Machine Learning (CCS4340)

Lab 03: Mean or Median Imputation

Mean / median imputation consists in replacing missing values by the variable's mean or median.

In this Lab, we will perform mean and median imputation utilizing pandas, Scikit-learn and Feature-engine.

In [1]:
```
import pandas as pd

# to split the data sets:
from sklearn.model_selection import train_test_split

# to impute missing data with sklearn:
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer

# to impute missing data with Feature-engine:
from feature_engine.imputation import MeanMedianImputer
```

Load Data

In [2]:
```
data = pd.read_csv("credit_approval_uci.csv")
data.head()
```

Out[2]:

	A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15	target
0	b	30.83	0.000	u	g	w	v	1.25	t	t	1	f	g	202.0	0	1
1	a	58.67	4.460	u	g	q	h	3.04	t	t	6	f	g	43.0	560	1
2	a	24.50	NaN	u	g	q	h	NaN	NaN	NaN	0	f	g	280.0	824	1
3	b	27.83	1.540	u	g	w	v	3.75	t	t	5	t	g	100.0	3	1
4	b	20.17	5.625	u	g	w	v	1.71	t	f	0	f	s	120.0	0	1

Split data in train and test sets

In [3]:
```
X_train, X_test, y_train, y_test = train_test_split(data.drop("target", axis=1), d
X_train.shape, X_test.shape
```

Out[3]:
```
((483, 15), (207, 15))
```

In [4]:
```
# Let's inspect the proportion of missing value per variable:

X_train.isnull().mean()
```

Out[4]:
```
A1     0.008282
A2     0.022774
A3     0.140787
A4     0.008282
A5     0.008282
A6     0.008282
A7     0.008282
A8     0.140787
A9     0.140787
A10    0.140787
A11    0.000000
A12    0.000000
A13    0.000000
A14    0.014493
A15    0.000000
dtype: float64
```

Select the variables to impute

In [5]:
```
numeric_vars = X_train.select_dtypes(exclude="O").columns.to_list()
numeric_vars
```

Out[5]:
```
['A2', 'A3', 'A8', 'A11', 'A14', 'A15']
```

Mean / median imputation with pandas

pd.fillna https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.fillna.html

In [6]:
```
# Learn the variables median values:

median_values = X_train[numeric_vars].median().to_dict()
median_values
```

Out[6]:
```
{'A2': 28.835, 'A3': 2.75, 'A8': 1.0, 'A11': 0.0, 'A14': 160.0, 'A15': 6.0}
```

In [7]:
```
# Replace missing data by the median:

X_train = X_train.fillna(value = median_values)  # Fill NA/NaN values using the sp
X_test = X_test.fillna(value = median_values)
```

In [8]:
```
# Corroborate absense of missing values:

X_train[numeric_vars].isnull().sum()
#X_test[numeric_vars].isnull().sum()
```

Out[8]:
```
A2     0
A3     0
A8     0
A11    0
A14    0
A15    0
dtype: int64
```

Mean / median imputation with Scikit-learn

SimpleImputer https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html

In [16]:
```
# Split data into train and test set:

X_train, X_test, y_train, y_test = train_test_split(data.drop("target", axis=1), d
numeric_vars
```

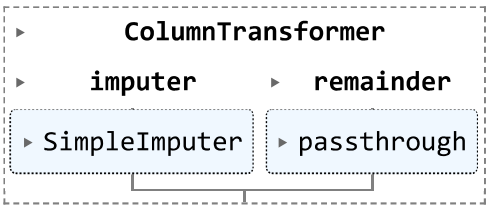
Out[16]:
```
['A2', 'A3', 'A8', 'A11', 'A14', 'A15']
```

In [17]:
```
# Make a list with the non-numerical variables:

remaining_vars = [var for var in X_train.columns if var not in numeric_vars]
remaining_vars
```

Out[17]:
```
['A1', 'A4', 'A5', 'A6', 'A7', 'A9', 'A10', 'A12', 'A13']
```

In [18]:
```
# Set up the imputer to replace missing data with the median:

imputer = SimpleImputer(strategy="median")

# Indicate which variables to impute:

ct = ColumnTransformer([("imputer", imputer, numeric_vars)], remainder="passthroug

# Find the median value per variable:
ct.fit(X_train)
```

Out[18]:



In [19]:
```
# Check the median that will be used in the imputation:
ct.named_transformers_.imputer.statistics_
```

Out[19]:
```
array([ 28.835,   2.75 ,   1.   ,   0.   , 160.   ,   6.   ])
```

In [20]:
```
# Replace missing data:
X_train = ct.transform(X_train)
X_test = ct.transform(X_test)

X_train
```

Out[20]:
```
array([[46.08, 3.0, 2.375, ..., 't', 't', 'g'],
       [15.92, 2.875, 0.085, ..., 'f', 'f', 'g'],
       [36.33, 2.125, 0.085, ..., 't', 'f', 'g'],
       ...,
       [19.58, 0.665, 1.665, ..., 'f', 'f', 'g'],
       [22.83, 2.29, 2.29, ..., 't', 't', 'g'],
       [40.58, 3.29, 3.5, ..., 'f', 't', 's']], dtype=object)
```

In [21]:
```
# Convert returned array to a pandas dataframe:

X_train = pd.DataFrame(X_train, columns=numeric_vars + remaining_vars,)
X_train.head()
```

Out[21]:

	A2	A3	A8	A11	A14	A15	A1	A4	A5	A6	A7	A9	A10	A12	A13
0	46.08	3.0	2.375	8.0	396.0	4159.0	a	u	g	c	v	t	t	t	g
1	15.92	2.875	0.085	0.0	120.0	0.0	a	u	g	q	v	f	f	f	g
2	36.33	2.125	0.085	1.0	50.0	1187.0	b	y	p	w	v	t	t	f	g
3	22.17	0.585	0.0	0.0	100.0	0.0	b	y	p	ff	ff	f	f	f	g
4	57.83	7.04	14.0	6.0	360.0	1332.0	b	u	g	m	v	t	t	t	g

In [27]:
```
# Corroboratw absence of missing values:

X_train[numeric_vars].isnull().sum()
```

Out[27]:
```
A2     0
A3     0
A8     0
A11    0
A14    0
A15    0
dtype: int64
```

In [30]:
```
# Convert returned array to a pandas dataframe:
X_test = pd.DataFrame(X_test, columns = numeric_vars +remaining_vars)
X_test.head()
```

Out[30]:

	A2	A3	A8	A11	A14	A15	A1	A4	A5	A6	A7	A9	A10	A12	A13
0	45.83	10.5	5.0	7.0	0.0	0.0	a	u	g	q	v	t	t	t	g
1	64.08	20.0	17.5	9.0	0.0	1000.0	b	u	g	x	h	t	t	t	g
2	31.25	3.75	0.625	9.0	181.0	0.0	a	u	g	cc	h	t	t	t	g
3	39.25	9.5	6.5	14.0	240.0	4607.0	b	u	g	m	v	t	t	f	g
4	26.17	2.0	0.0	0.0	276.0	1.0	a	u	g	j	j	f	f	t	g

In [31]:
```
# Corroborate absence of missing values:

X_test[numeric_vars].isnull().sum()
```

Out[31]:
```
A2     0
A3     0
A8     0
A11    0
A14    0
A15    0
dtype: int64
```

Mean / Median imputation with Feature-engine

MeanMedianImputer https://feature-engine.readthedocs.io/en/latest/api_doc/imputation/MeanMedianImputer.html

In [32]:
```
# Split data into train and test set:

X_train, X_test, y_train, y_test = train_test_split(data.drop("target", axis=1), d
```

In [33]:
```
# Set up the imputer to replace missing data with the median:

imputer = MeanMedianImputer(imputation_method="median", variables=numeric_vars)

# Find the median values:

imputer.fit(X_train)
```

Out[33]:
```
MeanMedianImputer(variables=['A2', 'A3', 'A8', 'A11', 'A14', 'A15'])
```

In [34]:
```
# The median values per variable:

imputer.imputer_dict_
```

Out[34]:
```
{'A2': 28.835, 'A3': 2.75, 'A8': 1.0, 'A11': 0.0, 'A14': 160.0, 'A15': 6.0}
```

In [36]:
```
# Replace missing data with the median:
X_train = imputer.transform(X_train)
X_test = imputer.transform(X_test)
```

In [37]:
```
# Corroborate absence of missing values:

X_train[numeric_vars].isnull().sum()
```

Out[37]:
```
A2     0
A3     0
A8     0
A11    0
A14    0
A15    0
dtype: int64
```

In [38]:
```
# Corroborate absence of missing values:

X_test[numeric_vars].isnull().sum()
```

Out[38]:
```
A2     0
A3     0
A8     0
A11    0
A14    0
A15    0
dtype: int64
```