THE iSCALE

Industries Helping Hands

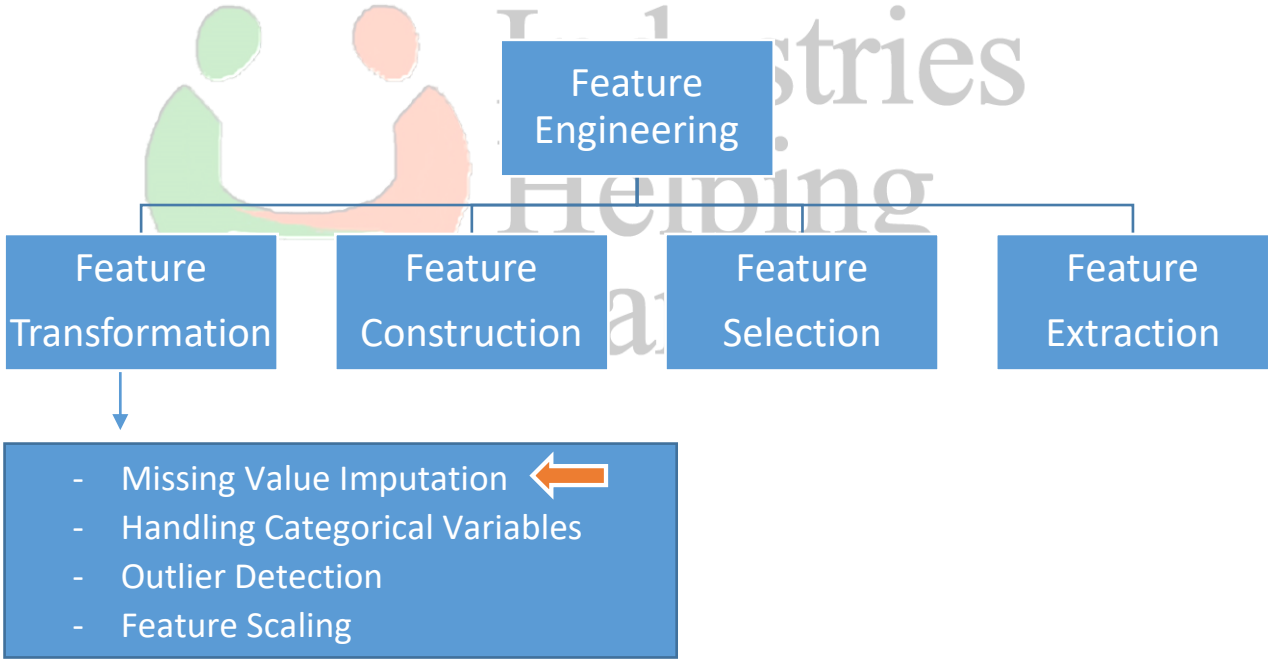
Data Science | 30 Days of Machine Learning | Day - 11

Educator Name: Nishant Dhote
Support Team: **+91-7880-113-112**

----Today Topics | Day 11----

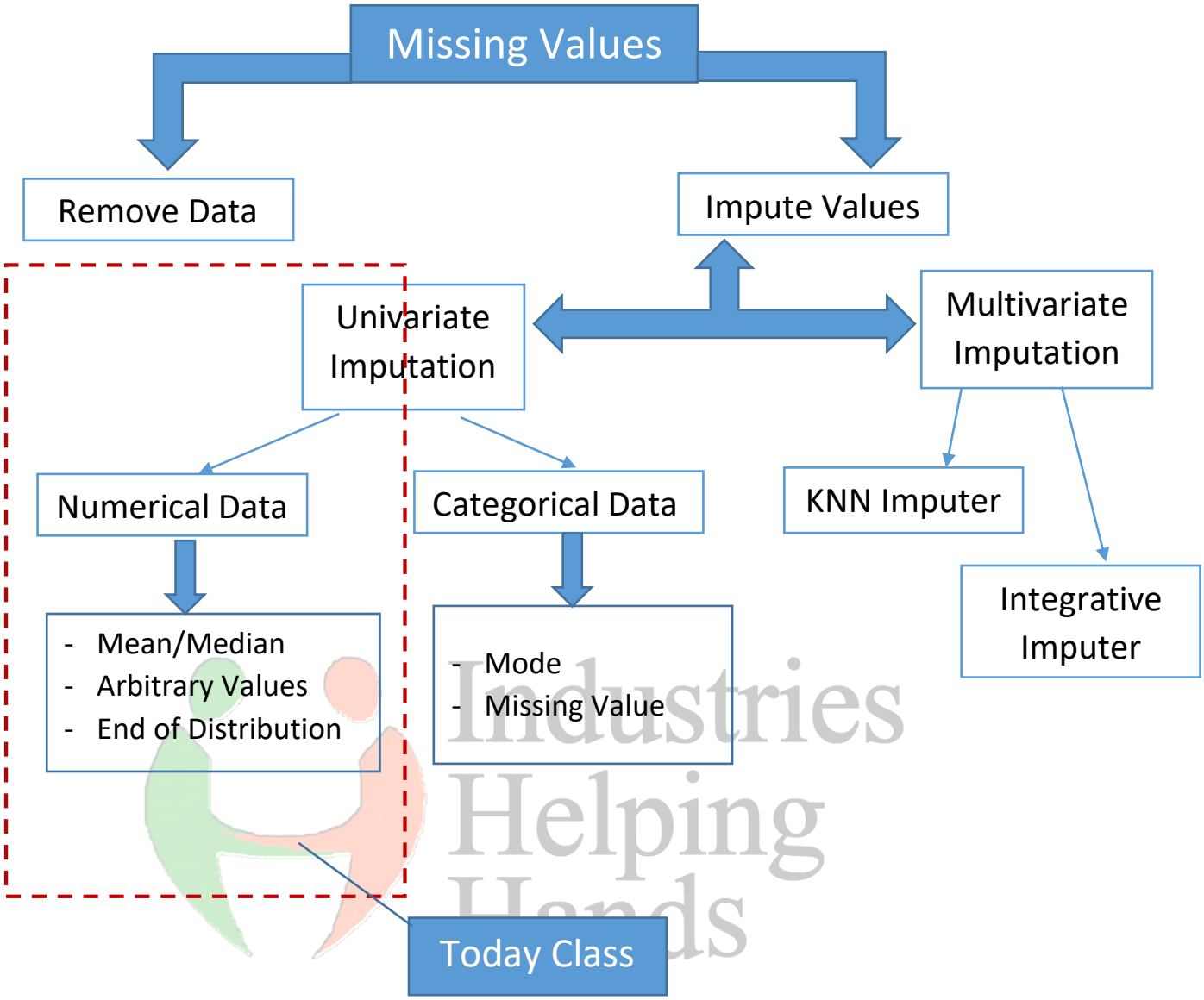
- Univariate Imputation in Numerical Data
- Difference between univariate imputation and multivariate imputation
- Mean or Median Imputation
- Arbitrary Value Imputation
- End of Distribution
- Random value Imputation

Dataset Link GitHub: https://github.com/TheiScale/30_Days_Machine_Learning/



- Two Important ways to handle missing values in the dataset:

1. Deleting rows with missing values (Remove Missing Values)
2. Impute missing values (Fill)
   - Univariate (Numerical & Categorical Removal)
   - Multivariate (KNN & Iterative Imputer)



Univariate Imputation in Numerical Data

What is the difference between univariate imputation and multivariate imputation?

Univariate imputation implies that we are only considering the values of a single column when performing imputation. Multivariate imputation, on the other hand, involves taking into account other features in the dataset when performing imputation.

Univariate Imputation

C1	C2	C3	C4	C5
	--			

Multivariate Imputation

C1	C2	C3	C4	C5
	--			

We cover below topics

- Mean/Median
- Arbitrary Values
- End of Distribution

Mean or Median Imputation-

Mean imputation (MI) is one such method in which the mean of the observed values for each variable is computed and the missing values for that variable are imputed by this mean.

Mean / Median imputation: example

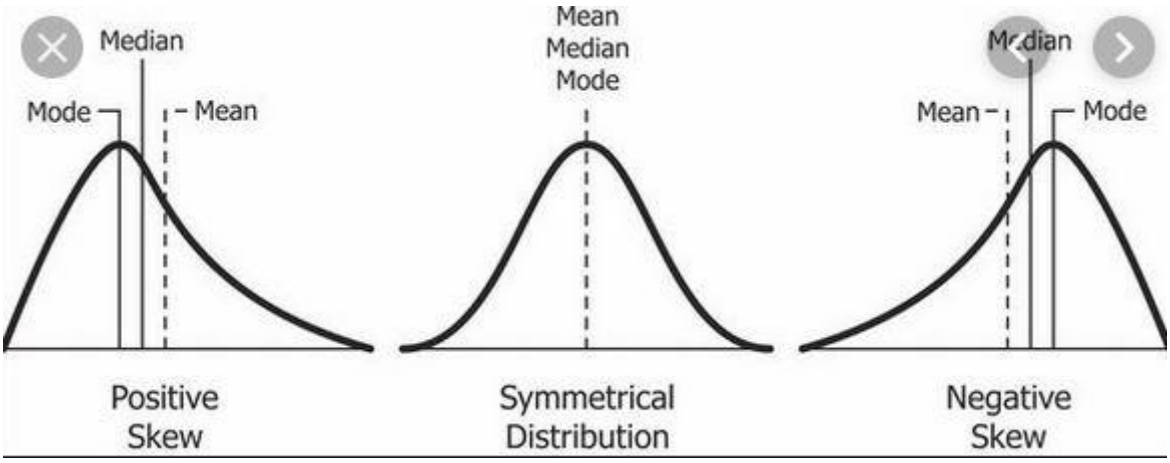
Price
100
90
50
40
20
100
60
120
200

Mean = 86.66

Median = 90

Price
100
90
50
40
20
100
86.66
60
120
86.66
200

When will we use mean value and median value?



Average	Advantages	Disadvantages
Mean	All the data is used to find the answer	Very large or very small numbers can distort the answer
Median	Very big and very small values don't affect it	Takes a long time to calculate for a very large set of data
Mode or modal class	The only average we can use when the data is not numerical	1. There may be more than one mode 2. There may be no mode at all if none of the data is the same 3. It may not accurately represent the data

<Start Coding | Mean – Median - imputation>

#Import Libraries

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

#Import sklearn Libraries

```
from sklearn.model_selection import
train_test_split
from sklearn.impute import SimpleImputer
from sklearn.compose import ColumnTransformer
```

#Import Dataset

```
df = pd.read_csv('titanic_toy.csv')
```

```
df.head()
```

```
df.info(
```

#Perform Train Test Split

```
X = df.drop(columns=['Survived'])
```

```
y = df['Survived']
```

```
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.2,random_state=2
)
```

```
X_train.shape, X_test.shape
```

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

#Calculate Mean and Median (Age | Fare)

```
mean_age = X_train['Age'].mean()
median_age = X_train['Age'].median()

mean_fare = X_train['Fare'].mean()
median_fare = X_train['Fare'].median()
```

#Create new column & impute missing values

```
X_train['Age_median'] =
X_train['Age'].fillna(median_age)
X_train['Age_mean'] =
X_train['Age'].fillna(mean_age)

X_train['Fare_median'] =
X_train['Fare'].fillna(median_fare)
X_train['Fare_mean'] =
X_train['Fare'].fillna(mean_fare

---
X_train.sample(8)
```

#Review Variance

```
print('Original Age variable variance: ',
X_train['Age'].var())
print('Age Variance after median imputation: ',
X_train['Age_median'].var())
print('Age Variance after mean imputation: ',
X_train['Age_mean'].var())
```

```
print('Original Fare variable variance: ',
X_train['Fare'].var())
print('Fare Variance after median imputation: ',
X_train['Fare_median'].var())
print('Fare Variance after mean imputation: ',
X_train['Fare_mean'].var())
```

#Changes in Distribution in Age

```
fig = plt.figure()
ax = fig.add_subplot(111)

# original variable distribution
X_train['Age'].plot(kind='kde', ax=ax)

# variable imputed with the median
X_train['Age_median'].plot(kind='kde', ax=ax,
color='red')

# variable imputed with the mean
X_train['Age_mean'].plot(kind='kde', ax=ax,
color='green')

# add legends
lines, labels = ax.get_legend_handles_labels()
ax.legend(lines, labels, loc='best')
```

#Changes in Distribution in Fare

```
fig = plt.figure()
ax = fig.add_subplot(111)

# original variable distribution
X_train['Fare'].plot(kind='kde', ax=ax)

# variable imputed with the median
```

```
X_train['Fare_median'].plot(kind='kde', ax=ax,
color='red')

# variable imputed with the mean
X_train['Fare_mean'].plot(kind='kde', ax=ax,
color='green')

# add legends
lines, labels = ax.get_legend_handles_labels()
ax.legend(lines, labels, loc='best')
```

#Check Covariance

```
X_train.cov()
```

#Check Correlation

```
X_train.corr()
```

#Box Plot for Age

```
X_train[['Age', 'Age_median',
'Age_mean']].boxplot()
```

#Box Plot for Fare

```
X_train[['Fare', 'Fare_median',
'Fare_mean']].boxplot()
```

- **Arbitrary Value Imputation:** Arbitrary value imputation consists of replacing all occurrences of missing values within a variable with an arbitrary value.

For Categorical Data-

Using the word 'Missing' as a category to replace the missing values

ID	City	Degree	Married ?
1	Lisbon	NaN	0
2	Berlin	Bachelor	1
3	Lisbon	NaN	1
4	Lisbon	Bachelor	1
5	Berlin	Bachelor	0
6	Lisbon	Bachelor	0
7	Berlin	Masters	1
8	Berlin	No Degree	0
9	Berlin	Masters	1
10	Madrid	Masters	1

ID	City	Degree	Married ?
1	Lisbon	Missing	0
2	Berlin	Bachelor	1
3	Lisbon	Missing	1
4	Lisbon	Bachelor	1
5	Berlin	Bachelor	0
6	Lisbon	Bachelor	0
7	Berlin	Masters	1
8	Berlin	No Degree	0
9	Berlin	Masters	1
10	Madrid	Masters	1

For Numerical Data-

Arbitrary Value = -1

ID	City	Age	Married ?
1	Lisbon	25	0
2	Berlin	25	1
3	Lisbon	30	1
4	Lisbon	30	1
5	Berlin	18	0
6	Lisbon	NaN	0
7	Berlin	30	1
8	Berlin	NaN	0
9	Berlin	25	1
10	Madrid	25	1

ID	City	Age	Married ?
1	Lisbon	25	0
2	Berlin	25	1
3	Lisbon	30	1
4	Lisbon	30	1
5	Berlin	18	0
6	Lisbon	-1	0
7	Berlin	30	1
8	Berlin	-1	0
9	Berlin	25	1
10	Madrid	25	1

<Start Coding | Arbitrary Value Imputation >

#Import Libraries

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

#Import Dataset

```
df = pd.read_csv('titanic_toy.csv')
```

```
df.head()
```

```
df.info(
```

#Perform Train Test Split

```
X = df.drop(columns=['Survived'])
y = df['Survived']
```

```
X_train,X_test,y_train,y_test =
train_test_split(X,y,test_size=0.2,random_state=2
)
```

```
X_train['Age_99'] = X_train['Age'].fillna(99)
X_train['Age_minus1'] = X_train['Age'].fillna(-1)

X_train['Fare_999'] = X_train['Fare'].fillna(999)
X_train['Fare_minus1'] = X_train['Fare'].fillna(-
1)
```

#Create New Column and Replace Value (Age-99 | Fare-999)

```
X_train['Age_99'] = X_train['Age'].fillna(99)
X_train['Age_minus1'] = X_train['Age'].fillna(-1)

X_train['Fare_999'] = X_train['Fare'].fillna(999)
X_train['Fare_minus1'] = X_train['Fare'].fillna(-
1)
```

#Review variance

```
print('Original Age variable variance: ',
X_train['Age'].var())
print('Age Variance after 99 use imputation: ',
X_train['Age_99'].var())
print('Age Variance after -1 use imputation: ',
X_train['Age_minus1'].var())

print('Original Fare variable variance: ',
X_train['Fare'].var())
print('Fare Variance after 999 use imputation: ',
X_train['Fare_999'].var())
print('Fare Variance after -1 use imputation: ',
X_train['Fare_minus1'].var())
```

#Review Distribution - Age:

```
fig = plt.figure()
ax = fig.add_subplot(111)

# original variable distribution
X_train['Age'].plot(kind='kde', ax=ax)

# variable imputed with the median
X_train['Age_99'].plot(kind='kde', ax=ax,
color='red')

# variable imputed with the mean
X_train['Age_minus1'].plot(kind='kde', ax=ax,
color='green')

# add legends
lines, labels = ax.get_legend_handles_labels()
ax.legend(lines, labels, loc='best')
```

#Review Distribution –Fare:

```
fig = plt.figure()
ax = fig.add_subplot(111)

# original variable distribution
X_train['Fare'].plot(kind='kde', ax=ax)

# variable imputed with the median
X_train['Fare_999'].plot(kind='kde', ax=ax,
  color='red')

# variable imputed with the mean
```

```
X_train['Fare_minus1'].plot(kind='kde', ax=ax,
  color='green')

# add legends
lines, labels = ax.get_legend_handles_labels()
ax.legend(lines, labels, loc='best')
```

#Check Covariance

```
X_train.cov()
```

#Check Correlation

```
X_train.corr()
```

Day 11: Curious Data Minds

How generative AI-ML and Voice Technology will transform?

https://www.fastcompany.com/90909940/speak-listen-do-how-generative-ai-and-voice-technology-will-transform-how-we-live-and-work

Indian Singer Arjit Singh patent voice

https://www.moneycontrol.com/news/trends/arijit-singh-aims-to-patent-his-voice-says-ai-is-a-tool-not-threat-video-12023771.html#google_vignette

https://www.hindustantimes.com/entertainment/bollywood/the-ethics-of-using-ai-to-generate-dead-artists-voices-industry-experts-weigh-in-on-copyright-and-privacy-issues-101687521121515.html