VARUVAN VADIVELAN INSTITUTE OF TECHNOLOGY

IBM : NAAN MUDHALVAN

PHASE : 3

DEVELOPMENT PART 1

TECHNOLOGY : DATA ANALYTICS

PROJECT TITLE : COVID-19 CASES ANALYSIS

Data Preprocessing :

- ✓ Some factors reduce the classification performance in artificial intelligence-based diagnostic systems realized by using biomedical datasets. For this reason, various data preprocessing techniques suitable for datasets are used to increase the classification performance.

- ✓ A data analysis directly depends on both the data preprocessing step and the techniques chosen for this purpose. 24 While the importance of data preprocessing is so evident, it is very important to find the most suitable data preprocessing techniques for the study to be carried out.

- ✓ In this study carried out in this direction, certain data preprocessing techniques on blood tests of infected and noninfected individuals were analyzed and the effect of these techniques on the diagnosis of COVID-19 was examined.

In this study, the dataset was applied to encode categorical values with one-hot encoding, min-max feature scaling in the range of [0–1], filling the missing data using KNN and MICE methods and data balancing with SMOTE method.

Some stems is there in preprocessing :

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data.
- Splitting dataset into training and test set
- Feature scaling

Get the Dataset :

To create a Jupyter model , the first thing required is a dataset as a jupyter model completely works on data. The Collected data for a particular problem in a proper format is Known as the Dataset.

PROGRAM :

```
 # Data Preprocessing

import numpy as np
import pandas as pd
```

```
# Data Analysis

import plotly.express as px
import missingno as msno

# Feature Selection

import scipy.stats as stats
from scipy.stats import chi2_contingency

# Data Modeling

from sklearn.model_selection import train_test_split
from imblearn.under_sampling import RandomUnderSampler
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier

# Model Evaluation & saving the model

from sklearn.metrics import classification_report, confusion_matrix, ConfusionMatrixDisplay, recall_score, accuracy_score, precision_score, f1_score
import pickle
```

Reading The Data

```
# Loading the Data
```

```
data = pd.read_csv("../input/covid19-dataset-for-year-
2020/covid_data_2020-2021.csv")

data.head()
```

```
test_date    cough        fever sore_throat        shortness_of_breath
        head_ache corona_result     age_60_and_above     gender
        test_indication

0       2021-10-11 0      0      0      0      0      Negative    Yes    female
        Other

1       2021-10-11 0      0      0      0      0      Negative    Yes    male  Other

2       2021-10-11 0      0      0      0      0      Negative    No     female
        Other

3       2021-10-11 0      0      0      0      0      Negative    Yes    female
        Other

4       2021-10-11 0      0      0      0      0      Negative    Yes    female
        Other
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>

RangeIndex: 5861480 entries, 0 to 5861479

Data columns (total 10 columns):

 #   Column              Dtype

---  ------              -----

 0   test_date           object

 1   cough               int64

 2   fever               int64

 3   sore_throat         int64

 4   shortness_of_breath  int64

 5   head_ache           int64

 6   corona_result       object

 7   age_60_and_above    object
 8   gender              object
 9   test_indication      object
dtypes: int64(5), object(5)
memory usage: 447.2+ MB
Dataset has 5861480 records and 10 features.
This is a Binary Classification Problem.
```

```
# Checking the levels for categorical features

def show(data):
  for i in data.columns[1:]:
    print("Feature: {} with {} Levels".format(i,data[i].unique()))

show(data)
```

```
Feature: cough with [0 1] Levels
Feature: fever with [0 1] Levels
Feature: sore_throat with [0 1] Levels
Feature: shortness   of  breath   with [0 1] Levels
Feature: head ache with [0 1] Levels
Feature: corona result with ['Negative' 'Positive'] Levels
Feature: age 60 and above with ['Yes' 'No'] Levels
Feature: gender with ['female' 'male'] Levels
Feature: test indication with ['Other' 'Contact with confirmed' 'Abroad'] Levels
```

Target Feature is Corona result.

Data is completely Categorical except test date feature.

OUTPUT :

Classification Report for Train Data

```
          precision    recall  f1-score   support

       0       1.00      0.96      0.98    188938
       1       0.94      1.00      0.97    113501

    accuracy                           0.98    302439
   macro avg       0.97      0.98      0.98    302439
weighted avg       0.98      0.98      0.98    302439

------------------------------------------------------------------------
Recall on Train Data:  0.9994
Specificity on Train Data:  0.9641
Accuracy on Train Data:  0.9773
Precision on Train Data:  0.9435
F1 Score on Train Data:  0.9707
------------------------------------------------------------------------

Classification Report for Test Data

          precision    recall  f1-score   support

       0       1.00      0.96      0.98     81097
       1       0.94      1.00      0.97     48520

    accuracy                           0.98    129617
   macro avg       0.97      0.98      0.98    129617
weighted avg       0.98      0.98      0.98    129617
```

Importing Libraries :

In orders to perform data preprocessing using python, we need to import some predefined python libraries . These libraries are used to perform some specific jobs. These are three specific libraries that we will use for data preprocessing.

NUMPY :

Numpy python library is usesd for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in python. It also supports to add large, multidimensional arrays and matrices. So, in python, we can import it as:

import numpy as np

Here we have used nm, which is a short name for Numpy, and it will be used in the whole program

Matplotlib:

The second library is matplotlib, which is a python 2D plotting library, and with this library, we need to import a sub- library Pyplot. This library is used to plot any type of chats in python for the code .

Import matplotlib pyplot as mtp

Pandas :

The last library is the Pandas library , which is one the most famous Python libraries and used for importing and managing and the Dataset .

It's an open-sources data manipulation and analysis library .

It will be imported as below

Import pandas as pd

Handling Missing data :

The next step of data preprocessing is to

Handle missing data in the datasets . If our dataset contains

Some missing data, then it may create a huge problem for our jupyter model . Hence it's necessary to handle missing values present in the dataset .

Ways to handle missing data :

There are mainly two ways to handle missing data ,which are:

By deleting the particular row :

The first way is used to commonly deal with null values in the ways , we just delete the specific row or column which consist null values . But this way is not so efficient and removing data may lead to loss of information which will not give accurate output .

By calculating the mean :

In this way,we will calculate the mean of that column or row which contains any missing value and will put it on the placeof missing value.This strategy is useful for the features which have numeric data such as age,salary,year,etc. Here we will use this approach.

To handle missing values,we will use Scikit-let :

Encoding Categorial data :

Categorial data is which has some categories sucn as , in our dataset;

There are two categorical varible, **Country** and **Purchased.**

Since Jupyter model completely works on mathematics and numbers,but if our dataset would have a categorical varible,then it may create trouble while building the model.So it is necessary to encode these categorical variables into numbers.

For Country variable :

Firstly,we will convert the country variables into categorical data. So to do this,we will use **LabelEncoder()**class from **preprocessing** library **learn** library in our code,which contains various laibraries for building Jupyter models. Here we will use **Imputer** class of **sklearn.preprocessing** library.

```
# categorical data
 # for Country Variable
  From sklearn.preprocessing import
  LabelEncoder
  Label_encoder_x= LabelEncoder()
  X[:,0]= label_encoder_x.fit_transform(x[:,0])
```

Splitting the Dataset into the Tranining set and Test set :

In Jupyter model data preprocessing we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our Jupyter model.

Suppose , if we have given training to our Jupyter model by a data set and we test it by a completely different data set. Then , it will create difficult for our model to understand the correlations between the models.

If we train our model vary well and its training accuracy is also very high , but we provide a new data set to it , then it will decrease the performance.

So we always try to may a Jupyter model which performances well with the training set and also with the dataset . Here , we can define these dataset as :

Training set:

A subset of dataset to train the Jupter model , and we already know the output.

Test set :

A subset of dataset to test the Jupter model, and by using the test set, model predicts the output.

For splitting the dataset, we will use the below lines of the code.

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,
test_size=0,.2,rabdom_state=0)
```

Feature Scaling

Feature scaling is the final step of data preprocessing in jupyter . It is a technique to standardize the independent variables of the dataset in a specific range . In feature scaling , we put our variables in the same range and in the same scale so that no any variable dominate the other variable .