Report

भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

PCA on Large Datasets

EP4130: Data Science and Analysis Project

Student: Shashank Shanbhag

Roll Number: CS20BTECH11061

Student: Satwik Arawalli

Roll Number: ME20BTECH11058

Abstract

Principal Component Analysis (PCA) is a widely used technique for dimensionality reduction in large datasets. In this project report, we explore the application of PCA on large datasets, one of them being the Labeled Faces in the Wild (LFW) dataset. We present a step-by-step approach to using PCA for feature extraction and dimensionality reduction, followed by a demonstration of its usefulness in face recognition tasks. Additionally, we explore other potential applications of PCA in large datasets, including outlier detection and image compression. Our results demonstrate that PCA can effectively reduce the dimensionality of large datasets while preserving essential features and providing valuable insights.

Contents

1. Introduction

PCA (Principal Component Analysis) is an unsupervised learning algorithm that is widely used for dimensionality reduction and feature extraction on large datasets. With the advent of big data, handling large datasets has become a challenging task in the field of data analysis. PCA helps in reducing the number of features while retaining most of the information, thereby enabling faster processing of the data. This technique has been successfully applied to various fields such as image processing, signal processing, genomics, and finance, among others.

In this project, we explore the use of PCA on large datasets with a focus on the LFW (Labeled Faces in the Wild) dataset. The LFW dataset is a collection of more than 13,000 face images of 5,749 people collected from the internet. With such a large dataset, performing tasks such as face recognition, classification, and clustering becomes computationally expensive. PCA can be used to extract the most important features from the dataset and reduce the dimensionality, thereby making the analysis more efficient.

In this report, we provide a step-by-step guide on using PCA on the LFW dataset. We first introduce the LFW dataset and its properties. We then explain the PCA algorithm and how it can be used for feature extraction and dimensionality reduction. We also provide code examples for performing PCA on the LFW dataset and demonstrate its effectiveness in reducing the dimensionality of the dataset while retaining the most important information. Finally, we discuss some additional applications of PCA on large datasets.

2. Background

Principal Component Analysis (PCA) is a popular statistical technique used for data dimensionality reduction, visualization, and feature extraction. PCA is a linear transformation method that identifies the directions of the maximum variance in high-dimensional data and projects it onto a lower-dimensional space while preserving the essential information in the data. PCA is widely used in various fields such as finance, image processing, natural language processing, genetics, and many more. PCA is especially useful in dealing with large datasets, where it can reduce the computational complexity and memory requirements of data analysis tasks. In this project, we explore the application of PCA on large datasets, specifically the Labeled Faces in the Wild (LFW) dataset, and investigate its performance in tasks such as face recognition, image compression, and feature extraction.

3. Algorithm

Principal Component Analysis (PCA) is a dimensionality reduction technique used to reduce the number of features in high-dimensional data by finding a lower-dimensional representation of the data that retains as much information as possible.

Given a dataset with n observations and p features, PCA works by finding a new set of p uncorrelated variables called principal components (PCs), which are linear combinations of the original p features. The first PC captures the largest variance in the data, the second PC captures the second-largest variance orthogonal to the first PC, and so on.

The principal components are found by decomposing the covariance matrix of the dataset or the Singular Value Decomposition (SVD) of the dataset. The PCs are then ranked in descending order of the amount of variance they explain, and the user selects the number of components to retain.

Algorithm 1 PCA Algorithm

Require: X: Data matrix, k: Number of principal components

1: $\mu \leftarrow$ Mean of each feature in X
2: $X \leftarrow X - \mu$
3: $S \leftarrow \frac{1}{n} X^T X$
4: $\lambda, V \leftarrow$ Eigenvalues and eigenvectors of S
5: $\Lambda \leftarrow$ Diagonal matrix of eigenvalues sorted in descending order
6: $V_k \leftarrow$ First k eigenvectors in V
7: $Z \leftarrow XV_k$
8: **return** Z, Λ, V_k

Here, X is the data matrix with n observations and d features, μ is the mean vector of each feature, S is the covariance matrix of the centered data, λ and V are the eigenvalues and eigenvectors of S, Λ is a diagonal matrix of the eigenvalues sorted in descending order, V_k is the matrix of the first k eigenvectors, and Z is the transformed data matrix with the first k principal components.

3.1 Demonstration using Toy example

```
import numpy as np
from sklearn.decomposition import PCA
X = np.array([[-1, -1], [-2, -1], [-3, -2], [1, 1], [2, 1], [3, 2]])
pca = PCA(n_components=2)
pca.fit(X)
PCA(n_components=2)
print(pca.explained_variance_ratio_)
```

✓ 0.4s

```
[0.99244289 0.00755711]
```

Figure 1: Toy example

4. Datasets

We use two datasets to explain how PCA is useful.

- LFW Dataset
- S&P500 companies Dataset

4.1 LFW Dataset

All the plots are generated for a large subset of data taken from LFW Dataset.

4.1.1 Classification using KNN after PCA

We take LFW Dataset in such a way that the extracted dataset will only retain pictures of people that have at least 50 different pictures. Some of the images in the dataset can be seen in Figure 2.

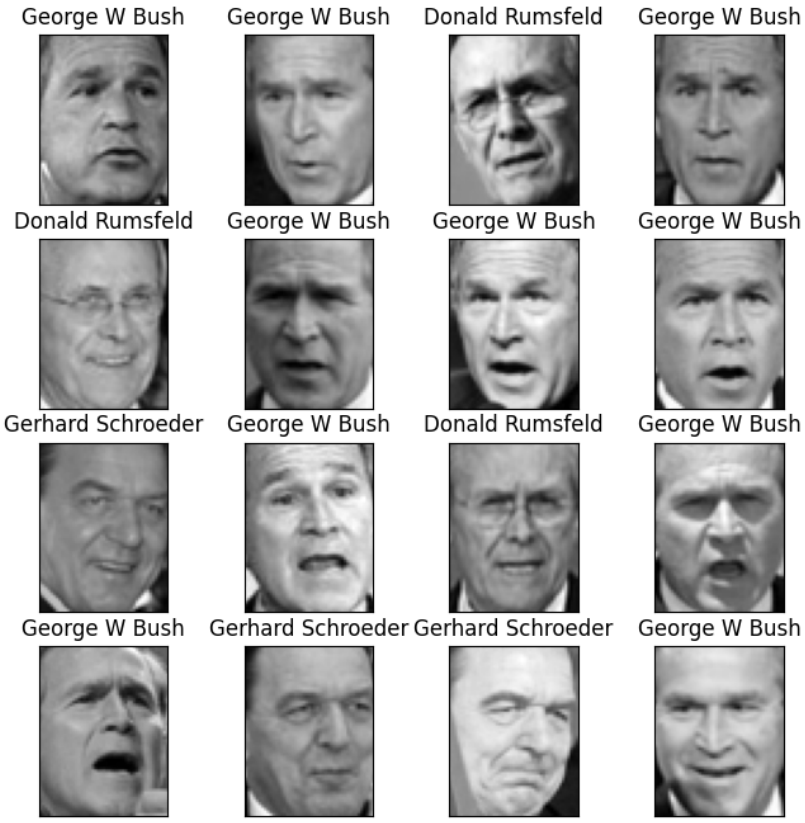

Figure 2: Visualization of images in dataset

	precision	recall	f1-score	support
Donald Rumsfeld	0.92	0.43	0.59	28
George W Bush	0.81	0.99	0.89	137
Gerhard Schroeder	0.90	0.36	0.51	25
accuracy			0.83	190
macro avg	0.88	0.59	0.66	190
weighted avg	0.84	0.83	0.80	190

Figure 3: Classification Report

We apply PCA to achieve dimensionality reduction and then we train KNN classifier with k=5 on training dataset. We achieve an accuracy of 83%. The results of the classification can be seen in Figure 3.

The first 25 eigenfaces learned by PCA can be seen in Figure 4. The plot of the explained variance ratio for each principal component helps in determining how many principal components to keep for dimensionality reduction. It can be seen in Figure 5.

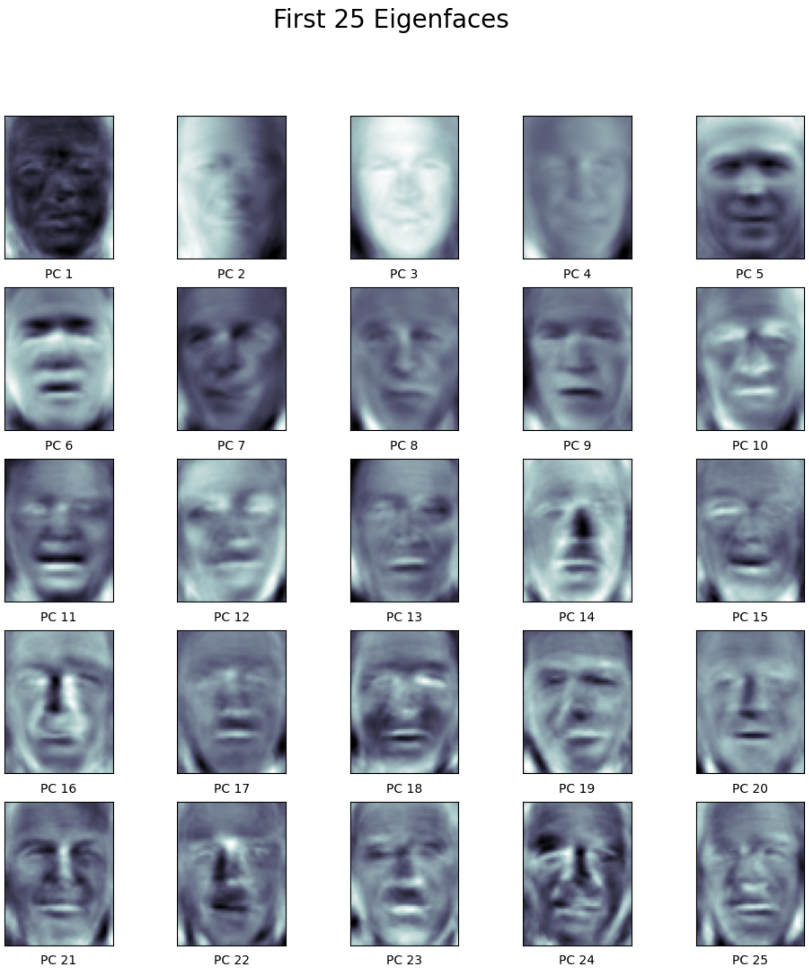

Figure 4: The first 25 eigenfaces learned by PCA

We can see the classification of KNN classifier after PCA on some images of test set in Figure 6. To understand how eigenvalues behave with increase in number of principal components, we use a plot named Scree plot, which can be seen in Figure 7. This plot gives us the eigenvalues in decreasing order. The scree plot can help you determine how many principal components to keep for dimensionality reduction. We can see the plot of cumulative explained variance with number of principal components in Figure 8. We can understand PCA better by observing how classification accuracy varies with number of principal components, which can be seen Figure 9. All these plots help us in choosing the optimal number of principal components for dimensionality reduction.
We see how PCA is successful in capturing essential information by comparing original images with reconstructed images with varying number of principal components in Figure 11. This also shows to what extent we can lose information after applying PCA. If you have a large number of features in your dataset, you can use PCA to select a subset of the most important features. PCA can help you identify the features

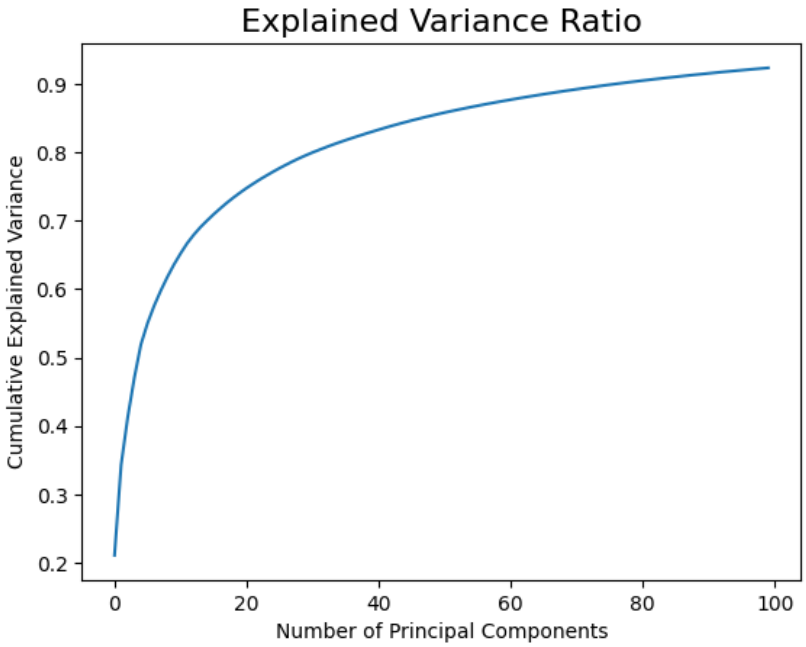

Figure 5: Explained variance vs No. of principal components

that explain the most variance in your data, and you can select the top N features for your analysis.

4.1.2 Visualization in 3D

We have used 100 PC's to visualize training dataset. So any data can be visualized using PCA. This can be seen in Figure 10. Each point represents a face image and is colored based on the person it belongs to.

4.1.3 Face clustering using PCA

We get principal components using PCA and then apply KMeans algorithm based on the principal components obtained. It first projects the face images onto the first few principal components, then applies KMeans clustering with the number of clusters set to 10. Results can be seen in Figure 12. You can change the values of number of components and k to see the effect of using different numbers of principal components and clusters. PCA can be used as a pre-processing step for clustering algorithms. By projecting the data onto the first few principal components, you can reduce the dimensionality of the data and make clustering algorithms more efficient.

4.1.4 Outlier Detection using PCA

We take LFW Dataset in such a way that the extracted dataset will only retain pictures of people that have at least 70 different pictures. We apply PCA after normalizing the data and remove outliers using Mahalanobis distance. PCA can detect outliers by identifying data points that are far from the rest of the data along one or more principal components. We find 76 outliers in the dataset taken.

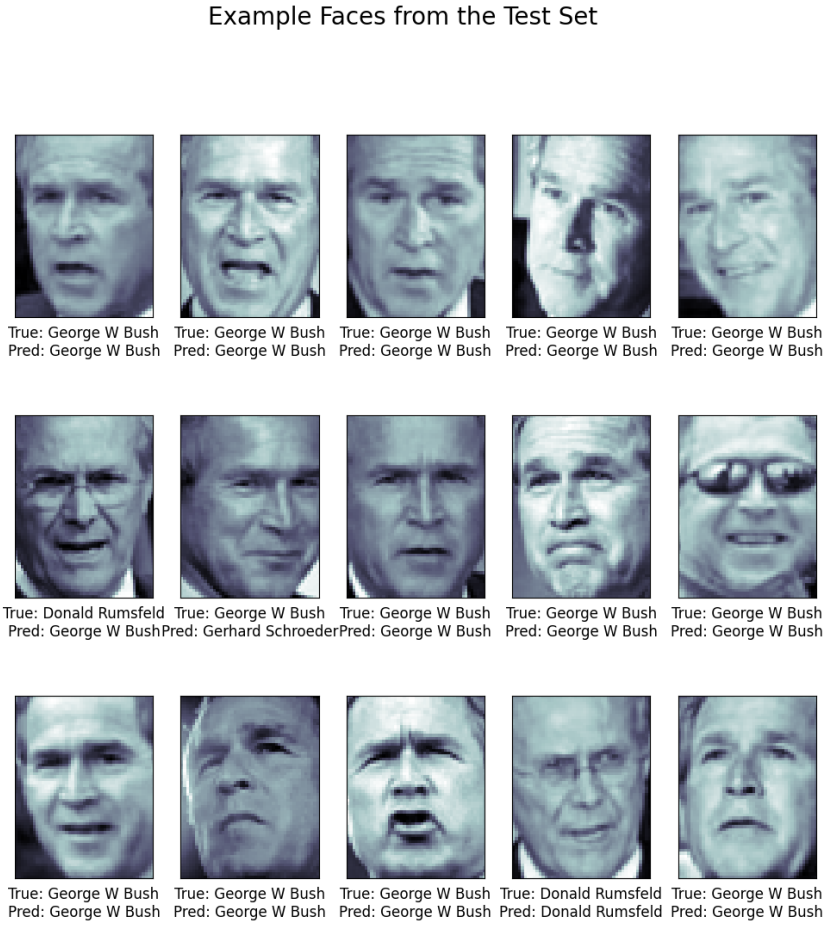

Figure 6: Example Faces from the Test Set with true and predicted label

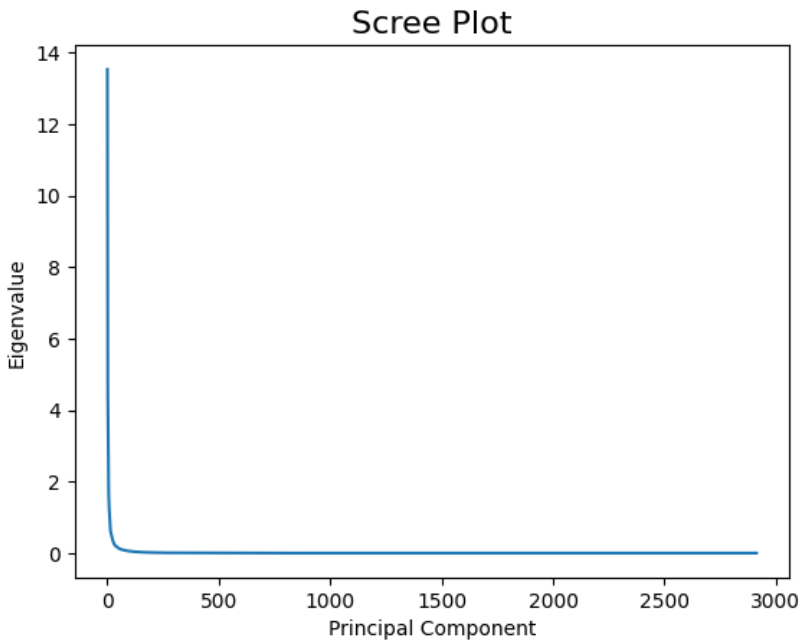

Figure 7: Scree plot

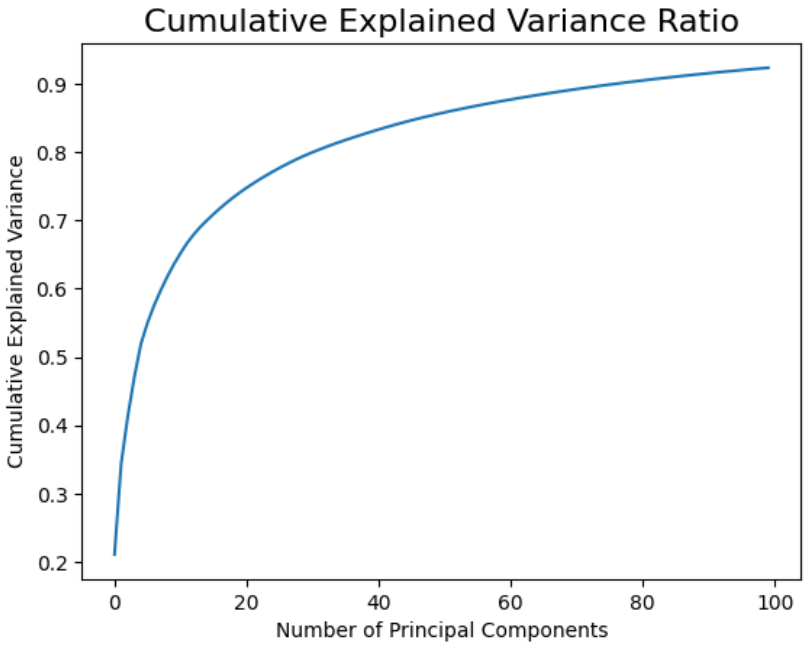

Figure 8: Cumulative explained variance VS No. of principal components

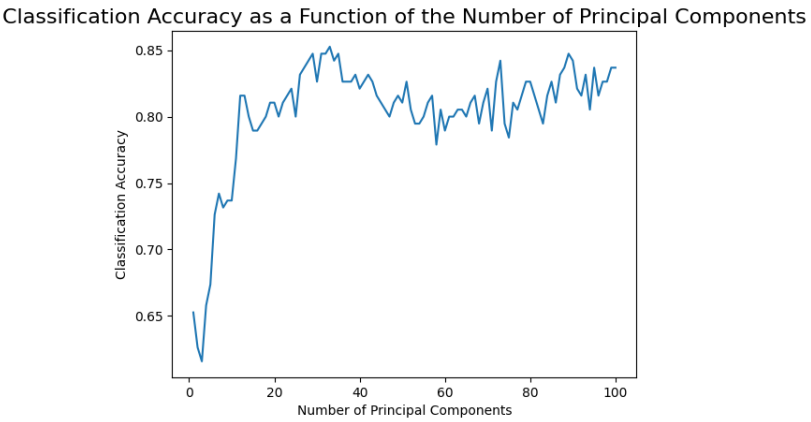

Figure 9: Classification accuracy VS No. of principal components

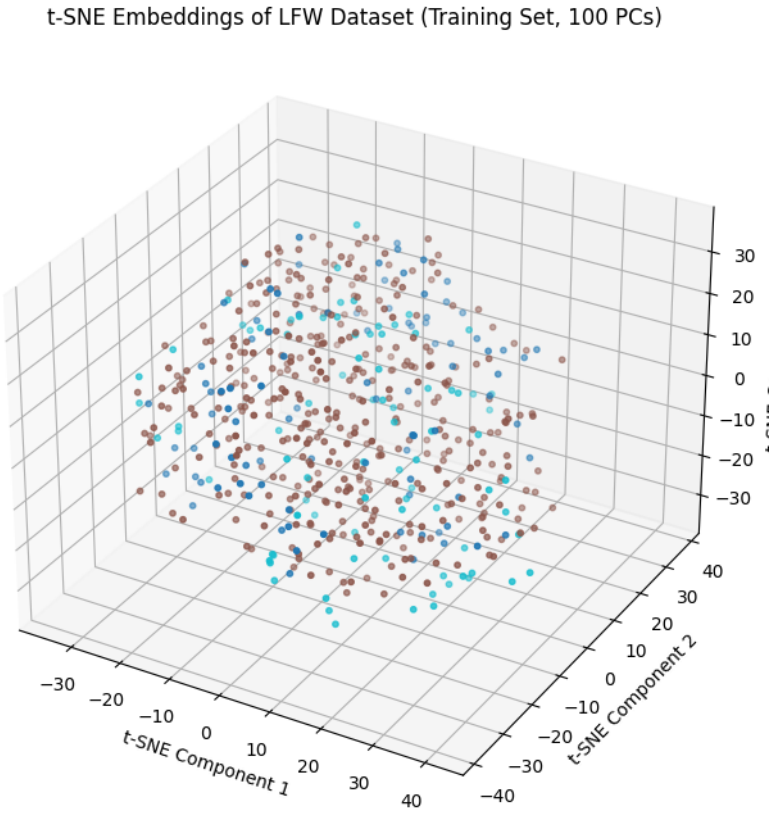

Figure 10: Visualize the principal components in 3D

4.1.5 Data Compression using PCA

We take the dataset with no outliers and compress the data using PCA. The result of compressing 10 images taken from the dataset can be seen in Figure 13. This can be useful if you have a large dataset and want to reduce its size without losing too much information. PCA can transform your high-dimensional data into a lower-dimensional space while preserving as much variance as possible.

4.1.6 Dimensionality Reduction and Visualization

We visualize the data into two dimensions, ensuring the ability to understand some patterns in high-dimensional data. It can be seen for 3 people in LFW Dataset in Figure 14. You can use PCA to create scatter plots or heatmaps of your data, or to create interactive visualizations that allow you to explore your data in different ways.

4.2 S&P500 companies dataset

The SP 500 dataset refers to a collection of historical data on the performance of the Standard Poor's 500 index, which is a stock market index that measures the performance of 500 large-cap publicly traded companies in the United States.
The SP 500 dataset typically includes a variety of financial data, including daily or monthly closing prices, total returns, dividend yields, market capitalization, and other market metrics, covering a period of several decades.

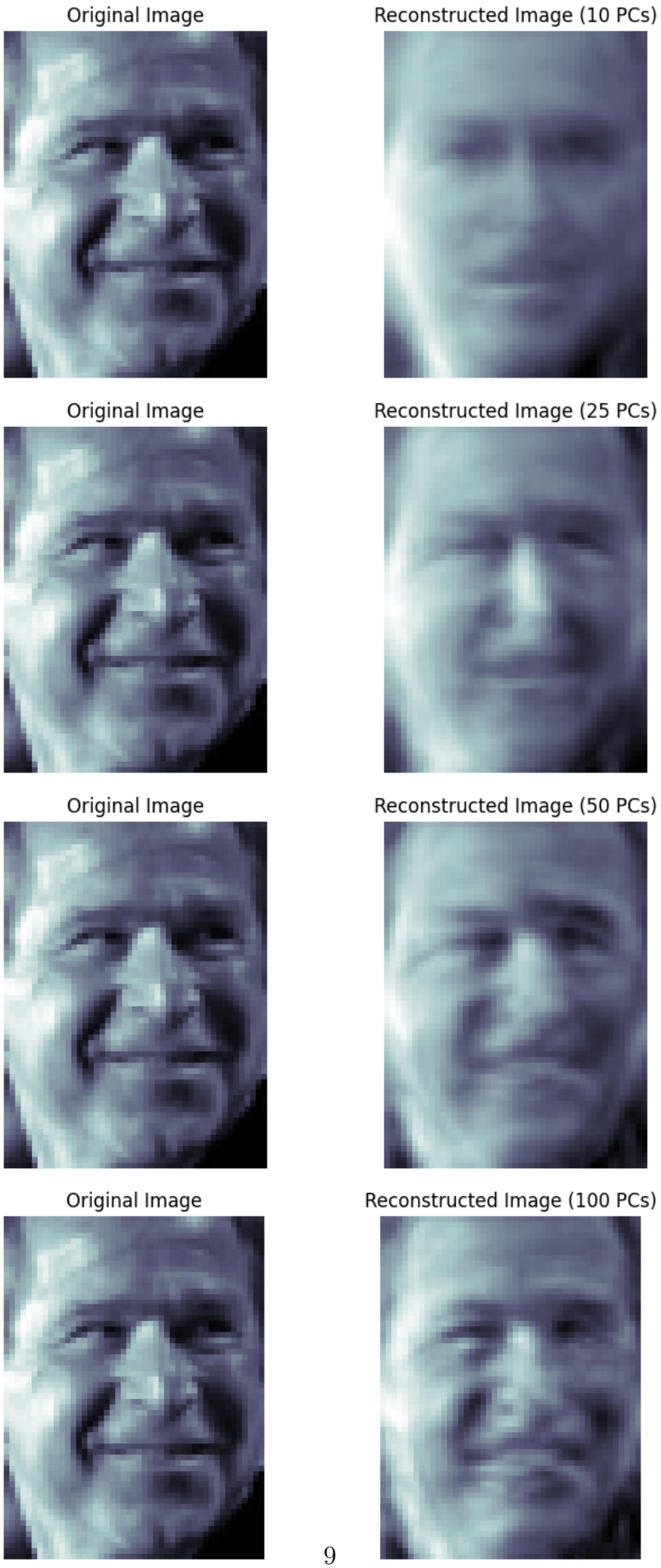

Figure 11: Original VS Reconstructed image

Figure 12: Clusters created based on principal components

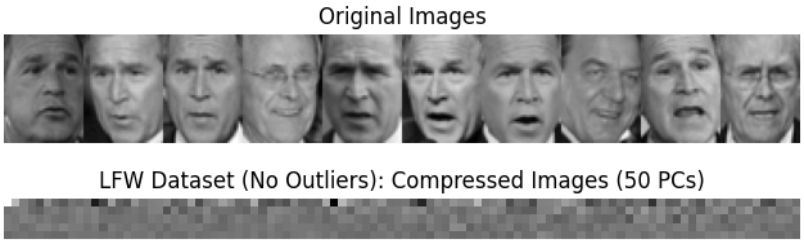

Figure 13: Image compression

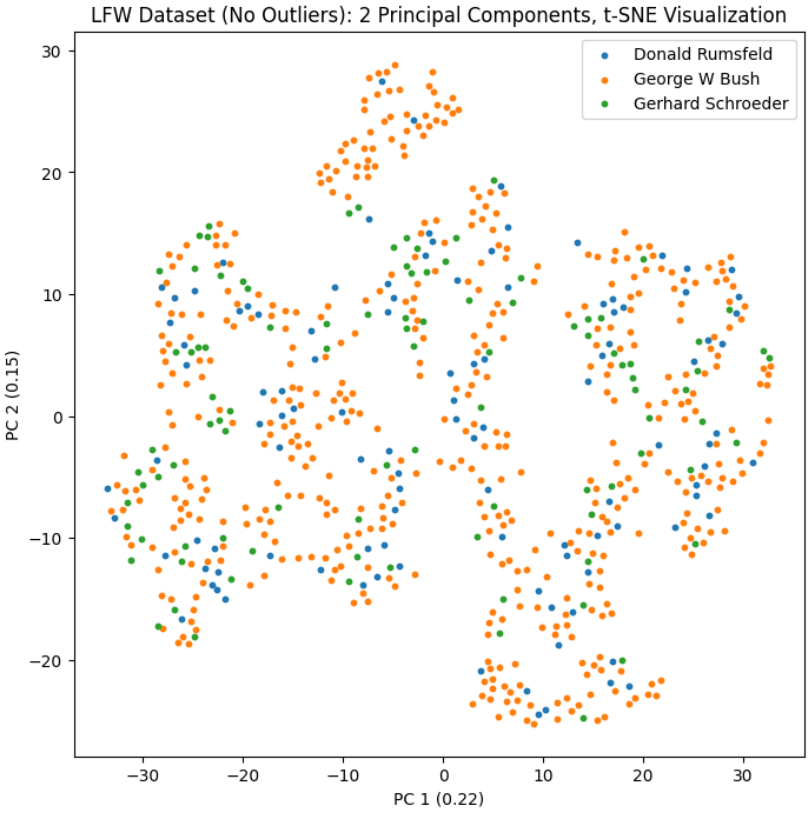

Figure 14: Dimensionality reduction

We will be using data from 1st Jan 2020. The dataset will contain closing prices of stocks from around 500 companies from the start date till today.

Lets try to visualize the daily returns and the cumulative daily returns of the stocks. It is apparent that the data is very cluttered and nothing meaningful can be interpreted from the plots. Hence we will apply PCA on this data and use the principle components to perform some analysis.

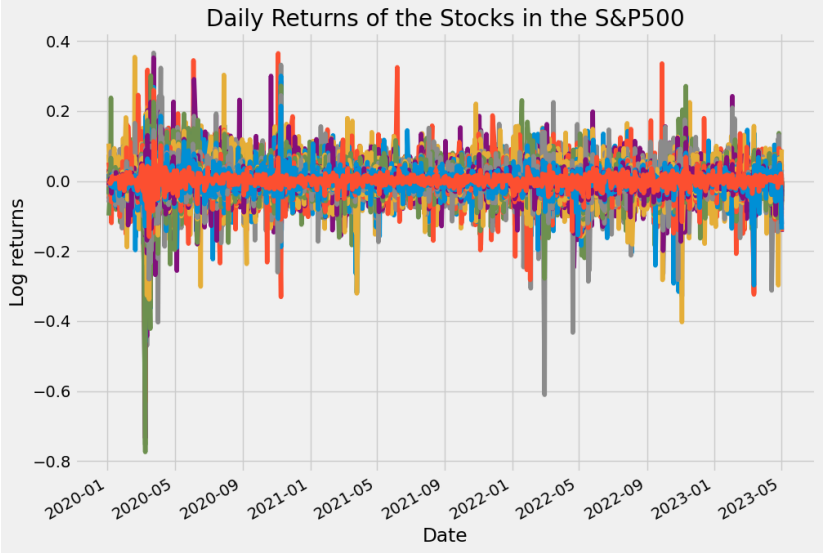

Figure 15: Log Daily returns

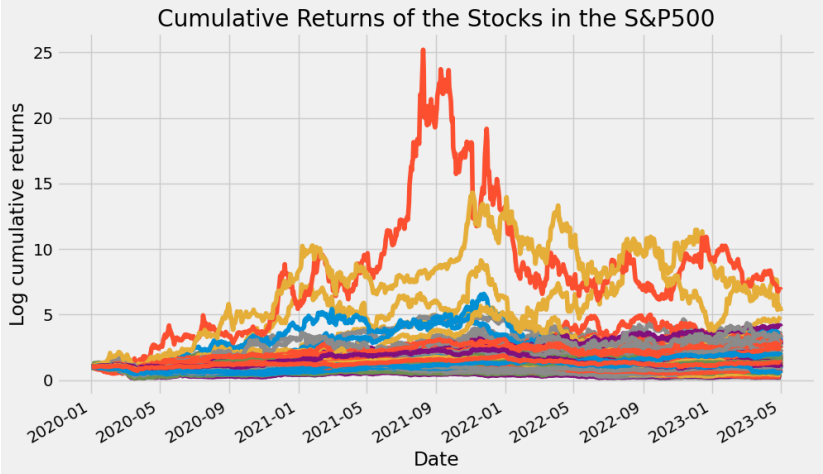

Figure 16: Cumulative Daily returns

4.2.1 Capital Asset Pricing Model (CAPM)

The Capital Asset Pricing Model (CAPM) is a financial model used to estimate the expected return on an investment, given its level of risk. It is based on the principle that investors require a higher return for taking on more risk, and that the expected return of an investment should be proportional to its risk level relative to the overall market.

The returns of a stock are decomposed into 3 main categories

1. Risk-free returns - these are the returns on an asset that is risk free (government bonds)
2. Market factor returns - the market factor monitors the state of the overall stock market as a whole and is often measured through an index such as the SP500.
3. Idiosyncratic returns - the returns specific to a stock and dependent on company factors.

4.2.2 Portfolio replication

The first principle components approximate the market factor. The code for finding and plotting the first principle components is given shown below.

```
from sklearn.decomposition import PCA

pca = PCA(1).fit(rs.fillna(0))

pc1 = pd.Series(index=rs.columns, data=pca.components_[0])

pc1.plot(figsize=(10,6), xticks=[], grid=True, title='First Principal Component of the S&P500')
plt.tight_layout()
```

Figure 17: Code for PCA

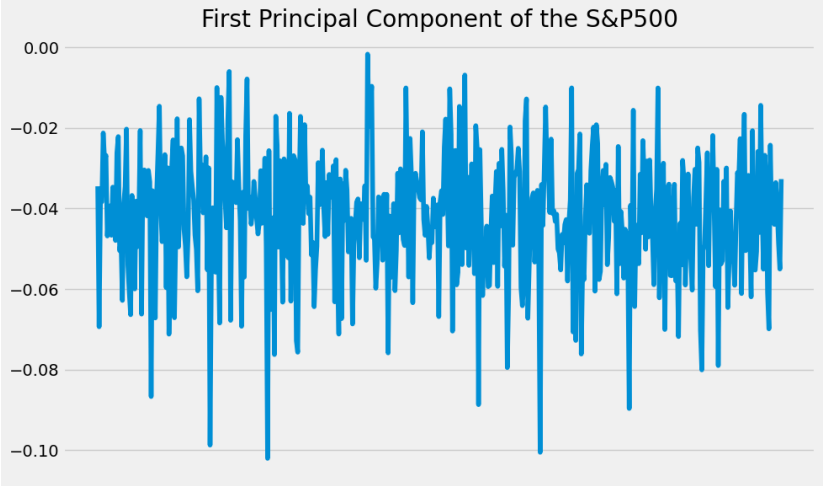

Figure 18: First principle components

If we formulate a portfolio of stocks by allocating the cash proportionally to the 1st principal component (i.e. linear combination of the input data), we can replicate the returns of the SP500 approximately (i.e. the primary driver of stock returns).

4.2.3 Analyzing impact of covid19 using PCA

let's look a the 1st principal component, and select the stocks that have the most and the least negative PCA weights.

As seen in image 22 the most negative stocks are in the tourism and the energy sector. This makes sense since COVID19 heavily impacted the travelling business, as

```
weights = abs(pc1)/sum(abs(pc1))
myrs = (weights*rs).sum(1)

rs_df = pd.concat([myrs, prices.apply(np.log).diff(1)], axis = 1)
rs_df.columns = ["PCA Portfolio", "S&P500"]

rs_df.dropna().cumsum().apply(np.exp).plot(subplots=True, figsize=(10,6), grid=True, linewidth=3)
plt.tight_layout()
plt.savefig('tmp.png')
✓ 0.9s
```

Figure 19: Code for portfolio replication

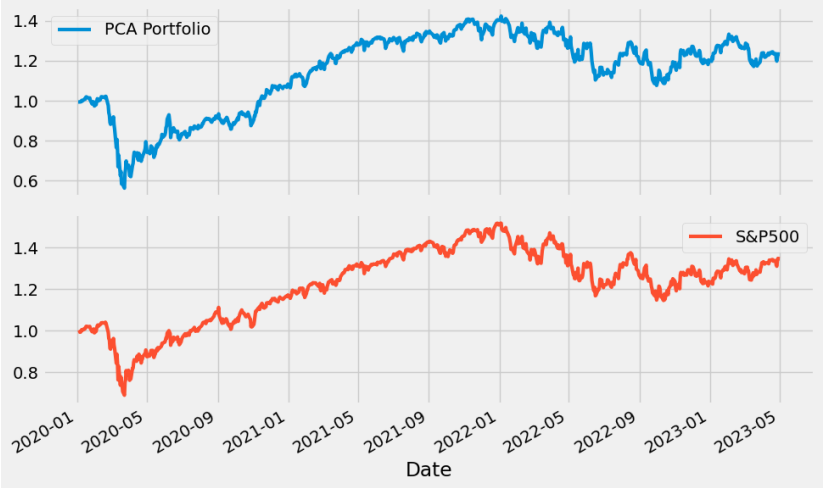

Figure 20: Portfolio replication

```
fig, ax = plt.subplots(2,1, figsize=(10,6))
pc1.nsmallest(10).plot.bar(ax=ax[0], color='green', grid=True, title='Stocks with Most Negative PCA Weights')
pc1.nlargest(10).plot.bar(ax=ax[1], color='blue', grid=True, title='Stocks with Least Negative PCA Weights')

plt.tight_layout()
```

Figure 21: Code for selecting PCA weights

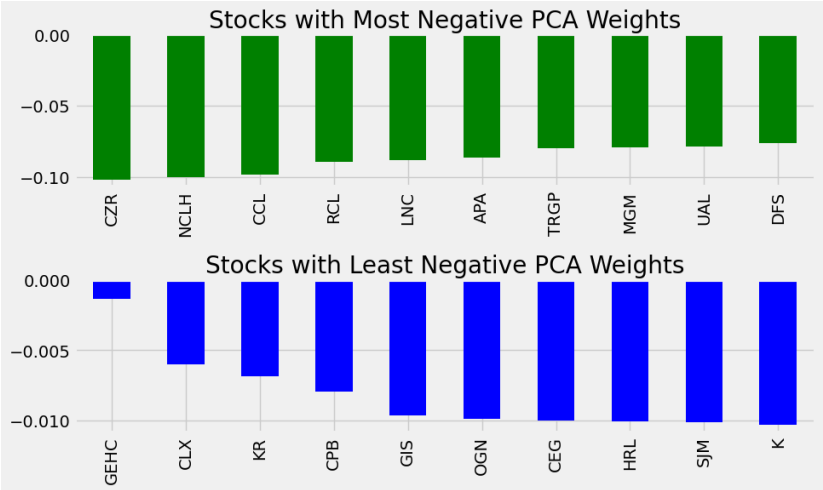

Figure 22: Analyzing stocks with PCA weights

well as the energy companies that provide fuel for those businesses. On the other hand, the least impacted companies fall into the consumer goods sectors, which also makes sense since this sector benefited from the boost in sales of consumer goods due to the quarantine measures.

5. Applications of PCA

1. Data Visualization and Dimensionality Reduction
2. Data compression
3. Pre-processing data
4. Feature Extraction

6. Advantages and Disadvantages of PCA

6.1 Advantages

1. Removes correlated features
2. Improves algorithm performance
3. Reduces overfitting
4. Improves visualization

6.2 Disadvantages

1. Independent variables become less interpretable
2. Data standardization is must before PCA
3. Information Loss

7. Conclusion

We looked at using Principle Component analysis to

1. Perform classification, visualize in 3D, Detect outliers, Data compression and Dimensionality reduction and Visualization
2. Perform stock market and portfolio analysis for SP500 dataset

We see that PCA is a very useful tool for analyzing large datasets containing a high number of dimensions/features per observation, increasing the interpretability of data

while preserving the maximum amount of information, and enabling the visualization of multidimensional data.

8. Code

Both the code and the images used in the report can be accessed at PCA on Large Datasets.

9. References

1. Wikipedia
2. LFW Dataset
3. Scree Plot
4. PCA Algorithm
5. PCA on LFW Dataset
6. SP500 dataset
7. sklearn.decomposition.PCA