Certificate Course in Machine Learning using Python [6 Weeks]

Pre-processing of data for Machine Learning

Pre-processing

Step 4: Taking care of Missing Data in Dataset

- In Python, specifically Pandas, NumPy and Scikit-Learn, missing values are represented as NaN.
- Values with a NaN value are ignored from operations like sum, count, etc.
- Missing values are specified with NaN. Python will recognize only NaNs as missing.
- Any other missing values such as space, .(dot), *, $ or # will not be recognized by the Python as missing values.
- Missing values other than NaN are handled by na_values parameter of read_csv().

na_values - handles non NaN values in a DataFrame.

For example: In the Data_for_proprocessing.csv file, the missing values are represented by '#'. The '#' can be replaced with NaN as

```
dataset=pd.read_csv('Data_for_preprocessing.csv', na_values=[' #','NULL'])
```

- Here, na_values=[' #','NULL'] specifies that the # and NULL values are treated as NaN.
- We can specify any symbol as missing value in na_values. The symbol depends upon the dataset being used.

Checking Missing Values

isnull()

isnull() function is used to check missing values in a data frame. It returns Boolean values which are True for NaN values.

- **Checking entire data frame**

```
print(dataset.isnull())
```

- **Checking Age column only**

```
print(dataset['Age'].isnull())
```

- **Counting missing values from each column**

```
print(dataset.isnull.sum())
```

Replacing missing values

- **A Simple Option**: Drop Columns or rows with Missing Values

dropna(): dropna() function is used to drop Rows/Columns with NaN values.

- **To drop columns with missing values:**

```
X=X.dropna(axis=1)
```

Now, the column which has NaN values will be dropped from the X dataframe.

- **To drop rows with missing values:**

```
X=X.dropna()
```

Now, all the rows with NaN values are dropped from the X dataframe.

- **A Better Option: Imputation**

The Imputer() class can take a few parameters —

- **missing_values:** The missing_values placeholder which has to be imputed. By default is NaN.
- **strategy :** The data which will replace the NaN values from the dataset. The strategy argument can take the values – 'mean'(default), 'median', 'most_frequent'.
- **axis :** We can either assign it 0 or 1. 0 to impute along columns and 1 to impute along rows.

Note: Imputer class works on numbers, not strings.

- **For numerical values, the simplest method is to replace the missing numerical values with mean.**

```
from sklearn.preprocessing import Imputer
fill_NaN = Imputer(missing_values='NaN', strategy='mean', axis=0)
X[['Age','Salary']]= fill_NaN.fit_transform(X[['Age','Salary']])
print (X)
```

Note: Since 'Age' and 'Salary' column contains numerical values. So the missing values of 'Age' and 'Salary' column is replaced by their mean.

- **For Categorical values, count the occurrences of each category and replace the missing values with high frequency values.**

- **Count frequency of each category**

```
#Imputing missing values of categorical column 'Country'
#Counting frequency of each category in 'Country' Column using value_counts()
X['Country'].value_counts()
```

Output:
```
France  4
Spain   3
Germany 1
```

- **Replace the missing values with highest frequency value**

Output suggests that the most frequent value is 'France'. So replace the NaN values of 'Country' Column with 'France'.

```
#Replacing the NaN values with 'France'
X['Country'].fillna('France', inplace=True)
```

- Checking missing values again,

```
X.isnull().sum()
```

Output: Country 0

Age 0

Salary 0

Previous Next

PREVIOUS ACTIVITY

◄ Data File: Admission_Predict.csv: Day 10

Jump to...

NEXT ACTIVITY

Python code: Regression Problem:Apply Different Algorithm: Day 11 ►

Stay in touch

Contact Us

http://nielit.gov.in/gorakhpur/

abhinav@nielit.gov.in or ajay.verma@nielit.gov.in