18IT053

Data Pre-processing using Scikit-learn

1. Standardization
2. normalization
3. encoding
4. discretization
5. imputation of missing values

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import preprocessing
```

```
data = pd.read_csv("MagicBricks.csv")
data.head()
```

	Area	BHK	Bathroom	Furnishing	Locality	Parking	Price	Status
0	800.0	3.0	2.0	Semi-Furnished	Rohini Sector 25	1.0	6500000.0	Ready_to_move
1	750.0	2.0	2.0	Semi-Furnished	J R Designers Floors, Rohini Sector 24	1.0	5000000.0	Ready_to_move
2	950.0	2.0	2.0	Furnished	Citizen Apartment, Rohini Sector 13	1.0	15500000.0	Ready_to_move
3	600.0	2.0	2.0	Semi-Furnished	Rohini Sector 24	1.0	4200000.0	Ready_to_move
					Rohini			

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1259 entries, 0 to 1258
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Area         1259 non-null   float64
 1   BHK          1259 non-null   float64
 2   Bathroom     1257 non-null   float64
 3   Furnishing   1254 non-null   object
```

```
 4   Locality     1259 non-null   object
 5   Parking      1226 non-null   float64
 6   Price        1259 non-null   float64
 7   Status       1259 non-null   object
 8   Transaction  1259 non-null   object
 9   Type         1254 non-null   object
 10  Per_Sqft     1018 non-null   float64
dtypes: float64(6), object(5)
memory usage: 108.3+ KB
```

Encoding

```
from sklearn.preprocessing import LabelEncoder , OneHotEncoder
```

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

```
Ready_to_move    1184
Almost_ready       75
Name: Status, dtype: int64
```

LABEL ENCODER

```
le = LabelEncoder()
data['Status'] = le.fit_transform(data['Status'])
```

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

```
1    1184
0      75
Name: Status, dtype: int64
```

```
le.classes_
```

```
array(['Almost_ready', 'Ready_to_move'], dtype=object)
```

ONE HOT ENCODER

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

```
Resale          781
New_Property    478
Name: Transaction, dtype: int64
```

```
one_hot = OneHotEncoder()
transformed_data = one_hot.fit_transform(data['Transaction'].values.reshape(-1,1)).toarray
```

```
one_hot.categories_
```

```
[array(['New_Property', 'Resale'], dtype=object)]
```

```
transformed_data = pd.DataFrame(transformed_data ,columns = ['Area', 'BHK'])
transformed_data.head()
```

	Area	BHK
0	1.0	0.0
1	1.0	0.0
2	0.0	1.0
3	0.0	1.0
4	1.0	0.0

```
transformed_data.iloc[90 , ]
```

```
Area    0.0
BHK     1.0
Name: 90, dtype: float64
```

```
data['Transaction'][90]
```

```
'Resale'
```

Normalization & Standardization

```
numeric_columns = [c for c in data.columns if data[c].dtype != np.dtype('O')]
```

```
len(numeric_columns) , len(data.columns)
```

```
(7, 11)
```

```
temp_data = data[numeric_columns]
temp_data
```

	Area	BHK	Bathroom	Parking	Price	Status	Per_Sqft
0	800.0	3.0	2.0	1.0	6500000.0	1	NaN
1	750.0	2.0	2.0	1.0	5000000.0	1	6667.0
2	950.0	2.0	2.0	1.0	15500000.0	1	6667.0

Normalization

```
from sklearn.preprocessing import StandardScaler , MinMaxScaler
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
normalizer = MinMaxScaler()
```

```
temp_data.dropna(axis = 1 , inplace = True)

normalized_data = normalizer.fit_transform(temp_data)

pd.DataFrame(normalized_data , columns = temp_data.columns)
```

	Area	BHK	Price	Status
0	0.031806	0.222222	0.023013	1.0
1	0.029746	0.111111	0.016736	1.0
2	0.037986	0.111111	0.060669	1.0
3	0.023566	0.111111	0.013389	1.0
4	0.025626	0.111111	0.021757	1.0
...	...	...	...	...
1254	0.168507	0.333333	0.225941	1.0
1255	0.042106	0.222222	0.048117	1.0
1256	0.034896	0.222222	0.069038	1.0
1257	0.039634	0.111111	0.043933	1.0
1258	0.454103	0.222222	0.073222	1.0

1259 rows × 4 columns

Standardization

```
standard_scaler = StandardScaler()
```

```
standardized_data = standard_scaler.fit_transform(temp_data)
```

```
pd.DataFrame(standardized_data , columns = temp_data.columns)
```

	Area	BHK	Price	Status
0	-0.425188	0.213130	-0.578591	0.251684
1	-0.457087	-0.835038	-0.637205	0.251684
2	-0.329490	-0.835038	-0.226904	0.251684
3	-0.552785	-0.835038	-0.668466	0.251684
4	-0.520886	-0.835038	-0.590313	0.251684
...	...	...	...	...
1254	1.691650	1.261298	1.316608	0.251684
1255	-0.265691	0.213130	-0.344133	0.251684
1256	-0.377339	0.213130	-0.148752	0.251684
1257	-0.303970	-0.835038	-0.383209	0.251684
1258	6.114170	0.213130	-0.109675	0.251684

1259 rows × 4 columns

Handling With Missing Values

```
data.isnull().sum()
```

```
Area             0
BHK              0
Bathroom         2
Furnishing       5
Locality         0
Parking         33
Price            0
Status           0
Transaction      0
Type             5
Per_Sqft       241
dtype: int64
```

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

```
241
```

Simple Imputer

```
from sklearn.impute import SimpleImputer
```

```
imputer = SimpleImputer(missing_values=np.nan , strategy='mean')
```

```
Per_Sqft = imputer.fit_transform(data['Per_Sqft'].values.reshape(-1,1))
```

```
pd.DataFrame(Per_Sqft).isnull().sum()
```

```
0    0
dtype: int64
```

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

```
241
```

Discretization

```
from sklearn.preprocessing import KBinsDiscretizer
```

```
temp_data.head()
```

	Area	BHK	Price	Status
0	800.0	3.0	6500000.0	1
1	750.0	2.0	5000000.0	1
2	950.0	2.0	15500000.0	1
3	600.0	2.0	4200000.0	1
4	650.0	2.0	6200000.0	1

Quantile Discretization Transform

```
trans = KBinsDiscretizer(n_bins =10 , encode = 'ordinal' , strategy='quantile')
new_data = trans.fit_transform(temp_data)
```

```
pd.DataFrame(new_data,columns = temp_data.columns )
```

	Area	BHK	Price	Status
0	2.0	2.0	2.0	0.0
1	2.0	1.0	2.0	0.0
2	3.0	1.0	5.0	0.0
3	1.0	1.0	1.0	0.0
4	1.0	1.0	2.0	0.0

Uniform Discretization Transform

1254	9.0	3.0	9.0	0.0

```
trans = KBinsDiscretizer(n_bins =10 , encode = 'ordinal' , strategy='uniform')
new_data = trans.fit_transform(temp_data)

pd.DataFrame(new_data,columns = temp_data.columns )
```

	Area	BHK	Price	Status
0	0.0	2.0	0.0	9.0
1	0.0	1.0	0.0	9.0
2	0.0	1.0	0.0	9.0
3	0.0	1.0	0.0	9.0
4	0.0	1.0	0.0	9.0
...	...	...	...	...
1254	1.0	3.0	2.0	9.0
1255	0.0	2.0	0.0	9.0
1256	0.0	2.0	0.0	9.0
1257	0.0	1.0	0.0	9.0
1258	4.0	2.0	0.0	9.0

1259 rows × 4 columns

KMeans Discretization Transform

```
trans = KBinsDiscretizer(n_bins =10 , encode = 'ordinal' , strategy='kmeans')
new_data = trans.fit_transform(temp_data)

pd.DataFrame(new_data,columns = temp_data.columns )
```

	Area	BHK	Price	Status
0	0.0	2.0	0.0	1.0
1	0.0	1.0	0.0	1.0
2	0.0	1.0	1.0	1.0
3	0.0	1.0	0.0	1.0
4	0.0	1.0	0.0	1.0
...	...	...	...	...
1254	3.0	3.0	3.0	1.0
1255	0.0	2.0	1.0	1.0
1256	0.0	2.0	1.0	1.0
1257	0.0	1.0	1.0	1.0
1258	5.0	2.0	1.0	1.0

1259 rows × 4 columns

✓ 0s completed at 6:08 PM