Veermata Jijabai Technological Institute, Mumbai 400019

Experiment No.: 03

Aim: To perform data cleaning and preparing for operations

Name: Kiran K Patil

Enrolment No.: 211070904

Branch: Computer Engineering

Batch: D

Theory:

1. **Handling Missing Data:**
    - Missing data can be addressed through techniques like:
        - **Imputation**: Replacing missing values with estimates (e.g., mean, median, mode).
        - **Deletion**: Removing records with missing values (be cautious not to lose too much information).
        - Treating missing values as a separate category when applicable (e.g., "unknown" category for categorical data).

2. **Data Transformation:**
    - Transform data to make it suitable for analysis by:
        - Converting data types, such as dates to datetime objects.
        - Encoding categorical variables using one-hot encoding or label encoding.
        - Normalizing or scaling numeric features for consistent scales.
        - Creating new features through feature engineering to capture more meaningful information.

3. **Data Standardization:**
    - Standardize data by:
        - Mean-centering numeric features, ensuring their means are close to zero.
        - Scaling features to unit variance, so they have similar ranges.
        - Important for algorithms sensitive to feature scales (e.g., support vector machines).

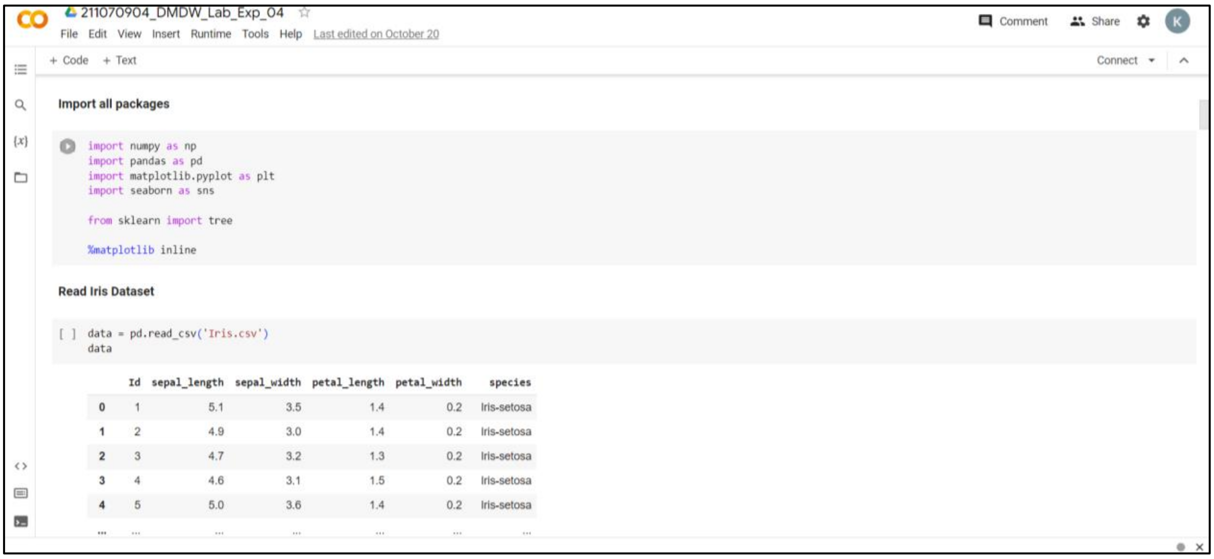
4. **Feature Selection:**
    - Select relevant features by:
        - Using techniques like Recursive Feature Elimination (RFE) to iteratively remove less important features.
        - Ranking features by importance from tree-based models like Random Forest or Gradient Boosting.

5. **Data Splitting:**
   - Divide your data into subsets for model development and evaluation:
     - Training set: Used to train machine learning models.
     - Validation set: Used for hyperparameter tuning and model selection.
     - Test set: Reserved for final model evaluation to assess generalization performance.

6. **Handling Imbalanced Data:**
   - Address imbalanced class distributions by:
     - Oversampling: Creating more instances of the minority class to balance the dataset.
     - Undersampling: Reducing instances of the majority class to balance the dataset.
     - Using synthetic data generation methods like Synthetic Minority Over-sampling Technique (SMOTE) to create synthetic instances of the minority class.

These processes ensure that your data is cleaned, transformed, standardized, and appropriately prepared for analysis or machine learning. The specific techniques you choose within each step will depend on the characteristics of your data and the objectives of your project.

Implementation:



211070904_DMDW_Lab_Exp_04

File Edit View Insert Runtime Tools Help Last edited on October 20

Comment Share

+ Code + Text Connect

Import all packages

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn import tree

%matplotlib inline
```

Read Iris Dataset

```
data = pd.read_csv('Iris.csv')
data
```

	Id	sepal_length	sepal_width	petal_length	petal_width	species
0	1	5.1	3.5	1.4	0.2	Iris-setosa
1	2	4.9	3.0	1.4	0.2	Iris-setosa
2	3	4.7	3.2	1.3	0.2	Iris-setosa
3	4	4.6	3.1	1.5	0.2	Iris-setosa
4	5	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...	...

211070904_DMDW_Lab_Exp_04

File Edit View Insert Runtime Tools Help Last edited on October 20

Comment Share

+ Code + Text Connect

Define Colunms

```
col_names = ['id', 'sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']

data.columns = col_names

col_names
```

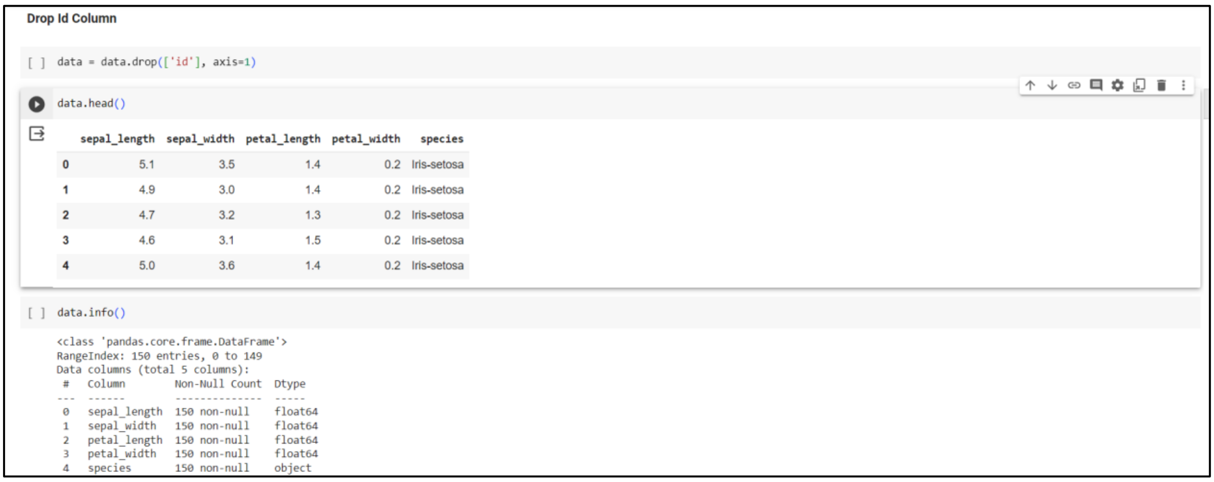
```
['id', 'sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species']
```

Drop Id Column

```
data = data.drop(['id'], axis=1)
```

```
data.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa



Drop Id Column

```
data = data.drop(['id'], axis=1)
```

```
data.head()
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object
```

Checking the target categorical counts

```
[ ] data['species'].value_counts()

    Iris-setosa        50
    Iris-versicolor    50
    Iris-virginica     50
    Name: species, dtype: int64
```

Check missing values in variables

```
[ ] data.isnull().sum()

    sepal_length    0
    sepal_width     0
    petal_length    0
    petal_width     0
    species         0
    dtype: int64
```

```
target_col = ['species']
```

```
[ ] X = data.drop(['species'], axis=1)

    y = data['species']
```

Split dataset into train and test

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, random_state = 42)
```

Check datatypes

```
[ ] X_train.dtypes

    sepal_length    float64
    sepal_width     float64
    petal_length    float64
    petal_width     float64
    dtype: object
```

Conclusion:

In conclusion, effective data cleaning and preprocessing are essential for ensuring data accuracy and preparing it for analysis or machine learning. Addressing missing data, transforming features, standardizing data, selecting relevant features, splitting data, and handling imbalanced datasets are key processes that improve the quality and usability of the data for downstream tasks. Properly cleaned and preprocessed data significantly enhances the reliability and performance of analytical and machine learning models.