CLASSIFIER COMPARISON FOR RECESSION PREDICTION USING IMBALANCED FINANCIAL DATA

A thesis proposal to the faculty of the Graduate School of

Western Carolina University in partial fulfillment of the requirements for the degree of Master of Science in Technology

By

Uzzal Basu

Director: Dr. Yanjun Yan

Professor

School of Engineering and Technology

Committee Members:

Dr. Paul Yanik, School of Engineering and Technology

Dr. Chaitanya Bora, School of Engineering and Technology

May 2025

ACKNOWLEDGEMENTS

I am deeply grateful for the patience and guidance of my advisor, Dr. Yanjun Yan, particularly during the early stages of my research work when I had limited knowledge in this field. Under her mentorship, I gained much knowledge about machine learning. Her constructive feedback helped me refine my ideas and improve my work. I express my sincere gratitude for her unwavering support which has strengthened my confidence as a researcher.

I would also like to extend my heartfelt gratitude to Dr. Paul Yanik, and Dr. Chaitanya Borra for their invaluable guidance and encouragement.

Furthermore, I am grateful for the financial support that Western Carolina University provided me through graduate and summer assistantships.

Lastly, I offer my warmest regards and thanks to all my family members for their continued support in pursuing my dreams.

TABLE OF CONTENTS

TABLE OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

RF	Random Forest
LR	Logistic Regression
DT	Decision Tree
SVM	Support Vector Machine
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
SMOTE	Synthetic Minority Oversampling TEchnique

ABSTRACT

CLASSIFIER COMPARISON FOR RECESSION PREDICTION USING IMBALANCED FINANCIAL DATA

Uzzal Basu, MSET

Western Carolina University (May 2025)

Advisor: Dr. Yanjun Yan

Recession predictions are essential for policymakers, businesses, and investors to prepare for economic downturns. However, economic datasets are often highly imbalanced with recessions being a less frequent event, making it challenging for machine learning (ML) models to make accurate predictions. This research explored various ways to use machine learning (ML) classifiers for recognizing financial recessions and forecasting them one quarter ahead, addressing the particular challenge of data imbalance. This study began with a raw dataset containing quarterly economic indicators, which were then extended into a monthly time-series dataset through interpolation methods to match the monthly recession data. Various machine learning algorithms such as Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), Support Vector Machines (SVM), and Multilayer Perceptron (MLP) were applied to recognition and forecasting financial recession using Monte Carlo simulations. This study systematically evaluated the performance of various classifiers by applying different data balancing techniques, including classic bootstrapping, weighted bootstrapping, and SMOTE (Synthetic Minority Oversampling TEchnique) to both original and interpolated datasets. Additionally, principal component analysis (PCA) was applied to explore its impact on classifier performance. This study not only evaluated the impact of different data balancing techniques but also analyzed various

classifiers' performance before and after applying PCA, providing a comprehensive evaluation of balancing methods and dimensionality reduction, offering valuable insights for economic forecasting and data-driven decision-making.

CHAPTER ONE: INTRODUCTION

In recent years, machine learning has made significant progress in enabling data-driven solutions to complex problems across various fields. From healthcare and finance to weather forecasting and transportation [1] machine learning algorithms are pioneering innovations. Before machine learning became popular, the finance industry relied on a “rule-based approach” to handle various financial problems. However, as online transactions increased and customer data volumes grew, machine learning started to attract more attention in the financial world. It has transformed the financial sector by providing powerful tools for analyzing complex data and improving decision-making processes. In stock market investments, machine learning algorithms are employed to forecast market trends [2], predict stock prices, and identify profitable trading opportunities. These algorithms analyze vast amounts of historical market data, including price movements, trading volumes, and macroeconomic indicators, to uncover patterns and correlations that inform future market behavior. Beyond investments, machine learning techniques are vital in early risk prediction [3] and control. Machine learning models help identify risky investments early, enabling academics and investors to make smarter decisions and maximize returns [4]. Additionally, machine learning plays a fundamental role in credit card fraud detection by analyzing transaction patterns to identify fraudulent activities [5]. The recession of a country depends on many economic indicators such as GDP, inflation rates, employment rates, income taxes, etc. Machine learning plays a crucial role in forecasting recessions by analyzing these indicators to accurately predict their occurrence. The world economy goes up and down over time, and financial recessions can cause big challenges for businesses, governments, and individuals. Therefore, accurately predicting recessions is essential for minimizing their impact and planning effective economic strategies. However, datasets used for recession prediction are often imbalanced, with the non-recession instances happening much more frequently than recession instances, making it challenging to develop reliable predictive models as traditional methods are struggling with the complexities of modern economics, particularly when challenged by scenarios with imbalanced data. Handling imbalanced datasets requires specialized techniques to ensure that the model does not become biased towards the majority class and can effectively learn from the minority class examples. By experimenting with various classifiers, including Random forest (RF), Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), and Multilayer Perceptron (MLP), this study aims to identify the most effective algorithms for predicting financial recessions of a country and to identify patterns that could signal impending recessions, allowing businesses, policymakers and financial institutions to make informed decisions.

CHAPTER TWO: LITERATURE REVIEW

Financial datasets often face the challenge of class imbalance, where events like recessions or economic crises are much less frequent than periods of stability. This imbalance can make it difficult for models to accurately predict the outcomes, as they tend to favor the majority class. To address this, researchers have used various techniques such as data balancing, feature selection, and advanced machine learning algorithms, etc. Many studies have focused on improving model performance, but some are limited to specific classifiers. For instance, S. R. R., L. Narayanaa T., and S. Jhansi Ida [3] developed a logistic regression-based model utilizing variables such as GDP, inflation, and employment rates, achieving 78% accuracy. This study focused on forecasting unemployment rates using logistic regression without exploring additional classifiers or dimensionality reduction techniques to assess the impact of the variables on unemployment rates. D. Jin [6] introduced the use of Fuzzy clustering algorithms, such as the C-means (FCM) algorithm, for managing high-dimensional time series data. Similarly, S. Sa'adah, T. H. Liong and Adiwijaya [7] developed a genetic algorithm to forecast economic crises in Indonesia using time series analysis on GDP and inflation data. It did not compare multiple classifiers, which could have provided a more robust evaluation of economic indicators. Some studies have applied advanced machine learning classifiers, to predict recessions. For example, N. Zyatkov and O. Krivorotko [8] conducted quantitative analyses of U.S. macroeconomic indicators to predict recessions over 6, 12, and 24 month horizons. Their research focused on nine variables, including GDP, and utilized fully connected neural networks providing the most accurate predictions with minimal errors. Similarly, N. Sh. Mehdiyev et al. [9] used recurrent neural networks (RNNs) to forecast recession indices using historical data, including the YCS and NYSE indexes. However, these studies did not use any balancing techniques to address the imbalanced datasets, which limited the robustness of their findings. Some studies have focused on handling imbalanced datasets using various methods. For example, D. Varmedja et al. [10] achieved a high accuracy of 99.96% using Random Forest (RF) with the Synthetic Minority Oversampling TEchnique (SMOTE) to handle highly imbalanced datasets. The study also analyzed 31 input variables and applied feature exploration. Similarly, S. O. Moepya, S. S. Akhoury, and F. V. Nelwamondo [11] conducted their study using data from the South African market. They applied feature exploration techniques such as Principal Component Analysis (PCA) and Factor Analysis (FA) to identify key features, concluding that eight principal components explained 90% of the dataset's variance. Their findings showed that Support Vector Machines (SVM) outperformed Naive Bayes (NB) and K-Nearest Neighbors (KNN) in cost-sensitive classifications, achieving satisfactory sensitivity at cost ratios between 10:1 and 30:1. However, the study requires more classifiers due to discarded features to avoid missing values. Dennis et al. [12] addressed the challenges of highly imbalanced datasets by exploring various resampling techniques, including SMOTE, K-Means SMOTE, and Borderline SMOTE for oversampling, as well as Near-Miss and All-KNN for undersampling. They applied PCA, identifying Logistic Regression (LR) as the best performing classifier in their study. Their findings showed that oversampling generally performed better than undersampling, though each method required manual adjustments for optimal results. Additionally, the undersampling techniques involved removing some instances of the majority class, which could lead to the loss of valuable data and potentially mislead the algorithm.

While previous studies have made valuable contributions to recession forecasting and financial analysis, many relied on a limited set of variables, or single classifiers, or lacked effective techniques for addressing highly imbalanced datasets. In contrast, our research focused on two primary tasks: recognition and forecasting. We studied a highly imbalanced dataset where a recession constitutes only 14.76% of the total dataset. To address this, we implemented a wide range of machine learning classifiers, including RF, LR, DT, SVM, and MLP, combined with various data balancing techniques such as SMOTE, unweighted bootstrapping, and weighted bootstrapping. These methods ensured a more balanced class distribution for training in the machine learning models, leading to improved reliability and accuracy. Additionally, PCA was employed to enhance feature relevance and optimize model performance. Furthermore, this study not only evaluated the impact of different data balancing techniques on the dataset but also analyzed their performance after applying PCA, providing a comprehensive evaluation of balancing methods and dimensionality reduction.

CHAPTER THREE: METHODOLOGY

3.1 Two tasks: Recognition and Forecasting

This study focused on two primary tasks: recognition and forecasting of financial recessions. Recognition involves identifying patterns in economic data to classify whether a given period corresponds to a recession or not. Forecasting, on the other hand, aims to predict future recessions one quarter ahead of time based on economic indicators. For each instance in the training data, the recession status for the next quarter, along with current and past economic data from the previous year, was used as a single sample for training the forecasting models.

3.2 Data Pre-Processing

Data pre-processing involves cleaning and preparing the raw data for analysis. The original financial dataset had 18 variables from publicly available sources from the Federal Reserve Bank of St. Louis, as shown in Table 3.1. The shorthand notations of these variables are listed in the first column, while their meanings and definitions are explained in the second column.

3.2.1 Original Dataset

In the original dataset shown in Figure 3.1, we gathered quarterly data from 1967 to 2019. It contained 210 rows and 18 different variables capturing a wide range of information. We considered 16 specific variables as our inputs, which are closely related to the recession of a country. The output variable is the 'Recession' variable. Note that we chose not to include the 'Date' variable as an input as it was not an economic variable, but it was used to ensure proper data alignment.

Table 3.1: Variables used in the dataset

List of variables	Description
GDPC1	Real Gross Domestic Product, Billions of Chained 2012 Dollars, Quarterly, Seasonally Adjusted Annual Rate
A939RX0Q048S BEA	Real Gross Domestic Product Per Capita, Chained 2012 Dollars, Quarterly, Seasonally Adjusted Annual Rate.
GDP	Gross Domestic Product, Billions of Dollars, Quarterly, Seasonally Adjusted Annual Rate.
GDPPC	Gross Domestic Product Per Capita, Dollars, Quarterly, Seasonally Adjusted Annual Rate.
CAPEXP	Capital Expenditures Domestic Nonfinancial Sectors, Millions of Dollars, Quarterly, Not Seasonally Adjusted.
EMP	All Employees, Total Nonfarm, Thousands of Persons, Quarterly, Seasonally Adjusted.
INCTAX	U.S Individual Income Tax: Tax Rates for Regular Tax: Highest Bracket, Percent, Annual, Not Seasonally Adjusted.
RGOVINV	Real Government Consumption Expenditures and Gross Investment, Billions of Chained 2012 Dollars, Quarterly, Seasonally Adjusted Annual Rate.
TOT	Gross Domestic Product: Terms of Trade Index, Index, Quarterly, Seasonally Adjusted.
RINV	Real Gross Private Domestic Investment, Billions of Chained 2012 Dollars, Quarterly, Seasonally Adjusted Annual Rate.
RPCE	Real Personal Consumption Expenditures, Billions of Chained 2012 Dollars, Quarterly, Seasonally Adjusted Annual Rate.
LABCOST	Business Sector: Unit Labor Costs, Percent Change From Quarter One Year Ago, Quarterly, Seasonally Adjusted.
SAVING	Personal Saving as a Percentage of Disposable Personal Income, Percent, Quarterly, Seasonally Adjusted Annual Rate.
CPI	Consumer Price Index for All Urban Consumers: All Items, Index 1982-1984=100, Quarterly, Seasonally Adjusted.
3MOINT	3-Month Treasury Bill: Secondary Market Rate, Percent, Quarterly, Not Seasonally Adjusted.
UNEMPR	Civilian Unemployment Rate, Percent, Quarterly, Seasonally Adjusted.
Recession	Output Variable of a Country.

The original dataset organized data quarterly within each year, with specific dates representing each quarterly period. Entries were recorded for the first quarter of each year (starting on January 1st), the second quarter (April 1st), the third quarter (July 1st), and the fourth quarter (October 1st). Notably, the “Recession” was available for every year spanning from 1967 to 2019.

1	Date	GDPC1	39RX0Q048SE	GDP	GDPPC	CAPEXP	EMP	INCTAX	RGOVINV
2	1967-01-01	4535.591	22911	844.170	4264	64403	65455	70.000	1543.706
3	1967-04-01	4538.370	22869	848.983	4278	65229	65612	70.000	1537.300
4	1967-07-01	4581.309	23020	865.233	4348	67607	66065	70.000	1554.384
5	1967-10-01	4615.853	23129	881.439	4417	68005	66609	70.000	1566.194
6	1968-01-01	4709.993	23551	909.387	4547	69209	67105	75.250	1594.198
7	1968-04-01	4788.688	23889	934.344	4661	73187	67704	75.250	1602.467
8	1968-07-01	4825.799	24009	950.825	4731	73932	68313	75.250	1607.267
9	1968-10-01	4844.779	24039	968.030	4803	74095	68984	75.250	1608.726
10	1969-01-01	4920.605	24365	993.337	4919	76795	69681	77.000	1612.372
11	1969-04-01	4935.564	24383	1009.020	4985	78067	70345	77.000	1607.344
12	1969-07-01	4968.164	24475	1029.956	5074	79851	70884	77.000	1610.604
13	1969-10-01	4943.935	24284	1038.147	5099	76672	71149	77.000	1588.938
14	1970-01-01	4936.594	24189	1051.200	5151	75120	71311	71.750	1581.297
15	1970-04-01	4943.600	24148	1067.375	5214	77639	71167	71.750	1562.913
16	1970-07-01	4989.159	24288	1086.059	5287	79338	70978	71.750	1569.519
17	1970-10-01	4935.693	23945	1088.608	5281	75562	70574	71.750	1570.250
18	1971-01-01	5069.746	24520	1135.156	5490	81917	70844	70.000	1547.285
19	1971-04-01	5097.179	24581	1156.271	5576	85885	71179	70.000	1543.730

Figure 3.1: Original Dataset

Figure 3.2 provides a comprehensive overview of the trends of the 16 inputs economic variables alongside the recession variable. The data, visualized in dB scale over time, highlights patterns and fluctuations spanning the period from 1967 to 2019. The red dots in the figure indicate the recession events reflected in the economic variables during specific years.

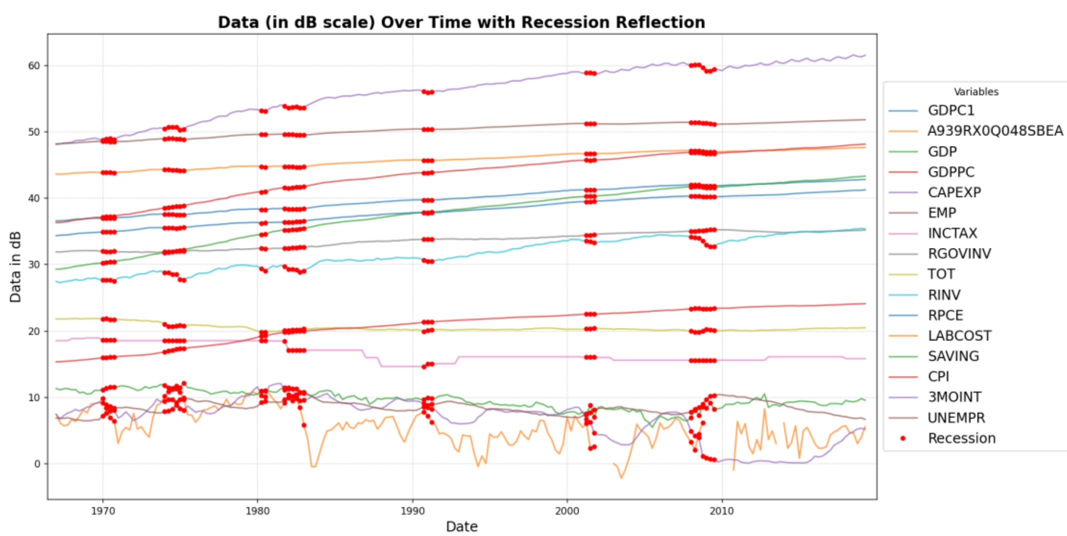

Figure 3.2: Data (in dB scale) over time. The red dots on the data lines indicate the occurrence of recession.

3.2.2 Interpolated Dataset

Interpolation is a mathematical technique used to estimate unknown values that lie between known data points. As shown in Figure 3.3 [13] it involves creating a function or curve that passes through the known data points and then using this function to predict the values of points within the range of the known data. There are various methods of interpolation, including linear interpolation, polynomial interpolation, spline interpolation, and nearest-neighbor interpolation.

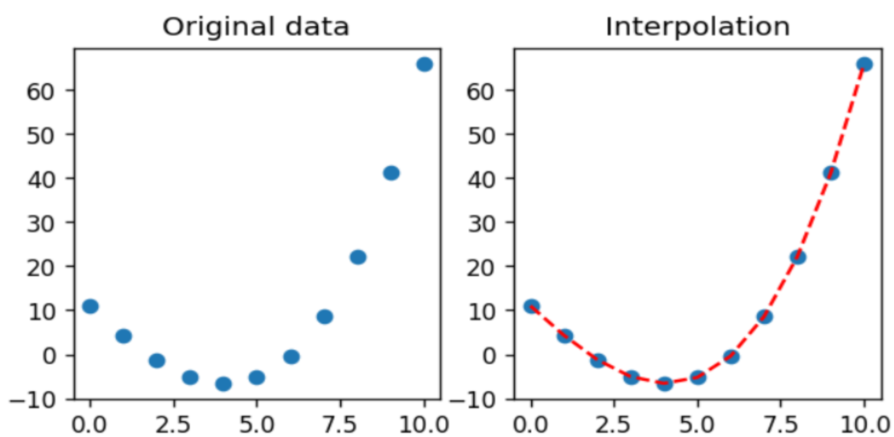

Figure 3.3: Developing interpolation data from original data

The original dataset is the raw dataset consisting of 210 rows of quarterly economic indicators spanning from 1967 to 2019. Since the recession data was available monthly, we applied linear interpolation to the economic indicators to align them with the monthly recession data and enlarge the dataset. As shown in Figure 3.4, after the interpolation of the original dataset, we expanded the data to include all months from January 1967 to December 2019. As a result, the dataset now comprises 628 rows with entries for every month over the entire period.

1	Date	GDPC1	RX0Q048S	GDP	GDPPC	CAPEXP	EMP	INCTAX	RGOVINV
2	1/1/1967	4535.591	22911	844.17	4264	64403	65455.33	70	1543.71
3	2/1/1967	4536.517	22897	845.774	4268.7	64678.3	65507.56	70	1541.57
4	3/1/1967	4537.444	22883	847.379	4273.3	64953.7	65559.78	70	1539.44
5	4/1/1967	4538.37	22869	848.983	4278	65229	65612	70	1537.3
6	5/1/1967	4552.683	22919.3	854.4	4301.3	66021.7	65763	70	1542.99
7	6/1/1967	4566.996	22969.7	859.816	4324.7	66814.3	65914	70	1548.69
8	7/1/1967	4581.309	23020	865.233	4348	67607	66065	70	1554.38
9	8/1/1967	4592.824	23056.3	870.635	4371	67739.7	66246.44	70	1558.32
10	9/1/1967	4604.338	23092.7	876.037	4394	67872.3	66427.89	70	1562.26
11	10/1/1967	4615.853	23129	881.439	4417	68005	66609.33	70	1566.19
12	11/1/1967	4647.233	23269.7	890.755	4460.3	68406.3	66774.44	71.75	1575.53
13	12/1/1967	4678.613	23410.3	900.071	4503.7	68807.7	66939.56	73.5	1584.86
14	1/1/1968	4709.993	23551	909.387	4547	69209	67104.67	75.25	1594.2
15	2/1/1968	4736.225	23663.7	917.706	4585	70535	67304.56	75.25	1596.95
16	3/1/1968	4762.456	23776.3	926.025	4623	71861	67504.44	75.25	1599.71
17	4/1/1968	4788.688	23889	934.344	4661	73187	67704.33	75.25	1602.47
18	5/1/1968	4801.058	23929	939.838	4684.3	73435.3	67907.33	75.25	1604.07
19	6/1/1968	4813.429	23969	945.331	4707.7	73683.7	68110.33	75.25	1605.67

Figure 3.4: Interpolated Dataset

3.3 Data scaling and normalization

In the original dataset, we observed notable range differences in the numerical values of different variables. Some of the variables had extremely high values, while others had extremely low values. To ensure the optimal performance of our machine learning models, we normalized all the variables to have a zero mean and a standard deviation of 1 as shown in Figure 3.5 Additionally, the interpolated dataset was also normalized in the same way. Figure 3.6 illustrates a plot depicting the GDPC1 variable over time before scaling, with the starting year, 1967, as an example for interpolation. Each circular dot on this plot represents the original data point, while each cross symbolizes the interpolated data point. Figure 3.7 illustrated a plot depicting the GDPC1 variable over time after scaling starting from 1967 to 2019.

1	Date	GDPC1	39RX0Q048SE	GDP	GDPPC	CAPEXP	EMP	INCTAX	RGOVINV
2	1967-01-01	4535.591	22911	844.170	4264	64403	65455	70.000	1543.706
3	1967-04-01	4538.370	22869	848.983	4278	65229	65612	70.000	1537.300
4	1967-07-01	4581.309	23020	865.233	4348	67607	66065	70.000	1554.384
5	1967-10-01	4615.853	23129	881.439	4417	68005	66609	70.000	1566.194
6	1968-01-01	4709.993	23551	909.387	4547	69209	67105	75.250	1594.198
7	1968-04-01	4788.688	23889	934.344	4661	73187	67704	75.250	1602.467
8	1968-07-01	4825.799	24009	950.825	4731	73932	68313	75.250	1607.267
9	1968-10-01	4844.779	24039	968.030	4803	74095	68984	75.250	1608.726
10	1969-01-01	4920.605	24365	993.337	4919	76795	69681	77.000	1612.372
11	1969-04-01	4935.564	24383	1009.020	4985	78067	70345	77.000	1607.344
12	1969-07-01	4968.164	24475	1029.956	5074	79851	70884	77.000	1610.604
13	1969-10-01	4943.935	24284	1038.147	5099	76672	71149	77.000	1588.938
14	1970-01-01	4936.594	24189	1051.200	5151	75120	71311	71.750	1581.297
15	1970-04-01	4943.600	24148	1067.375	5214	77639	71167	71.750	1562.913
16	1970-07-01	4989.159	24288	1086.059	5287	79338	70978	71.750	1569.519
17	1970-10-01	4935.693	23945	1088.608	5281	75562	70574	71.750	1570.250
18	1971-01-01	5069.746	24520	1135.156	5490	81917	70844	70.000	1547.285
19	1971-04-01	5097.179	24581	1156.271	5576	85885	71179	70.000	1543.730

1	Date	GDPC1	A939RX0Q	GDP	GDPPC	CAPEXP	EMP	INCTAX	RGOVINV
2	1/1/1967	-1.41998	-1.56257	-1.23026	-1.36601	-1.28334	-1.79028	1.441442	-1.35605
3	4/1/1967	-1.41934	-1.56655	-1.22945	-1.36522	-1.28123	-1.78402	1.441442	-1.3664
4	7/1/1967	-1.40952	-1.55224	-1.22675	-1.36131	-1.27515	-1.76594	1.441442	-1.33878
5	10/1/1967	-1.40162	-1.5419	-1.22405	-1.35745	-1.27413	-1.74422	1.441442	-1.31969
6	1/1/1968	-1.38008	-1.50191	-1.21939	-1.35018	-1.27105	-1.72445	1.778313	-1.27441
7	4/1/1968	-1.36208	-1.46987	-1.21523	-1.3438	-1.26089	-1.70052	1.778313	-1.26104
8	7/1/1968	-1.35359	-1.45849	-1.21249	-1.33989	-1.25898	-1.67621	1.778313	-1.25328
9	10/1/1968	-1.34925	-1.45565	-1.20962	-1.33586	-1.25857	-1.64943	1.778313	-1.25092
10	1/1/1969	-1.33191	-1.42475	-1.20541	-1.32937	-1.25167	-1.62163	1.890604	-1.24503
11	4/1/1969	-1.32848	-1.42305	-1.20279	-1.32568	-1.24841	-1.59511	1.890604	-1.25315
12	7/1/1969	-1.32103	-1.41432	-1.19931	-1.32071	-1.24385	-1.57363	1.890604	-1.24788
13	10/1/1969	-1.32657	-1.43243	-1.19794	-1.31931	-1.25198	-1.56302	1.890604	-1.28291
14	1/1/1970	-1.32825	-1.44143	-1.19577	-1.3164	-1.25595	-1.55658	1.553732	-1.29527
15	4/1/1970	-1.32665	-1.44532	-1.19307	-1.31288	-1.24951	-1.56232	1.553732	-1.32499
16	7/1/1970	-1.31622	-1.43205	-1.18996	-1.30879	-1.24517	-1.56986	1.553732	-1.31431
17	10/1/1970	-1.32845	-1.46456	-1.18954	-1.30913	-1.25482	-1.58599	1.553732	-1.31313
18	1/1/1971	-1.29779	-1.41006	-1.18178	-1.29744	-1.23857	-1.57521	1.441442	-1.35026
19	4/1/1971	-1.29152	-1.40428	-1.17826	-1.29263	-1.22843	-1.56184	1.441442	-1.35601

Figure 3.5: Normalization before and after for the original data

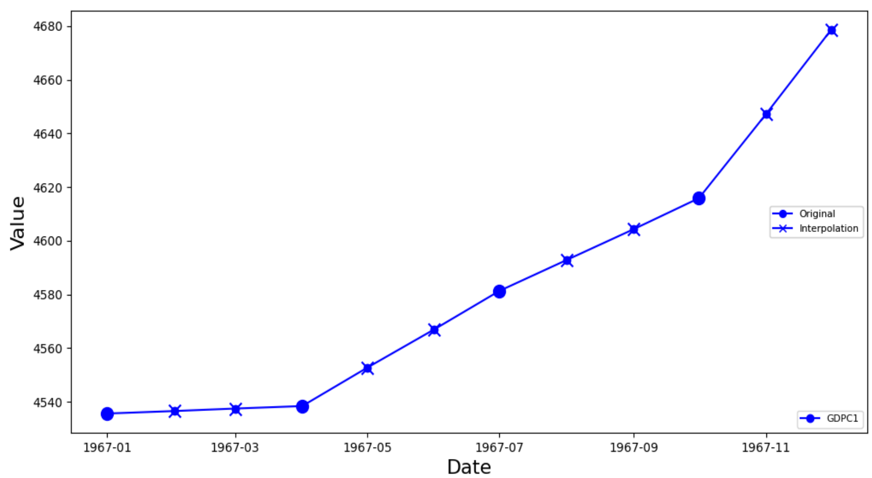

Figure 3.6: Plot for one example variable over time before scaling

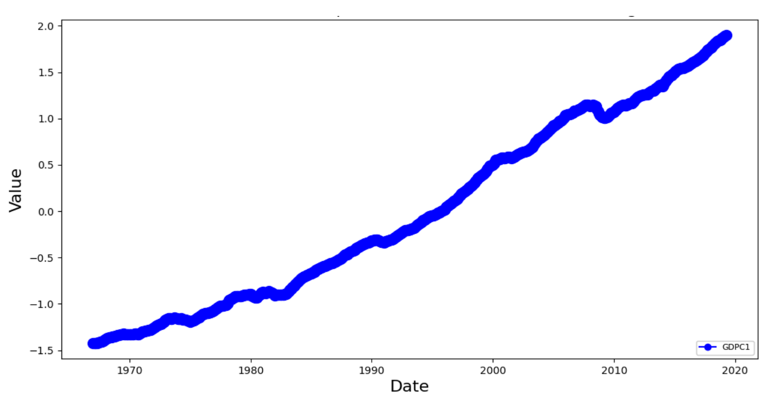

Figure 3.7: Plot for one example variable over time after scaling

3.4 Feature Exploration:

Figure 3.8 is a widely used dimensionality reduction technique in machine learning and data analysis. It aims to transform a high-dimensional dataset into a lower dimensional one while retaining the most significant information [14]. These principal components are linear combinations of the original variables representing the axes along which the data varies most.

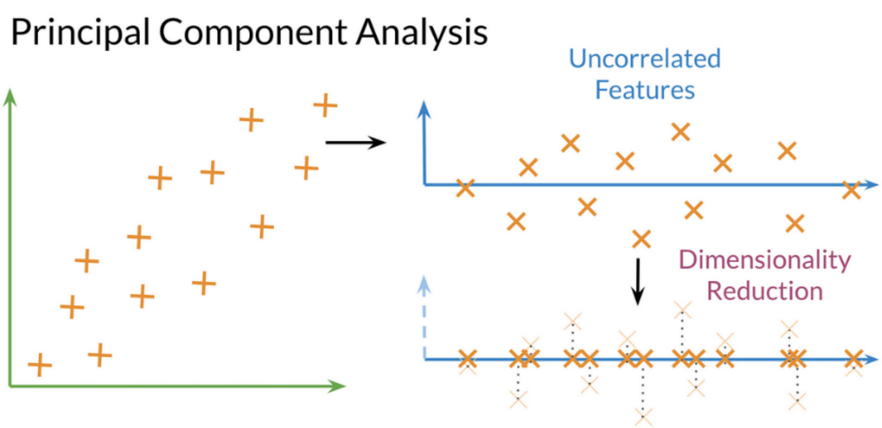

Figure 3.8: Principal component analysis techniques (PCA)

By reducing dimensionality, PCA simplifies the data structure making it easier to visualize and analyze. It also helps to avoid overfitting machine learning models by focusing on the most informative variables and removing noise or redundant information. Principal component analysis (PCA) was applied to both the original and interpolated datasets as a feature extraction technique to reduce dimensionality. As illustrated in Figure 3.7, the explained variance ratio (bars) and cumulative explained variance (line) show that the first few principal components capture the majority of the variance in the data.

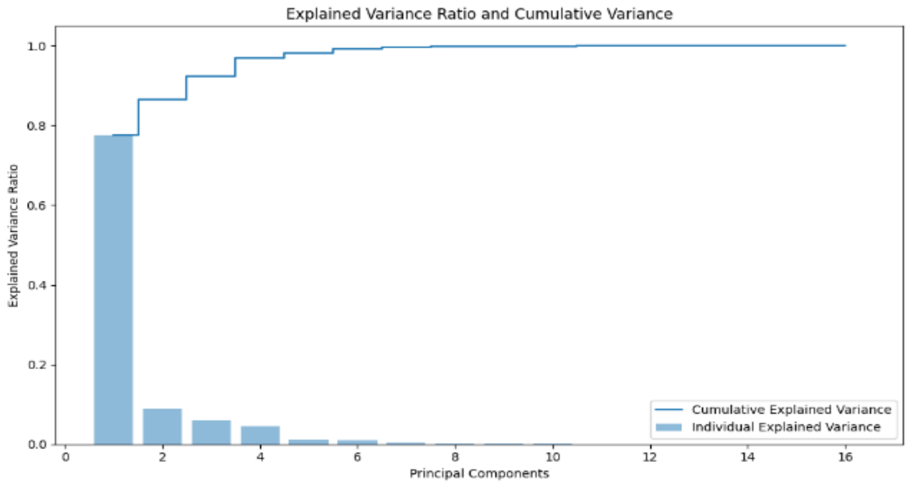

Figure 3.7: Explained variance ratio of PCA on the original data

From the given result, PC1 explains a significant portion of the variance, contributing approximately 77.66% of the total variance in the original dataset. The PC2 contributes an additional 8.79%, while the PC3 and PC4 components add 5.92% and 4.62%, respectively. Beyond PC4, the additional component contributes only a minimal amount of total variance. After applying PCA with the original and interpolated dataset for recognition and forecasting tasks, the number of variables is reduced from 16 to 4. As shown in Figure 3.8, the newly transformed dataset consists of four principal components (PC1 through PC4), derived from the original dataset 16 dimensions after normalization.

PC1	PC2	PC3	PC4	Recession
-5.16511	-2.45443	0.689752	-0.11393	0
-4.9833	-2.54956	1.07812	-0.39954	0
-5.00803	-2.51663	1.063276	-0.41825	0
-5.03524	-2.46654	1.013827	-0.37532	0
-5.07725	-2.57993	0.723588	-0.22691	0
-5.07557	-2.70026	0.694871	-0.29315	0
-4.94575	-2.62827	0.442281	-0.37194	0
-5.04747	-2.63257	0.113705	-0.17098	0
-4.97783	-2.69439	0.131324	-0.34686	0
-5.10915	-2.61715	-0.29622	0.033227	0
-5.27501	-2.53992	-0.32575	0.27541	0
-5.31444	-2.48818	-0.4277	0.268056	0
-5.33852	-2.07694	-0.52234	0.53832	1
-5.21541	-1.9702	0.342259	0.341519	1
-5.08065	-1.71609	0.848075	0.224323	1
-5.01597	-1.47841	1.281714	0.318962	1

Figure 3.8. Applied PCA on the original dataset

3.5 Handle Imbalanced Data

When working with machine learning, having an imbalanced dataset can be a big challenge. This happens when one class has way more data than the other, making it harder for the model to learn from the smaller class. As a result, the model may struggle to predict the minority class correctly, leading to poor performance. To solve this problem, different techniques can be used to balance the data. This section will explain three methods: Unweighted Bootstrapping, and Weighted Bootstrapping and SMOTE (Synthetic Minority Over-sampling TEchnique). These techniques help adjust the data so that machine learning models can better recognize patterns in both the majority and minority classes, leading to more accurate predictions.

3.5.1 Bootstrapping with unchanged class weights

Bootstrapping is a resampling technique used in statistics to estimate the sampling distribution of a statistic by repeatedly sampling from the original dataset with replacement. As depicted in Figure 3.11 [15]. This method involves randomly selecting samples from the original dataset with replacements to create multiple new datasets of the same size as the original.

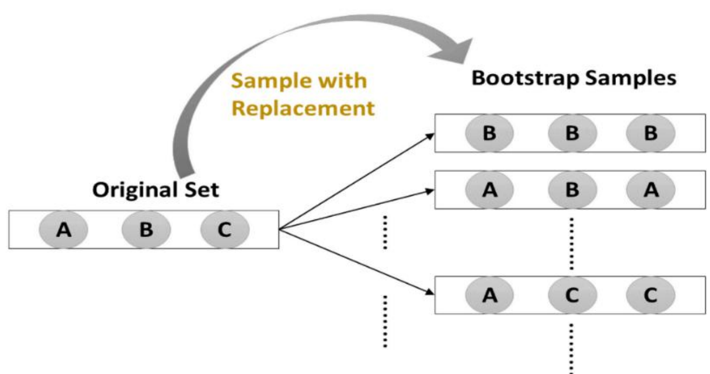

Figure 3.11: Bootstrapping data generation

By generating multiple bootstrap samples and calculating the statistic of interest of each sample, one can obtain an empirical estimate of the sampling distribution of the statistic without making any assumptions. Bootstrapping is commonly used to estimate the standard error, confidence intervals, and other statistical properties of a sample statistic, particularly when the result is unknown or irregular. For both tasks, the bootstrapping technique was employed to address the issue of imbalanced data in the original and interpolated datasets. This method involves randomly selecting samples from the dataset with replacements to create a new dataset of the same size. For instance, for the recognition task, the interpolated dataset initially contained 628 rows, with recession and non-recession proportions of 14.49% and 85.51%, respectively. After applying the bootstrapping technique, the dataset remained at 628 rows, but the recession proportion varied between 11.46% to 18.47%, while non-recession proportion ranged from 88.54% to 81.53% across 100 bootstrapped datasets. In Monte Carlo simulations, each bootstrapped dataset is randomly generated. To examine the effect of a larger training dataset, we also generated datasets twice the length of the original and interpolated datasets, resulting in 420 rows for the original and 1256 rows for the interpolated datasets. The weights of the two classes were still unchanged. Additionally, this technique was applied to PCA-transformed datasets to effectively handle the imbalanced data.

3.5.2 Weighted Bootstrapping

In the context of bootstrapping data generation, adjusting the weight refers to modifying the distribution of certain categories within the dataset. By assigning different weights to specific categories, such as recession and non-recession data, we can control their relative frequencies in the resampled dataset. For instance, if we increase the weight assigned to recession data, it means that more instances of recession will be included in the resampled dataset compared to non-recession instances. Conversely, reducing the weight of a category decreases its representation in the resampled dataset. This manipulation allows us to rebalance the data set according to our specific objectives. Previously, our approach involved uncontrolled data generation through bootstrapping on the original and interpolated datasets. To address the challenge of imbalanced class distributions (recession and non-recession), we assigned a weight of 30% to the recession data during bootstrapping for both datasets. This adjustment ensured that the dataset generated contained 30% recession data and 70% non-recession data. For instance, in forecasting tasks using the original data, the double-length bootstrapped datasets initially had proportions of 14.81% recession and 85.19% non-recession.

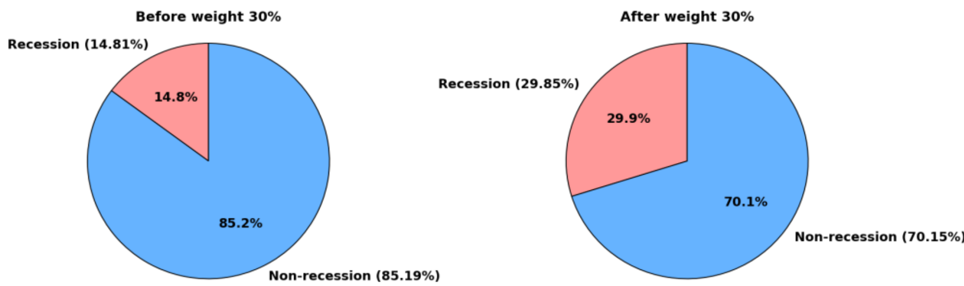

Figure 3.12: Class distribution before and after bootstrapping with 30% weight of the minority class

After applying weighted bootstrapping at 30%, the proportion of recession data increased to 29.85%, while non recession data decreased to 70.15% as shown in Figure 3.12. Similarly, a weight of 50% was assigned to the recession data during bootstrapping, resulting in datasets with equal class distribution: 50% recession and 50% non-recession for recognition and forecasting tasks across both datasets. Additionally, this technique was also applied to PCA transformed datasets to effectively handle imbalanced data.

3.5.3 SMOTE (Synthetic Minority Over-Sampling TEchnique)

Synthetic Minority Oversampling TEchnique (SMOTE) addresses the challenge of class imbalance by creating new synthetic examples with similar distributions as shown in Figure 3.9, for the minority class. Instead of simply duplicating existing examples, SMOTE creates new ones by combining similar examples [16]. This helps prevent overfitting, which can happen if the same examples are repeated too many times.

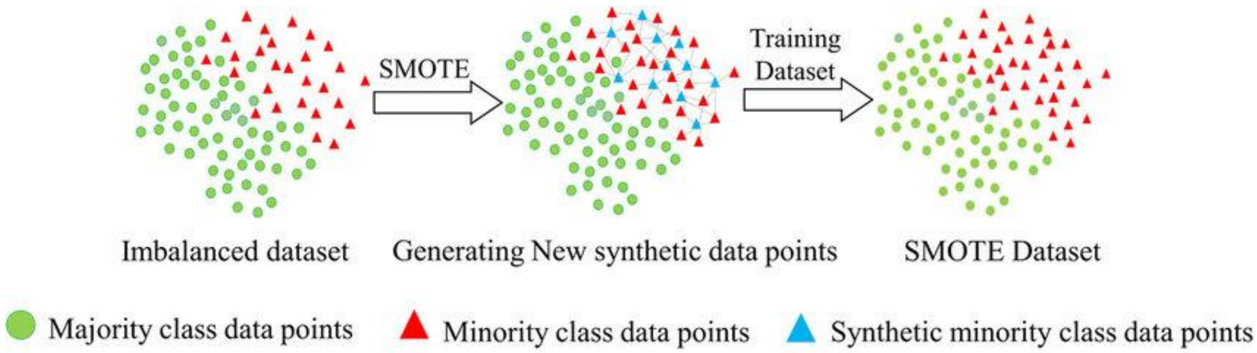

Figure 3.9: Illustration of SMOTE Technique for Balancing Imbalanced Datasets

SMOTE generates synthetic samples for the minority class, balancing the dataset and enabling machine learning models to learn from both classes more effectively. This technique was applied to both the original and interpolated datasets for recognition and forecasting tasks. For example, in the recognition task, the original dataset comprised 210 rows, with the recession class accounting for 14.76% and the non-recession class for 85.24%.

Figure 3.10: Class distribution before and after SMOTE technique

After applying SMOTE, synthetic samples were generated for the minority class (recession), resulting in a balanced dataset with 358 rows, where both recession and non-recession classes constituted 50%, as shown in Figure 3.10 Additionally, SMOTE was applied to PCA-transformed datasets.

CHAPTER FOUR: MODEL DEVELOPMENT AND EVALUATION PROCESS

4.1 Data Splitting

Figure 4.1 shows the general process followed in our research. The process begins with a dataset that goes through various data pre-processing techniques to prepare it for model training and testing. In this study, data was divided into training and testing sets across all datasets, including the original, interpolated, PCA-transformed, and those created using balancing techniques such as SMOTE, unweighted bootstrapping, and weighted bootstrapping.

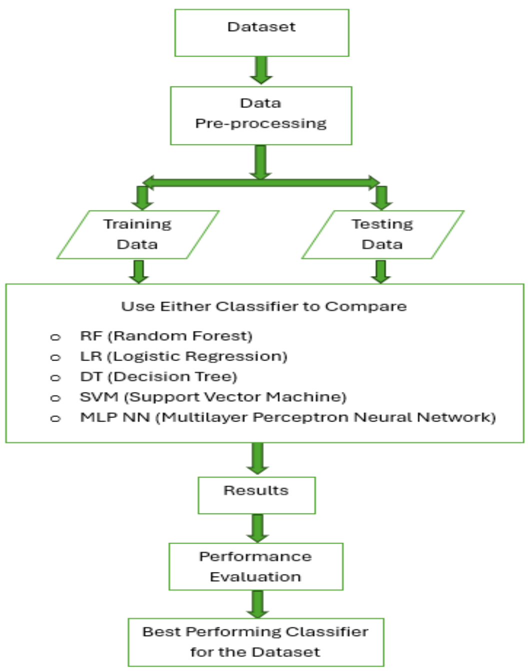

Figure 4.2: Schematic diagram of showing the steps of research

For each dataset, 70% of the data was used for training, while 30% was reserved for testing. For example, the original dataset with 210 rows was split into 147 rows for training and 63 rows for testing. This random split was repeated in each Monte Carlo simulation to ensure the models were properly evaluated and could effectively predict economic recessions. Different machine learning models were used in this study, including Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), Support Vector Machine (SVM), and Multilayer Perceptron Neural Network (MLP NN). The results from these models were then evaluated using accuracy, F1 score, and other important measures. In the end, the best-performing model was selected based on these results.

This figure shows 4.2, the different data balancing techniques before PCA used to improve model performance on the original and interpolated datasets. Three methods were applied: unweighted bootstrapping, weighted bootstrapping and SMOTE. Unweighted bootstrapping kept the original class distribution while increasing data size by creating datasets with the same length as the original and double the length.

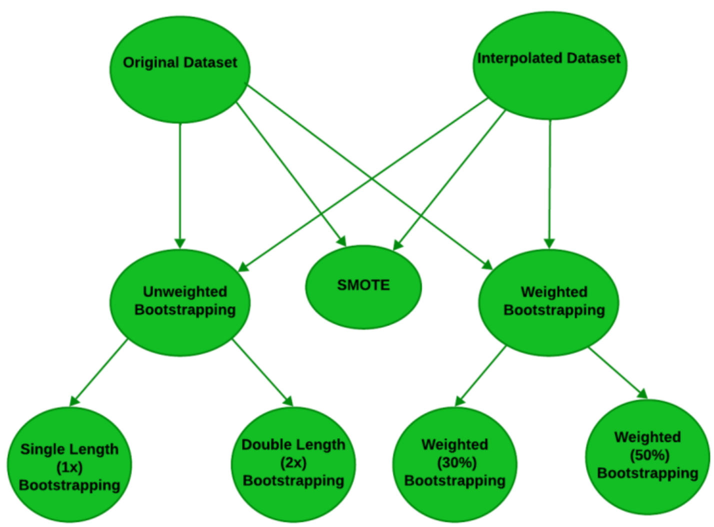

Figure 4.2: Schematic diagram of showing the steps of data balancing techniques before PCA

Weighted bootstrapping adjusted the data by giving more importance to recession cases, using 30% and 50% weighting. SMOTE (Synthetic Minority Oversampling TEchnique) added new synthetic data points to improve the representation of recession cases. These techniques were applied to help the models learn better from both datasets, improving their accuracy and overall performance in recognizing and predicting economic changes. After applying PCA (Principal Component Analysis) to the original and interpolated datasets, we generated PCA-transformed datasets. We then applied the same data balancing techniques to these PCA-transformed datasets as shown in Figure 4.3 to see how PCA-transformed data responds to different balancing techniques and how it affects model performance.

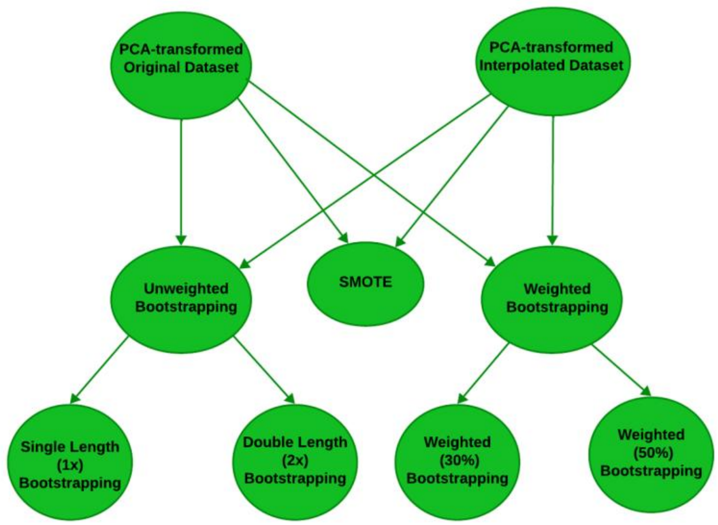

Figure 4.3: Schematic diagram of showing the steps of data balancing techniques after PCA

4.2 Algorithm Selection

In this research we explored a variety of machine learning algorithms to identify the most suitable model to predict recession. The algorithms include Support Vector Machines (SVM) [17], Logistic Regression (LR) [2], Decision Tree (DT) [13], Random Forest (RF) [5], and Multilayer Perceptron Neural Network (MLP NN) [5]. By comparing these models, we aimed to identify the most effective approach for accurate recession prediction. During the training process, the model parameters were adapted to minimize the difference between the predicted outputs and the actual outputs in the training dataset.

4.2.1 Random Forest (RF) Classifier

Random Forest is a popular ensemble learning method used for both classification and regression tasks. It combines the predictions of multiple individual decision trees to produce a more accurate and robust final prediction. It consists of a collection of decision trees, each trained on a random subset of the training data and using a random subset of the variables. During prediction, the individual tree predictions [18] are averaged or majority-voted to produce the final output as shown in Figure 4.4.

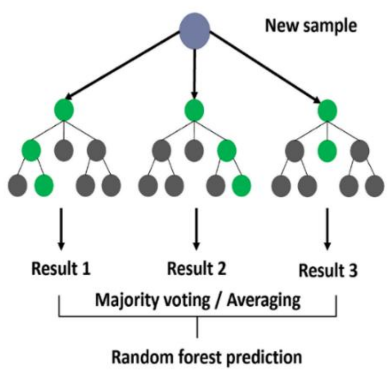

Figure 4.4: Visualization of random forest classifier

In the context of predicting a recession in a country using a dataset with 16 variables, the Random Forest Classifier is a powerful tool. Unlike a single decision tree, Random Forest is more like a diverse group of decision trees, each offering its perspective on the prediction. It works by combining the insights of multiple decision trees, each considering subsets of the variables, to collectively make predictions about the likelihood of a recession. Random Forest effectively deals with the complexities of the dataset, providing strong predictions. This collaborative method not only enhances prediction accuracy but also guards against overfitting, ensuring the model remains dependable even when faced with complex relationships among the variables. By setting this parameter to 'balanced' we instructed the random forest classifier to automatically adjust the class weights inversely proportional to their frequencies in the input data. This helps mitigate the effects of class imbalance and ensures fair representation of minority classes.

4.2.2 Decision Tree (DT) Classifier

Decision Tree Classifier is a popular machine learning model that uses a tree-like structure to make decisions based on the values of input features. It can be used for both classification and regression tasks.

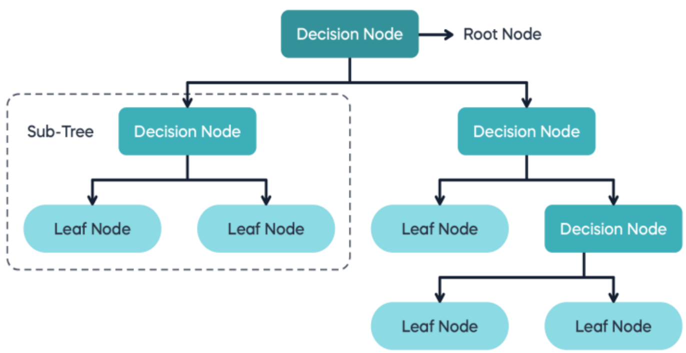

Figure 4.5: Visualization of decision tree classifier

Decision Trees are hierarchical structures composed of nodes, where each node represents a decision based on the value of a variable. As shown in Figure 4.5, the top node, known as the root node, [19] corresponds to the best feature that splits the dataset into subsets with the highest homogeneity or the greatest reduction in impurity. At each node of the tree, a decision is made based on the value of a specific feature, partitioning the dataset into subsets recursively until a stopping criterion is met. The terminal nodes of the tree, called leaf nodes, represent the final decision or prediction. One of the key advantages of DT is their interpretability, as the decision paths within the tree can be easily visualized and understood, providing insights into the underlying patterns in the data.

4.2.3 Logistic Regression (LR) Classifier

Logistic Regression is a widely used statistical technique for binary classification tasks. It is a linear model that models the probability of an instance belonging to a particular class. In the context of predicting a recession in a country with a dataset containing 16 variables, LR functions by seeking a decision boundary-essentially a line or boundary- within the multidimensional space of the variables.

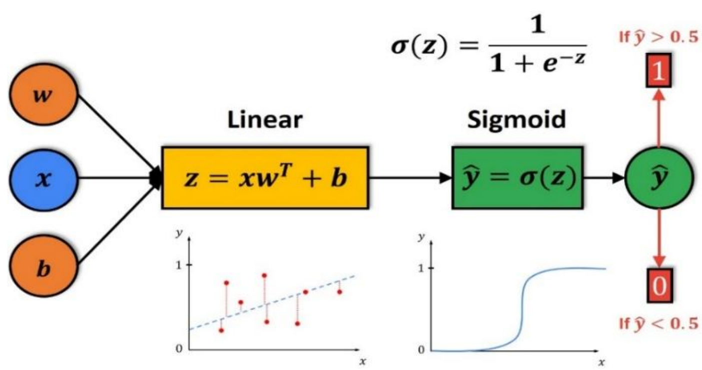

Figure 4.6: Visualization of logistic regression classifier

This boundary aims to effectively separate instances associated with economic recession from those not in recession. It achieves this by analyzing the relationships among our input variables, such as GDP growth rate, unemployment rate, inflation rate, and the likelihood of a recession occurring. Unlike traditional linear regression (LR), which forecasts exact values, logistic Regression estimates the probability that a country will experience a recession, typically ranging between 0 and 1. As shown in Figure 4.6, after determining these probabilities, Logistic Regression [20] applies a threshold, often at 0.5, to classify each instance as either indicative of an impending recession or not. This process involves minimizing the disparity between its predictions and the actual outcomes observed in the training data, ensuring a robust fit to the data.

4.2.4 Support Vector Machine (SVM) Classifier

SVM is a versatile classifier that can be used for both binary and multiclass classification, as well as regression tasks. In terms of financial data prediction, the Support Vector Machine (SVM) is a powerful tool.

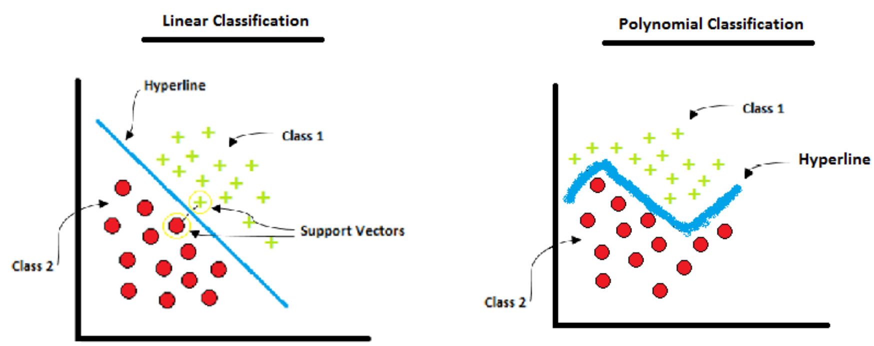

Figure 4.7: Visualization of support vector machine classifier

SVM operates by identifying the optimal hyperplane that best segregates instances representing recessionary periods from those indicating economic stability while maximizing the margin between classes. As shown in Figure 4.7, this hyperplane [21] serves as the decision boundary, distinguishing between different economic states with the widest possible gap. SVM identifies support vectors, which are instances closest to the decision closest to the decision boundary and utilizes them to define the hyperplane. In classification tasks, SVM assigns labels to new instances based on their position relative to the decision boundary. Notably, SVM can handle complex relationships among variables by employing kernal functions. In this study, we used kernal functions like the 'radial basis function' (RBF) in the SVM building code. This enables SVM to capture intricate patterns, even amidst complex data distributions. Like Random Forest Classifier (RF), by setting the parameter to 'balanced' we instructed the SVM to automatically adjust the class weights inversely proportional to their frequencies in the input data. This helps mitigate the effects of class imbalance and ensures fair representation of minority classes.

4.2.5 Multilayer Perceptron Neural Network (MLP NN) Classifier

MLP is a type of artificial neural network with multiple hidden layers. It is capable of learning complex relationships within data and is widely used for various tasks, including image recognition, natural language processing, and financial data prediction. Multilayer Perceptron Neural Network (MLP) encompasses various key components to effectively learn from data and make accurate predictions. As shown in Figure 4.8, its network [22] architecture consists of interconnected layers, including an input layer, and one or more hidden layers, where each neuron processes inputs as a weighted sum followed by an activation function to produce outputs.

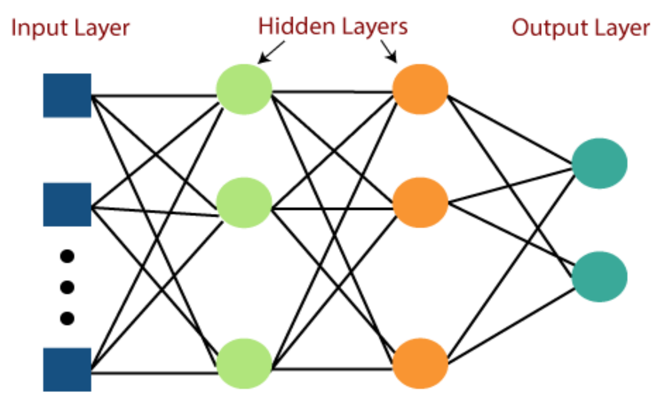

Figure 4.8: Visualization of multilayer perceptron classifier

Unlike Logistic Regression, MLP employs activation functions like sigmoid, tanh (the ratio of sinh and cosh), or relu (Rectified Linear Unit) to capture complex relationships between variables and predictions. Through forward propagation, input data traverses the network, with each layer applying transformations and activation functions to generate predictions. In this study, the hidden layer size is set to 128 and 64 indicating that the MLP consists of two hidden layers with 128 and 64 neurons, respectively. The 'relu' activation function is utilized, which is known for introducing non-linearity to the network. The ‘alpha’ parameter is set to 0.0001, controlling the regularization strength to prevent overfitting. The maximum iteration parameter is set to 1000, specifying the maximum number of iterations for the optimization algorithm. Overall, this configuration creates a robust MLP classifier with multiple hidden layers and appropriate activation functions, suitable for learning complex patterns and making accurate predictions.

4.3 Model Evaluation

Confusion matrices are common tools in evaluating the performance of classification models, offering a comprehensive overview of predicted versus actual class labels. As shown in Table 4.1, these matrices present a tabular representation where each row corresponds to the actual classes, and each column corresponds to the predicted classes. The main diagonal of the confusion matrix represents instances that are correctly classified, while off-diagonal elements indicate misclassifications.

Table 4.1: Confusion matrices

	Predicted Negative	**Predicted Positive**
Actual Negative	True Negative (TN)	False Positive (FP)
Actual Positive	False Negative (FN)	True Positive (TP)

By analyzing the values in the confusion matrices [5], accuracy, precision, recall, and F1 score are calculated to assess the model effectiveness. Accuracy measures the proportion of correctly classified instances out of the total instances in (1):

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{1}$$

where TP is the number of samples that are true positive, TN is true negative, FP is false positive, and FN is false negative.

Precision assesses the proportion of true positive predictions out of all positive predictions made by the model in (2):

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

Recall measures the proportion of true positive predictions out of all actual positives in the dataset in (3) :

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

F1 score is the harmonic means of precision and recall. It provides a balanced metric between precision and recall in (4) :

$$\text{F1 Score} = 2 \,.\, \frac{precision \cdot recall}{precision + recall} \tag{4}$$

The F1 score is important to evaluate classifiers on imbalanced datasets because it balances precision and recall, otherwise even a dummy classifier that always favors the decision in the majority class will yield a high accuracy, which alone is not meaningful. As our dataset is imbalanced, both accuracy and F1 score will serve as the primary metrics for evaluating the performance of a machine learning classifier.

CHAPTER FIVE: EXPERIMENTS AND RESULTS

In this section, we will evaluate the performance of various classifiers in recognition and forecasting tasks by systematically applying different data balancing techniques, including SMOTE, unweighted bootstrapping, and weighted bootstrapping, to both the original and interpolated datasets. In addition, we will explore the impact of PCA on the classifiers when applied with these balancing techniques to the original and interpolated datasets. The subplots in Figures 5.1 to 5.2 and 5.4 to 5.5 are organized in the same way with results from the recognition task in the upper row, and results from the forecasting task in the bottom row. Then the accuracy is reported in the left column, and the F1 score is reported in the right column. The x-axis includes baseline performance as the first categorical bin without using data balancing technique and a list of data-balancing techniques from the second categorical bin to the right as a comparison. Each data line, with a unique line style and color, represents a classifier in comparison. The differences between Figures 5.1 to 5.2 and 5.4 to 5.5 are that Figure 5.1 is on the results using the original quarterly data before PCA is applied, while Figure 5.2 is on the results after PCA is applied on the quarterly data. Figure 5.4 is on the results using the interpolated monthly data before PCA is applied, while Figure 5.5 is on the results after PCA is applied on the monthly data. The results are presented in detail below.

5.1 Original data before applying PCA for recognition and forecasting tasks

In Figure 5.1 for recognition tasks in the upper row, the MLP classifier (orange line with a cross marker) consistently outperformed all other classifiers across all techniques. In terms of data-balancing technique, the weighted bootstrapping method at 50% weight was the best technique (the right-most one), with the MLP classifier achieving the highest accuracy of 99.07% and an F1-score of 99.09%. SVM and LM did not perform well even with this weighted bootstrapping method at 50% weight.

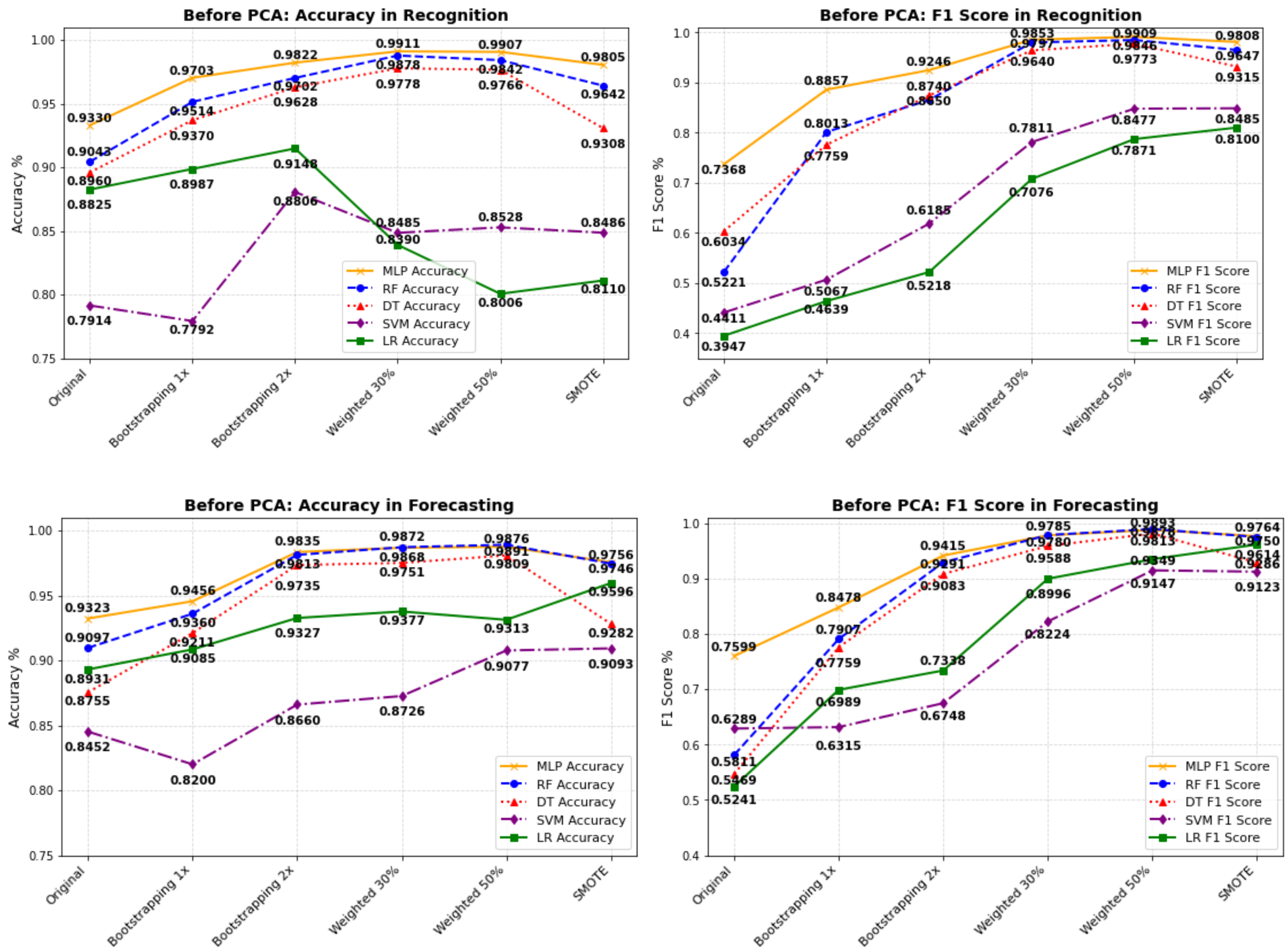

Figure 5.1: Classifier accuracy and F1 score for recognition (top) and forecasting (bottom) using original data before PCA

For forecasting tasks, the MLP classifier again excelled. The only exception is that under the weighted bootstrapping techniques (with 30% or 50% weight), RF outperformed other classifiers, but the margin between RF and MLP is tiny. In forecasting, SVM recorded the lowest accuracy of 82% and F1score of 63.15% with bootstrapping-original length (1x).

5.2 Original data after applying PCA with balancing techniques for recognition and forecasting tasks

As shown in Figure 5.2, after applying PCA, nearly all classifiers experienced declines in both accuracy and F1-scores for each of the data-balancing techniques, and for both recognition and forecasting tasks. RF (blue dashed line with a circle marker), instead of MLP, was shown to be the best performing classifier for recognition and forecasting, although with a small margin above MLP.

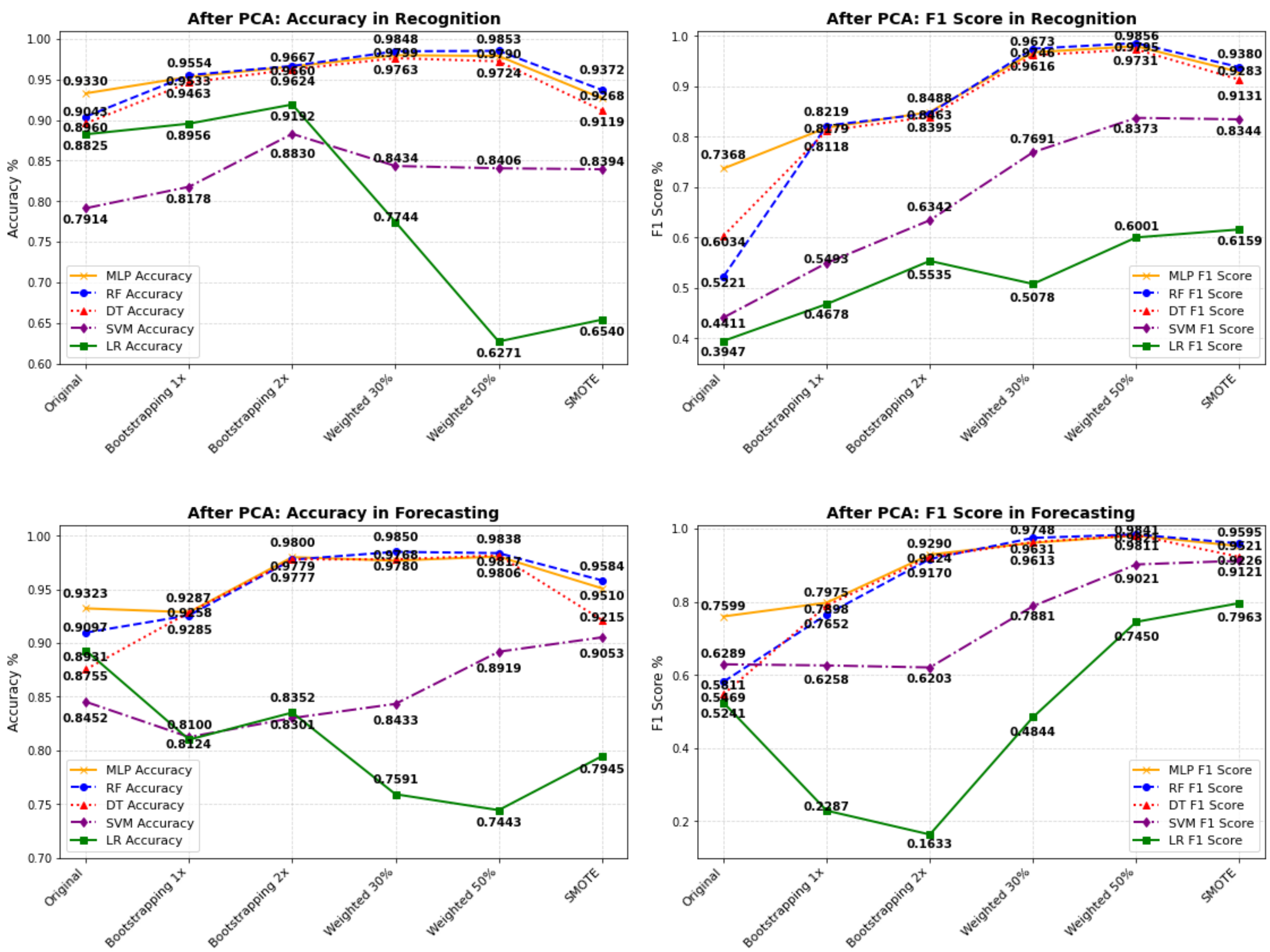

Figure 5.2: Classifier accuracy and F1 score for recognition (top) and forecasting (bottom) using original data after PCA

The only case when PCA improved performance was for DT in forecasting using weighted bootstrapping at 50% weight, where DT sees a slight increase in accuracy from 98.09% before PCA to 98.17% after PCA, and its F1-score rising from 98.13% to 98.21%, again with a small margin.

5.3 Analysis of Principal Component Analysis (PCA) weights for feature contribution on the original quarterly data

Principal Component Analysis (PCA) was applied to reduce the dimensionality of the dataset while retaining the most significant variance within the data. Figure 5.3 below presents the weight distribution for the first four principal components (PC1 to PC4) on the original quarterly recognition data which collectively captures a substantial portion of the dataset's variance. PC1 is mainly influenced by economic indicators like GDPC1 (real gross domestic product), A939RX0Q048SBEA (real gross domestic product per capita), GDP (gross domestic product), GDPPC (gross domestic product per capita), CAPEXP (capital expenditures domestic nonfinancial sectors), and EMP (All Employees total nonfarm), which have strong positive effects. On the other hand, INCTAX (U.S individual income tax), TOT (gross domestic product: terms of trade index), and LABCOST (unit labor costs) show strong negative effects, suggesting they move in the opposite direction to these economic trends. PC2 is strongly impacted by UNEMPR (civilian unemployment rate), showing that unemployment rates play a major role, while TOT (gross domestic product: terms of trade) has a large negative effect, indicating it behaves differently from unemployment trends.

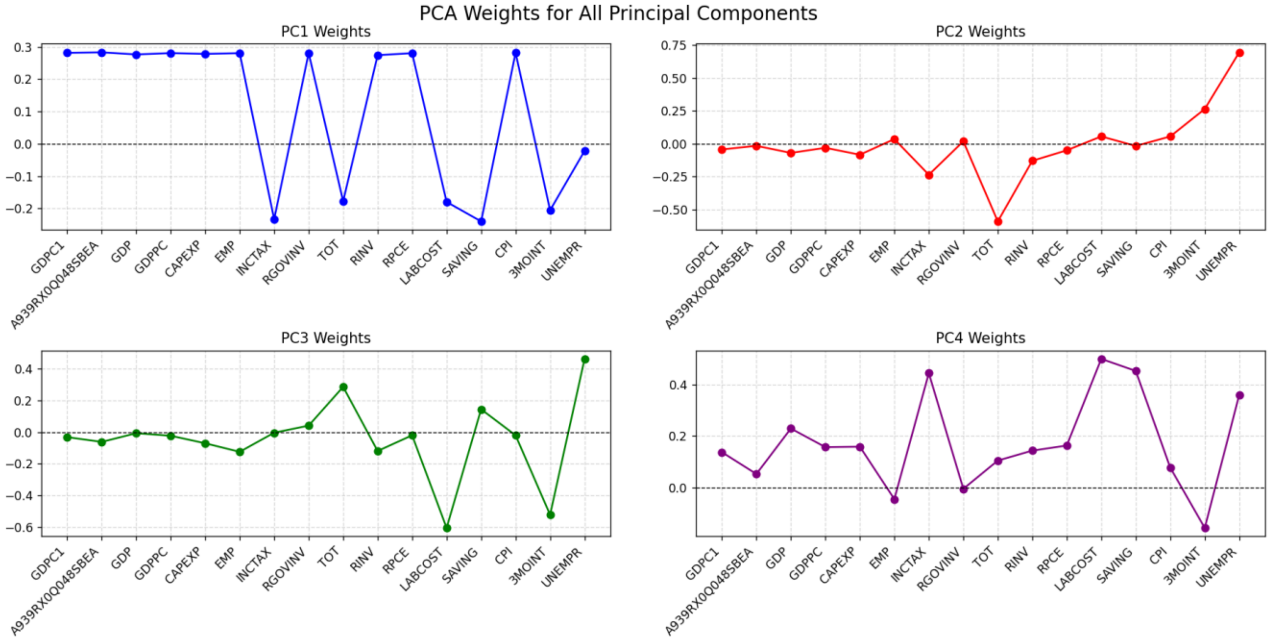

Figure 5.3: PCA weight distribution for first four principal components on original quarterly recognition data

In PC3, TOT and UNEMPR contribute positively, while LABCOST and SAVING (personal saving as a percentage of disposal personal income) have negative effects, suggesting this component is linked to financial stability and labor market conditions. PC4 is driven by positive contributions from INCTAX, SAVING, and UNEMPR, while 3MOINT (3-Month Treasury Bill) shows a sharp negative effect, highlighting the influence of taxation, savings patterns, and short-term interest rates.

We also analyzed the weights of the four principal components for the forecasting task and found that the results were mostly similar to the recognition analysis.

5.4 Interpolated data before applying PCA for recognition and forecasting

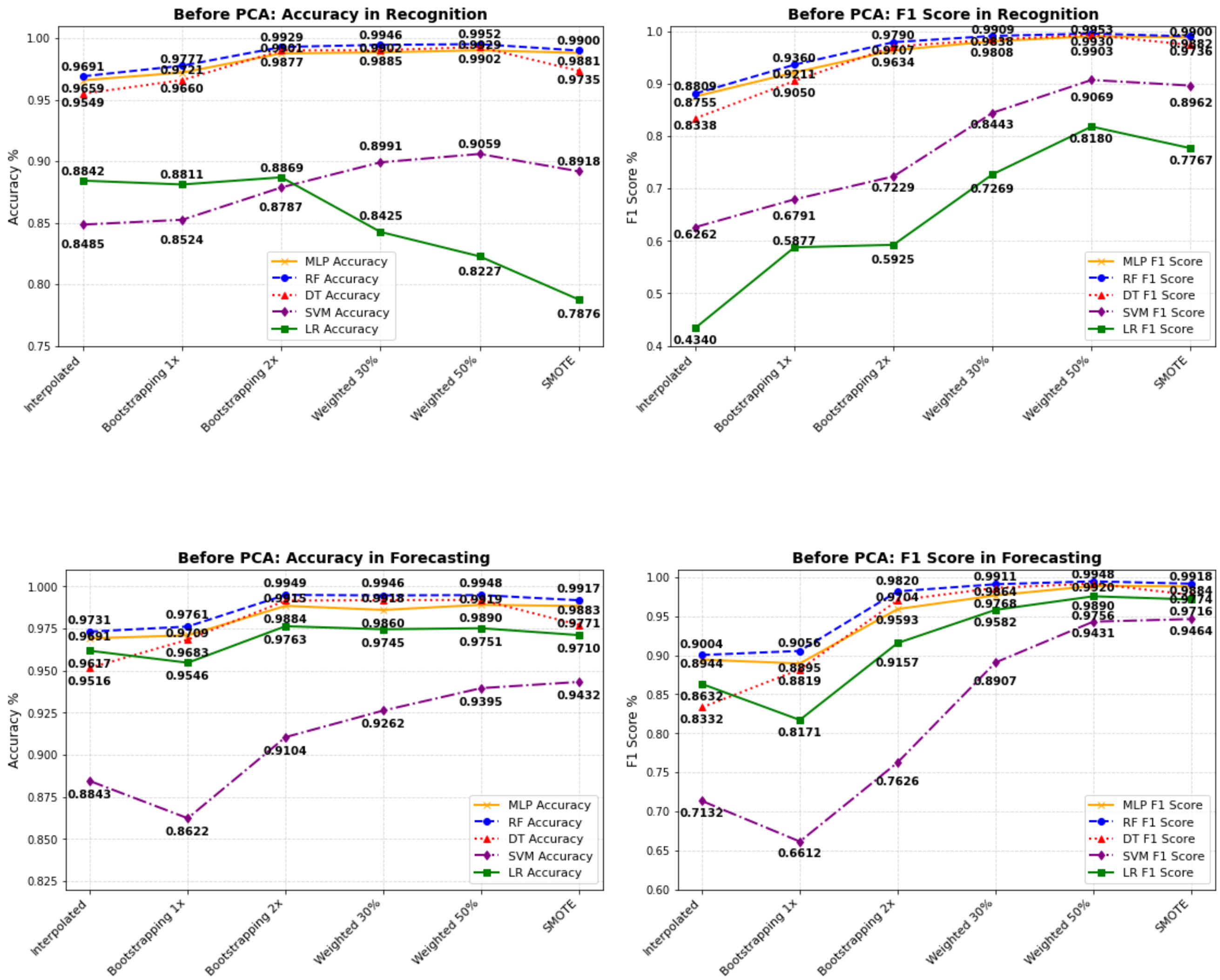

Figure 5.4: Classifier accuracy and F1 score for recognition (top) and forecasting (bottom) using interpolated data before PCA

Figure 5.4 presented the results using interpolated monthly data before PCA, where the RF classifier demonstrated superior performance for both recognition and forecasting tasks using weighted bootstrapping at 50% weight. In recognition, RF achieved the highest accuracy of 99.52% and an F1-score of 99.53%. In forecasting, RF excelled again, achieving the highest accuracy of 99.48% and an F1-score of 99.48%. LR and SVM recorded the lowest performance.

5.5 Interpolated data after applying PCA with balancing techniques for recognition and forecasting

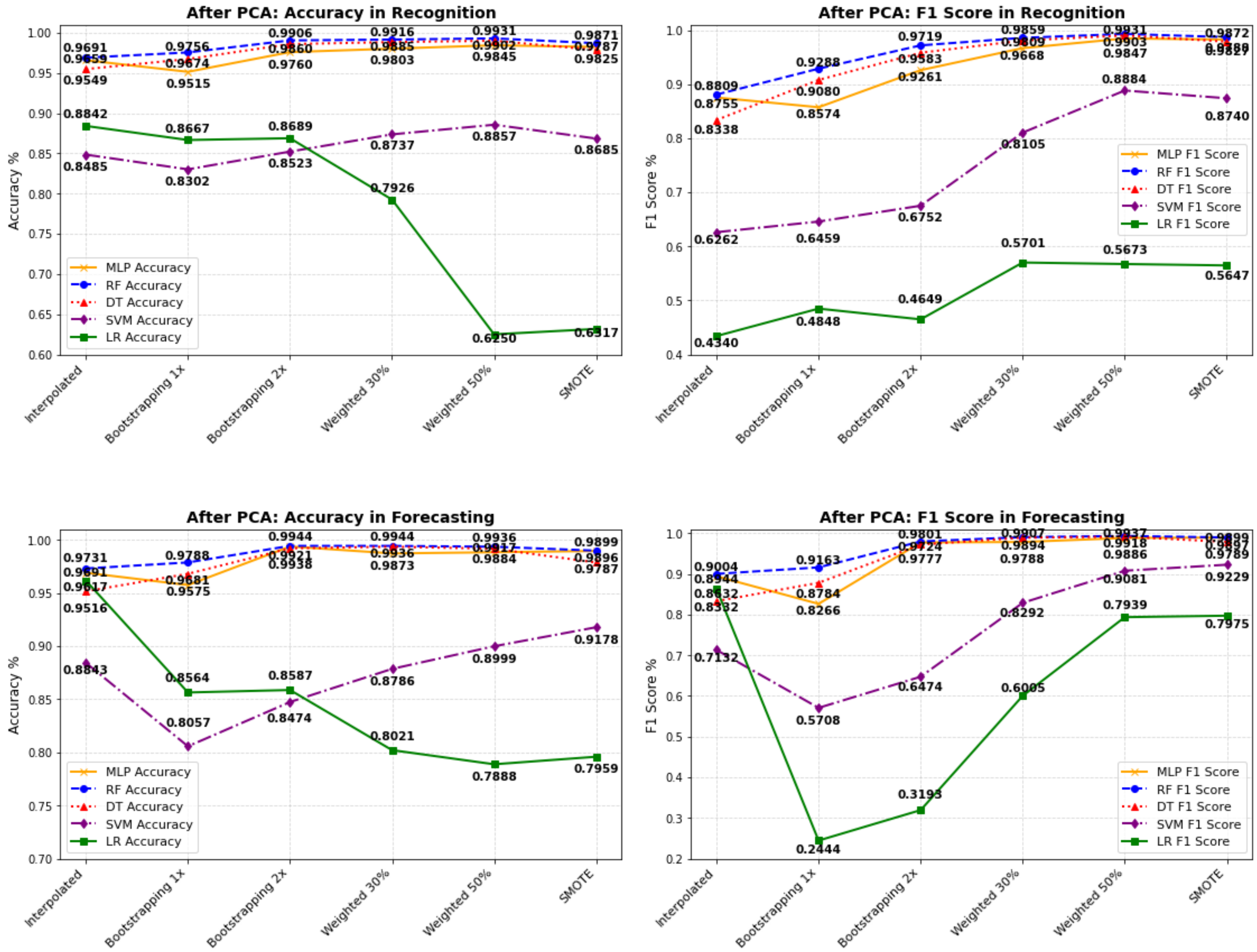

Figure 5.5: Classifier accuracy and F1 score for recognition (top) and forecasting (bottom) using interpolated data after PCA

Figure 5.5 presented the result using interpolated monthly data after PCA, where RF experienced a slight decrease in performance, with the highest accuracy dropping from 99.52% to 99.31% and the highest F1 scores from 99.53% to 99.31%, yet it still excelled over other classifiers. Other classifiers, including LR, DT, SVM, and MLP, also showed declines in both accuracy and F1-

scores after applying PCA with various balancing techniques on the interpolated data, for both recognition and forecasting tasks.

Following this analysis, Tables 5.1 to 5.8 present detailed performance metrics for both recognition and forecasting tasks across original and interpolated datasets, before and after applying PCA. These tables provide insights into how different classifiers perform under various data balancing techniques, including SMOTE, bootstrapping, and weighted bootstrapping. By comparing these results, we can assess the impact of data preprocessing and dimensionality reduction on classification accuracy and model robustness.

5.6 Analysis of Principal Component Analysis (PCA) weights for feature contribution on the interpolated monthly data

Figure 5.6 shows the weight distribution for the first four principal components (PC1 to PC4) on the interpolated monthly recognition data, which together explains a large part of the data's variation. Similar to the results from the original quarterly data, PC1 is mostly influenced by economic factors like GDPC1(real gross domestic product), A939RX0Q048SBEA (real gross domestic product per capita), GDP (gross domestic product), GDPPC (gross domestic product per capita), and CAPEXP (capital expenditures domestic nonfinancial sectors), which have strong positive effects. In contrast, INCTAX (U.S. individual income tax), TOT (gross domestic product terms of trade), and LABCOST (unite labor costs) show negative effects, suggesting they move in the opposite direction to these economic indicators. PC2 is strongly influenced by UNEMPR (civilian unemployment rate) play a major role, while TOT (gross domestic product terms of trade) has a strong negative effect, showing an opposite trend to unemployment.

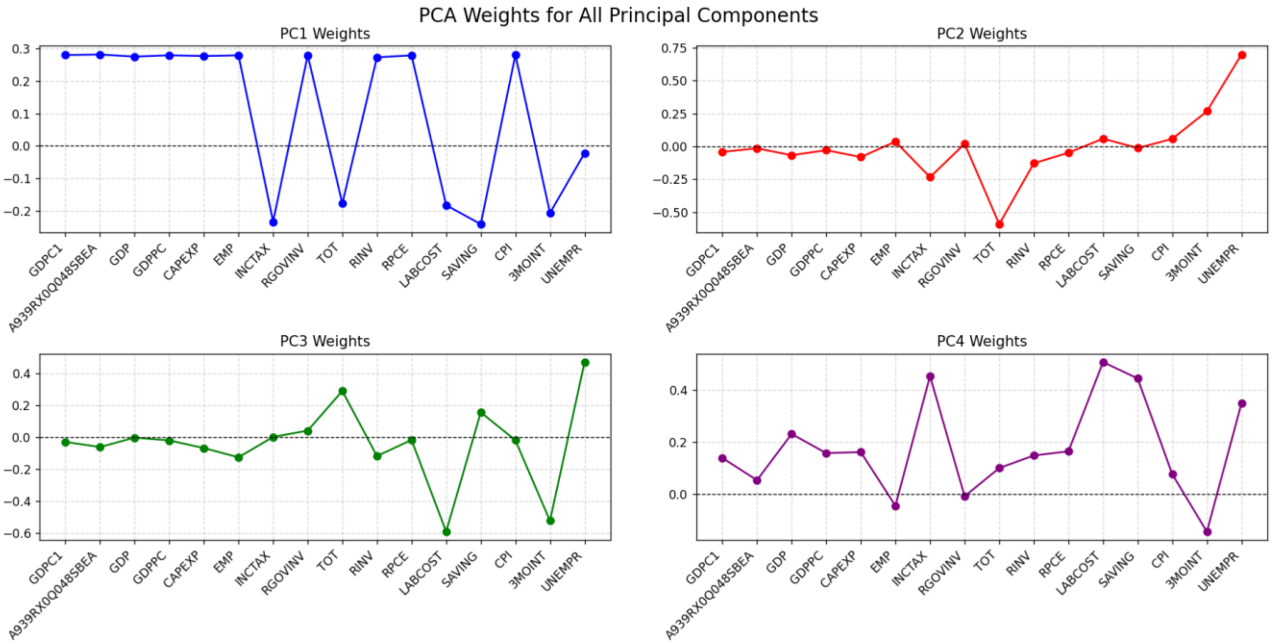

Figure 5.6: PCA weight distribution for first four principal components on interpolated monthly recognition data

In PC3, positive contributions from TOT and UNEMPR are offset by negative effects from LABCOST and SAVING, suggesting this component reflects financial stability and job market trends. PC4 highlights positive effects from INCTAX, SAVING, and UNEMPR, while 3MOINT shows a strong negative impact, indicating the influence of taxes, savings behavior, and short-term interest rates. Overall, these results show that despite changing the data from quarterly to monthly, the key influential features remain similar. These PCA results show that economic growth indicators, employment rates, and financial stability are key factors affecting the data, providing useful insights for identifying important features in predictive mode.

We also examined the weights of the four principal components for the forecasting task and found that the results were mostly similar to the recognition analysis.

5.7 Comparison of Principal Component weights between original quarterly and interpolated monthly datasets for recognition tasks

Figure 5.7 shows a comparison of the principal component (PC) weights from PC1 to PC4 between the original and interpolated recognition data. Each subplot represents one principal component, and the bars show how much each variable contributes to that component. The orange bars represent the original data, and the purple bars represent the interpolated data. Across all four components, the shapes and heights of the bars are very similar between the two datasets.

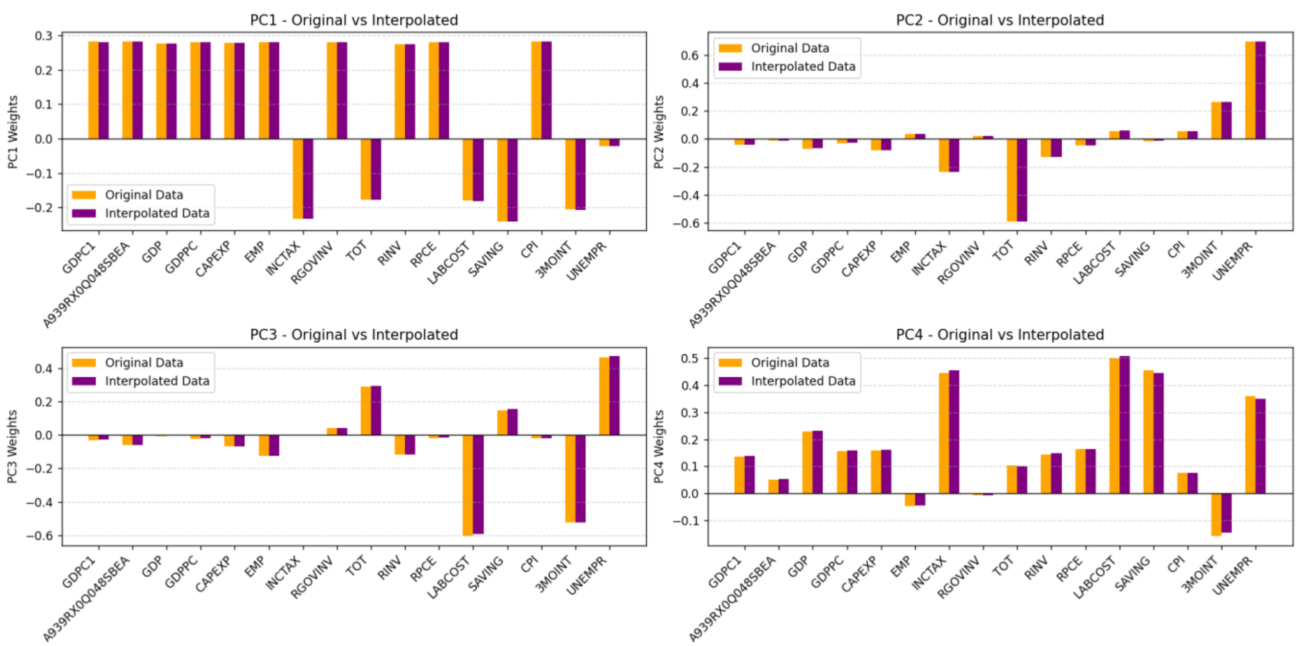

Figure 5.7: Comparison of principal component weights for original and interpolated recognition datasets

This means that the interpolation process did not significantly change the patterns in the data when we extended our data from quarterly to monthly. For example, variables like "UNEMPR" and "SAVING" continue to play important roles in both datasets, especially in PC2, PC3, and PC4.

Although there are a few small differences in some variables, the overall structure and trends remain the same.

Similarly, we also compared the principal component weights of the original and interpolated data for the forecasting task and found that across all four components, the shapes and heights of the bars were very similar between the two datasets.

Table 5.1: Performance Analysis of Classifiers on Recognition of Original Data Before PCA

Classifier	Average	Analysis of Recognition Original Data Before PCA					
		Original Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[53.4 0.6] [5.43 3.57]	[53.51 0.49] [2.57 6.43]	[109.90 0.10] [3.65 12.35]	[87.36 0.64] [0.9 37.1]	[61.08 1.92] [0.07 62.93]	[51.33 2.67] [1.2 52.8]
	Accuracy	0.9043	0.9514	0.9702	0.9978	0.9842	0.9642
	Precision	0.8859	0.9379	0.9931	0.9839	0.9710	0.9529
	Recall	0.3967	0.7144	0.7719	0.9763	0.9989	0.9778
	F1-score	0.5221	0.8013	0.8650	0.9797	0.9846	0.9647
Logistic Regression (LR)	Confusion metrics	[53.07 0.93] [6.47 2.53]	[53.7 0.3] [6.08 2.92]	[109.19 0.81] [9.93 6.07]	[81.01 6.99] [13.29 24.71]	[54.25 8.75] [16.37 46.63]	[44.07 9.93] [10.48 43.52]
	Accuracy	0.8825	0.8987	0.9148	0.8390	0.8006	0.8110
	Precision	0.7605	0.9179	0.8947	0.7821	0.8434	0.8175
	Recall	0.2811	0.3244	0.3794	0.6503	0.7402	0.8059
	F1-score	0.3947	0.4639	0.5218	0.7076	0.7871	0.8100
Decision Tree (DT)	Confusion metrics	[51.35 2.65] [3.9 5.1]	[52.11 1.89] [2.08 6.92]	[108.77 1.23] [2.58 13.42]	[85.98 2.02] [0.78 37.22]	[60.16 2.84] [0.11 62.89]	[49.66 4.34] [3.13 50.87]
	Accuracy	0.8960	0.9370	0.9628	0.9778	0.9766	0.9308
	Precision	0.6808	0.8071	0.9231	0.9501	0.9576	0.9227
	Recall	0.5667	0.7689	0.8387	0.9795	0.9983	0.9420
	F1-score	0.6034	0.7759	0.8740	0.9640	0.9773	0.9315
Support Vector Machine (SVM)	Confusion metrics	[44.72 9.28] [3.86 5.14]	[42.01 11.99] [1.92 7.08]	[98.86 11.14] [3.91 12.09]	[72.90 15.10] [3.99 34.01]	[55.45 7.55] [11.0 52.0]	[45.77 8.23] [8.12 45.88]
	Accuracy	0.7914	0.7792	0.8806	0.8485	0.8528	0.8486
	Precision	0.3877	0.3768	0.5344	0.6958	0.8748	0.8508
	Recall	0.5711	0.7867	0.7556	0.8950	0.8254	0.8496
	F1-score	0.4411	0.5067	0.6185	0.7811	0.8477	0.8485
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[52.59 1.41] [2.81 6.19]	[53.62 0.38] [1.49 7.51]	[109.68 0.32] [1.92 14.08]	[87.40 0.6] [0.52 37.48]	[61.83 1.17] [0.0 63.0]	[52.11 1.89] [0.22 53.78]
	Accuracy	0.9330	0.9703	0.9822	0.9911	0.9907	0.9805
	Precision	0.8247	0.9544	0.9804	0.9850	0.9821	0.9665
	Recall	0.6878	0.8344	0.8800	0.9863	1.0000	0.9959
	F1 -score	0.7368	0.8857	0.9246	0.9853	0.9909	0.9808

Table 5.2: Performance Analysis of Classifiers on Recognition of Original Data After PCA

Classifier	Average	Analysis of Recognition Original Data After PCA					
		Original Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[53.45 0.55] [5.99 3.01]	[53.44 0.56] [2.25 6.75]	[109.92 0.08] [4.12 11.88]	[87.07 0.93] [0.99 37.01]	[61.24 1.76] [0.09 62.91]	[49.9 4.1] [2.68 51.32]
	Accuracy	0.8962	0.9554	0.9667	0.9848	0.9853	0.9372
	Precision	0.8683	0.9354	0.9943	0.9764	0.9734	0.9273
	Recall	0.3344	0.7500	0.7425	0.9739	0.9986	0.9504
	F1-score	0.4617	0.8219	0.8463	0.9746	0.9856	0.9380
Logistic Regression (LR)	Confusion metrics	[53.45 0.55] [7.33 1.67]	[53.36 0.64] [5.94 3.06]	[109.38 0.62] [9.56 6.44]	[82.83 5.17] [23.26 14.74]	[43.58 19.42] [27.57 35.43]	[40.55 13.45] [23.92 30.08]
	Accuracy	0.8749	0.8956	0.9192	0.7744	0.6271	0.6540
	Precision	0.8313	0.8410	0.9265	0.7485	0.6477	0.6953
	Recall	0.1856	0.3400	0.4025	0.3879	0.5624	0.5570
	F1-score	0.2781	0.4678	0.5535	0.5078	0.6001	0.6159
Decision Tree (DT)	Confusion metrics	[50.2 3.8] [4.22 4.78]	[52.39 1.61] [1.77 7.23]	[108.72 1.28] [3.46 12.54]	[85.86 2.14] [0.84 37.16]	[59.78 3.22] [0.26 62.74]	[48.60 5.4] [4.11 49.89]
	Accuracy	0.8727	0.9463	0.9624	0.9763	0.9724	0.9119
	Precision	0.5812	0.8399	0.9157	0.9473	0.9518	0.9042
	Recall	0.5311	0.8033	0.7837	0.9779	0.9959	0.9239
	F1-score	0.5358	0.8118	0.8395	0.9616	0.9731	0.9131
Support Vector Machine (SVM)	Confusion metrics	[45.72 8.28] [3.94 5.06]	[44.56 9.44] [2.04 6.96]	[98.61 11.39] [3.35 12.65]	[73.40 14.60] [5.13 32.87]	[54.09 8.91] [11.17 51.83]	[46.73 7.27] [10.08 43.92]
	Accuracy	0.8060	0.8178	0.8830	0.8434	0.8406	0.8394
	Precision	0.3924	0.4316	0.5374	0.6953	0.8550	0.8600
	Recall	0.5622	0.7733	0.7906	0.8650	0.8227	0.8133
	F1-score	0.4503	0.5493	0.6342	0.7691	0.8373	0.8344
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[51.06 2.94] [4.43 4.57]	[53.26 0.74] [2.20 6.80]	[109.46 0.54] [3.74 12.26]	[86.29 1.71] [0.82 37.18]	[60.53 2.47] [0.17 62.83]	[48.75 5.25] [2.66 51.34]
	Accuracy	0.8830	0.9533	0.9660	0.9799	0.9790	0.9268
	Precision	0.6184	0.9201	0.9624	0.9579	0.9629	0.9082
	Recall	0.5078	0.7556	0.7662	0.9784	0.9973	0.9507
	F1 -score	0.5402	0.8179	0.8488	0.9673	0.9795	0.9283

Table 5.3: Performance Analysis of Classifiers on Forecasting of Original Data Before PCA

Classifier	Average	Analysis of Forecasting Original Data Before PCA					
		Original Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[52.26 0.74] [4.86 4.14]	[50.26 0.74] [3.23 7.77]	[105.89 0.11] [2.21 15.79]	[86.08 0.92] [0.67 36.33]	[60.70 1.30] [0.05 61.95]	[50.46 1.99] [0.68 51.87]
	Accuracy	0.9097	0.9360	0.9813	0.9872	0.9891	0.9746
	Precision	0.8741	0.9222	0.9943	0.9760	0.9799	0.9639
	Recall	0.4600	0.7064	0.8772	0.9819	0.9992	0.9871
	F1-score	0.5811	0.7907	0.9291	0.9785	0.9893	0.9750
Logistic Regression (LR)	Confusion metrics	[51.52 1.48] [5.15 3.85]	[49.41 1.59] [4.08 6.92]	[103.67 2.33] [6.01 11.99]	[81.55 5.45] [2.28 34.72]	[54.40 7.60] [0.92 61.08]	[48.27 4.18] [0.06 52.49]
	Accuracy	0.8931	0.9085	0.9327	0.9377	0.9313	0.9596
	Precision	0.7562	0.8454	0.8565	0.8674	0.8903	0.9272
	Recall	0.4278	0.6291	0.6661	0.9384	0.9852	0.9989
	F1-score	0.5241	0.6989	0.7338	0.8996	0.9349	0.9614
Decision Tree (DT)	Confusion metrics	[49.50 3.5] [4.22 4.78]	[48.58 2.42] [2.47 8.53]	[104.51 1.49] [1.79 16.21]	[85.01 1.99] [1.10 35.90]	[59.77 2.23] [0.14 61.86]	[48.25 4.20] [3.34 49.21]
	Accuracy	0.8755	0.9211	0.9735	0.9751	0.9809	0.9282
	Precision	0.6005	0.8015	0.9247	0.9492	0.9658	0.9225
	Recall	0.5311	0.7755	0.9006	0.9703	0.9977	0.9364
	F1-score	0.5469	0.7759	0.9083	0.9588	0.9813	0.9286
Support Vector Machine (SVM)	Confusion metrics	[44.42 8.58] [1.02 7.98]	[41.42 9.58] [1.58 9.42]	[90.38 15.62] [1.00 17.00]	[71.84 15.16] [0.64 36.36]	[51.38 10.62] [0.83 61.17]	[45.84 6.61] [2.91 49.64]
	Accuracy	0.8452	0.8200	0.8660	0.8726	0.9077	0.9093
	Precision	0.4936	0.5062	0.5282	0.7091	0.8536	0.8836
	Recall	0.8867	0.8564	0.9444	0.9827	0.9866	0.9446
	F1-score	0.6289	0.6315	0.6748	0.8224	0.9147	0.9123
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[51.05 1.95] [2.25 6.75]	[49.15 1.85] [1.52 9.48]	[105.23 0.77] [1.28 16.72]	[85.96 1.04] [0.6 36.40]	[60.51 1.49] [0.05 61.95]	[49.92 2.53] [0.03 52.52]
	Accuracy	0.9323	0.9456	0.9835	0.9868	0.9876	0.9756
	Precision	0.7912	0.8501	0.9593	0.9731	0.9770	0.9549
	Recall	0.7500	0.8618	0.9289	0.9838	0.9992	0.9994
	F1 -score	0.7599	0.8478	0.9415	0.9780	0.9878	0.9764

Table 5.4: Performance Analysis of Classifiers on Forecasting of Original Data After PCA

Classifier	Average	Analysis of Forecasting Original Data After PCA					
		Original Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[51.85 1.15] [4.76 4.24]	[49.67 1.33] [3.27 7.73]	[105.48 0.52] [2.25 15.75]	[85.98 1.02] [0.84 36.16]	[60.09 1.91] [0.1 61.9]	[49.15 3.3] [1.07 51.48]
	Accuracy	0.9047	0.9258	0.9777	0.9850	0.9838	0.9584
	Precision	0.8127	0.8689	0.9712	0.9737	0.9705	0.9409
	Recall	0.4711	0.7027	0.8750	0.9773	0.9984	0.9796
	F1-score	0.5789	0.7652	0.9170	0.9748	0.9841	0.9595
Logistic Regression (LR)	Confusion metrics	[51.14 1.86] [7.14 1.86]	[48.35 2.65] [9.13 1.87]	[101.46 4.54] [15.90 2.10]	[79.92 7.08] [22.79 14.21]	[45.77 16.23] [15.48 46.52]	[41.15 11.30] [10.28 42.27]
	Accuracy	0.8548	0.8100	0.8352	0.7591	0.7443	0.7945
	Precision	0.5379	0.4699	0.3686	0.6798	0.7428	0.7918
	Recall	0.2067	0.1700	0.1167	0.3841	0.7503	0.8044
	F1-score	0.2795	0.2287	0.1633	0.4844	0.7450	0.7963
Decision Tree (DT)	Confusion metrics	[49.65 3.35] [3.95 5.05]	[49.12 1.88] [2.55 8.45]	[104.82 1.18] [1.56 16.44]	[85.57 1.43] [1.3 35.7]	[59.87 2.13] [0.14 61.86]	[47.56 4.89] [3.35 49.20]
	Accuracy	0.8823	0.9285	0.9779	0.9780	0.9817	0.9215
	Precision	0.6302	0.8311	0.9378	0.9628	0.9673	0.9113
	Recall	0.5611	0.7682	0.9133	0.9649	0.9977	0.9362
	F1-score	0.5786	0.7898	0.9224	0.9631	0.9821	0.9226
Support Vector Machine (SVM)	Confusion metrics	[43.43 9.57] [1.14 7.86]	[40.78 10.22] [1.41 9.59]	[85.85 20.15] [0.92 17.08]	[68.55 18.45] [0.98 36.02]	[49.09 12.91] [0.49 61.51]	[43.58 8.87] [1.07 51.48]
	Accuracy	0.8273	0.8124	0.8301	0.8433	0.8919	0.9053
	Precision	0.4572	0.4928	0.4628	0.6630	0.8278	0.8542
	Recall	0.8733	0.8718	0.9489	0.9735	0.9921	0.9796
	F1-score	0.5966	0.6258	0.6203	0.7881	0.9021	0.9121
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[50.27 2.73] [3.02 5.98]	[48.77 2.23] [2.19 8.81]	[105.04 0.96] [1.52 16.48]	[85.17 1.83] [1.05 35.95]	[59.64 2.36] [0.05 61.95]	[48.79 3.66] [1.49 51.06]
	Accuracy	0.9073	0.9287	0.9800	0.9768	0.9806	0.9510
	Precision	0.7058	0.8116	0.9505	0.9528	0.9640	0.9341
	Recall	0.6644	0.8009	0.9156	0.9716	0.9992	0.9717
	F1 -score	0.6742	0.7975	0.9290	0.9613	0.9811	0.9521

Table 5.5: Performance Analysis of Classifiers on Recognition of Interpolated Data Before PCA

Classifier	**Average**	**Analysis of Recognition Interpolated Data Before PCA**					
		Interpolated Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[161.02 0.98] [4.86 22.14]	[153.62 1.38] [2.84 31.16]	[312.08 0.92] [1.74 62.26]	[263.01 0.99] [1.06 111.94]	[186.77 1.68] [0.12 188.43]	[159.24 2.21] [1.03 160.52]
	Accuracy	0.9691	0.9777	0.9929	0.9946	0.9952	0.9900
	Precision	0.9602	0.9595	0.9857	0.9914	0.9912	0.9865
	Recall	0.8200	0.9165	0.9728	0.9906	0.9994	0.9936
	F1-score	0.8809	0.9360	0.9790	0.9909	0.9953	0.9900
Logistic Regression (LR)	Confusion metrics	[158.60 3.40] [18.48 8.52]	[150.37 4.63] [17.85 16.15]	[303.17 9.83] [32.80 31.20]	[238.48 25.52] [33.87 79.13]	[159.92 28.53] [38.31 150.24]	[134.98 26.47] [42.12 119.43]
	Accuracy	0.8842	0.8811	0.8869	0.8425	0.8227	0.7876
	Precision	0.7343	0.7854	0.7660	0.7571	0.8409	0.8194
	Recall	0.3156	0.4750	0.4875	0.7003	0.7968	0.7393
	F1-score	0.4340	0.5877	0.5925	0.7269	0.8180	0.7767
Decision Tree (DT)	Confusion metrics	[158.65 3.35] [5.18 21.82]	[151.90 3.10] [3.33 30.67]	[311.41 1.59] [2.15 61.85]	[261.62 2.38] [1.31 111.69]	[185.82 2.63] [0.04 188.51]	[156.84 4.61] [3.94 157.61]
	Accuracy	0.9549	0.9660	0.9901	0.9902	0.9929	0.9735
	Precision	0.8706	0.9129	0.9759	0.9795	0.9863	0.9718
	Recall	0.8081	0.9021	0.9664	0.9884	0.9998	0.9756
	F1-score	0.8338	0.9050	0.9707	0.9838	0.9930	0.9736
Support Vector Machine (SVM)	Confusion metrics	[136.56 25.4] [3.2 23.8]	[131.74 23.26] [4.63 29.37]	[272.02 40.98] [4.74 59.26]	[235.95 28.05] [9.99 103.01]	[168.57 19.88] [15.59 172.96]	[137.18 24.27] [10.67 150.88]
	Accuracy	0.8485	0.8524	0.8787	0.8991	0.9059	0.8918
	Precision	0.4894	0.5622	0.5955	0.7881	0.8978	0.8623
	Recall	0.8815	0.8638	0.9259	0.9116	0.9173	0.9340
	F1-score	0.6262	0.6791	0.7229	0.8443	0.9069	0.8962
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[159.61 2.39] [4.06 22.94]	[152.87 2.13] [3.14 30.86]	[311.19 1.81] [2.82 61.18]	[261.67 2.33] [2.01 110.99]	[185.11 3.34] [0.36 188.19]	[158.34 3.11] [0.73 160.82]
	Accuracy	0.9659	0.9721	0.9877	0.9885	0.9902	0.9881
	Precision	0.9104	0.9383	0.9717	0.9797	0.9827	0.9811
	Recall	0.8496	0.9076	0.9559	0.9822	0.9981	0.9955
	F1 -score	0.8755	0.9211	0.9634	0.9808	0.9903	0.9882

Table 5.6: Performance Analysis of Classifiers on Recognition of Interpolated Data After PCA

Classifier	Average	Analysis of Recognition Interpolated Data After PCA					
		Interpolated Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[160.73 1.27] [7.10 19.90]	[154.04 0.96] [3.65 30.35]	[311.97 1.03] [2.5 61.50]	[262.35 1.65] [1.53 111.47]	[186.09 2.36] [0.25 188.30]	[158.65 2.80] [1.36 160.19]
	Accuracy	0.9557	0.9756	0.9906	0.9916	0.9931	0.9871
	Precision	0.9426	0.9712	0.9838	0.9856	0.9877	0.9829
	Recall	0.7370	0.8926	0.9609	0.9865	0.9987	0.9916
	F1-score	0.8229	0.9288	0.9719	0.9859	0.9931	0.9872
Logistic Regression (LR)	Confusion metrics	[158.82 3.18] [20.41 6.53]	[151.79 3.21] [21.99 12.01]	[305.99 7.01] [42.40 21.60]	[246.78 17.22] [60.97 52.03]	[142.82 45.63] [95.74 92.81]	[126.69 34.76] [84.21 77.34]
	Accuracy	0.8749	0.8667	0.8689	0.7926	0.6250	0.6317
	Precision	0.6979	0.8041	0.7657	0.7526	0.6711	0.6912
	Recall	0.2419	0.3532	0.3375	0.4604	0.4922	0.4787
	F1-score	0.3483	0.4848	0.4649	0.5701	0.5673	0.5647
Decision Tree (DT)	Confusion metrics	[158.16 3.84] [6.02 20.98]	[152.39 2.61] [3.55 30.45]	[310.79 2.21] [3.08 60.92]	[261.42 2.58] [1.74 111.26]	[185.18 3.27] [0.41 188.14]	[157.21 4.24] [2.64 158.91]
	Accuracy	0.9478	0.9674	0.9860	0.9885	0.9902	0.9787
	Precision	0.8504	0.9242	0.9659	0.9776	0.9830	0.9742
	Recall	0.7770	0.8956	0.9519	0.9846	0.9978	0.9837
	F1-score	0.8079	0.9080	0.9583	0.9809	0.9903	0.9788
Support Vector Machine (SVM)	Confusion metrics	[134.76 27.24] [3.9 23.10]	[127.76 27.24] [4.85 29.15]	[263.71 49.29] [6.41 57.59]	[227.66 36.34] [11.27 101.73]	[162.16 26.29] [16.81 171.74]	[132.89 28.56] [13.93 147.62]
	Accuracy	0.8352	0.8302	0.8523	0.8737	0.8857	0.8685
	Precision	0.4632	0.5221	0.5424	0.7379	0.8678	0.8383
	Recall	0.8556	0.8574	0.8998	0.9003	0.9108	0.9138
	F1-score	0.5987	0.6459	0.6752	0.8105	0.8884	0.8740
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[159.14 2.86] [7.74 19.26]	[152.17 2.83] [6.33 27.67]	[310.83 2.17] [6.87 57.13]	[261.09 2.91] [4.52 108.48]	[183.65 4.80] [1.05 187.50]	[156.80 4.65] [1.01 160.54]
	Accuracy	0.9439	0.9515	0.9760	0.9803	0.9845	0.9825
	Precision	0.8774	0.9125	0.9641	0.9742	0.9752	0.9720
	Recall	0.7133	0.8138	0.8927	0.9600	0.9944	0.9937
	F1 -score	0.7820	0.8574	0.9261	0.9668	0.9847	0.9827

Table 5.7: Performance Analysis of Classifiers on Forecasting of Interpolated Data Before PCA

Classifier	Average	Analysis of Forecasting Interpolated Data Before PCA					
		Interpolated Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[157.04 0.96] [4.02 22.98]	[159.01 0.99] [3.43 21.57]	[316.58 0.42] [1.46 51.54]	[257.87 1.13] [0.85 110.15]	[183.18 1.82] [0.12 184.88]	[155.1 2.35] [0.27 157.28]
	Accuracy	0.9731	0.9761	0.9949	0.9946	0.9948	0.9917
	Precision	0.9628	0.9573	0.9921	0.9900	0.9903	0.9854
	Recall	0.8511	0.8628	0.9725	0.9923	0.9994	0.9983
	F1-score	0.9004	0.9056	0.9820	0.9911	0.9948	0.9918
Logistic Regression (LR)	Confusion metrics	[155.37 2.63] [4.45 22.55]	[157.62 2.38] [6.01 18.99]	[313.31 3.69] [5.08 47.92]	[252.36 6.64] [2.8 108.2]	[177.05 7.95] [1.27 183.73]	[149.85 7.6] [1.53 156.02]
	Accuracy	0.9617	0.9546	0.9763	0.9745	0.9751	0.9710
	Precision	0.8989	0.8943	0.9302	0.9427	0.9587	0.9538
	Recall	0.8352	0.7596	0.9042	0.9748	0.9931	0.9903
	F1-score	0.8632	0.8171	0.9157	0.9582	0.9756	0.9716
Decision Tree (DT)	Confusion metrics	[157.47 4.53] [4.42 22.58]	[157.2 2.80] [3.06 21.94]	[315.29 1.71] [1.43 51.57]	[256.73 2.27] [0.77 110.23]	[182.06 2.94] [0.06 184.94]	[152.4 5.05] [2.15 155.4]
	Accuracy	0.9516	0.9683	0.9915	0.9918	0.9919	0.9771
	Precision	0.8370	0.8933	0.9688	0.9800	0.9845	0.9688
	Recall	0.8363	0.8776	0.9730	0.9931	0.9997	0.9864
	F1-score	0.8332	0.8819	0.9704	0.9864	0.9920	0.9774
Support Vector Machine (SVM)	Confusion metrics	[137.18 20.82] [0.59 26.41]	[134.90 25.10] [0.39 24.61]	[283.91 33.09] [0.06 52.94]	[231.68 27.32] [0.00 111.0]	[162.62 22.38] [0.0 185.0]	[139.63 17.82] [0.06 157.49]
	Accuracy	0.8843	0.8622	0.9104	0.9262	0.9395	0.9432
	Precision	0.5636	0.5003	0.6175	0.8034	0.8925	0.8987
	Recall	0.9781	0.9844	0.9989	1.0000	1.0000	0.9996
	F1-score	0.7132	0.6612	0.7626	0.8907	0.9431	0.9464
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[154.97 3.03] [2.68 24.32]	[157.63 2.37] [3.02 21.98]	[314.98 2.02] [2.26 50.74]	[255.70 3.30] [1.88 109.12]	[181.86 3.14] [0.94 184.06]	[154.24 3.21] [0.49 157.06]
	Accuracy	0.9691	0.9709	0.9884	0.9860	0.9890	0.9883
	Precision	0.8965	0.9089	0.9647	0.9711	0.9834	0.9801
	Recall	0.9007	0.8792	0.9574	0.9831	0.9949	0.9969
	F1 -score	0.8944	0.8895	0.9593	0.9768	0.9890	0.9884

Table 5.8: Performance Analysis of Classifiers on Forecasting of Interpolated Data After PCA

Classifier	Average	Analysis of Forecasting Interpolated Data After PCA					
		Interpolated Data	**Bootstrapping (same length)**	**Bootstrapping (double length)**	**Weighted (30%) Bootstrapping**	**Weighted (50%) Bootstrapping**	**SMOTE**
Random Forest (RF)	Confusion metrics	[156.87 1.13] [4.58 22.42]	[159.27 0.73] [3.19 21.81]	[316.59 0.41] [1.65 51.35]	[257.68 1.32] [0.76 110.24]	[182.78 2.22] [0.15 184.85]	[155.36 2.09] [1.09 156.46]
	Accuracy	0.9691	0.9788	0.9944	0.9944	0.9936	0.9899
	Precision	0.9547	0.9691	0.9924	0.9884	0.9882	0.9869
	Recall	0.8304	0.8724	0.9689	0.9932	0.9992	0.9931
	F1-score	0.8853	0.9163	0.9801	0.9907	0.9937	0.9899
Logistic Regression (LR)	Confusion metrics	[152.89 5.11] [19.90 7.10]	[154.06 5.94] [20.62 4.38]	[305.29 11.71] [40.58 12.42]	[241.52 17.48] [55.75 55.25]	[141.31 43.69] [34.47 150.53]	[124.0 33.45] [30.84 126.71]
	Accuracy	0.8648	0.8564	0.8587	0.8021	0.7888	0.7959
	Precision	0.5930	0.4296	0.5180	0.7624	0.7758	0.7917
	Recall	0.2630	0.1752	0.2343	0.4977	0.8137	0.8043
	F1-score	0.3573	0.2444	0.3193	0.6005	0.7939	0.7975
Decision Tree (DT)	Confusion metrics	[154.76 3.24] [4.56 22.44]	[157.64 2.36] [3.55 21.45]	[315.62 1.38] [1.54 51.46]	[257.25 1.75] [0.63 110.37]	[182.23 2.77] [0.29 184.71]	[152.90 4.55] [2.15 155.40]
	Accuracy	0.9578	0.9681	0.9921	0.9936	0.9917	0.9787
	Precision	0.8778	0.9052	0.9747	0.9846	0.9854	0.9718
	Recall	0.8311	0.8580	0.9709	0.9943	0.9984	0.9864
	F1-score	0.8509	0.8784	0.9724	0.9894	0.9918	0.9789
Support Vector Machine (SVM)	Confusion metrics	[132.90 25.10] [1.15 25.85]	[125.31 34.69] [1.26 23.74]	[261.88 55.12] [1.34 51.66]	[261.22 42.78] [2.15 108.85]	[150.18 34.82] [2.21 182.79]	[134.25 23.20] [2.70 154.85]
	Accuracy	0.8581	0.8057	0.8474	0.8786	0.8999	0.9178
	Precision	0.5105	0.4096	0.4852	0.7188	0.8406	0.8702
	Recall	0.9574	0.9496	0.9747	0.9806	0.9881	0.9829
	F1-score	0.6645	0.5708	0.6474	0.8292	0.9081	0.9229
Multilayer Perceptron Neural Network (MLP)	Confusion metrics	[155.09 2.91] [5.63 21.37]	[154.15 1.85] [6.01 18.99]	[316.57 0.43] [1.87 51.13]	[256.55 2.45] [2.26 108.74]	[181.39 3.61] [0.67 184.33]	[154.78 2.67] [0.61 156.94]
	Accuracy	0.9538	0.9575	0.9938	0.9873	0.9884	0.9896
	Precision	0.8864	0.9196	0.9921	0.9783	0.9809	0.9834
	Recall	0.7915	0.7596	0.9647	0.9796	0.9964	0.9961
	F1 -score	0.8320	0.8266	0.9777	0.9788	0.9886	0.9897

CHAPTER SIX: DISCUSSION OF THE RESULTS

This paper analyzed the effect of multiple factors that impacted recognition (at the present time) or forecasting (in a quarter ahead in the future) using imbalanced financial data. These factors include (1) various classifiers (MLP, RF, DT, SVM, and LR), (2) various data balancing techniques (SMOTE with a default 50% weight of the minority class, bootstrapping at the same data size or length (1x), bootstrapping at twice the length (2x), bootstrapping at twice the length and the weight for the minority class is 30%, bootstrapping at twice the length and the weight for the minority class is 50%), (3) using the original quarterly financial data or the interpolated monthly data, and (4) applying PCA on the data to extract features as the input to the classifiers or not. This analysis highlights distinct performance patterns among different classifiers.

Among all the classifiers, RF and MLP consistently emerged as the best performers across various data balancing techniques in terms of accuracy and F1 score. The MLP classifier performed better than RF with the original dataset because the original dataset had fewer instances (31 recession instances 31 and 179 non-recession instances). In such cases, MLP's ability to learn complex relationships and patterns makes it more effective, with or without data balancing techniques. However, using PCA features derived from the original data with data balancing, RF began to outperform MLP. PCA reduces the number of variables by keeping only the most important features and removing features with the smaller variance. This condensed data works well with RF, which is good at finding patterns in structured data. With the interpolated monthly dataset, RF consistently outperformed MLP in every scenario. RF performs well when there is more data to analyze because it works by creating multiple decision trees based on patterns and splits the data. More data allows RF to make more accurate and robust predictions. The interpolated dataset has more instances (91 recession instances and 537 non-recession instances) than the original dataset, which gave RF an advantage.

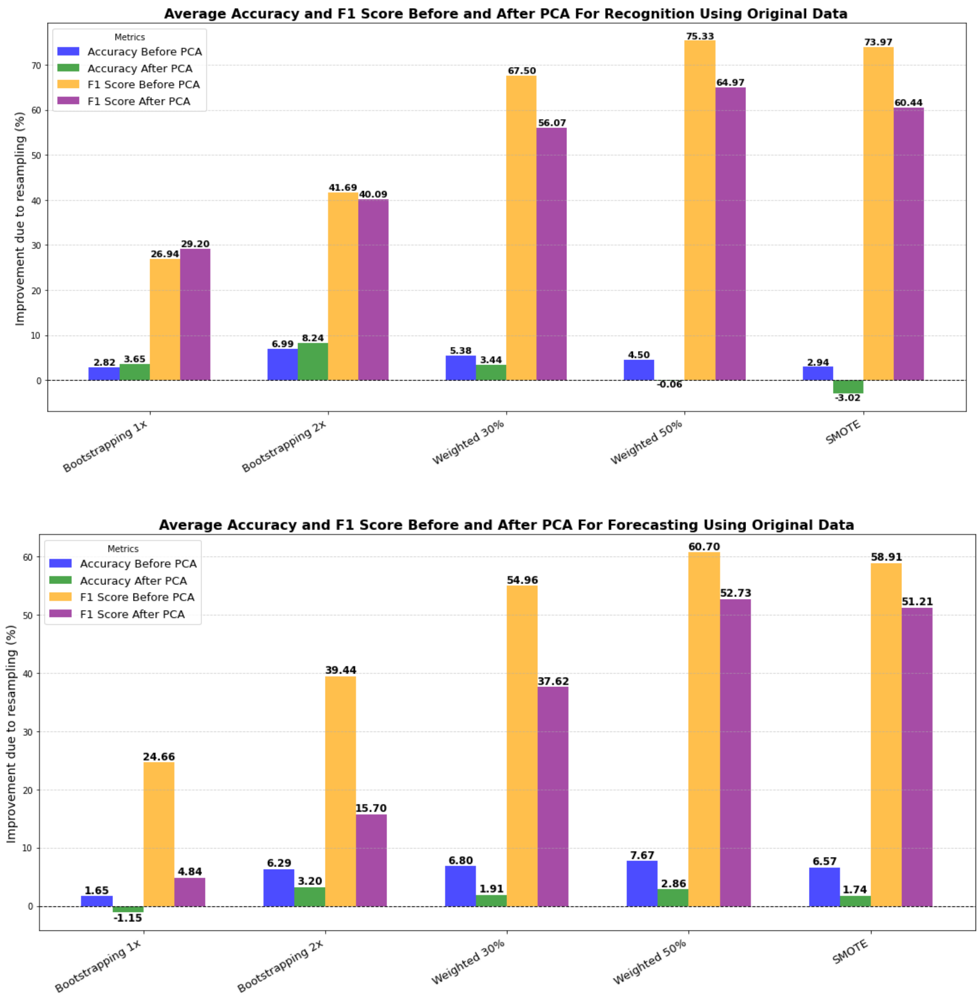

Figure 6.1: Effect of PCA, applied on original quarterly data, shown as the average improvement in classifier accuracy and F1 score due to data balancing techniques for recognition (top) and forecasting (bottom)

Figures 6.1 and 6.2 were generated by averaging the improvement in accuracy and F1 score of the performance of all the classifiers using various data balancing techniques, including SMOTE, bootstrapping-original length (1x) and bootstrapping-double length (2x), weighted bootstrapping (at 30% or 50% weight), using the original quarterly and the interpolated monthly economic data, respectively. Then such averages are compared between before PCA and after PCA to visualize the effect of PCA.

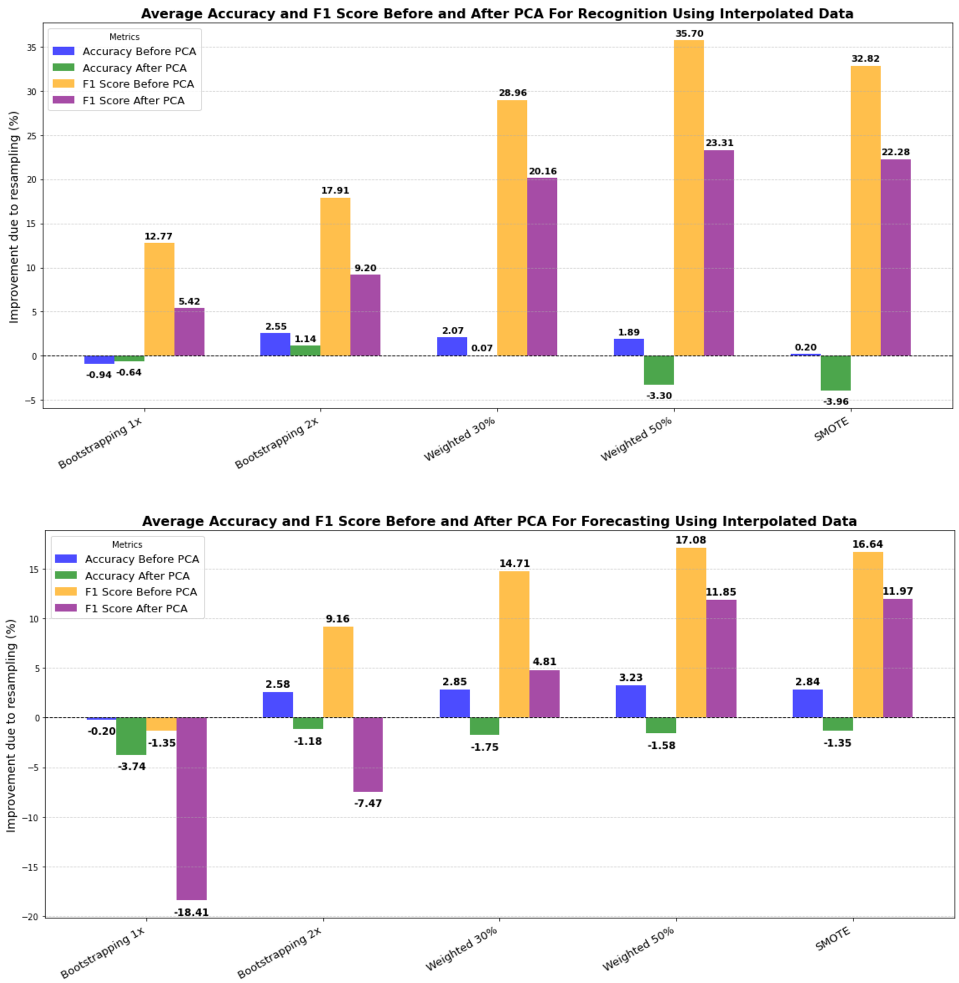

Figure 6.2: Effect of PCA, applied on interpolated monthly data, shown as the average improvement in classifier accuracy and F1 score due to data balancing techniques for recognition (top) and forecasting (bottom)

Among all the other data balancing techniques, bootstrapping at 50% weight and SMOTE are the most effective approaches. The weighted bootstrapping at 50% weight achieved the highest improvement in F1 score from before PCA is applied, with 75.33% improvement for recognition and 60.70% improvement for forecasting using original data, and 35.70% improvement for recognition and 17.08% improvement for forecasting using interpolated data. After PCA, however, all the F1 scores declined (purple bars are lower than orange bars). In terms of accuracy improvement, weighted bootstrapping at 50% weight achieved 4.50% improvement from baseline for recognition and 7.67% for forecasting using original data and 1.89% for recognition and 3.23% for forecasting using interpolated data before PCA. After PCA, accuracy declined across most of the techniques in both the datasets (green bars are lower than blue bars). Overall, SMOTE and weighted bootstrapping methods (30% and 50%) consistently outperformed normal bootstrapping techniques, with weighted bootstrapping at 50% emerging as the best technique in terms of both F1 score and accuracy. PCA reduced the performance of all classifiers across these data-balancing techniques.

CHAPTER SEVEN: CONCLUSION AND FUTURE WORK

This research highlights the essential role of selecting appropriate classifiers and data balancing techniques to optimize model performance on highly imbalanced datasets. Data balancing techniques, when applied to both the original and interpolated datasets, noticeably improved the performance of all classifiers, enabling them to learn more effectively from the data. Among these techniques, weighted bootstrapping at 50% weight was particularly effective, achieving the highest F1 scores for both recognition and forecasting tasks before and after PCA using original and interpolated datasets. However, PCA reduced the performance overall of all classifiers. This research provides useful information for policymakers and government bodies, helping them to create better policies to deal with economic challenges arising from highly imbalanced datasets.

Future research will focus on improving the interpretability of Random Forest (RF) and Multilayer Perceptron (MLP) models by examining their design and decision-making processes, alongside incorporating deep learning models. Further study will investigate the observed declines in performance with PCA and evaluate alternative dimensionality reduction techniques that may further enhance the classifier performance.

REFERENCES

[1] M. A. Rahman, S. Jamal, and H. Taheri, “Remote Condition Monitoring of Rail tracks using Distributed Acoustic Sensing (DAS): A Deep CNN-LSTM-SW based Model,” Green Energy Intell. Transp., p. 100178, Jan. 2024, doi: 10.1016/j.geits.2024.100178.

[2] B. Omar, B. Zineb, A. Cortes Jofre, and D. Gonzalez Cortes, “A Comparative Study of Machine Learning Algorithms for Financial Data Prediction,” in 2018 International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Rabat, Morocco: IEEE, Nov. 2018, pp. 1–5. doi: 10.1109/ISAECT.2018.8618774.

[3] S. R. R, L. N. T, and S. J. Ida, “Forecasting Unemployment Recession Rates: A Logistic Regression Approach Utilizing Comprehensive Economic Indicators,” in 2023 International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering (RMKMATE), Chennai, India: IEEE, Nov. 2023, pp. 1–6. doi: 10.1109/RMKMATE59243.2023.10368645.

[4] W. Yang, “A Novel Intelligent Prediction Model for Corporate Distress Using Machine Learning and Support Vector Machine,” in 2023 International Conference on Networking, Informatics and Computing (ICNETIC), Palermo, Italy: IEEE, May 2023, pp. 162–166. doi: 10.1109/ICNETIC59568.2023.00040.

[5] I. Vejalla, S. P. Battula, K. Kalluri, and H. K. Kalluri, “Credit Card Fraud Detection Using Machine Learning Techniques,” in 2023 2nd International Conference on Paradigm Shifts in Communications Embedded Systems, Machine Learning and Signal Processing (PCEMS), Nagpur, India: IEEE, Apr. 2023, pp. 1–4. doi: 10.1109/PCEMS58491.2023.10136040.

[6] D. Jin, “Research on Financial Time Series Analysis and Prediction Algorithm Based on Fuzzy Clustering Algorithm,” in 2022 2nd International Conference on Networking, Communications and Information Technology (NetCIT), Manchester, United Kingdom: IEEE, Dec. 2022, pp. 350–353. doi: 10.1109/NetCIT57419.2022.00089.

[7] S. Sa’adah, T. H. Liong, and Adiwijaya, “Prediction system of economic crisis in Indonesia using time series analysis and system dynamic optimized by genetic algorithm,” in 2012 International Conference on System Engineering and Technology (ICSET), Bandung: IEEE, Sep. 2012, pp. 1–6. doi: 10.1109/ICSEngT.2012.6339339.

[8] N. Zyatkov and O. Krivorotko, “Forecasting Recessions in the US Economy Using Machine Learning Methods,” in 2021 17th International Asian School-Seminar "Optimization Problems of Complex Systems (OPCS), Novosibirsk, Russian Federation: IEEE, Sep. 2021, pp. 139–146. doi: 10.1109/OPCS53376.2021.9588678.

[9] N. Sh. Mehdiyev, Babek. G. Guirimov, and R. R. Aliyev, “New neuro-fuzzy approach to recession prediction,” in 2009 Fifth International Conference on Soft Computing, Computing with Words and Perceptions in System Analysis, Decision and Control, Famagusta, Cyprus: IEEE, Sep. 2009, pp. 1–4. doi: 10.1109/ICSCCW.2009.5379422.

[10] D. Varmedja, M. Karanovic, S. Sladojevic, M. Arsenovic, and A. Anderla, "Credit Card Fraud Detection - Machine Learning methods," in 2019 18th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina: IEEE, Mar. 2019, pp. 1–5. doi: 10.1109/INFOTEH.2019.8717766.

[11] S. O. Moepya, S. S. Akhoury, and F. V. Nelwamondo, "Applying Cost-Sensitive Classification for Financial Fraud Detection under High Class-Imbalance," in 2014 IEEE International Conference on Data Mining Workshop, Shenzhen, China: IEEE, Dec. 2014, pp. 183–192. doi: 10.1109/ICDMW.2014.141.

[12] Dennis, I. R. Budianto, R. K. Azaria, and A. A. S. Gunawan, "Machine Learning-based Approach on Dealing with Binary Classification Problem in Imbalanced Financial Data," in 2021 International Seminar on Machine Learning, Optimization, and Data Science (ISMODE), Jakarta, Indonesia: IEEE, Jan. 2022, pp. 152–156. doi: 10.1109/ISMODE53584.2022.9742834.

[13] Lepot, M.; Aubin, J.-B.; Clemens, F.H.L.R. Interpolation in Time Series: An Introductive Overview of Existing Methods, Their Performance Criteria and Uncertainty Assessment. Water 2017, 9, 796. https://doi.org/10.3390/w9100796.

[14] Erick Odhiambo Omuya, George Onyango Okeyo, Michael Waema Kimwele, Feature Selection for Classification using Principal Component Analysis and Information Gain, Expert Systems with Applications, Volume 174, 2021, 114765, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.114765.

[15] B. Efron. "Bootstrap Methods: Another Look at the Jackknife." The Annals of Statistics, 7(1) 1-26 January 1979, https://doi.org/10.1214/aos/1176344552.

[16] Fernández, Alberto et al. "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary." J. Artif. Intell. Res. 61 (2018): 863-905.

[17] M. Somvanshi, P. Chavan, S. Tambade, and S. V. Shinde, "A review of machine learning techniques using decision tree and support vector machine," in 2016 International Conference on Computing Communication Control and automation (ICCUBEA), Pune: IEEE, Aug. 2016, pp. 1–7. doi: 10.1109/ICCUBEA.2016.7860040.

[18] T. M. Khoshgoftaar, M. Golawala and J. V. Hulse, "An Empirical Study of Learning from Imbalanced Data Using Random Forest," 19th IEEE International Conference on Tools with Artificial Intelligence(ICTAI 2007), Patras, Greece, 2007, pp. 310-317, doi: 10.1109/ICTAI.2007.46.

[19] B. Charbuty and A. Abdulazeez, "Classification Based on Decision Tree Algorithm for Machine Learning", JASTT, vol. 2, no. 01, pp. 20–28, Mar. 2021, doi: 10.38094/jastt20165.

[20] Reed, L.J., & Berkson, J. The Application of the Logistic Function to Experimental Data. The Journal of Physical Chemistry, 1929, 33, 760-779, doi:10.1021/J150299A014.

[21] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt and B. Scholkopf, "Support vector machines," in IEEE Intelligent Systems and their Applications, vol. 13, no. 4, pp. 18-28, July-Aug. 1998, doi: 10.1109/5254.708428.

[22] M.-C. Popescu, V. E. Balas, L. Perescu-Popescu, and N. E. Mastorakis, “Multilayer perceptron and neural networks,” WSEAS Transactions on Circuits and Systems archive, vol. 8, no. 7, pp. 579–588, Jul. 2009, doi: 10.5555/1639537.1639542.