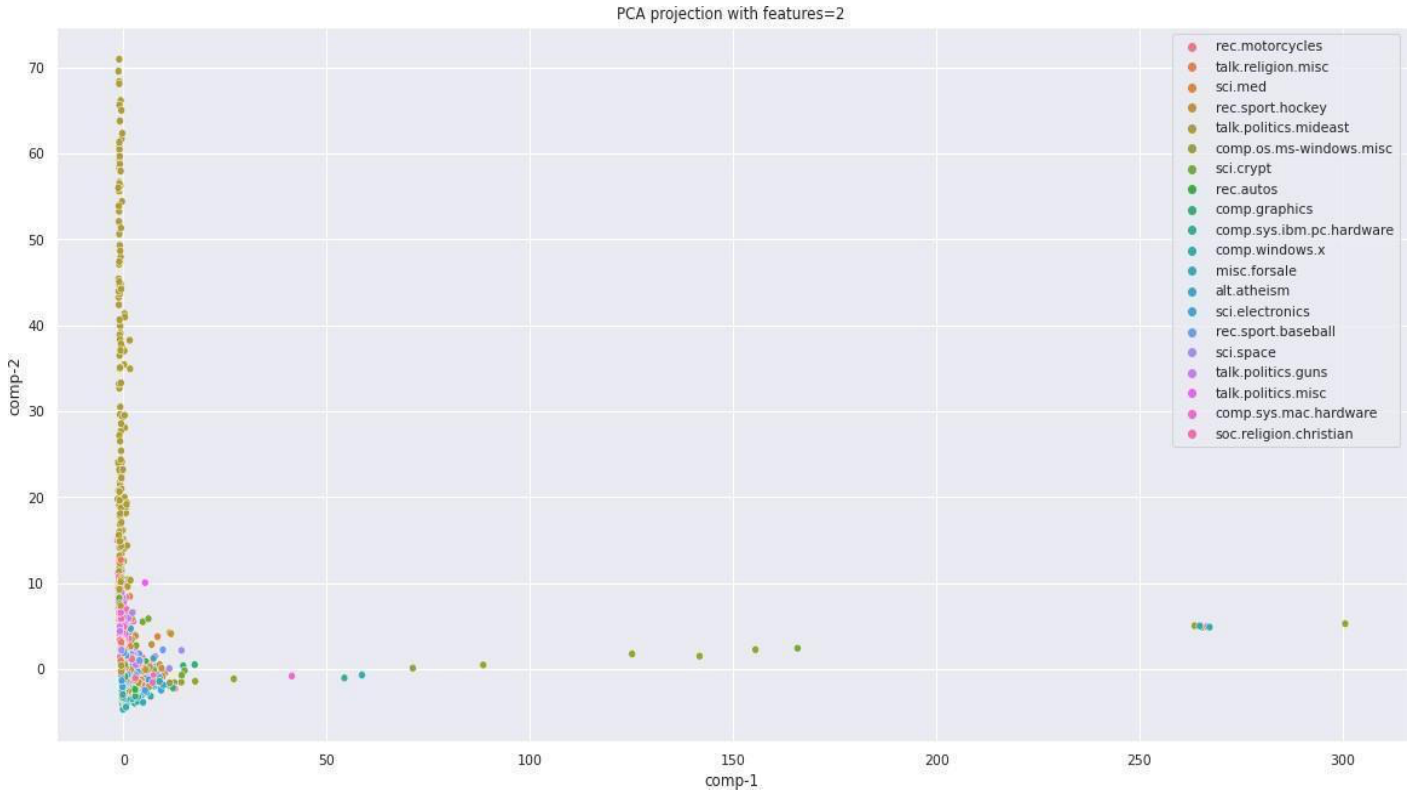

Figure 5: This is a plot after using PCA to reduce the data dimension from the original 9k to 2 dimensions

However, in Figure 6, LDA's 2-dimensional reduction seems to be better as we see

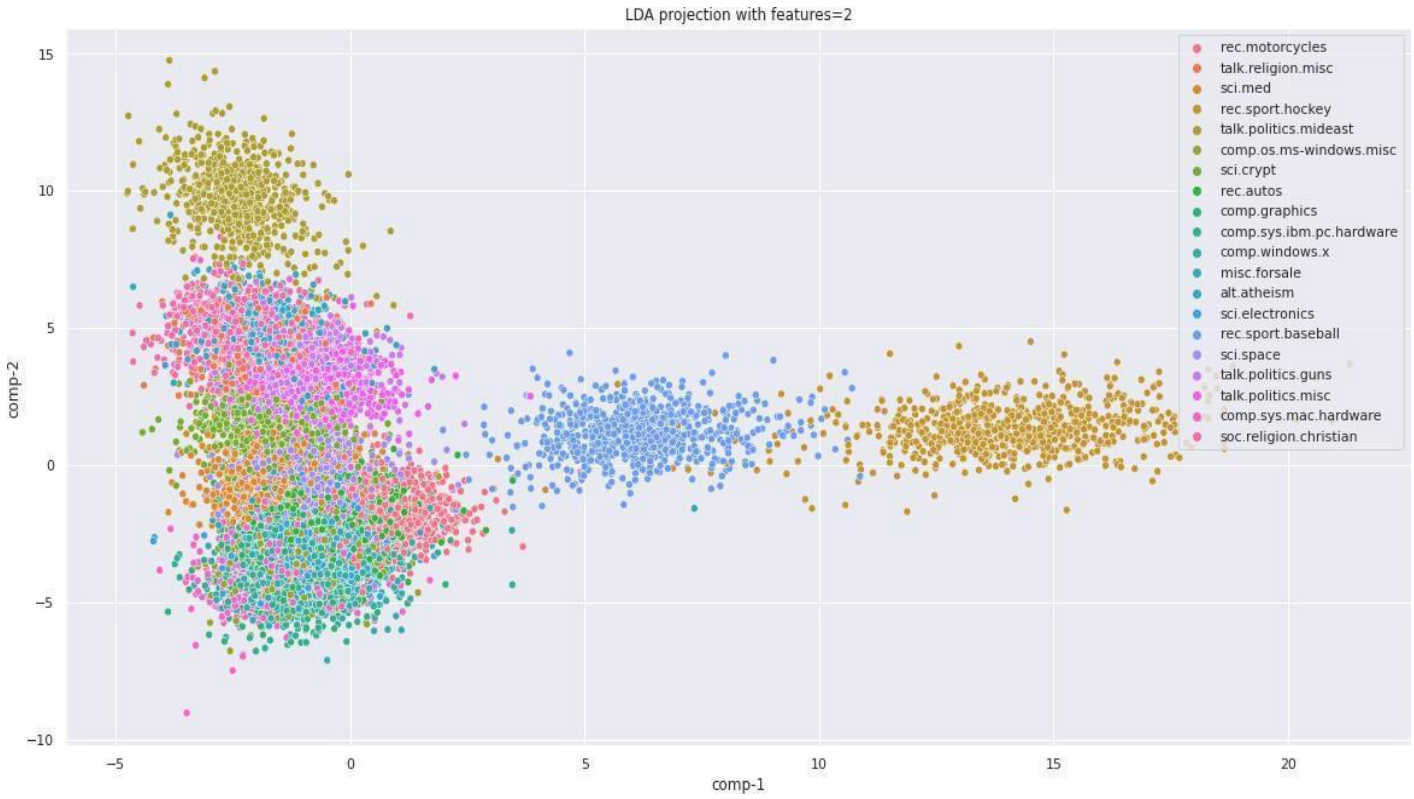

Figure 6: This is a plot after using LDA to reduce the data dimension from the original 9k to 2 dimensions

In addition to exploring linear dimensionality reduction techniques, our analysis delved into the realm of nonlinear methods, specifically Kernel PCA and Uniform Manifold Approximation and Projection (UMAP). These techniques offer alternative approaches to dimensionality reduction, particularly suited for capturing nonlinear relationships within complex datasets. Kernel PCA was subjected to rigorous testing across varying numbers of components, including 2, 512, 1024, and 2048, aiming to comprehensively evaluate its performance across a range of dimensionality reduction settings. Meanwhile, UMAP underwent experimentation with dimensions of 2, 32, 64, and 128 components, with particular attention paid to identifying an optimal dimensionality for achieving peak performance.

In Figure 7 using Kernel PCA doesn't seem very beneficial as we don't see any distinction.

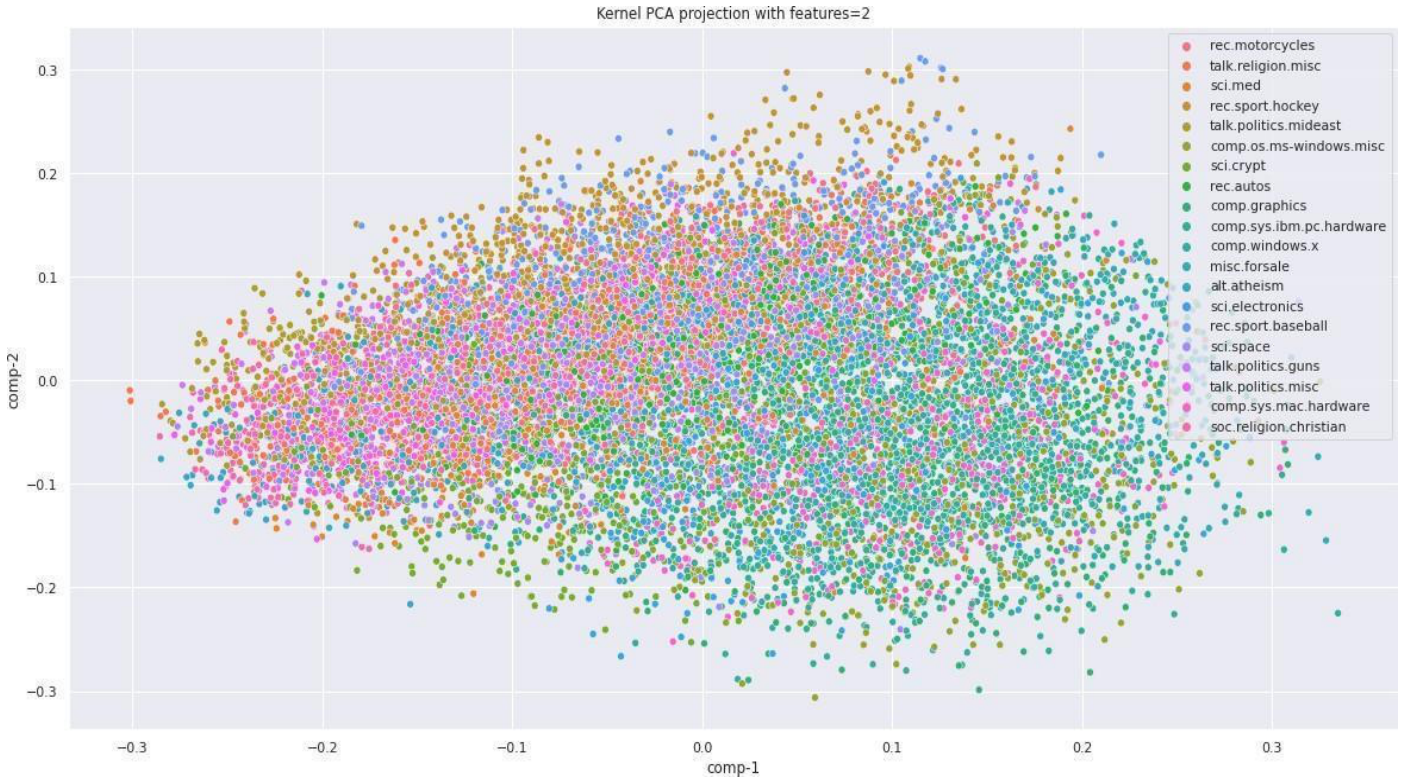

Figure 7: This is a plot after using Kernel PCA to reduce the data dimension from the original 9k to 2 dimensions

It's worth noting that the component selections differed for UMAP due to the unique characteristics of its performance curve. Through iterative experimentation, I observed a notable peak in performance around 32 features, with deviations in either direction resulting in reduced accuracy.

Here, UMap performs the best to separate classes as shown in Figure 8.

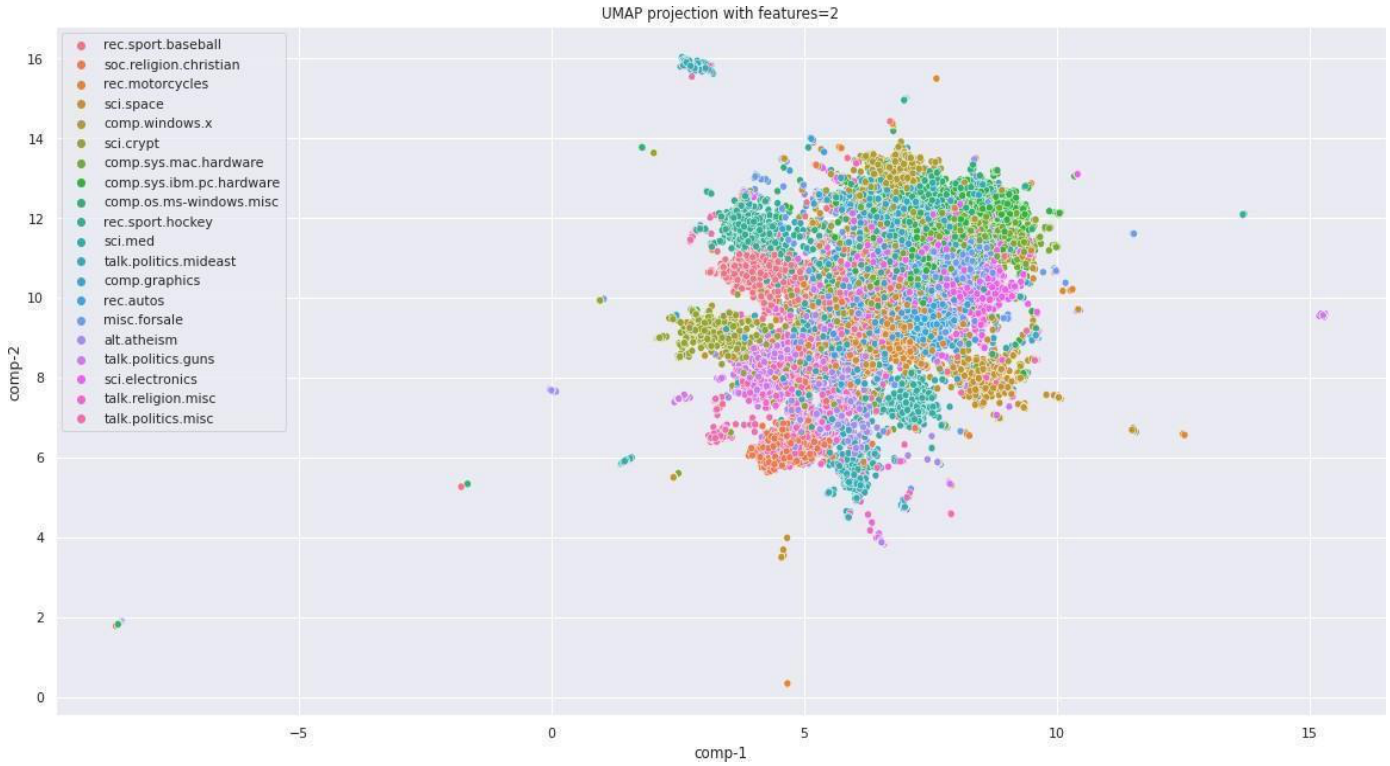

Figure 8: This is a plot after using UMap to reduce from the original 9k to 2 dimensions

In contrast, Kernel PCA exhibited a more stable performance across varying numbers of components, showcasing its robustness in capturing nonlinear patterns within the dataset.

To provide a comprehensive overview of the performance of Kernel PCA and UMAP in conjunction with different machine learning algorithms, the average results of reduced feature datasets were computed over 5 trial runs. These datasets were subsequently evaluated across Random Forests, Support Vector Machines (SVM), and K-means classifiers. The detailed outcomes of these experiments are meticulously documented and presented in Tables 7, 8, and 9 for Kernel PCA and Tables 10, 11, and 12 for UMAP, offering insights into the efficacy of these nonlinear dimensionality reduction techniques across various classification models and dimensionality settings.

Table 7 presents the results of a classification report using the Random Forest algorithm with different feature counts reduced by Kernel PCA

Table 7: Kernel PCA - Random Forest

Features	Accuracy	Precision	Recall	Computation Time
9K	68.6%	0.682	0.686	102.42s
2	14.2±0.31%	0.142	0.142	**0.040s**
512	**68.8 ±0.43%**	**0.696**	**0.688**	0.732s
1024	68.2±0.48%	0.688	0.682	1.184s
2084	66.3±0.39%	0.674	0.663	1.596s

Table 8 presents the results of a classification report using the Support Vector Machine (SVM) algorithm with different feature counts reduced by Kernel PCA

Table 8: Kernel PCA - SVM

Features	Accuracy	Precision	Recall	Computation Time
9K	72.6%	0.747	0.726	225.67s
2	18.9%±0	0.192	0.189	**0.176s**
512	75.2%±0	0.76027	0.751	1.389s
1024	**76.4%±0**	**0.7755**	**0.764**	4.802s
2084	76.1%±0	0.774	0.760	9.257s

Table 9 presents the results of a classification report using KMeans algorithm with different feature counts reduced by Kernel PCA

Table 9: Kernel PCA - KMeans

Features	Purity Score	Homogeneity Score	Completeness Score	Computation Time
9K	**0.775**	0.188	0.502	94.92s
2	0.184±0.012	0.183	0.224	**0.026s**
512	0.203±0.025	0.225	0.508	0.251s
1024	0.306±0.0489	0.314	0.530	0.544s
2048	0.334±0.04	**0.332**	**0.542**	1.129s

The effectiveness of UMAP in visualizing complex datasets and facilitating clustering tasks underscores its utility as a powerful dimensionality reduction technique.

Table 10 presents the results of a classification report using the Random Forest algorithm with different feature counts reduced by UMap.

Table 10: UMap - Random Forest

Features	Accuracy	Precision	Recall	Computation Time
9K	**68.6%**	**0.682**	**0.686**	102.42s
2	52.52±1.08%	0.524	0.525	**3.144s**
32	60.64±1.49%	0.622	0.606	10.763s
64	57.58±0.97%	0.607	0.576	17.178s
128	43.15±2.72%	0.534	0.432	27.175s

Conversely, the visualization results from UMAP showcased a notable separation and clear visibility of clusters within the reduced-dimensional space. This observation underscores UMAP's effectiveness in preserving the local and global structure of the dataset, facilitating the identification of distinct categories or clusters. Notably, UMAP demonstrated strong performance in conjunction with K-Means clustering, leveraging its ability to represent the underlying data distribution accurately.

Table 11 presents the results of a classification report using the Support Vector Machine (SVM) algorithm with different feature counts reduced by UMap.

Table 11: UMap - SVM

Features	Accuracy	Precision	Recall	Computation Time
9K	**72.6%**	**0.747**	**0.726**	225.67s
2	55.03±1.25%	0.558	0.551	**10.585s**
32	60.99±0.61%	0.618	0.61	10.83s
64	56.41±1.84%	0.589	0.564	14.898s
128	46.08±1.85%	0.571	0.461	21.597s

Within the context of Kernel PCA, the performance of different kernel functions, including sigmoid, radial basis function (RBF), and polynomial, was rigorously evaluated. These kernel functions serve as nonlinear transformations applied to the input data, enabling the discovery of complex relationships within the dataset. Interestingly, the analysis revealed that the sigmoid kernel function yielded superior results compared to RBF and polynomial kernels across various machine learning algorithms, including Random Forest, Support Vector Machines (SVM), and K-Means.

However, it's essential to note that UMAP exhibited slightly lower performance scores when utilized with SVMs compared to other dimensionality reduction algorithms. Despite this, the overall effectiveness of UMAP in visualizing and clustering high-dimensional datasets remains evident, highlighting its versatility and utility across a range of machine-learning tasks.

Table 12 presents the results of a classification report using the KMeans algorithm with different feature counts reduced by UMap.

Table 12: UMap - KMeans

Features	Purity Score	Homogeneity Score	Completeness Score	Computation Time
9K	**0.775**	0.188	0.502	94.92s
2	0.539±0.0056	0.419	0.459	**1.65s**
32	0.619±0.013	0.444	0.502	2.391s
64	0.607±0.0147	**0.445**	**0.523**	3.253s
128	0.55±0.0418	0.424	0.508	4.762s

Despite the stability exhibited by Kernel PCA, UMAP's performance exhibited variability across multiple runs, evidenced by a high standard deviation. However, this variability did not detract from the promising results obtained with UMAP, particularly in the realm of data visualization and K-means clustering.

In summary, visualization results demonstrate the effectiveness of dimensionality reduction techniques in revealing underlying dataset patterns. After preprocessing, both linear (PCA, LDA) and nonlinear methods (Kernel PCA, UMAP) were applied. Reduced datasets trained various ML models—Random Forest, SVM, K-Means. PCA consistently outperformed LDA, highlighting its versatility. Kernel PCA excelled over UMAP with Random Forest, and SVM, but UMAP was optimal for K-Means, showcasing its dimensionality projection and visualization capabilities. UMAP balances accuracy and efficiency, making it suitable for high-dimensional datasets. Although UMAP didn't match others in accuracy, its clustering performance and scatter plot separability suggest its potential for exploratory analysis. Despite LDA's lower accuracy, it performs well given the reduced feature space. Contextualizing performance based on task objectives and constraints is crucial. Dimensionality reduction trades accuracy for computational efficiency and mitigates overfitting. PCA and Kernel PCA expedite Random Forest and SVM computation while maintaining accuracy. UMAP enhances clustering tasks and aids exploratory analysis. Considering task-specific requirements is vital for optimizing model performance and interpretability. This analysis elucidates the nuanced

tradeoffs and benefits of dimensionality reduction techniques in real-world machine learning applications.

7. CONCLUSION

In conclusion, this study highlights the significant advancements in computational efficiency attained through dimensionality reduction techniques. The substantial reduction in model-building time presents opportunities for leveraging these techniques in scenarios prioritizing rapid response times, such as large-scale recommendation systems like those of Netflix and Amazon, without compromising overall accuracy.

However, it's crucial to recognize the tradeoff between computational speed and accuracy. In domains like banking and medical sciences, where precision is paramount, even minor decreases in accuracy can have significant repercussions. Therefore, a nuanced approach to model optimization is essential.

The potential of Uniform Manifold Approximation and Projection (UMAP) warrants further investigation, particularly in scenarios with larger feature sizes. Despite limitations in creating higher-dimensional features due to computational demands, UMAP demonstrated promising results in clustering and classification tasks, suggesting potential for further optimization with higher-dimensional feature spaces.

In summary, while dimensionality reduction techniques offer compelling benefits in computational efficiency and scalability, their adoption must consider the specific requirements and constraints of each application domain. Continued exploration and refinement of these techniques, alongside integration with advanced algorithms, hold promise for enhancing machine learning workflows and addressing complex real-world challenges.

8. FUTURE WORK

Future research endeavors could focus on evaluating the effectiveness of dimensionality reduction techniques in scenarios where minor fluctuations in accuracy over unseen data are acceptable. By exploring the robustness of these techniques to slight variations in accuracy, researchers can better understand their practical utility across diverse applications.

Furthermore, there is ample opportunity for further exploration of dimensionality

reduction techniques in conjunction with advanced algorithms such as Deep Neural Networks (DNNs), Convolutional Neural Networks (CNNs), and Recurrent Neural

Networks (RNNs). Investigating the synergies between dimensionality reduction and these complex algorithms could unlock new avenues for enhancing model performance and scalability across a range of domains.

9. COURSEWORK

a. CSCI 642 – Natural Language Processing

In Spring 2023, I enrolled in the Natural Language Processing course under Professor Dr. Maninder Singh, a decision that profoundly influenced my approach to feature engineering in text datasets for the current project. The course's Text Feature Engineering module provided invaluable insights into extracting meaningful features from textual data. Through hands-on assignments, I mastered techniques like Bag-of-Words, TF-IDF, and word embeddings (Word2Vec, GloVe), essential for preprocessing raw text from the 20 newsgroups dataset. Additionally, I learned to address text-specific challenges such as stop words, stemming, lemmatization, and text encoding issues, further enhancing my understanding. Leveraging these techniques, I effectively preprocessed the text data, optimized it for analysis, and applied dimensionality reduction methods and machine learning models. In summary, the course equipped me with essential skills and knowledge in text feature engineering, directly contributing to the success of the project. The practical experience gained proved invaluable in tackling complex challenges and deriving actionable insights from the data.

b. CSCI 540 – Artificial Intelligence

In Summer 2022, I enrolled in the Introduction to Artificial Intelligence course led by Professor Dr. Maninder Singh, an experience that profoundly influenced my understanding of dimensionality reduction techniques such as Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA). This course provided a comprehensive exploration of PCA and LDA, elucidating their principles, applications, and implications in machine learning. Through rigorous coursework and practical assignments, I gained a solid foundation in leveraging PCA to reduce the dimensionality of high-dimensional datasets while preserving essential information. Additionally, LDA introduced me to supervised dimensionality reduction, enabling me to effectively discriminate between classes in feature space. The knowledge and skills acquired in the Artificial Intelligence course served as a springboard for

further exploration into advanced dimensionality reduction techniques, a central focus of the current project. Inspired by the insights gleaned from PCA and LDA, I embarked on investigating alternative methods such as Uniform Manifold Approximation and Projection (UMAP) and Kernel PCA. These techniques, which build upon the principles learned in the course, offer unique advantages in capturing complex data structures and enhancing the performance of machine learning models. Thus, the Artificial Intelligence course provided the foundational knowledge and inspiration necessary to propel my exploration into dimensionality reduction techniques, ultimately shaping the trajectory of the current project.

c. CSCI 681 – Technical Presentation

In Spring 2023, I undertook CSCI 681 – Technical Presentation, instructed by Professor Jie Meichsner. This course was instrumental in honing my skills in crafting impactful presentations. Through comprehensive instruction, I learned to create engaging PowerPoint presentations and deliver articulate talks effectively. Moreover, emphasis was placed on meticulous documentation, highlighting its importance in professional settings. These acquired skills have greatly benefited my internship, where I regularly deliver presentations in the realm of process automation and development. Clear and engaging communication, as instilled by this course, remains indispensable in effectively conveying ideas and strategies.

d. Dimensionality Reduction: Machine Learning in Python - Udemy

Enrolling in the "Dimensionality Reduction: Machine Learning in Python" course on Udemy proved to be a pivotal step in enhancing my proficiency in visualization techniques, a crucial aspect of my project. Led by experienced instructors, the course equipped me with comprehensive knowledge and practical skills in mastering visualization and dimensionality reduction in Python. By delving into advanced concepts such as Principal Component Analysis (PCA), Locally Linear Embedding (LLE), t-distributed Stochastic Neighbor Embedding (t-SNE), Multidimensional Scaling (MDS), ISOMAP, and Fisher Discriminant Analysis, I gained a deep understanding of how to effectively visualize high-dimensional data and uncover underlying patterns. Moreover, the course provided invaluable insights into the application of robust Machine Learning techniques for dimensionality reduction, empowering me to apply these methods confidently in my project. Through hands-on challenges and assignments, I honed my problem-solving skills and developed efficient workflows, enabling me to approach visualization tasks with a structured and analytical mindset.

10. REFERENCES

[1] Jolliffe Ian T. and Cadima Jorge. “Principal component analysis: a review and recent developments”. In: *The Royal Society* 374 (2016). doi: https://doi.org/10.1098/rsta.2015. 0202.

[2] Alaa Tharwat et al. “Linear discriminant analysis: A detailed tutorial”. In: *AI Commun.* 30 (2017), pp. 169–190. doi: 10.3233/AIC-170729.

[3] Bernhard Scho¨lkopf, Alexander Smola, and Klaus Robert Mu¨ller. “Kernel principal component analysis”. English. In: Publisher Copyright: © Springer-Verlag Berlin Heidelberg 1997.; 7th International Conference on Artificial Neural Networks, ICANN 1997 ; Conference date: 08-10- 1997 Through 10-10-1997. 1997, pp. 583–588. ISBN: 3540636315. doi: 10.1007/bfb0020217.

[4] Mebarka Allaoui, Mohammed Lamine Kherfi, and Abdelhakim Cheriet. “Considerably Improv- ing Clustering Algorithms Using UMAP Dimensionality Reduction Technique: A Comparative Study”. In: *Image and Signal Processing*. Ed. by Abderrahim El Moataz et al. Cham: Springer International Publishing, 2020, pp. 317–325. isbn: 978-3-030-51935-3. doi: 10.1007/978-3- 030-51935-3_34.

[5] G. Thippa Reddy et al. “Analysis of Dimensionality Reduction Techniques on Big Data”. In: *IEEE Access* 8 (2020), pp. 54776–54788. doi: 10.1109/ACCESS.2020.2980942.