Educational Administration: Theory and Practice
2024, 30(6), 2576 -2590
ISSN: 2148-2403
https://kuey.net/
Research Article

Data Expedition: Travel Through Data Preprocessing, EDA And PCA

Kaushik Paul[1*] , Dipankar Basu[2]

[1*]Assistant Professor, Department :CSE, Brainware University,Barasat, West Bengal, India, E- mail id : kkpaul47@gmail.com
[2]Assistant Professor (HOD), Department :Computer Application, Swami Vivekananda Institute Of Modern Studies, Barrackpore, West Bengal, India, E- mail id : dipankarbasu89@gmail.com

Citation: Kaushik Paul , Dipankar Basu et.al (2024) Data Expedition: Travel Through Data Preprocessing, EDA And PCA *Educational Administration: Theory and Practice, 30(6),* 2576 -2590
Doi: 10.53555/kuey.v30i6.5828

ARTICLE INFO

ABSTRACT

Pre-processing is essential in order to improve the quality of the data and make it more suitable for specific tasks like data mining. It describes the steps taken to prepare data for analysis, such as cleaning, converting, and integrating it. This chapter focuses on the comprehensive analysis of the data collection process through web scraping techniques, preprocessed using some Python methods, and finally analyzed with the help of exploratory data analysis (EDA). Initially, different data collection methods are outlined, followed by preprocessing steps including statistical information, which determines the overall structure of the dataset considered. To build a machine learning (ML) model, some data pre-processing schemes are considered, such as handling missing or null values (N-V), outlier detection, and removing duplicates. Exploratory Data Analysis (EDA) is conducted at various levels, including univariate, bivariate, and multivariate analysis, to understand the relationships within the dataset. A dataset may contain a large number of feature variables, which can be merged into a smaller number of variables using principal component analysis (PCA). PCA reduces the complexity of the model that will be built using ML algorithms such as logistic regression, linear regression, etc. This chapter provides insights into the entire process of data analysis, from data collection to model evaluation, demonstrating the effectiveness of web scraping in extracting valuable information for predictive modeling. Subsequently, both logistic regression and linear regression models are constructed to predict target variables. Feature selection techniques are employed to identify the most influential variables, and principal component analysis (PCA) is utilized for dimensionality reduction. Finally, model performance is evaluated using confusion matrices for the logistic regression model and root-mean-squarederror for the linear regression model. In this work,the Python language is considered, which is an object-oriented, interpreted, and interactive programming language. It is open source with rich sets of libraries like Pandas, Numpy, Matplotlib, Seaborn, etc. For executing the Python code, JUPYTER NOTEBOOK is used, which provides a web-based application process and a rich media representation of the object.

***Keywords*:** pre-processing, exploratory data analysis (EDA), machine learning (ML),principal component analysis (PCA), Matplotlib, Seaborn, Numpy, Pandas, Jupyter Notebook.

INTRODUCTION

Machine learning (ML) is a revolutionary domain of artificial intelligence that delegates computers to grasp knowledge from data and enhance their capacity without explicit programming[1]. This centers on utilizing information and calculations to empower AI to parody the way that people learn, moderately progressing its accuracy. Enhancing computers with "machine knowledge" that can power intelligent applications is a long-standing goal for AI [2]. Data science is a fast-growing area in which machine learning plays a critical role. In data mining projects, algorithms are trained to classify data, provide predictions, and uncover new information using statistical techniques. Key growth indicators should be impacted by the actions taken by apps and businesses based on this data. With the creation and expansion of big data, there will likely be a greater need for scientists. They must assist in identifying the most important business questions and the associated data requirements. Data preprocessing [3,4] is one of the major phases within the knowledge discovery process. The global objective of data preprocessing is to remove unwanted variability or effects from the signal so that the useful information related to the property(ies) of interest can be used for efficient modeling [5]. Preprocessing of data involves various steps, including data cleaning, where missing values need to be filled, outliers should be identified, and smoothing out inconsistent and noisy data. In data integration, redundancy should be handled, and aggregation, generalization, normalization and attribute construction are performed in data transformation.Presently, the amount of generated data is growing exponentially following the emergence of thebig data phenomenon [6,17]. Data reduction techniques perform this simplification by selecting and deleting redundant and noisy features and/or instances, or by discretizing complex continuous feature spaces. This allows the input to maintain its original structure and meaning while at the same time obtaining a much more manageable size [8]. In [9], proper data preprocessing can eliminate changes in process or system conditions, as well as in data collection or transmission effects beforehand, which result in more parsimonious models evaluated by Famili et al. Growing amounts of data produced by modern process monitoring and data acquisition systems have resulted in correspondingly large data processing requirements, and therefore, efficient techniques for automatic data preprocessing are important [10]. Preprocessing the data for proper interpretation is a form of feature extraction that conditions the input data to allow easier subsequent feature extraction and increased resolution [11]. The use of principal components has been extensively studied[12]. The main goal of identifying principal components is to select proper attributes for data analysis. Identifying principal components involves checking the linear dependency among independent variables in a set of data attributes. According to Makiewicz et al. an important issue in principal component analysis is the interpretation of the component to help determine, after the reduction of the observation space, which initial variables have the greatest share in the variance of particular principal components [13]. Chatfield et al, showed that the EDA includes checks on data quality, the calculation of summary statistics and the plotting of appropriate graphs. The main objectives of EDA are data description and model formulation. As regards data description, it begins by summarizing the data and picking out the more important features. There are many situations where EDA is vital in generating hypotheses, building a suitable model and suggesting an appropriate statistical procedure to analyze a given data set[14]. In this chapter, a comprehensive analysis is being performed on the collected data set. Firstly, preprocessing techniques are used to increase the efficiency of the data set. EDA helps to understand the relationship between variables in data,while PCA reduces complexity through feature selection and dimensionality reduction. By constructing an evolutionary matrix for the logistic regression modeland rootmeansquarederror for the linear regression model, the performance of the model would be evaluated.

SYSTEM PROCESS MODEL

The framework or workflow of this analytics process model or system focuses on the following steps, each of which has its own prescribed task or significance in evaluating or understanding the functionality of the system. The commencement of the model starts with inserting unstructured data as input. Data preprocessing is crucial in preparing data for analysis. Most commonly, it involves null value treatment, outlier detection and duplication handling. Handling null values is essential to ensuring that the model can use all variables, detecting and treating outliers is important as they can negatively affect the accuracy of the model; and duplication handling improves data quality. The next EDA is performed to understand the relationship between variables within the data set. PCA helps to decrease complexity by using dimensionality reduction. The final step is to interpret the result of the analysis, and thus, the output data is ready to be used for model building. The complete flow of the data process is depicted in Fig. 2.1.

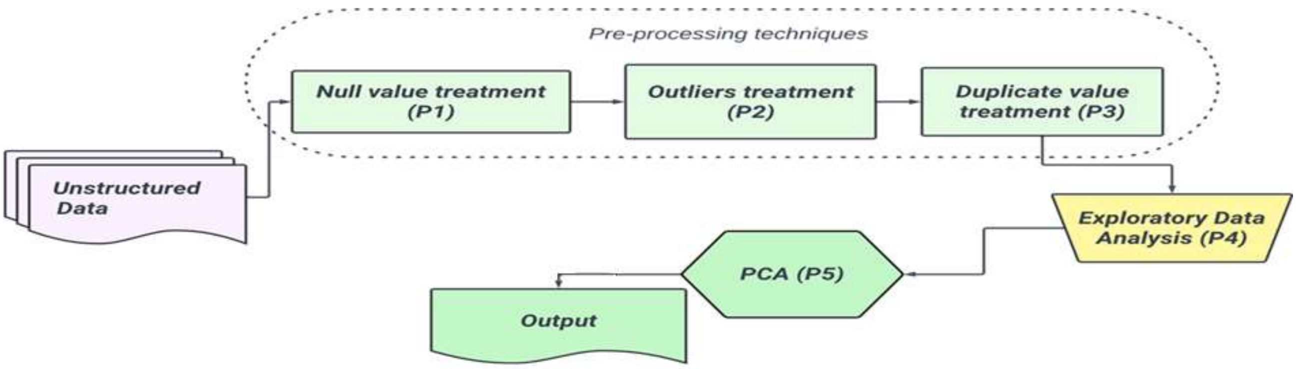

Figure 2.1: Block diagram of the system process model

PRE-PROCESSING

The data preprocessing can often have a significant impacton the generalization performance of the MLalgorithm [15].The elimination of noise instances is one of themost difficult problems in inductive ML [16]. Typically, the deleted instances have an excessive number of null feature values and are highly deviant.Theseoutliers are another name for deviant traits. Additionally, choosing a single sample from a large data set is a frequent strategy to deal with the impossibility of learning from very large data sets. Another problem that is frequently addressed in the data preparation stages is missing data handling.Feature selection is theprocess of identifyingand removing as much irrelevant and redundant informationas possible. This reducesthe dimensionality of the data andmay allow learning algorithms to operate faster and moreeffectively. Insome cases, accuracy in future classificationcan be improved.

NULL VALUE TREATMENT

In a dataset, the presence of empty cells, rows, and columns, referred to as null or missing values, leads to inconsistency in the dataset. Missing values may generate bias and affect the quality ofthe outcome [17,18]. The reason behind N-V could be that data does not exist, or data has been deleted accidentally, or the value is notrelevant to a particular case, could not be recorded whenthe data was collected, or is ignoredby users because ofprivacy concerns[19,20]. So, the detection of N-V is important in order to make the data set efficient for processing or ready for applying modeling stuff. Here, Python code is being used to detect N-V in data sets. Reading and detecting missing values in data sets are shown in Fig. 3.1.1.

```
train=pd.read_csv('train.csv')
test=pd.read_csv('test.csv')
print('Training data shape: ', train.shape)
print('Testing data shape: ', test.shape)
train.head()
```

```
Training data shape:  (891, 12)
Testing data shape:  (418, 11)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Figure 3.1.1: Head of the dataset

Figs. 3.1.2(a) and 3.1.2(b) show the count or percentage of missing values in every column of the train and test datasets,respectively which gives an idea about the distribution of N-V.

```
train_missing= missing_values_table(train)
train_missing
test_missing= missing_values_table(test)
test_missing
```

(a)

Your selected dataframe has 12 columns.
There are 3 columns that have missing values.

	Missing Values	% of Total Values
Cabin	687	77.1
Age	177	19.9
Embarked	2	0.2

(b)

Your selected dataframe has 11 columns.
There are 3 columns that have missing values.

	Missing Values	% of Total Values
Cabin	327	78.2
Age	86	20.6
Fare	1	0.2

Figure 3.1.2: Missing value summary of test dataset

From the above outcome,it can be seen that both the train and test sets have the same proportion of missing values.After detecting N-V in data sets, it is important to treat them too. There are some techniques,such as dropping methods, backward and forward filling techniques and statistical imputation,using which N-V can be treated. In the dropping method, either drop the rows or columns that contain N-V. Here, thedropna() function is being used to remove all the rows with N-V in the data frame,as depicted in figs.3.1.3(a) and 3.1.3(b), respectivelyfor the train and test datasets, respectively.

import pandas as pd
df = pd.read_csv('train.csv')
newdf = df.dropna()
print(newdf)
df = pd.read_csv('test.csv')
newdf1 = df.dropna()
print(newdf1)

(a)

	PassengerId	Survived	Pclass	\	Cabin	Embarked
1	2	1	1		C85	C
3	4	1	1		C123	S
6	7	0	1		E46	S
10	11	1	3		G6	S
11	12	1	1		C103	S
..	...	...	...		...	...
871	872	1	1		D35	S
872	873	0	1		B51 B53 B55	S
879	880	1	1		C50	C
887	888	1	1		B42	S
889	890	1	1		C148	C

[183 rows x 12 columns]

(b)

	PassengerId	Pclass	Name	\		Embarked
12	904	1	Snyder, Mrs. John Pillsbury (Nelle Stevenson)		12	S
14	906	1	Chaffee, Mrs. Herbert Fuller (Carrie Constance...		14	S
24	916	1	Ryerson, Mrs. Arthur Larned (Emily Maria Borie)		24	C
26	918	1	Ostby, Miss. Helene Ragnhild		26	C
28	920	1	Brady, Mr. John Bertram		28	S
..	...	...	...		...	...
404	1296	1	Frauenthal, Mr. Isaac Gerald		404	C
405	1297	2	Nourney, Mr. Alfred (Baron von Drachstedt")"		405	C
407	1299	1	Widener, Mr. George Dunton		407	C
411	1303	1	Minahan, Mrs. William Edward (Lillian E Thorpe)		411	Q
414	1306	1	Oliva y Ocana, Dona. Fermina		414	C

[87 rows x 11 columns]

Figure 3.1.3: (a) Dropping method for handling N-V in train dataset (b) Dropping method for handling N-V in test dataset.

Nevertheless, this approach has certain disadvantages, including the potential to lose important data, reduce the sample size, or introduce bias into the data distribution.In backward and forward filling,the missing values are replaced with the next available observation and the most recent available observation, respectively, the figs. 3.1.4(a) and 3.1.4(b) illustrate the forward and backward filling processes, respectively.

df = pd.read_csv('train.csv')
df.ffill(axis = 1)
df = pd.read_csv('test.csv')
df.bfill(axis = 1)

(a)

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.25	7.25	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.925	7.925	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.05	8.05	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0	13.0	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	female	1	2	W./C. 6607	23.45	23.45	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.75	7.75	Q

891 rows × 12 columns

(b)

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	Q	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0	S	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	Q	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	S	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	S	S
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	0	0	0	A.5. 3236	8.05	S	S
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9	C105	C
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.25	S	S
416	1308	3	Ware, Mr. Frederick	male	0	0	0	359309	8.05	S	S
417	1309	3	Peter, Master. Michael J	male	1	1	1	2668	22.3583	C	C

418 rows × 11 columns

Figure 3.1.4: (a)Forward-filling for handling N-V in train dataset (b) Backward-filling for handling N-V in test dataset

i.Inthe statistical (mode) imputation method, the most frequently occurring value of the dataset replaces the N-Vs. It is preferred if the data is a string(object) or numeric. The corresponding Python code is shown below. Figure3.1.5 depicts the result after mode imputation to null values.

ii.df = pd.read_csv('test.csv')
mode_values = df.mode().iloc[0]
df_filled = df.fillna(mode_values)
print(df_filled)
df = pd.read_csv('train.csv')
mode_values = df.mode().iloc[0]
df_filled = df.fillna(mode_values)
print(df_filled)

(a)

```
     PassengerId  Pclass                                          Name  \        Cabin Embarked
0            892       3                              Kelly, Mr. James   0    B57 B59 B63 B66        Q
1            893       3              Wilkes, Mrs. James (Ellen Needs)   1    B57 B59 B63 B66        S
2            894       2                     Myles, Mr. Thomas Francis   2    B57 B59 B63 B66        Q
3            895       3                              Wirz, Mr. Albert   3    B57 B59 B63 B66        S
4            896       3  Hirvonen, Mrs. Alexander (Helga E Lindqvist)   4    B57 B59 B63 B66        S
..           ...     ...                                           ...  ..                ...      ...
413         1305       3                            Spector, Mr. Woolf  413  B57 B59 B63 B66        S
414         1306       1                  Oliva y Ocana, Dona. Fermina  414             C105        C
415         1307       3                  Saether, Mr. Simon Sivertsen  415  B57 B59 B63 B66        S
416         1308       3                           Ware, Mr. Frederick  416  B57 B59 B63 B66        S
417         1309       3                      Peter, Master. Michael J  417  B57 B59 B63 B66        C

[418 rows x 11 columns]
```

(b)

```
     PassengerId  Survived  Pclass  \  Parch              Ticket     Fare    Cabin Embarked
0              1         0       3         0           A/5 21171   7.2500  B96 B98        S
1              2         1       1         0            PC 17599  71.2833      C85        C
2              3         1       3         0  STON/O2. 3101282   7.9250  B96 B98        S
3              4         1       1         0              113803  53.1000     C123        S
4              5         0       3         0              373450   8.0500  B96 B98        S
..           ...       ...     ...       ...                 ...      ...      ...      ...
886          887         0       2         0              211536  13.0000  B96 B98        S
887          888         1       1         0              112053  30.0000      B42        S
888          889         0       3         2          W./C. 6607  23.4500  B96 B98        S
889          890         1       1         0              111369  30.0000     C148        C
890          891         0       3         0              370376   7.7500  B96 B98        Q

[891 rows x 12 columns]
```

Figure 3.1.5:Mode imputation for handling N-V in (a) train dataset, (b) testdataset

The drawback of mode imputation isthat it skews the histograms and also underestimates the variance in the data. It changes the statistical nature of the data. However, it is the most common method of data imputation.After treating N-Vwith any of these methods, the count or percentage of N-Vof the feature variables becomes zero.

OUTLIERS DETECTION

An outlier is an observation that is numerically distant from the rest of the data. The intuitive definition ofan outlier would be an observation that deviates so much fromother observations as to arouse suspicionsthat it was generated by different mechanisms[21].Therefore, outliersmay generate errors in the EDA process [22].Multiple reasons cause outliers to appear in a dataset, such as equipment malfunction, data misunderstood or formulated incorrectly, or an unclear response misread by the user. A typing error is an inaccuracy that happens whenerrors in interpretation occur when data is copied or transcribed, either manually or by a computer. Sampling frame errors also occur when a unit that is not part of the target population is unintentionally included in the sample.Outlierscan have deleterious effects on statistical analyses. First, they generally serve to increase error variance andreduce the power ofstatistical tests. Second, if non-randomly distributed, they can decrease normality (and inmultivariate analyses, violate assumptions ofsphericity and multivariate normality),altering the odds of making bothType I and Type II errors. Third, they can seriously bias or influence estimatesthat may be ofsubstantive interest [23]. So, detecting and handling outlier values in the dataset is crucial in order to make data pre-processing effective.Here,the Pandas preloaded data frame (the diabetes dataset) is used to detect the outliers that arose during the data analysis step.Some python code associated with importing libraries, dataset as well as the reading of the data is mentioned below:

```
import sklearn
from sklearn.datasets import load_diabetes
import pandas as pd
diabetics = load_diabetes()
column_name = diabetics.feature_names
df_diabetics = pd.DataFrame(diabetics.data)
df_diabetics.columns = column_name
print(df_diabetics.head())
```

	age	sex	bmi	bp	s1	s2	s3
0	0.038076	0.050680	0.061696	0.021872	-0.044223	-0.034821	-0.043401
1	-0.001882	-0.044642	-0.051474	-0.026328	-0.008449	-0.019163	0.074412
2	0.085299	0.050680	0.044451	-0.005670	-0.045599	-0.034194	-0.032356
3	-0.089063	-0.044642	-0.011595	-0.036656	0.012191	0.024991	-0.036038
4	0.005383	-0.044642	-0.036385	0.021872	0.003935	0.015596	0.008142

	s4	s5	s6
0	-0.002592	0.019907	-0.017646
1	-0.039493	-0.068332	-0.092204
2	-0.002592	0.002861	-0.025930
3	0.034309	0.022688	-0.009362
4	-0.002592	-0.031988	-0.046641

Fig 3.2.1: Dataset head view

The detection and removal of outliers can be done using visualization or statistical approach. Box plot, scatter plot, z-score and IQR (Inter Quartile Range) methods are used for detecting and removing outliers in the data set.

i. Box plot

With just a basic box and whiskers, it efficiently and effectively captures the data summary. Boxplot uses the 25th, 50th, and 75th percentiles to summarize sample datawith knowledge about quartiles, medians, and outliers of the data set. The following codes are used in this regard:

```
import seaborn as sns
sns.boxplot(df_diabetics['bmi'])
```

Infigure 3.2.2(a), the dotted points represent the outliers of the data set, which are removed in figure 3.2.2(b)using the Box plot.

```
def removal_box_plot(df, column, threshold):
removed_outliers = df[df[column] <=threshold]
sns.boxplot(removed_outliers[column])
plt.show()
return removed_outliers
threshold_value = 0.12
no_outliers = removal_box_plot(df_diabetics, 'bmi', threshold_value)
```

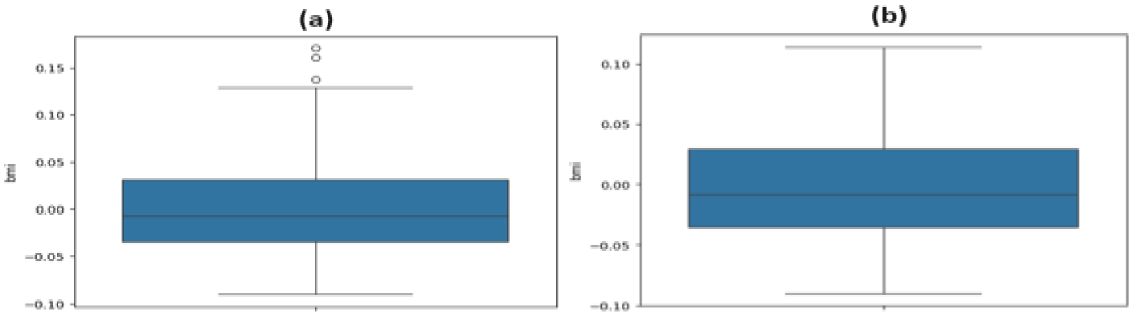

Figure 3.2.2: (a) Boxplot before outlier handle (b) Boxplot after outlier handle

ii. Scatterplot

The scatter plot is the collection of points that shows values for two variables. In figure 3.2.3,it can be seen thatmost of the data points are in the bottom left corner, but a few points arepresent near the top rightcorner of the graph. Those points in the top right corner can be regarded as outliers. Outlier detection using scatterplot is depicted in figure 3.2.3(a) using the following Python code:

```
fig, ax = plt.subplots(figsize=(6, 4))
ax.scatter(df_diabetics['bmi'], df_diabetics['bp'])
plt.show()
```

Now the removal of outliers using scatterplot needs some conditions, and the conditions are considered $bmi > 0.12 \; or \; bmi < 0.8$ after close observation in scatterplot. The following Python codes are used to remove the outliers using the scatterplot method, which is illustrated in figure 3.2.3(b).

```
outlier_indices = np.where((df_diabetics['bmi'] > 0.12) & (df_diabetics['bp'] < 0.8))
no_outliers = df_diabetics.drop(outlier_indices[0])
fig, ax_no_outliers = plt.subplots(figsize=(6, 4))
ax_no_outliers.scatter(no_outliers['bmi'], no_outliers['bp'])
plt.show()
```

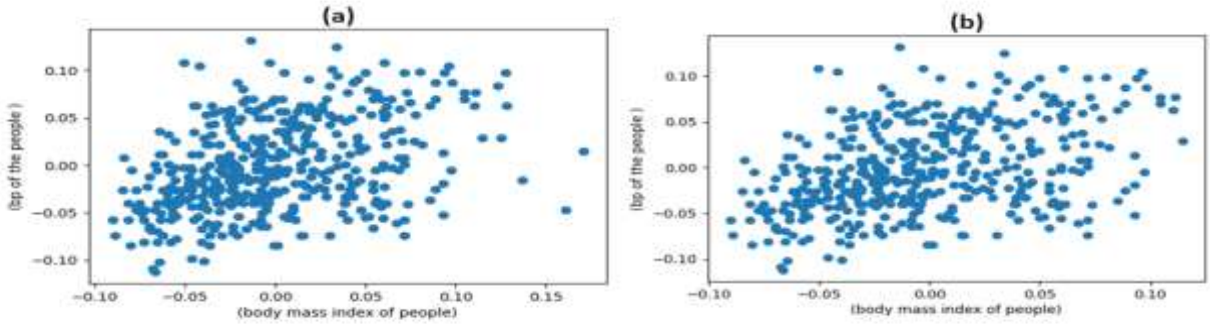

Figure 3.2.3: (a) Scatter-plot before outlier handle (b) Scatter-plot after outlier handle

iii.Z-score

Z-score describes any data point by finding their relationship with the standard deviation and mean of the group of data points. Z-score is finding the distribution of data where the mean is 0 and the standard deviation is 1. The following Python code is used to find the Z-score function defined in the Scipy library to detect the outliers. Figure3.2.4(a) illustrates the z-score corresponding to the 'age' variable in the dataset.

```
from scipy import stats
import numpy as np
z = np.abs(stats.zscore(df_diabetics['age']))
print(z)
```

To remove the outliers using z-score, a threshold value (here it is 2) needs to be set. "np.where()" is used to identify the position where the absolute Z score is greater than the specified threshold. It shows the position of outliers in any particular feature variable based on the Z-score criteria. The following Python code checks the data frame shape before and after removal using the Z-score method, and the output is depicted in 3.2.4(b).

```
threshold_z = 2
outlier_indices = np.where(z >threshold_z)[0]
no_outliers = df_diabetics.drop(outlier_indices)
print("Original DataFrame Shape:", df_diabetics.shape)
print("DataFrame Shape after Removing Outliers:", no_outliers.shape)
```

(a)

```
0       0.800500
1       0.039567
2       1.793307
3       1.872441
4       0.113172
          ...
437     0.876870
438     0.115937
439     0.876870
440     0.956004
441     0.956004
Name: age, Length: 442, dtype: float64
```

(b)

```
Original DataFrame Shape: (442, 10)
DataFrame Shape after Removing Outliers: (426, 10)
```

Figure 3.2.4: (a) Z-score value of the 'age', (b) Outliers before and after handling

iv.IQR

IQR is also known as the midspread or middle 50%, it is the measure of statistical dispersion, being equal to the difference between 75th and 25th percentiles, or between upper and lower quartiles, IQR = Q3 − Q1.Here we are calculating the interquartile range (IQR) for the ‘bmi’ column in the DataFrame. It first computes the Q1 and Q3 using the midpoint method, then calculates the IQR.

Q1 = np.percentile(df_diabetics['bmi'], 25, method='midpoint')
Q3 = np.percentile(df_diabetics['bmi'], 75, method='midpoint')
IQR = Q3 - Q1
print(IQR)

The above python code generates an IQR value equal to 0.06520763. The next stepis to define the base value in order to define the outlier, where $Upper\ bound\ =\ Q3\ +\ 1.5 * IQR$ and $lower\ bound\ =\ Q1 - 1.5 * IQR$.

upper_array = np.array(df_diabetics['bmi'] >= upper)
print("Upper Bound:", upper)
print(upper_array.sum())
lower_array = np.array(df_diabetics['bmi'] <= lower)
print("Lower Bound:", lower)
print(lower_array.sum())

The above code in Python returns the upper bound and lower bound equal to 0.128790 and -0.132040, respectively. This IQR value is used to remove outliers in any particular column. The corresponding Python code illustrated below:

df_diabetes.drop(index=upper_array, inplace=True)
df_diabetes.drop(index=lower_array, inplace=True)
print("New Shape: ", df_diabetes.shape)

Python returns after executing the above codes as: Old Shape: (442,10) and New Shape: (439,10).

DUPLICATE VALUE TREATMENT

Duplicated records that refer to the same entity with variations in their values represent a commonerror in datasets and are dealt with duplicate detection methods [24,25]. It can affect the quality, performance, and reliability of models. Depending on the application task, duplicate detection is referred to in the bibliography under different terms [26]. While integrating data from multiplesources, the amount of data increases,and data isalso duplicated [27] for various reasons, such as human errors, data entry mistakes, data merging or appending, web scraping, or data collection methods.Here, a dataset named ‘calls for service’ from thenew Orleans platform is used. Some Python code associated with importing libraries, datasets and reading the data is mentioned below:

import pandas as pd
twenty15_df = pd.read_csv("Calls_for_Service_2015.csv")
twenty15_df.head()

	NOPD_Item	Type_	TypeText	Priority	MapX	MapY	TimeCreate	TimeDispatch	TimeArrive	TimeClosed	Disposition	DispositionText	BLOCK_ADDRESS	Zip	PoliceDistrict	Location
0	A0000115	56	SIMPLE CRIMINAL DAMAGE	1D	3682553	532626	01/01/2015 12:00:34 AM	01/01/2015 01:24:47 AM	01/01/2015 01:41:20 AM	01/01/2015 01:41:30 AM	UNF	UNFOUNDED	007XX Orleans Ave	70116.0	8	(29.95850519, -90.06470624)
1	A0000215	21	COMPLAINT OTHER	1H	3682368	532820	01/01/2015 12:00:36 AM	NaN	01/01/2015 12:00:36 AM	01/01/2015 01:31:54 AM	NAT	Necessary Action Taken	Bourbon St & Orleans Ave	70116.0	8	(29.95904477, -90.06528204)
2	A0000415	94	DISCHARGING FIREARM	1A	3686245	546280	01/01/2015 12:01:47 AM	01/01/2015 01:20:19 AM	NaN	01/01/2015 01:32:38 AM	UNF	UNFOUNDED	Clematis St & Acacia St	70122.0	3	(29.99593586, -90.05256561)
3	A0000515	107	SUSPICIOUS PERSON	2A	3687521	537825	01/01/2015 12:02:22 AM	01/01/2015 12:08:17 AM	01/01/2015 12:13:19 AM	01/01/2015 12:24:40 AM	GOA	GONE ON ARRIVAL	026XX N Robertson St	70117.0	5	(29.97264816, -90.04883217)
4	A0000615	21	COMPLAINT OTHER	1H	3682082	529645	01/01/2015 12:02:44 AM	NaN	NaN	01/01/2015 01:22:17 AM	VOI	VOID	003XX Canal St	70130.0	8	(29.95032257, -90.06629572)

Figure 3.2.5: Head of the dataset

In handling duplicate data, the first step is to identify and quantify it. Depending on the type and structure of the data, different tools and techniques are available to detect duplicate data. Here we are using Pandas in Python to check duplicate rows or columns in a dataframe using the following method:

twenty15_df.duplicated()

Eliminating duplicate data is the easiest and most direct method of handling it. In addition to increasing the effectiveness and precision of models, this can lower noise and redundancy in the dataset. In Pandas,df.drop_duplicates() method is there to remove duplicate rows or columns, specifying the subset, keep, and inplace arguments. Figure 3.2.6 illustrates the dataset after the treatment.

twenty15_df.drop_duplicates()

	NOPD_Item	Type_	TypeText	Priority	MapX	MapY	TimeCreate	TimeDispatch	TimeArrive	TimeClosed	Disposition	DispositionText	BLOCK_ADDRESS	Zip	PoliceDistrict	Location
0	A0000115	56	SIMPLE CRIMINAL DAMAGE	1D	3682553	532626	01/01/2015 12:00:34 AM	01/01/2015 01:24:47 AM	01/01/2015 01:41:20 AM	01/01/2015 01:41:30 AM	UNF	UNFOUNDED	007XX Orleans Ave	70116.0	8	(29.95850519, -90.06470624)
1	A0000215	21	COMPLAINT OTHER	1H	3682368	532820	01/01/2015 12:00:36 AM	NaN	01/01/2015 12:00:36 AM	01/01/2015 01:31:54 AM	NAT	Necessary Action Taken	Bourbon St & Orleans Ave	70116.0	8	(29.95904477, -90.06528204)
2	A0000415	94	DISCHARGING FIREARM	1A	3686245	546280	01/01/2015 12:01:47 AM	01/01/2015 01:20:19 AM	NaN	01/01/2015 01:32:38 AM	UNF	UNFOUNDED	Clematis St & Acacia St	70122.0	3	(29.99593586, -90.05256541)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
432875	L3543015	62A	BURGLAR ALARM, SILENT	2C	[illegible]	530196	12/31/2015 11:59:01 PM	01/01/2016 12:01:41 AM	NaN	01/01/2016 12:33:52 AM	NAT	Necessary Action Taken	002XX Saint Charles Ave	70130.0	8	(29.95187716, -90.06873467)
432876	L3543115	94F	FIREWORKS	1A	3664996	518461	12/31/2015 11:59:11 PM	NaN	NaN	01/01/2016 12:48:27 AM	DUP	DUPLICATE	0068X Desmond St	70115.0	2	(29.92216334, -90.12061804)
432877	L3543215	94F	FIREWORKS	1A	3673236	541500	12/31/2015 11:59:32 PM	01/01/2016 01:34:48 AM	NaN	01/01/2016 01:36:45 AM	GOA	GONE ON ARRIVAL	Sherwood Forest Dr & City Park Ave (3H02)	70118.0	3	(29.96519616, -90.09082263)

432878 rows × 16 columns

Figure 3.2.6: Dataset after duplicate value treatment from the whole dataset

Column wise duplicate value treatment can also be implemented using the following Python code as illustrated in Figure 3.2.7.

twenty15_df.drop_duplicates(['TypeText'])

	NOPD_Item	Type_	TypeText	Priority	MapX	MapY	TimeCreate	TimeDispatch	TimeArrive	TimeClosed	Disposition	DispositionText	BLOCK_ADDRESS	Zip	PoliceDistrict	Location
0	A0000115	56	SIMPLE CRIMINAL DAMAGE	1D	3682553	532626	01/01/2015 12:00:34 AM	01/01/2015 01:24:47 AM	01/01/2015 01:41:20 AM	01/01/2015 01:41:30 AM	UNF	UNFOUNDED	007XX Orleans Ave	70116.0	8	(29.95850519, -90.06470624)
1	A0000215	21	COMPLAINT OTHER	1H	3682368	532820	01/01/2015 12:00:36 AM	NaN	01/01/2015 12:00:36 AM	01/01/2015 01:21:54 AM	NAT	Necessary Action Taken	Bourbon St & Orleans Ave	70116.0	8	(29.95904477, -90.06528204)
2	A0000415	94	DISCHARGING FIREARM	1A	3686245	546280	01/01/2015 12:01:47 AM	01/01/2015 01:20:19 AM	NaN	01/01/2015 01:32:38 AM	UNF	UNFOUNDED	Clematis St & Acacia St	70122.0	3	(29.99593586, -90.05256541)
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
353824	L2600015	88	CRIME AGAINST NATURE	1D	37369000	2512814	10/21/2015 08:22:18 PM	NaN	10/21/2015 08:22:18 PM	10/21/2015 09:34:11 PM	RTF	REPORT TO FOLLOW	Tricou St & Saint Claude Ave	70117.0	5	(0, 0)
396155	K3210015	62S	SAFE BURGLARY	1C	3681883	531587	11/28/2015 09:23:13 AM	11/28/2015 09:25:46 AM	11/28/2015 09:23:12 AM	11/28/2015 09:41:09 AM	RTF	REPORT TO FOLLOW	Conti St & Royal St	70130.0	8	(29.95567030, -90.06665815)
422403	L2440915	84	PANDERING	1H	3708730	551528	12/22/2015 08:53:56 AM	12/22/2015 09:02:23 AM	12/22/2015 09:09:53 AM	12/22/2015 12:54:46 PM	RTF	REPORT TO FOLLOW	0XXX Chef Menteur Hwy	70127.0	7	(30.01515612, -89.98130288)

151 rows × 16 columns

Figure 3.2.7: Dataset after duplicate value treatment from 'TypeText' feature

EDA

Exploratory Data Analysis (EDA) is a well-established statistical tradition that provides conceptual and computational tools for discovering patterns to foster hypothesis development and refinement [28].
It is an important initial stepfor any knowledge discovery process[29] in which data scientists interactively explore unfamiliar datasets by issuing asequence of analysis operations (e.g., filter, aggregation, andvisualization). The goal of EDA is to discover patterns in data. But in broad outline, it includes checks on data quality, the calculation of summary statistics, the plotting of appropriate graphs, and perhaps the use ofmore complicated data-analytic techniques such asprincipal component analysis.
This chapter analyzesEDA on the "bank marketing campaign" dataset. First,it is needed to refine the raw data through various stages like preprocessing, feature engineering, which includes data integration, analysis, cleaning, transformation and dimension reductionetc. The preprocessing methods were already discussed in previous sections of this chapter.The python code associated with importing libraries and datasets, as well as the reading and analysis of the data, is mentioned below:

import pandas as pd, numpy as np
import matplotlib.pyplot as plt, seaborn as sns
bdf= pd.read_csv("bank_marketing_updated_v1.csv")
bdf.head()

	customerid	age	salary	balance	marital	jobedu	targeted	default	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	response
0	1	58.0	100000	2143	married	management,tertiary	yes	no	yes	no	unknown	5	may, 2017	261 sec	1	-1	0	unknown	no
1	2	44.0	60000	29	single	technician,secondary	yes	no	yes	no	unknown	5	may, 2017	151 sec	1	-1	0	unknown	no
2	3	33.0	120000	2	married	entrepreneur,secondary	yes	no	yes	yes	unknown	5	may, 2017	76 sec	1	-1	0	unknown	no
3	4	47.0	20000	1506	married	blue-collar,unknown	no	no	yes	no	unknown	5	may, 2017	92 sec	1	-1	0	unknown	no
4	5	33.0	0	1	single	unknown,unknown	no	no	no	no	unknown	5	may, 2017	198 sec	1	-1	0	unknown	no

Figure 4.1: Dataset (bdf) Head view

In the dataset, there are multiple types of data types (numerical, categorical, ordinal etc.).We need to have ideas about those. Before proceeding to analysis, the preprocessing of the dataset is required for the missing value (in the 3.1 section) and handling of outliers (in the 3.2 section). If there are anyunnecessary features, drop them. In order to make the analytical process smooth, standardization of values needs to be performed, where we standardize units, scale values if required, remove extra characters etc.

UNIVARIATE ANALYSIS

Univariate analysis is thetype of quantitative data analysis. It's used to describe, summarize, and find patterns in the data from a single variable. Here,a univariate analysis of categorical unordered and categorical ordered datais observed. Unordered values like'marital status', 'job' whose analysis is given below:

```
bdf.marital.value_counts(normalize=True).plot.barh()
plt.show()
bdf.job.value_counts(normalize=True).plot.barh()
plt.plot()
```

The first two lines of the code are used to observe the horizontal bar plot of the "marital" feature. The last three lines generate the horizontal bar plot for the"job" feature. Figure 4.1.1(a) illustrates the bar plots where it is found that the married category has the largest response and the divorced category is the least class of the marital feature. Figure 4.1.1(b) also declares that the blue-collar and management classes have very high counts, while students and housemaidshave the least class in the "job" feature.

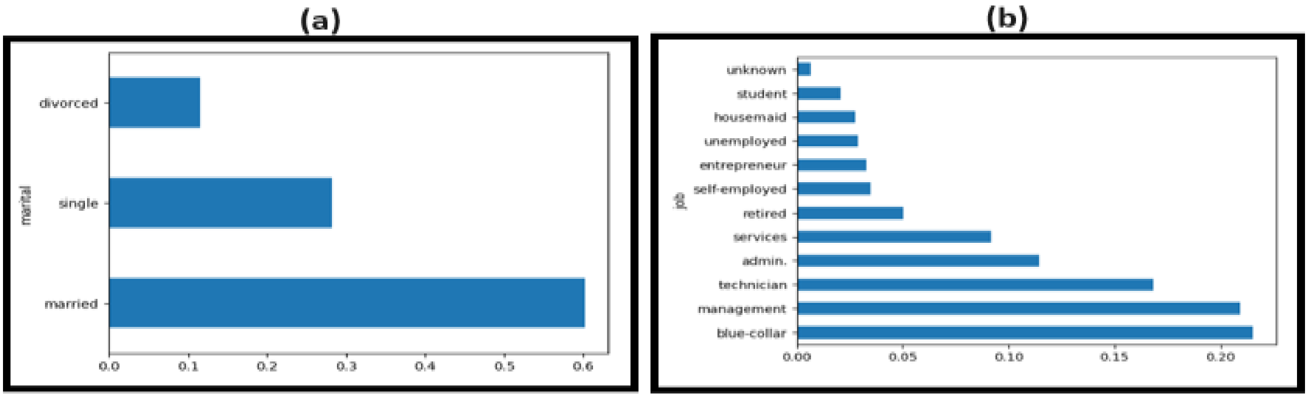

Figure 4.1.1: (a) horizontal bar chart for "marital" (b) horizontal bar chart for "job"

In the data set,some categorical ordered features are also present, such as "education", "poutcome" etc. which can be analyzed using univariate analysis. Here, pie-charts and bar chartshave been considered to analyze the categorical ordered features as depicted in figure 4.1.2(a-c).The following first two Python codes are used to get the piechart for the "education" feature and the last four codes generate the bar plots for the "poutcome" feature with and without the "unknown" class, respectively.

```
bdf.education.value_counts(normalize= True).plot.pie()
plt.show()
bdf.poutcome.value_counts(normalize= True).plot.bar()
plt.show()
bdf[-(bdf.poutcome=="unknown")].poutcome.value_counts(normalize= True).plot.bar()
plt.show()
```

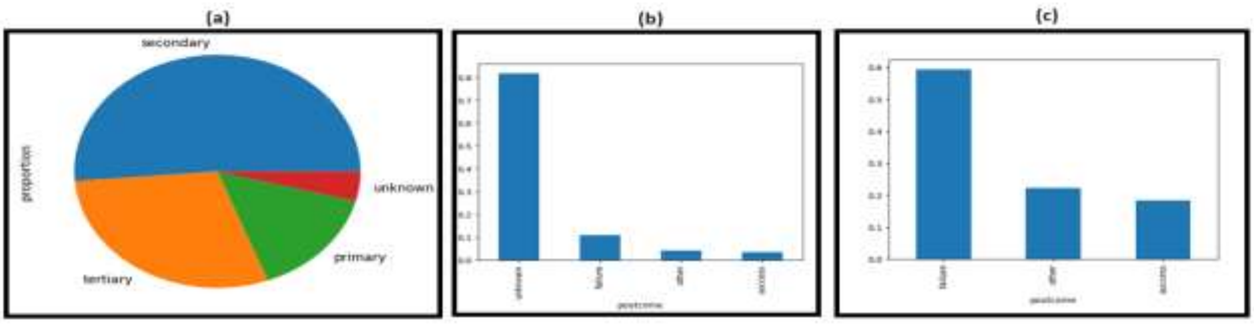

Figure 4.1.2: (a) Education pie-chart (b) poutcome with unknown bar chart (c) poutcome without unknown bar chart

BIVARIATE ANALYSIS

Bivariate analysis is one of the statistical analyses where the relation between two variables is observed (often denoted as x and y). It is used to find empirical relationships among bivariate data. Bivariate analysis is carried out by scatter plot, regression analysis, correlation coefficients etc. In this chapter, three types of bivariate analysis have been done. those are given below:

Numeric-Numeric Variable Analysis: Using a scatter plot the pattern of dependencies of two numeric values (balance and salary) is shown in Figure 4.2.1(a) and the corresponding Python codes are given below:

```
plt.scatter(bdf.salary, bdf.balance)
plt.show()
```

Numeric-CategoricVariableAnalysis: The dependencies of numeric (salary) and categorical (response) values aredepicted in figure 4.2.1(b) using a box plot and the Python code for the box plot is illustrated below.

```
sns.boxplot(data=bdf,x="response", y="salary")
plt.show()
```

Categorical-Categorical Variable Analysis: The bivariate analysis of two categorical variable'marital status' and'response rate' is shown in figure 4.2.1(c). First, it is needed to create a temporary variable of numeric data type where the response rate "yes" =1 and "no" =0, then the calculation of the mean of that temporary variable with different marital status categories is done. Here, a bar graph is used to show the dependencies of marital status with the average value of that temporary variable. Thefollowing three Python codes are used to do the bivariate analysis between marital status and response flag mean.

```
bdf["response_flag"]=np.where(bdf.response=="yes", 1, 0)
bdf.groupby(["marital"])["response_flag"].mean().plot.barh()
plt.show()
```

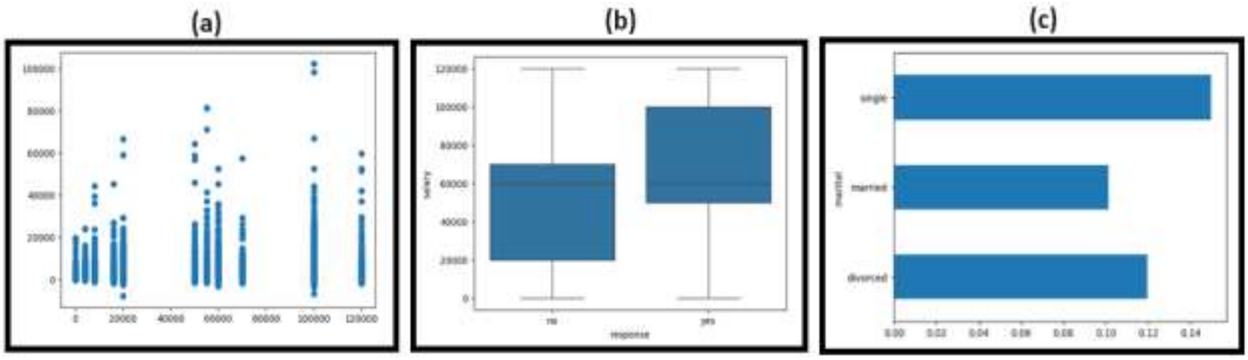

Figure 4.2.1:Bivariate analysis on (a) both numeric variables(b) numeric-categoric variable (c) both categoricvariables

MULTIVARIATE ANALYSIS

A statistical method for comprehending the connections between several variables at once is multivariate analysis [30]. Deeper insights beyond those obtained from univariate or bivariate analysis alone can be obtained by enabling researchers to examine intricate connections and patterns within their data sets. Multivariate analysis allows for the simultaneous examination of numerous variables, allowing for the detection of underlying patterns and trends. This facilitates more informed decision-making in a variety of domains, including economics, psychology, and biology.Here authors code for heatmaps using correlation matrix and pivot table which are shown below:

```
sns.heatmap( bdf[["salary","balance", "age"]].corr(), annot= True, cmap= "Reds")
plt.show()
res=pd.pivot_table(data=bdf, index="education", columns="marital", values="response_flag")
sns.heatmap(res, annot= True, cmap="RdYlGn")
plt.show()
res=pd.pivot_table(data=bdf, index="job", columns="marital", values="response_flag")
sns.heatmap(res, annot= True, cmap="RdYlGn")
plt.show()
```

The first two lines of code generate a heatmap using the correlation matrix among "salary", "balance" and "age" features. It has been observed that negligible correlations exist among the features considered. Negligible correlation insights are a good choice of feature for model building using machine learning algorithms. The heatmap is depicted in figure 4.3.1(a).

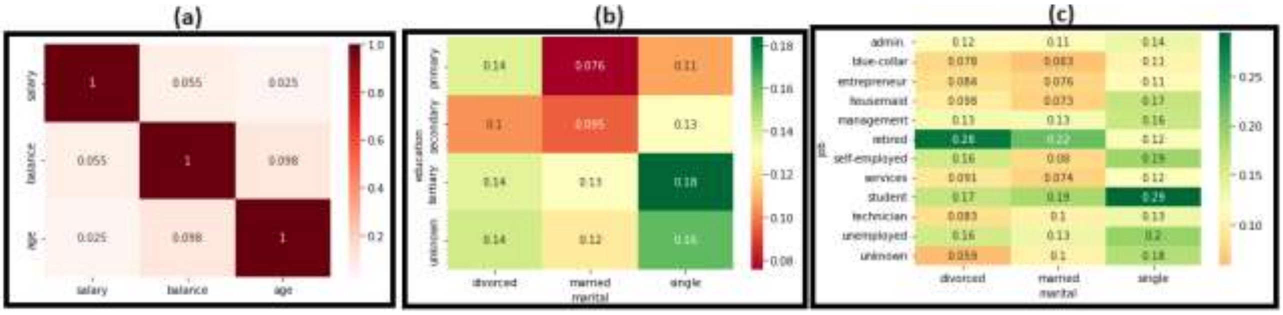

Figure 4.3.1: (a) heatmap for “salary”, “balance”, “age” features (b) heatmap for “education”, “marital”, “responseflag” (c) heatmap for “job”, “marital”, “responseflag”

Figure 4.3.1(b) illustrates the response flag values for all existing combinations between “education” and “marital”features, which can be found by coding the third, fourth and fifth lines of code mentioned above. There are some observations in terms of response flag found ranging from 10% to 18% for different combinations of marital and educational status. The last three lines of the code represent figure 4.3.1(c) where the heatmap generates between the combinations of marital and job status based on the response flag value. The response values lie between 0% to 30%.

PCA

Principal Component Analysis [32] is the general name for a technique that uses sophisticated underlying mathematical principles to transform a number of possibly correlatedvariables into a smaller number of variables called principal components.PCA is a multivariate technique that analyzes a data table in whichobservations are described by several inter-correlated quantitative dependent variables. Its goal is toextract important information from the statistical data, represent it as a set of new orthogonalvariables called principal components, and display the pattern of similarity between the observationsand the variables as points in spot maps.The central idea of principal componentanalysis is to reduce the dimensionality of a data set in which there are a large number of interrelatedvariables, while retaining as much as possible of the variation present in the data set.The process involves converting the original variables into a new set known as the principal components (PCs), which are uncorrelated and arranged so that the first few maintain the majority of the variation seen in all of the original variables.In this chapter, analysis of the principle component is done on the“iris” dataset, and the following code is related to reading and analyzing data, importing libraries, and datasets. The head of the dataset is depicted in figure 5.1.

```
from sklearn.datasets import load_iris
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
X = iris.data
y = iris.target
df = pd.DataFrame(X,columns=iris.feature_names)
df['Label']=y
df['Species']=df['Label'].map({0: 'setosa', 1: 'versicolor', 2: 'virginica'})
df.head()
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Label	Species
0	5.1	3.5	1.4	0.2	0	setosa
1	4.9	3.0	1.4	0.2	0	setosa
2	4.7	3.2	1.3	0.2	0	setosa
3	4.6	3.1	1.5	0.2	0	setosa
4	5.0	3.6	1.4	0.2	0	setosa

Figure 5.1: Head view of the iris detaset

Identifying the directions (principal components) that the data most frequently fluctuates along is how Principal Component Analysis (PCA) operates.PCA requires that the data be centered at 0. Centering the data at 0 (mean-centering) is important in PCA for some reasons, such as the removal of the mean, covariance calculation etc. But using sklearn, this could be done automatically.

```
pca = PCA()
X_pca = pca.fit_transform(X)
pca_df = pd.DataFrame(X_pca,columns=['PC1','PC2','PC3','PC4'])
df = pd.merge(df, pca_df, right_index=True, left_index=True)
```

To determine how much information each principal component retains from the original data, it is essential to look at the variance explained by each component. The significance of each component in capturing the variability of the dataset is ascertained by looking at the amount of variation that each primary component explains.In this case, nearly all of the variance (92.5%) is explained by PC1 alone.

```
print('Explained Variance Ratio')
for i in range(4):
print('PC{}: {}'.format(i+1,pca.explained_variance_ratio_[i]))
```

Visualizing data in one dimension in PCA helps to understand how much variance is captured by a single component and how the data points are distributed along this component. Now we are using PC1 to visualize the data in one dimension. From the strip plot depicted in figure 5.2, it is shown that the setosa category can be entirely distinguished from the other two by this component. Although the other two species are mostly separable, they experience some significant overlap, which could make classification difficult with PC1 alone.The corresponding Python code is written below:

```
sns.stripplot(x="PC1", y="Species", data=df,jitter=True)
plt.title('Iris Data Visualized in One Dimension');
```

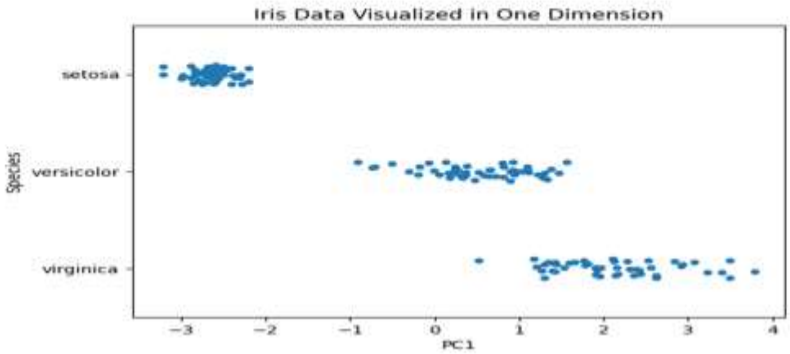

Figure 5.2: One dimension iris data with PCA1 only

In order to explain more variance,the required number of principal components should be known.

```
precent_of_variance_explained = .95
pca = PCA(n_components=precent_of_variance_explained)
pca_data = pca.fit_transform(X)
print("{} Principal Components are required to explain {} of the variation in this data.".format(pca.n_components_,precent_of_variance_explained))
```

Above python code returns that the 2 PCA are required to explain 0.95 % variation in the iris dataset. By plotting the correlation between the number of primary components and the variance explained, we are able to verify that two is a natural number of dimensions for our data.Below the lines, finally visualize the dataset with only 2 dimensions.Anlmplot can be seen in figure 5.3.

```
sns.lmplot(x='PC1',y='PC2',data=df,hue='Species',fit_reg=False)
plt.title('Iris Data Visualized in Two Dimensions');
plt.show()
```

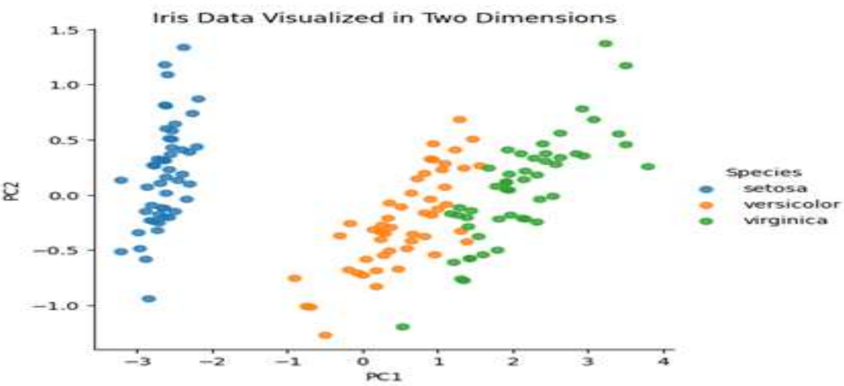

Figure 5.3: Two-dimension iris data with PCA1 and PCA2 only

Conclusion

This study leads us through an insightful excursion acrossthe essential stages of data analysis, from initial preprocessing to exploratory data analysis (EDA) and culminating in the powerful dimensionality reduction technique known as principal component analysis (PCA). Real-world data tend to be incomplete, inconsistent, noisy and missing. Thus data preprocessing is one of the important phases for data analysis. At the initial stage of the excursion, authors get into the preprocessing phase, which is important yet often overlooked.To prepare the dataset for analysis, authors clean up the data, deal with missing values and outlier treatments, and standardize or normalize our features. After preprocessing, authors commence the exploratory phase, where the chapter uncover hidden insights, patterns, and anomalies.Authors are able to make more informed decisions and generate assumptions by developing a more thorough comprehension of the structure of the data through statistical summaries, correlation analysis, and visualizations. Thenext step is principal component analysis, where authorsidentify its fundamental structure and minimize its dimensionality while keeping as much information as possible. PCA enables to extract the key features of the dataset.To sum up, "Data Odyssey" provides clarity and purpose as it navigates the complex terrain of PCA, EDA, and preprocessing, illuminating the way to successful data analysis. By embracing these fundamental methods, authors start on an expedition of exploration, discovering the unrealized potential of the data and redirecting the direction toward deeper insight and useful intelligence.

References

1. Goar, V., & Yadav, N. S. (2024). Foundations of machine learning. In *Intelligent Optimization Techniques for Business Analytics* (pp. 25-48). IGI Global.
2. D. Lenat, E. Feigenbaum: On the Thresholds of Knowledge. Artificial Intelligence 47(1-3), 1991
3. García, S., Luengo, J., & Herrera, F. (2015). Data preprocessing in data mining (Vol. 72, pp. 59-139). Cham, Switzerland: Springer International Publishing.
4. García, S., Luengo, J., & Herrera, F. (2016). Tutorial on practical tips of the most influential data preprocessing algorithms in data mining. Knowledge-Based Systems, 98, 1-29.
5. Mishra, P., Biancolillo, A., Roger, J. M., Marini, F., & Rutledge, D. N. (2020). New data preprocessing trends based on ensemble of multiple preprocessing techniques. TrAC Trends in Analytical Chemistry, 132, 116045.
6. Mayer-Schnberger, V., &Cukier, K. (2013). Big data: A revolution that will transform how we live, work, and think. Choice Rev. Online, 50, 50-6804.
7. García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., & Herrera, F. (2016). Big data preprocessing: methods and prospects. Big data analytics, 1, 1-22.
8. Ramírez-Gallego, S., Krawczyk, B., García, S., Woźniak, M., & Herrera, F. (2017). A survey on data preprocessing for data stream mining: Current status and future directions. Neurocomputing, 239, 39-57.
9. Famili, A., Shen, W. M., Weber, R., &Simoudis, E. (1997). Data preprocessing and intelligent data analysis. Intelligent data analysis, 1(1), 3-23.
10. G. Noriega and S. Pasupathy, Application of Kalman Filtering to Real-Time Preprocessing of Geophysical Data, IEEE Transactions on Geoscience and Remote Sensing, 30, 5 (1980), 897-910.
11. J.E Davis, B. Bakshik, K. Kosanovich, and M. Piovoso, Process Monitoring, Data Analysis and Data Interpretation, Proceedings of the Intelligent Systems in Process Engineering Conference, Snowmass, co (1995), 1-12
12. Z. Duszak and W.W. Loczkodaj, Using Principal Component Transformation in Machine Learning, Proceedings of International Conference on Systems Research, Informatics and Cybernetics, Baden-Baden, Germany (1994), 125-129.
13. Maćkiewicz, A., &Ratajczak, W. (1993). Principal components analysis (PCA). Computers & Geosciences, 19(3), 303-342.
14. Chatfield, C. (1986). Exploratory data analysis. European journal of operational research, 23(1), 5-13.
15. Kotsiantis, S. B., Kanellopoulos, D., &Pintelas, P. E. (2006). Data preprocessing for supervised leaning. International journal of computer science, 1(2), 111-117.
16. C. M. Teng. Correcting noisy data. In Proc. 16th International Conf. onMachine Learning, pages 239–248. San Francisco, 1999.
17. Qin, Y.S., et al. (2007). Semi-parametric Optimizationfor Missing Data Imputation. AppliedIntelligence, 2007,27(1): 79-88.
18. Zhang, C.Q., et al., (2007). An Imputation Method forMissing Values. PAKDD, LNAI, 4426, 2007: 1080-1087.
19. Somasundaram, R. S., &Nedunchezhian, R. (2011). Evaluation of three simple imputation methods for enhancing preprocessing of data with missing values. International Journal of Computer Applications, 21(10), 14-19.
20. Thomas Lumley, "Missing data", A Lecture Note, BIOST570, 2005-11-9.

21. Hawkins, D. M. (1980). Identification of outliers (Vol. 11). London: Chapman and Hall.
22. Ghosh, D., & Vogt, A. (2012, July). Outliers: An evaluation of methodologies. In Joint statistical meetings (Vol. 12, No. 1, pp. 3455-3460).
23. Osborne, J. W., &Overbay, A. (2019). The power of outliers (and why researchers should always check for them). Practical Assessment, Research, and Evaluation, 9(1), 6.
24. Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. 2007. Duplicate record detection: A survey.IEEE Trans. Knowl. Data Eng. 19, 1 (2007), 1–16.
25. Koumarelas, I., Jiang, L., & Naumann, F. (2020). Data Preparation for Duplicate Detection. Journal of Data and Information Quality, 12(3), 1–24.
26. Peter Christen. 2012. Data Matching—Concepts and Techniques for Record Linkage, Entity Resolution, and DuplicateDetection. Springer Data-Centric Systems and Applications.
27. Tamilselvi, J. J., &Gifta, C. B. (2011). Handling duplicate data in data warehouse for data mining. International Journal of Computer Applications, 15(4), 7-15.
28. Behrens, J. T. (1997). Principles and procedures of exploratory data analysis. Psychological methods, 2(2), 131.
29. Marbán, Ó., Mariscal, G., & Segovia, J. (2009). A data mining & knowledge discovery process model. In Data mining and knowledge discovery in real life applications. IntechOpen.
30. Mengual-Macenlle, N., Marcos, P. J., Golpe, R., & González-Rivas, D. (2015). Multivariate analysis in thoracic research. Journal of thoracic disease, 7(3), E2.
31. Milo, T., &Somech, A. (2020, June). Automating exploratory data analysis via machine learning: An overview. In Proceedings of the 2020 ACM SIGMOD international conference on management of data (pp. 2617-2622).
32. Richardson, M. (2009). Principal component analysis. URL: http://people. maths. ox. ac. uk/richardsonm/SignalProcPCA. pdf (last access: 3.5. 2013). AlešHladnik Dr., Ass. Prof., Chair of Information and Graphic Arts Technology, Faculty of Natural Sciences and Engineering, University of Ljubljana, Slovenia ales. hladnik@ ntf. uni-lj. si, 6(16), 4.
33. Mishra, S. P., Sarkar, U., Taraphder, S., Datta, S., Swain, D., Saikhom, R., ... & Laishram, M. (2017). Multivariate statistical data analysis-principal component analysis (PCA). International Journal of Livestock Research, 7(5), 60-78.