Data Science | 30 Days of Machine Learning | Day - 14

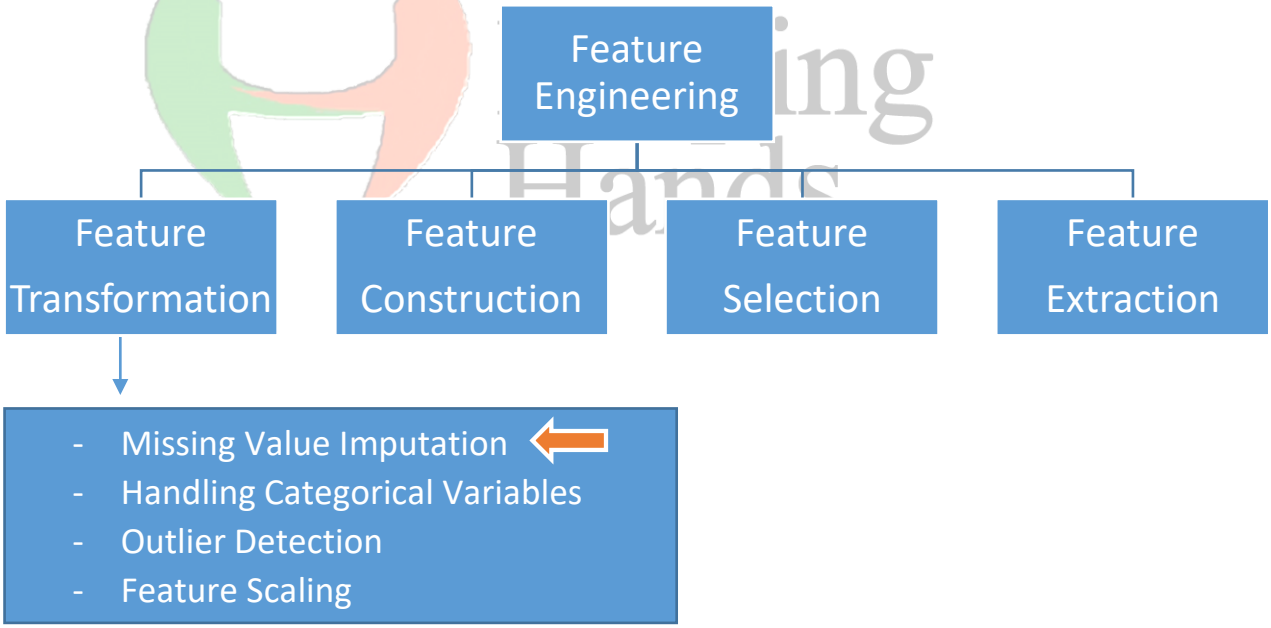
Educator Name: Nishant Dhote
Support Team: **+91-7880-113-112**

----Today Topics | Day 14----

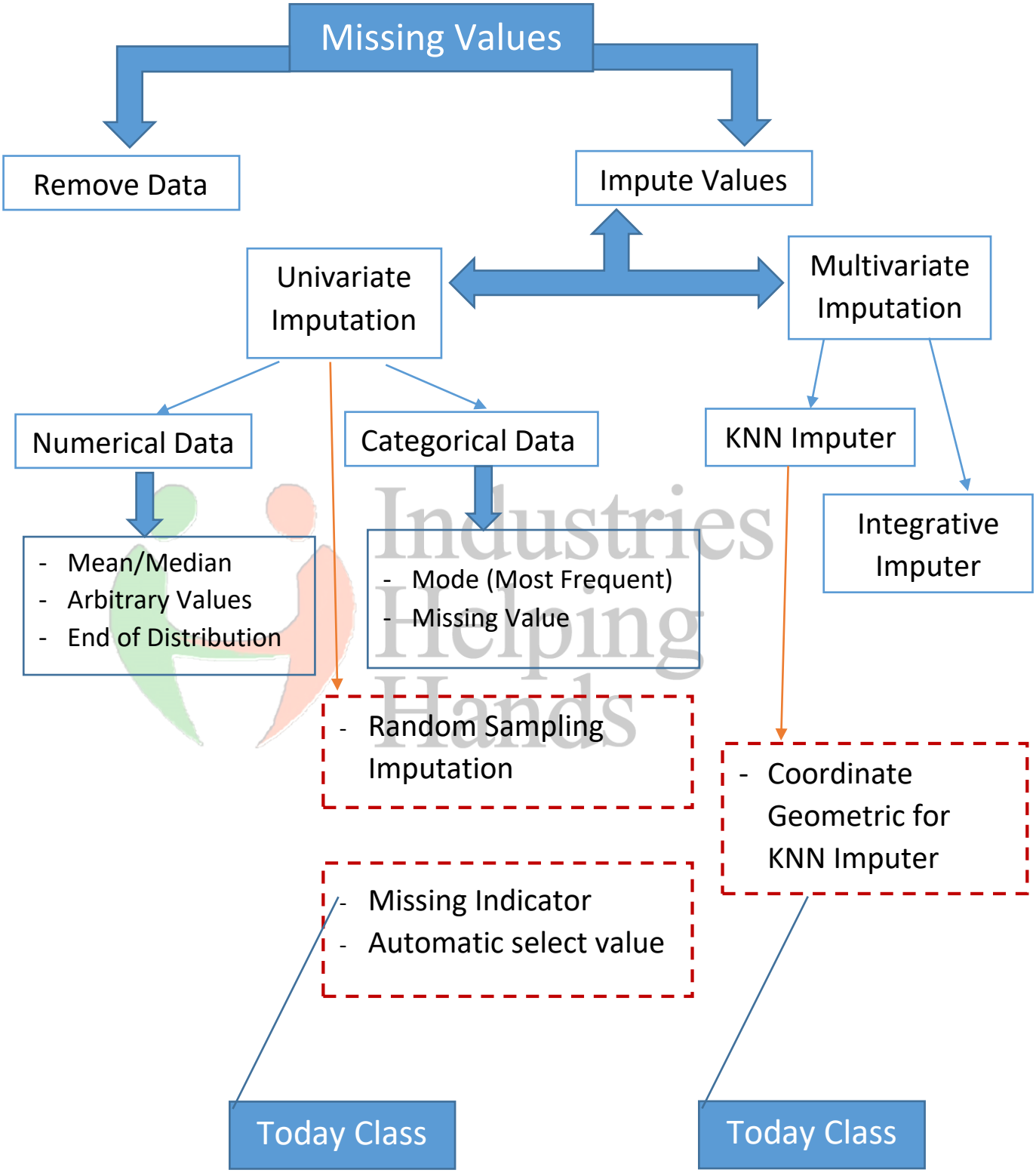
Feature Engineering (Missing Value Imputation)

- Missing Indicator in Univariate Imputation
- Automatic select value for Imputer parameter
- Coordinate Geometric for KNN Imputer
- Calculation in 2D and 3D Distance

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



Today's Topics:



- **Missing Indicator in Univariate Imputation**

The Missing Indicator Method (MIM), which adds indicator variables to indicate the missing pattern, can be used in conjunction with imputation to improve model performance. While commonly used in data science, MIM is surprisingly understudied from an empirical and especially theoretical perspective. In this paper, we show empirically and theoretically that MIM improves performance for informative missing values, and we prove that MIM does not hurt linear models asymptotically for uninformative missing values.

Example:

Age	Fare	Age-NA
15	58	False
18	70	False
21	58	False
N/A	55	True
22	58	False
15	55	False
18	70	False

Today Class we use Titanic Dataset

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

- <Start Coding | Missing Indicator Imputation >

#Import Libraries

```
import numpy as np
import pandas as pd

from sklearn.model_selection import
train_test_split

from sklearn.impute import
MissingIndicator,SimpleImputer
```

#Import Dataset

```
df =
pd.read_csv('train.csv',usecols=['Age','Fare','Su
rvived'])
----
df.head()
```

#Create X & Y

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

#Train & Test Split

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

```
----
```

```
X_train.head()
```

#Review Without "Missing Indicator Method" Technique

```
si = SimpleImputer()
```

```
X_train_trf = si.fit_transform(X_train)
```

```
X_test_trf = si.transform(X_test)
```

```
----
```

```
X_train_trf
```

#Call Logistic Regression

```
from sklearn.linear_model import
LogisticRegression
```

```
clf = LogisticRegression()
```

```
clf.fit(X_train_trf,y_train)
```

```
y_pred = clf.predict(X_test_trf)
```

```
from sklearn.metrics import accuracy_score
```

```
accuracy_score(y_test,y_pred)
```

#Define Missing Indicator

```
mi = MissingIndicator()

mi.fit(X_train)

----
MissingIndicator()

----
mi.features_
```

#Transform: Train Missing

```
X_train_missing = mi.transform(X_train)
----
X_train_missing
```

#Transform: Test Missing

```
X_test_missing = mi.transform(X_test)
----
X_test_missing
```

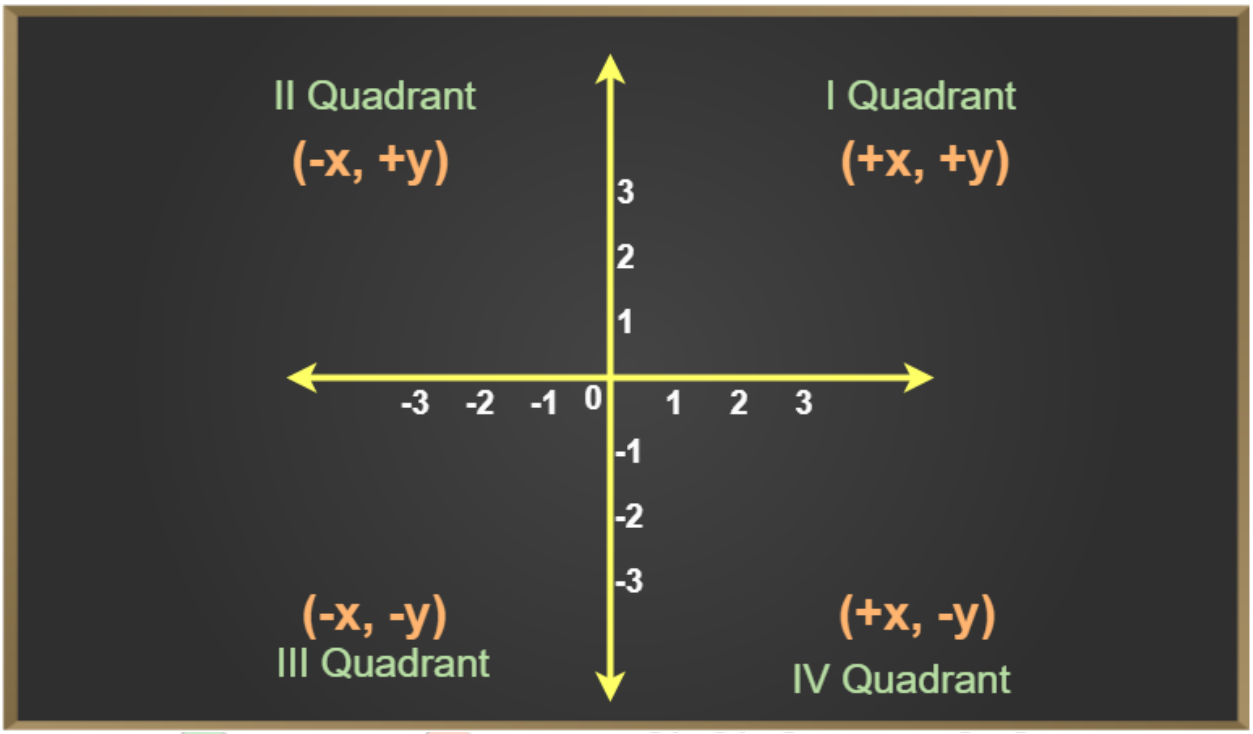
#Create New Column

```
X_train['Age_NA'] = X_train_missing
----
X_train
```

#Writing Code again and Check Accuracy

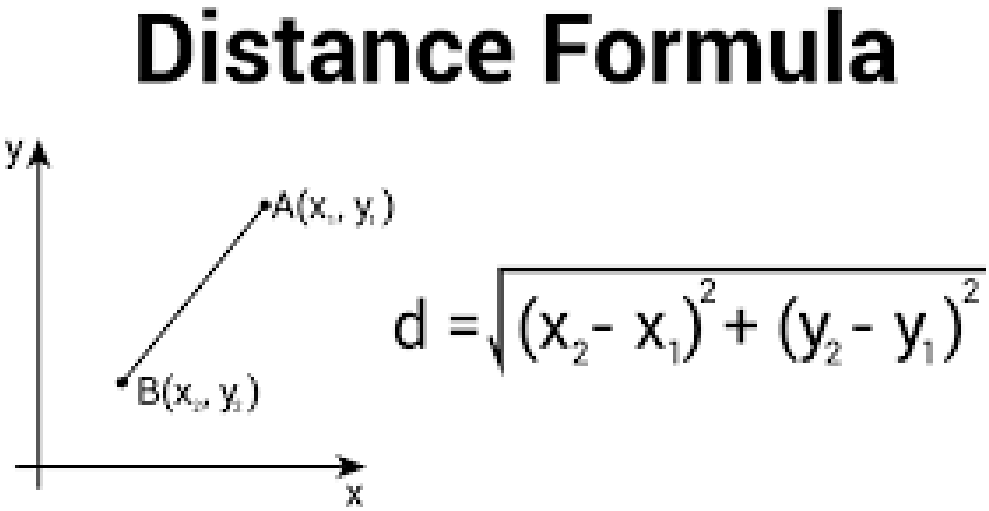
```
si = SimpleImputer()

X_train_trf2 = si.fit_transform(X_train)
X_test_trf2 = si.transform(X_test)


----
from sklearn.linear_model import
LogisticRegression

clf = LogisticRegression()

clf.fit(X_train_trf2,y_train)

y_pred = clf.predict(X_test_trf2)

from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

- **Coordinate Geometric for KNN Imputer**



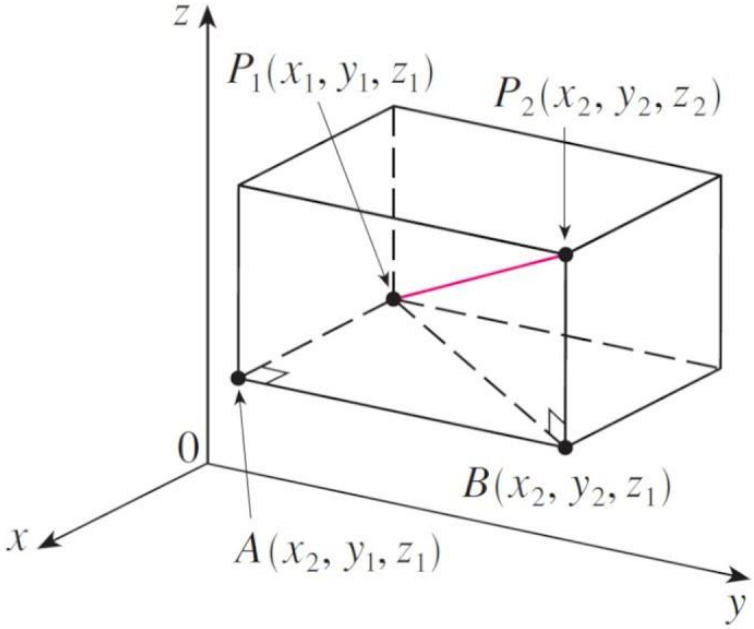
Basic Formulas : https://www.geeksforgeeks.org/coordinate-geometry/

2D – Distance Formula:



3D – Distance Formula:

$$|P_1P_2| = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$



Distance Formula in 3D

distance between points

2D: $\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}$

3D: $\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2}$

4D: $\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2 + (a_1 - a_0)^2}$

nD: $\sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2 + (a_1 - a_0)^2 + \cdots}$

Example:

S.No	Variable 1	Variable 2	Variable 3	Variable 4
1	28	--	48	22
2	--	40	37	24
3	34	22	55	26
4	26	--	30	--
5	50	20	49	--

Distance between 2 points formula

Consider two points A (x_1, y_1) and B(x_2, y_2) on the given coordinate axis. The distance between these points is given as:

Distance Between 2 Points Formula in 3D

The distance between two points $P(x_1, y_1, z_1)$ and $Q(x_2, y_2, z_2)$

$PQ = \sqrt{[(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2]}$

Distance Between 2 Points Formula Derivation in 3D

Let the points $P(x_1, y_1, z_1)$ and $Q(x_2, y_2, z_2)$ be referred to a system of rectangular axes OX,OY and OZ as shown in the figure.

sklearn.metrics.pairwise.nan_euclidean_distances

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.pairwise.nan_euclidean_distances.html#sklearn.metrics.pairwise.nan_euclidean_distances

Day 14: Curious Data Minds

Suggest Topic in Comment Box?