AI-driven exploration andprediction of company registration trends with registrar of companies-ROC-phase-3

TEAM MEMBER

623021106034:A.MUTHURAJA

PHASE-3 SUBMISSION DOCUMENT

INTRODUCTION:

Data preprocessing in AI-DRIVEN : 7 easy steps to follow.

1.Acquire the dataset.

2.Import all the crucial libraries.

3.Import the dataset.

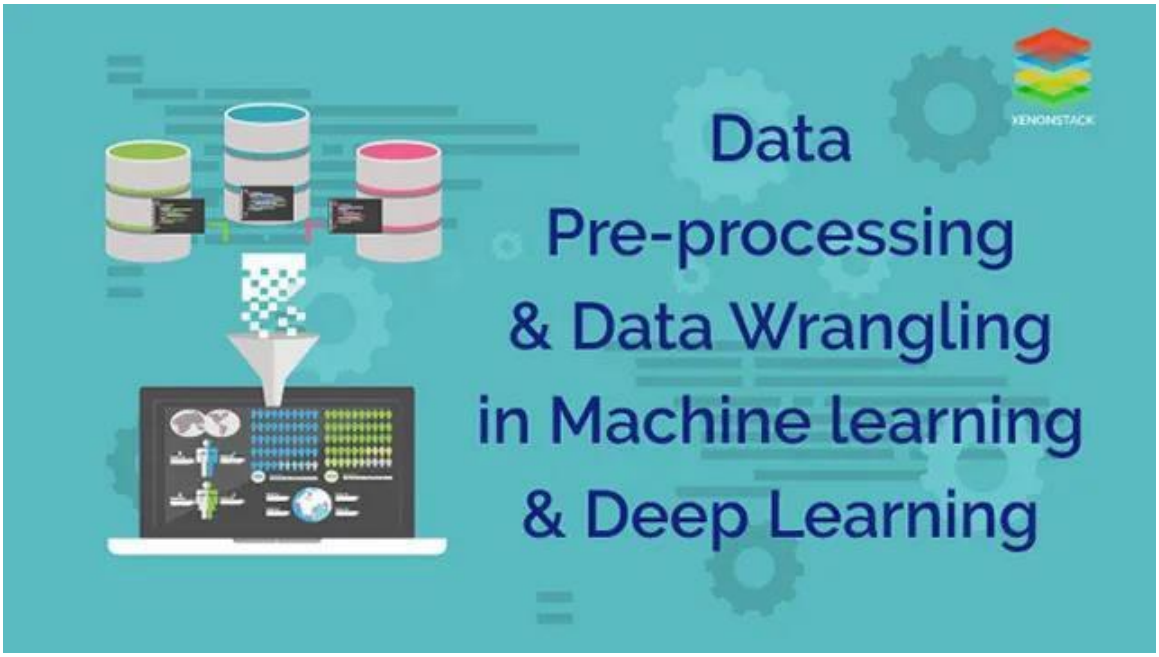
4.Identifying and handling the missing values.

5.Encoding the categorical data.

6.Splitting the dataset.

7.Feature scaling.

1.PRE-PROCESSING THE DATASET:



Introduction to Data Preparation and Preprocessing:

Deep learning and Machine learning are becoming more and more important in today's ERP (Enterprise Resource Planning). During the process of building the analytical model using Deep Learning or Machine Learning the data set is collected from various sources such as a file, database, sensors, and much more.

But, the collected data cannot be used directly for performing the analysis process. Therefore, to solve this problem Data Preparation is done. It includes two techniques that are listed below -

Data Preparation Architecture

Data Preparation process s an important part of Data Science. It includes two concepts such as Data Cleaning and Feature Engineering. These two are compulsory for achieving better accuracy and performance in the Machine Learning and Deep Learning projects.

What is Data Preprocessing?

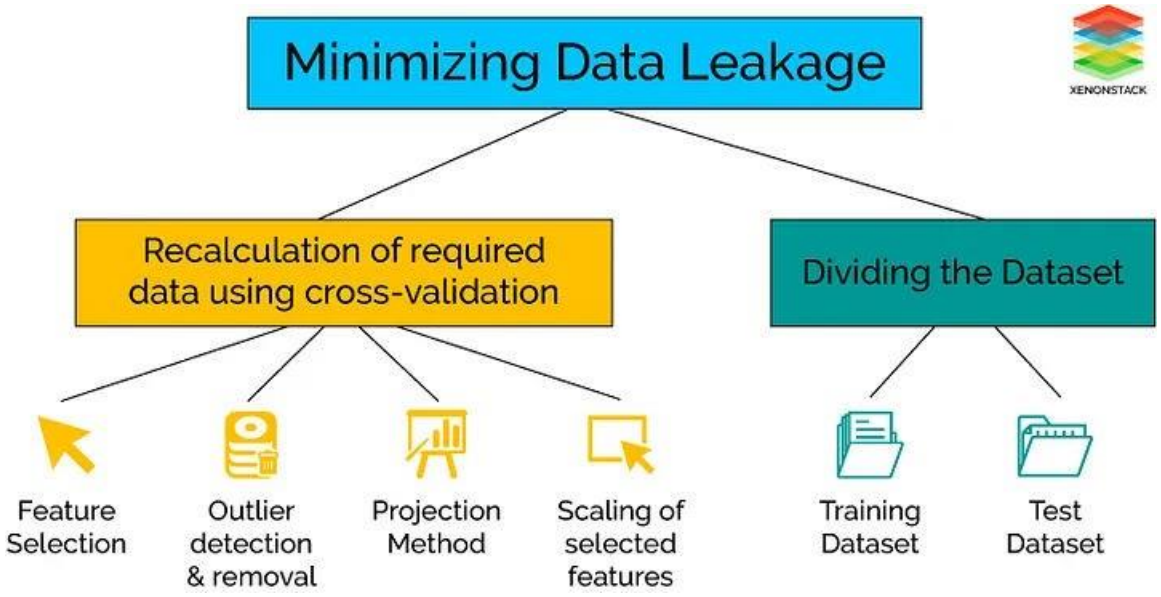
Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.

Therefore, certain steps are executed to convert the data into a small clean data set. This technique is performed before the execution of the Iterative Analysis. The set of steps is known as Data Preprocessing.

What Is Data Wrangling?

Data Wrangling is a technique that is executed at the time of making an interactive model. In other words, it is used to convert the raw data into the format that is convenient for the consumption of data.

This technique is also known as Data Munging. This method also follows certain steps such as after extracting the data from different data sources, sorting of data using the certain algorithms are performed, decompose the data into a different structured format and finally store the data into another database.



PROGRAM:

```
# Identify the quartiles
q1, q3 = np.percentile(df['Insulin'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (1.5 * iqr)
upper_bound = q3 + (1.5 * iqr)
# Drop the outliers
clean_data = df[(df['Insulin'] >= lower_bound)
                & (df['Insulin'] <= upper_bound)]


# Identify the quartiles
q1, q3 = np.percentile(clean_data['Pregnancies'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (1.5 * iqr)
upper_bound = q3 + (1.5 * iqr)
# Drop the outliers
clean_data = clean_data[(clean_data['Pregnancies'] >= lower_bound)
                        & (clean_data['Pregnancies'] <= upper_bound)]
```

```
# Identify the quartiles
q1, q3 = np.percentile(clean_data['Age'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (1.5 * iqr)
upper_bound = q3 + (1.5 * iqr)
# Drop the outliers
clean_data = clean_data[(clean_data['Age'] >= lower_bound)
                        & (clean_data['Age'] <= upper_bound)]


# Identify the quartiles
q1, q3 = np.percentile(clean_data['Glucose'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (1.5 * iqr)
upper_bound = q3 + (1.5 * iqr)
# Drop the outliers
clean_data = clean_data[(clean_data['Glucose'] >= lower_bound)
                        & (clean_data['Glucose'] <= upper_bound)]
```

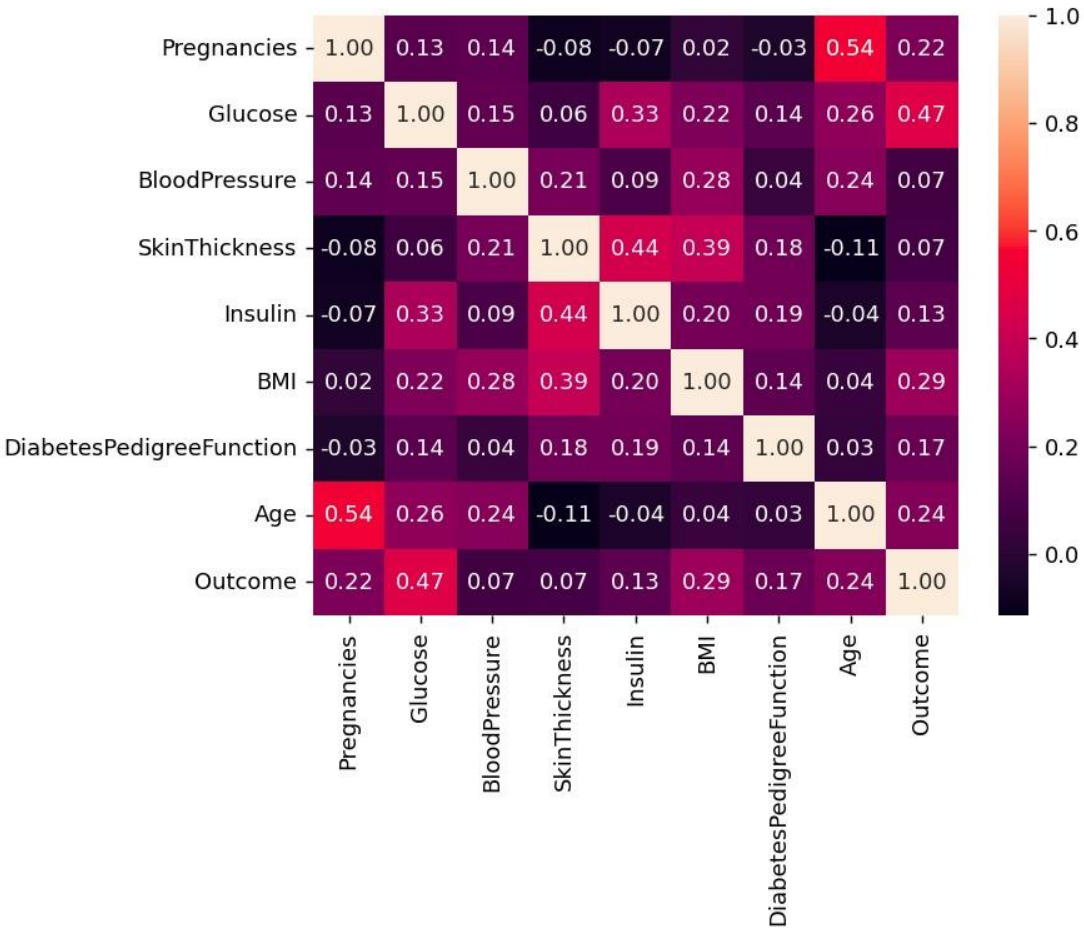
```
# Identify the quartiles
q1, q3 = np.percentile(clean_data['BloodPressure'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (0.75 * iqr)
upper_bound = q3 + (0.75 * iqr)
# Drop the outliers
clean_data = clean_data[(clean_data['BloodPressure'] >= lower_bound)
                        & (clean_data['BloodPressure'] <= upper_bound)]
```

```
# Identify the quartiles
q1, q3 = np.percentile(clean_data['BMI'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (1.5 * iqr)
upper_bound = q3 + (1.5 * iqr)
# Drop the outliers
clean_data = clean_data[(clean_data['BMI'] >= lower_bound)
                        & (clean_data['BMI'] <= upper_bound)]
```

```
# Identify the quartiles
q1, q3 = np.percentile(clean_data['DiabetesPedigreeFunction'], [25, 75])
# Calculate the interquartile range
iqr = q3 - q1
# Calculate the lower and upper bounds
lower_bound = q1 - (1.5 * iqr)
upper_bound = q3 + (1.5 * iqr)

# Drop the outliers
clean_data = clean_data[(clean_data['DiabetesPedigreeFunction'] >= lower_b
                        & (clean_data['DiabetesPedigreeFunction'] <= upper
```

OUTPUT:



- **Glucose 0.466581**
- **Outcome 1.000000**

- **BMI 0.292695**
- **Age 0.238356**
- **Pregnancies 0.221898**
- **DiabetesPedigreeFunction 0.173844**
- **Insulin 0.130548**

- **SkinThickness 0.074752**
- **BloodPressure 0.0**
- **array([[0.64 , 0.848, 0.15 , 0.907, -0.693, 0.204, 0.468, 1.426],**
- **[-0.845, -1.123, -0.161, 0.531, -0.693, -0.684, -0.365, -0.191],**
- **[1.234, 1.944, -0.264, -1.288, -0.693, -1.103, 0.604, -0.106],**
- **[-0.845, -0.998, -0.161, 0.155, 0.123, -0.494, -0.921, -1.042],**
- **[-1.142, 0.504, -1.505, 0.907, 0.766, 1.41 , 5.485**

2.LOADING OF DATASET:

```
def load_csv(filepath):
    data = []
    col = []
    checkcol = False
    with open(filepath) as f:
        for val in f.readlines():
            val = val.replace("\n","")
            val = val.split(',')
```

if checkcol is False: col = val checkcol = True

else:

data.append(val) df = pd.DataFrame(data=data, columns=col)

return df OUTPUT:

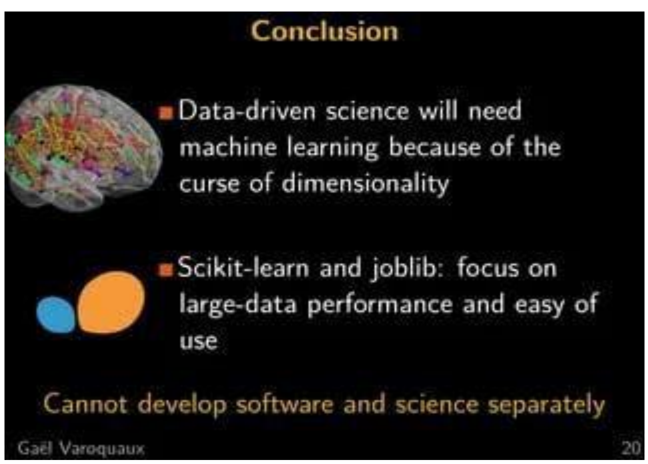
myData = load_csv('100 Sales Record.csv')

print(myData.head())

	Region	Country	Item Type	Sales Channel	Order Priority	Order Date	Order ID	Ship Date	Units Sold	Unit Price	Unit Cost	Total Revenue	Total Cost	Total Profit
0	Australia and Oceania	Tuvalu	Baby Food	Offline	H	5/28/2010	669165933	6/27/2010	9925	255.28	159.42	2533654.00	1582243.50	951410.50
1	Central America and the Caribbean	Grenada	Cereal	Online	C	8/22/2012	963881480	9/15/2012	2804	205.70	117.11	576782.80	328376.44	248406.36
2	Europe	Russia	Office Supplies	Offline	L	5/2/2014	341417157	5/8/2014	1779	651.21	524.96	1158502.59	933903.84	224598.75
3	Sub-Saharan Africa	Sao Tome and Principe	Fruits	Online	C	6/20/2014	514321792	7/5/2014	8102	9.33	6.92	75591.66	56065.84	19525.82
4	Sub-Saharan Africa	Rwanda	Office Supplies	Offline	L	2/1/2013	115456712	2/6/2013	5062	651.21	524.96	3296425.02	2657347.52	639077.50

CONCLUSION:

Dataset in Python is mostly used for manipulation of Gifs and other custom data which frames the entire dataset as per requirement.



In These two technology projects you will begin building your project by loading and preprocessing the dataset perform different analysis as needed.