Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Department of Computer Science and Engineering (Data Science)

Lab Manual

Subject: Foundations of Data Analysis Laboratory (DJ19DSL303)

Semester: III

Experiment 7

(Data Pre-processing)

NAME: Dev Patel **SAP ID: 60009200016**

Batch: K/K1 **Date: 11/01/2022**

Aim: Perform Data cleaning on a given dataset.

Theory: Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities for data to be duplicated or mislabelled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct. There is no one absolute way to prescribe the exact steps in the data cleaning process because the processes will vary from dataset to dataset.

After cleansing, a data set should be consistent with other similar data sets in the system. The inconsistencies detected or removed may have been originally caused by user entry errors, by corruption in transmission or storage, or by different data dictionary definitions of similar entities in different stores. Data cleaning differs from data validation in that validation almost invariably means data is rejected from the system at entry and is performed at the time of entry, rather than on batches of data.

Missing Values may involve removal of those entries (Usually if number of missing values is low and/or the field is important for analysis), estimated (If high correlation exists, low number of missing values), or that field/column may be dropped (large number of missing values and/or)

1. Remove duplicate or irrelevant observations
2. Fix structural errors
3. Filter unwanted outliers
4. Handle missing data
5. Validate

Dataset: Reservations.csv

Perform the following if required:

1. Remove Duplicate Values: All values containing the same values in all the columns have to be removed. As they duplicate values can lead to faulty predictions and analysis.
2. Imputation of missing values: Missing values can lead to data corruption and failure to record data. Hence, such values such be imputed from our dataset.
3. Remove outliers: Outliers have to be removed because they are unusual values that can distort statistical analysis and violate the predictions and assumptions made from them.
4. Correlation analysis: Correlation is the closeness of the relationship between two or more variables. We remove highly correlated features due storage and speed concerns.
5. Data Transformation: Data transformation is the process of changing the format, structure, or values of data. It acts as a power booster for the analytics process and helps you make better data-driven decisions.

Result:

We have performed the above 5 steps for data pre-processing. This is the link to the final dataset obtained as a result of performing the exercise.

https://github.com/d9vp/FDA/blob/main/Reservations%20Processed%20Data.xlsx

This was performed with the following code:

Code:

```
import pandas as pd
import numpy as np

from google.colab import drive
drive.mount('/content/drive')

df = pd.read_excel('/content/drive/MyDrive/Datasets/Reservations.xlsx')
df.head()
```

	hotel	is_canceled	lead_time	arrival_date_year	arrival_date_month	arrival_dat
0	Resort Hotel	0	342	2015	July	
1	Resort Hotel	0	737	2015	July	
2	Resort Hotel	0	7	2015	July	
3	Resort Hotel	0	13	2015	July	
4	Resort Hotel	0	14	2015	July	

1. Remove Duplicate Values

```
len(df.drop_duplicates())

# We drop the rows with same values in all its columns df
= df.drop_duplicates()

# Info of the modified, updated dataset df.info()
```

```
<class 'pandas.core.frame.DataFrame'> Int64Index:
87396 entries, 0 to 119389
Data columns (total 32 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
0   hotel                           87396 non-null  object
1   is_canceled                     87396 non-null  int64
2   lead_time                       87396 non-null  int64
3   arrival_date_year               87396 non-null  int64
4   arrival_date_month              87396 non-null  object
5   arrival_date_week_number        87396 non-null  int64
6   arrival_date_day_of_month       87396 non-null  int64
```

```
7   stays_in_weekend_nights         87396 non-null  int64
8   stays_in_week_nights            87396 non-null  int64
9   adults                          87396 non-null  int64
10  children                        87392 non-null  float64
11  babies                          87396 non-null  int64
12  meal                            87396 non-null  object
13  country                         86944 non-null  object
14  market_segment                  87396 non-null  object
15  distribution_channel            87396 non-null  object
16  is_repeated_guest               87396 non-null  int64
17  previous_cancellations          87396 non-null  int64
18  previous_bookings_not_canceled  87396 non-null  int64
19  reserved_room_type              87396 non-null  object
20  assigned_room_type              87396 non-null  object
21  booking_changes                 87396 non-null  int64
22  deposit_type                    87396 non-null  object
23  agent                           75203 non-null  float64
24  company                         5259 non-null   float64
25  days_in_waiting_list            87396 non-null  int64
26  customer_type                   87396 non-null  object
27  adr                             87396 non-null  float64
28  required_car_parking_spaces     87396 non-null  int64
29  total_of_special_requests       87396 non-null  int64
30  reservation_status              87396 non-null  object  31  reservation_status_date
    87396 non-null  object  dtypes: float64(4), int64(16), object(12) memory usage:
    22.0+ MB
```

2. Imputation of missing values

```
df.isnull().sum()
```

```
hotel                           0
is_canceled                     0
lead_time                       0
arrival_date_year               0
arrival_date_month              0
arrival_date_week_number        0
arrival_date_day_of_month       0
stays_in_weekend_nights         0
stays_in_week_nights            0
adults                          0
children                        4
babies                          0
meal                            0
country                       452
market_segment                  0
distribution_channel            0
is_repeated_guest               0
previous_cancellations          0
previous_bookings_not_canceled  0
reserved_room_type              0
assigned_room_type              0
booking_changes                 0
deposit_type                    0
agent                       12193
company                     82137
```

```
days_in_waiting_list              0
customer_type                     0
adr                               0
required_car_parking_spaces       0
total_of_special_requests         0
reservation_status                0
reservation_status_date           0
dtype: int64
```

```
# Agent and company seemed insignificant so they have been dropped.

df.drop(['agent', 'company'], axis=1, inplace=True) df.isnull().sum()
```

```
hotel                             0
is_canceled                       0
lead_time                         0
arrival_date_year                 0
arrival_date_month                0
arrival_date_week_number          0
arrival_date_day_of_month         0
stays_in_weekend_nights           0
stays_in_week_nights              0
adults                            0
children                          4
babies                            0
meal                              0
country                         452
market_segment                    0
distribution_channel              0
is_repeated_guest                 0
previous_cancellations            0
previous_bookings_not_canceled    0
reserved_room_type                0
assigned_room_type                0
booking_changes                   0
deposit_type                      0
days_in_waiting_list              0
customer_type                     0 adr
0 required_car_parking_spaces       0
total_of_special_requests         0
reservation_status                0
reservation_status_date           0
dtype: int64
```

```
df['country'].value_counts()
```

```
PRT    27453
GBR    10433
FRA     8837
ESP     7252
DEU         5387
...
KIR        1
CYM        1
SMR        1
AIA        1
GLP        1
Name: country, Length: 177, dtype: int64
```

```
# NaN values in the country column have been filled with the most frequently occurring va

df.fillna(df['country'].value_counts().index[0] , inplace=True)

# Binning with zero for any remaining NaN values

df.fillna(0) df.isnull().sum()
```

```
hotel                           0
is_canceled                     0
lead_time                       0
arrival_date_year               0
arrival_date_month              0
arrival_date_week_number        0
arrival_date_day_of_month       0
stays_in_weekend_nights         0
stays_in_week_nights            0
adults                          0
children                        0
babies                          0
meal                            0
country                         0
market_segment                  0
distribution_channel            0
is_repeated_guest               0
previous_cancellations          0
previous_bookings_not_canceled  0
reserved_room_type              0
assigned_room_type              0
booking_changes                 0
deposit_type                    0
days_in_waiting_list            0
customer_type                   0
adr                             0
required_car_parking_spaces     0
total_of_special_requests       0
reservation_status              0
reservation_status_date         0
dtype: int64
```

3. Remove Outliers

```
from scipy import stats
import numpy as np

# We remove outliers having abnormal values in number of days in waiting lists and specia
df=df[(np.abs(stats.zscore(df['days_in_waiting_list'])) < 3)]



df = df[(np.abs(stats.zscore(df['total_of_special_requests'])) < 3)] df.shape
```

```
(86181, 30)
```

4. Correlation Analysis

```
corr_matrix = df.corr(method='pearson').abs()

# We select upper triangle of correlation matrix upper =
corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.bool))

# To find correlation greater than 0.75 in features
to_drop = [column for column in upper.columns if any(upper[column] > 0.75)] print(to_drop)

# We did not find any highly correlated columns,hence, we do not remove any columns.
```

```
[]
```

5. Data Transformation

```
# We can reduce the number of attributes by clubbing together babies and number of childre
df = df[df['children'] != 'PRT'] df['total_children'] = df['children'].astype(int)
+df['babies'].astype(int) df.drop(['children', 'babies'], axis=1, inplace=True) df.info()
```

```
<class 'pandas.core.frame.DataFrame'> Int64Index:
86177 entries, 0 to 119389
Data columns (total 29 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
0   hotel                           86177 non-null  object
1   is_canceled                     86177 non-null  int64
2   lead_time                       86177 non-null  int64
3   arrival_date_year               86177 non-null  int64
4   arrival_date_month              86177 non-null  object
5   arrival_date_week_number        86177 non-null  int64
6   arrival_date_day_of_month       86177 non-null  int64
7   stays_in_weekend_nights         86177 non-null  int64
8   stays_in_week_nights            86177 non-null  int64
9   adults                          86177 non-null  int64
10  meal                            86177 non-null  object
11  country                         86177 non-null  object
12  market_segment                  86177 non-null  object
13  distribution_channel            86177 non-null  object
14  is_repeated_guest               86177 non-null  int64
15  previous_cancellations          86177 non-null  int64
16  previous_bookings_not_canceled  86177 non-null  int64
17  reserved_room_type              86177 non-null  object
18  assigned_room_type              86177 non-null  object
19  booking_changes                 86177 non-null  int64
20  deposit_type                    86177 non-null  object
21  days_in_waiting_list            86177 non-null  int64
22  customer_type                   86177 non-null  object
23  adr                             86177 non-null  float64
24  required_car_parking_spaces     86177 non-null  int64
25  total_of_special_requests       86177 non-null  int64
26  reservation_status              86177 non-null  object
```

```
27  reservation_status_date          86177 non-null  object    28  total_children
    86177 non-null  int64   dtypes: float64(1), int64(16), object(12) memory usage:
    19.7+ MB
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWarni
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
  This is separate from the ipykernel package so we can avoid doing imports until
/usr/local/lib/python3.7/dist-packages/pandas/core/frame.py:4174:  SettingWithCopyWar
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/u
errors=errors,
```

```
# We can make one column of date, month and year of arrival and drop the three individual

df['arrival_date'] = df['arrival_date_day_of_month'].astype(str)+ " " + df['arrival_date_m
df.drop(['arrival_date_day_of_month', 'arrival_date_month' , 'arrival_date_year' , 'arriva
```

```
# We convert arrival_date attribute to a date-time object

df['arrival_date'] = pd.to_datetime(df['arrival_date']) df.head()
```

	hotel	is_canceled	lead_time	stays_in_weekend_nights	stays_in_week_nights	ad
1	Resort Hotel	0	737	0	0	
2	Resort Hotel	0	7	0	1	
3	Resort Hotel	0	13	0	1	
4	Resort Hotel	0	14	0	2	

```
df.to_excel("Reservations Processed Data.xlsx") # Downloading the processed dataframe
```

56s completed at 7:25 PM