Feature Engineering

Feature Engineering is the process of using domain knowledge to extract features from raw data. These features can be used to improve the performance of machine learning algorithms.

What is Feature Scaling?

Feature scaling is a technique to standardize the independent features present in the data in a fixed range.

age	salary	purchase
70	10000	
40	36000	
30	40000	

10 scale

1000 scale

different scale

Why do we need feature scaling?

age	Salary	Purchase
50	83000	1
27	48000	0

if we use KNN and find the distance between $\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$ → age → 529

→ salary → 1225000000

→ dominant

Types of Feature Scaling

Feature Scaling

1. Standardization
2. Normalization
   a) Min-Max
   b) Robust Scaler

Standardization – Intuition

Also called as Z-score Normalization

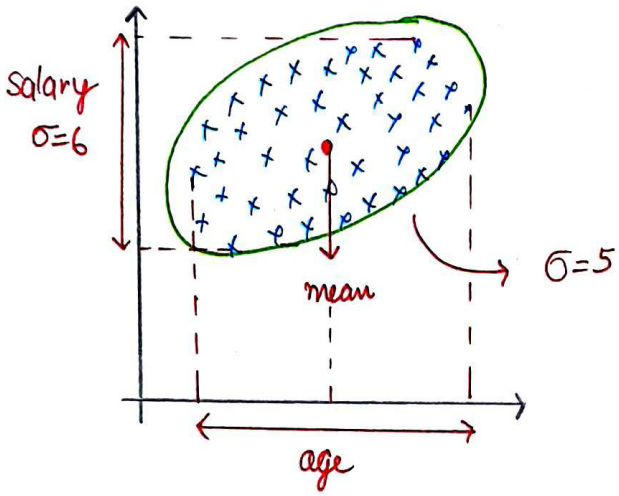
Age	salary
27	-
15	-
33	-
63	-
90	-
05	-
⋮	⋮
500 value	-

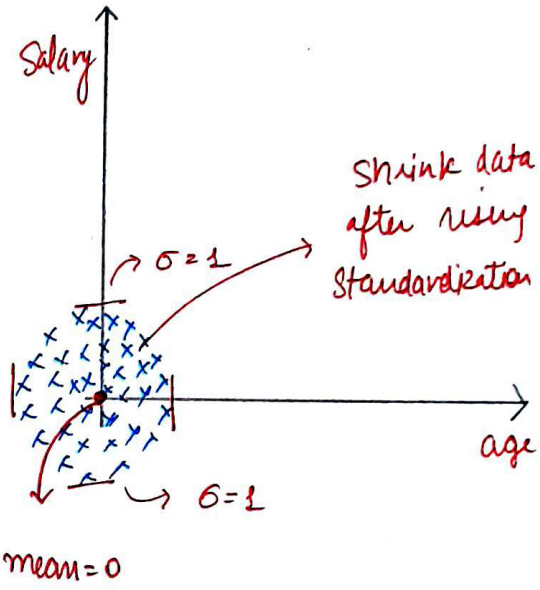
$$x_i' = \frac{x_i - \bar{x}}{\sigma}$$

$\bar{x}$ → mean

σ → Standard Deviation

* After scale the column and new scaled column $\mu = 0, \sigma = 1$ → Always (scaled col^n)





Impact of Outlier

There is not any impact of outlier on data.

When to use standardization?

1. K-Mean
2. K-Nearest Neighbours
3. PCA
4. Artificial Neural Network
5. Gradient Descent

Normalization

Normalization is a technique often applied as part of data preparation for machine learning. The goal of normalization is to change the values of numeric columns in the datasets to use a common scale, without distorting differences in the ranges of value or losing information.

1. Min Max scaling
2. Mean Normalization
3. Max
4. Robust scaling

MinMax Scaling

Weight (in kg)

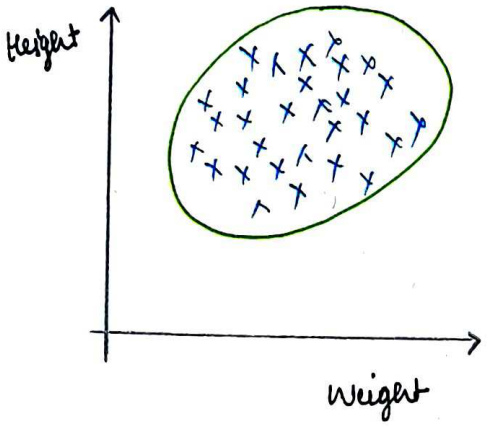
130
67
81
61
32
⋮

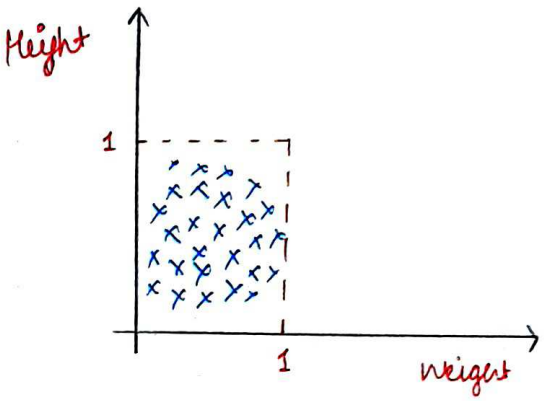
Normalize → Min Max Scaling →

$$X_i' = \frac{X_i - X_{min}}{X_{max} - X_{min}}$$

$$= \frac{130 - 32}{130 - 32} = 1$$

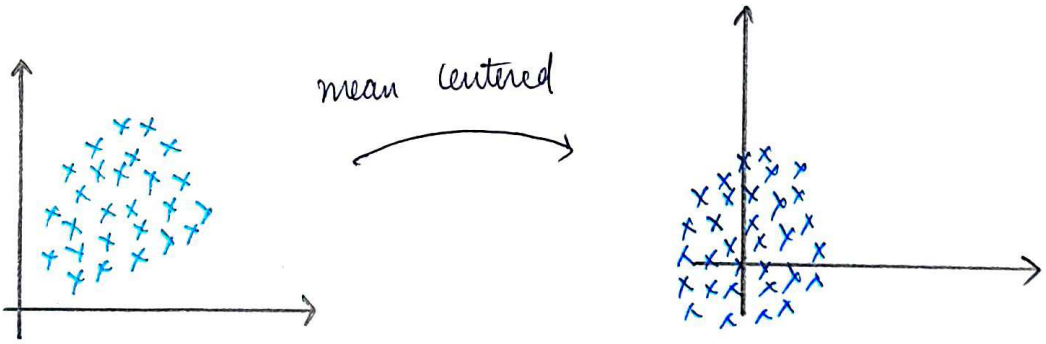
Height and weight





Mean Normalization

$$X_i' = \frac{X_i - X_{mean}}{X_{max} - X_{min}} \rightarrow [-1 \text{ to } 1]$$



Max Absolute Scaling

weight
200
100
300

$$X_i' = \frac{X_i}{|X_{max}|} \rightarrow$$

scikit learn

Max Abs scaler

use when data has many 0.

Robust Scaling

weight
200
300
100

$$X_i' = \frac{X_i - X_{median}}{IQR(75^{th}\ per - 25^{th}\ per)}$$

* If Data have lots of outlier. (Try)

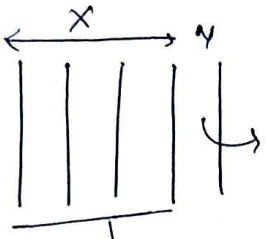
Normalization Vs standardization

1.> Is feature scaling required

2.> Min Max

image → CNN → 0-225 → min max

Encoding categorial Variables



Categorial → Yes / No
↳ Output data use Label encoding

In X categorial use
Ordinal encoding

Ordinal Encoding

Education

HS → 0
UG → 1
PG → 2
PG → 2
UG → 1
HS → 0
UG – 2

PG > UG > HS

* There is a order in data so we use Ordinal encoding Otherwise OneHot encoding can be use.

{ PG → 2
UG → 1
HS → 0 } → We have define this order.

OneHot Encoding

↳ dummy variable trap

color

Yellow
blue
Red

color-Y	color-B	color-R
1	0	0
0	1	0
0	0	1

n-1 cols
↳ accept

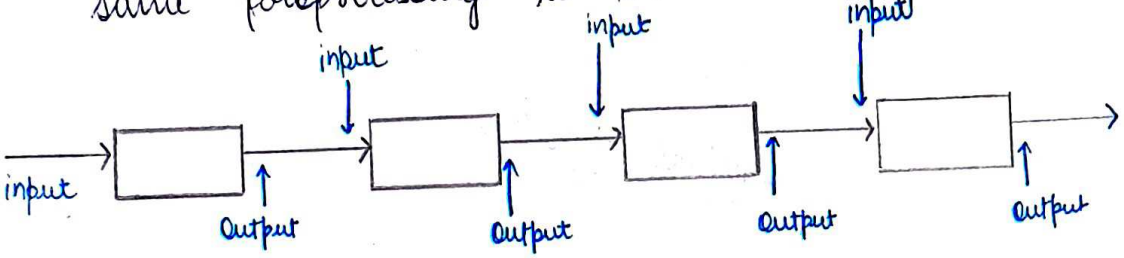
Multicollinearity

	X	
Y →	1	0 0
B →	0	1 0
R →	0	0 1

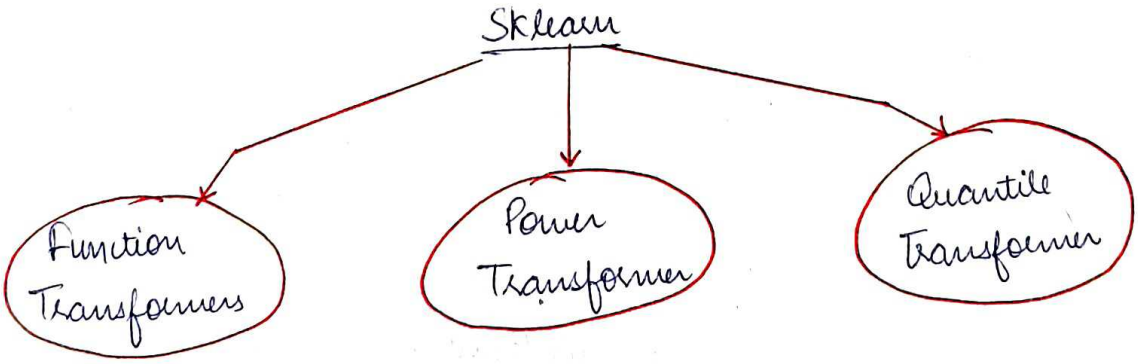
Scikit learn Pipelines

Pipelines chains together multiple steps so that the output of each step is used as input to the next step

Pipelines makes it easy to apply the same same preprocessing to train and test.



function Transformer

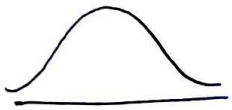


1. Log Transformer
2. Reciprocal
3. Square Root / Square
4. Custom

1. Box Cox
2. Yeo-Johnson

How to find if data is normal?

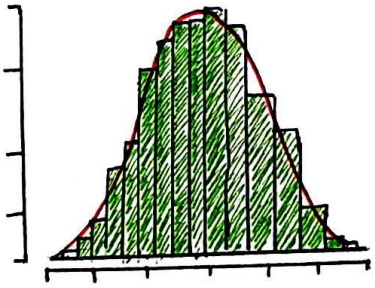
1) Sns.distplot



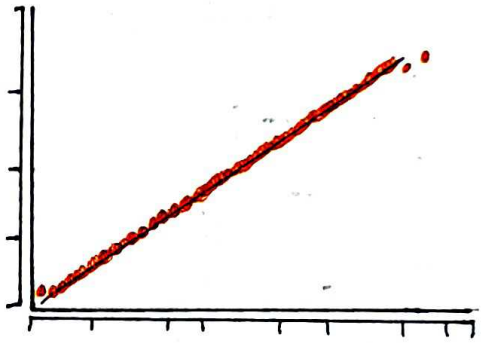
2) Pd.skew() = 0

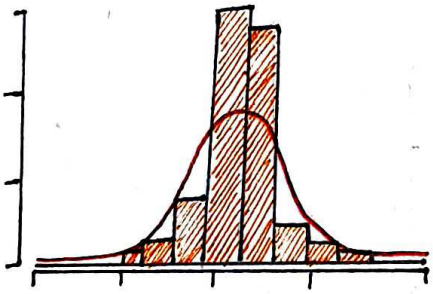
3) Q-Q plot

QQplot



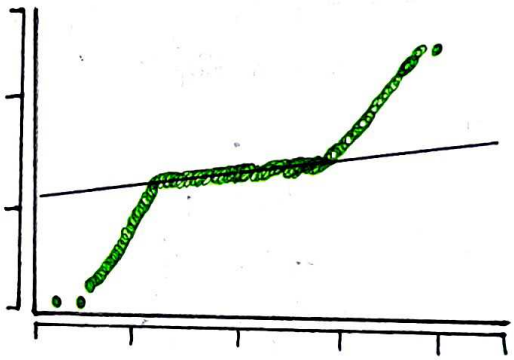
Normally distributed data

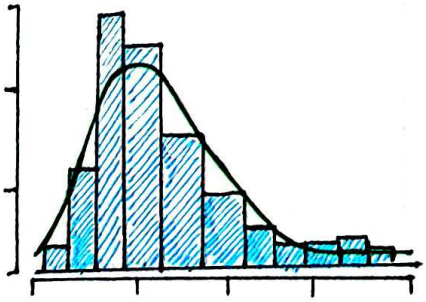




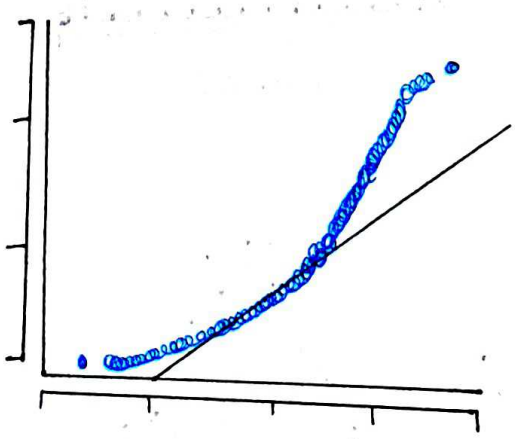
~~Skewed~~ ~~Data~~

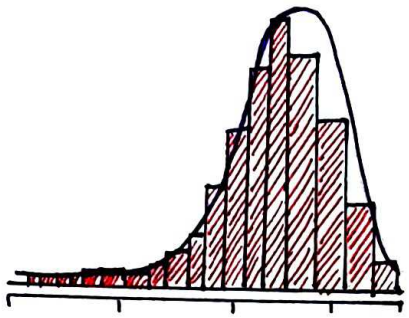
Data too peaked in middle



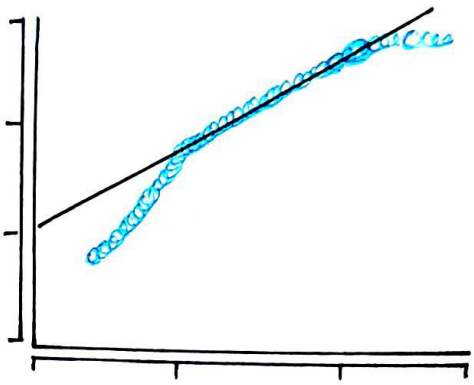


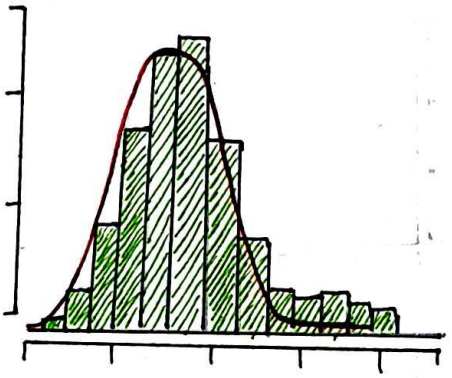
Skewed Data



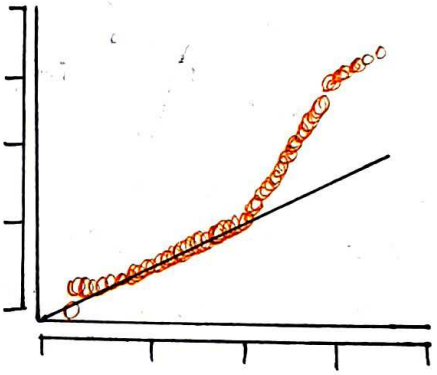


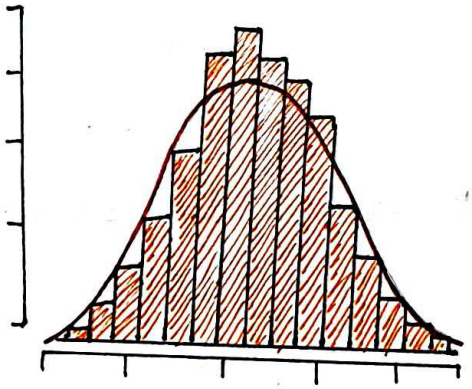
Skewed left



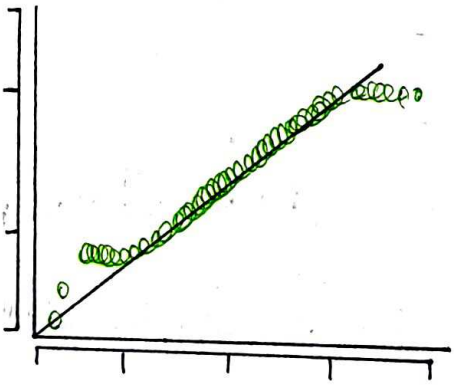


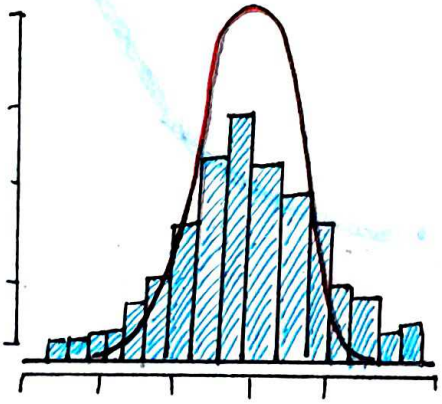
Skewed Right



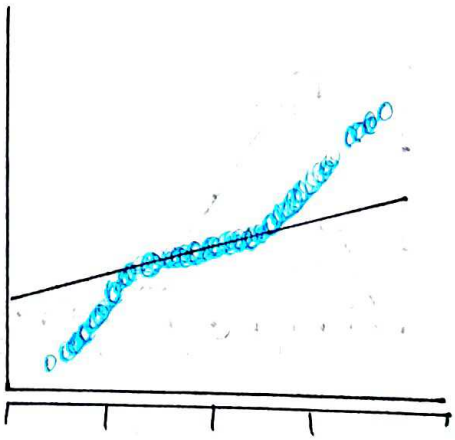


Thin tails





Fat Tails



Log Transform

Age

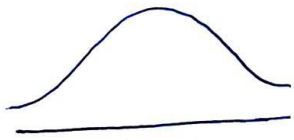
21 → (log) → np.log1p

35

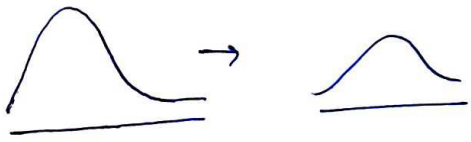
45

⋮

↳ * If data have 0 then always add 1 then perform log.



* log → not (-ve data)
* Right skewed $\xrightarrow{log}$ center



Other Transform

Reciprocal $1/x$ | Sq (x^2) ↓ Left skewed | Sqrt $(\sqrt{x})$

Power Transformer

Box Cox Transform

The exponent here is a variable called Lambda (λ) that varies over the range of -5 to 5, and in the process of searching, we examine all values of λ. Finally we choose the optimal value (resulting in the best approximation to a normal distribution) for your variable.

$$x_i^{(\lambda)} = \begin{cases} \dfrac{x_i^{\lambda} - 1}{\lambda} & \text{if } \lambda \neq 0, \\ \ln(x_i) & \text{if } \lambda = 0, \end{cases}$$

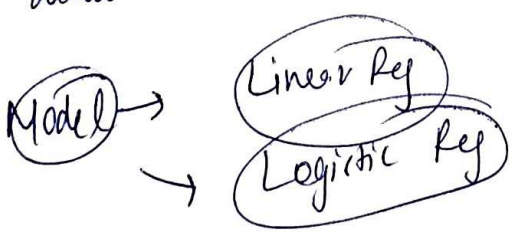
* Box cox → only applicable for $\boxed{n > 0}$

Yeo - Johnson Transform

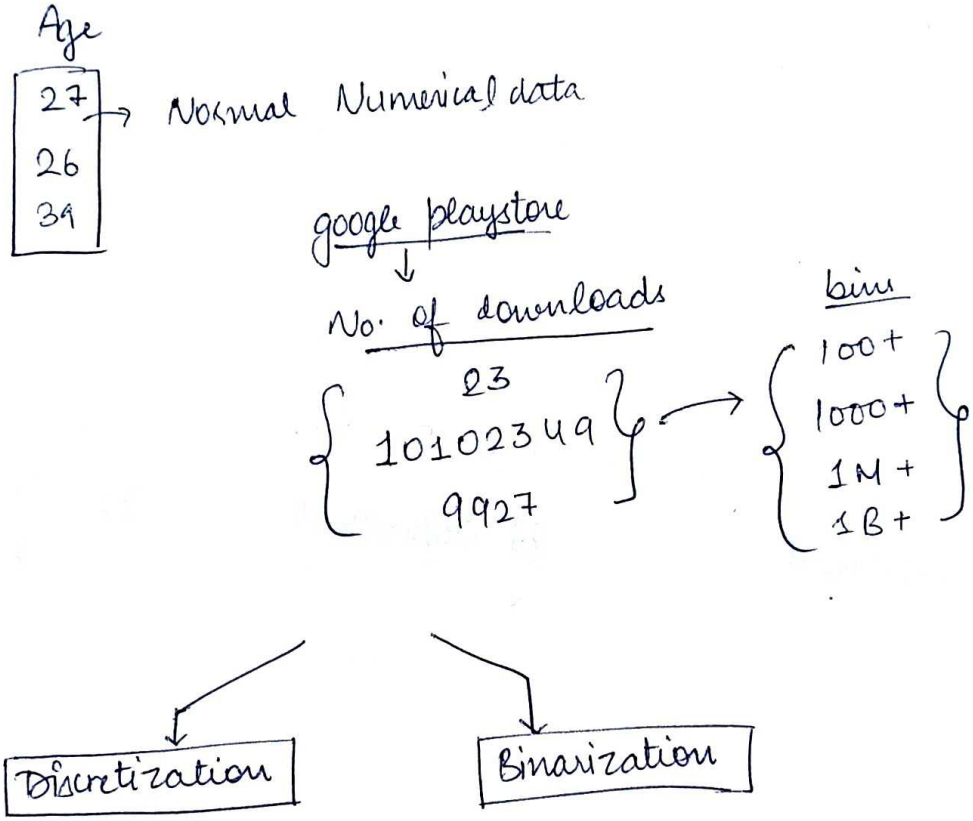
This transformation is somewhat of an adjustment to the Box-Cox transformation, by which we can apply it to negative numbers.

$$x_i^{(\lambda)} = \begin{cases} [(x_i+1)^{\lambda} - 1]/\lambda & \text{if } \lambda \neq 0, x_i \geq 0; \\ \ln(x_i)+1 & \text{if } \lambda = 0, x_i \geq 0. \\ -[(-x_i+1)^{2-\lambda} - 1]/(2-\lambda) & \text{if } \lambda \neq 2, x_i < 0; \\ -\ln(-x_i+1) & \text{if } \lambda = 2; x_i < 0 \end{cases}$$

* This transform can work with -ve and 0 and positive data.



Encoding Numerical Features



Discretization

Discretization is the process of tranforming continous variables into discrete variables by creating a set of contiguous intervals that span the range of the variable's values. Discretization is also called binning, where bin is an alternative name for interval.

Why use Discretization:

1. To handle Outliers
2. To improve the value spread

Age

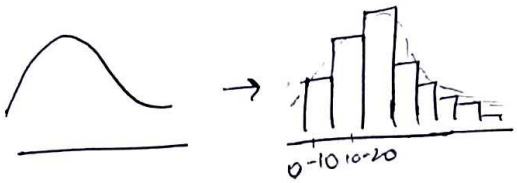
23, 42, 57, 81, ... 100

↓

0-10, 10-20, 20-30

↳ bins ↳ (6) ↳ (10)

↳ check how many value bet^{n} 0-10 (5)



Types of Discretization

1.] Unsupervised

a) Equal width (uniform)

b) Equal frequency (quantile)

c) K-means binning

2.] Supervised

a) Decision Tree (Binning)

3.] Custom Binning

Equal width / uniform Binning

Age

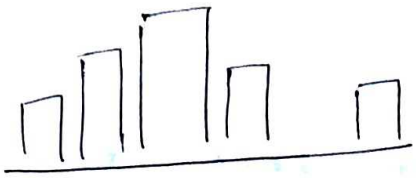
27, 32, 84, 56, ... max 100

Bins = 10

$$\frac{max - min}{bins} = \frac{100 - 0}{10} = 10$$

(0-10), (10-20), (20-30) .... (90-100)

Frequency: 5, 16, 17, ..., 5



→ equal binning

age	age-trf	age-label
43.0	5.0	(40.21, 48.16)
44.0	5.0	(40.21, 48.16)
15.0	1.0	(8.3, 16.3)
30.0	3.0	(24.9, 32.2)
35.0	4.0	(32.2, 40.21)
35.0	0.0	(0.4, 8.3)
2.0	2.0	(16.3, 24.9)
18.0	0.0	(0.4, 8.3)

Advantage

1.) Outlier handle

2.) No change spread

Equal Frequency / Quantile Binning

intervals = 10

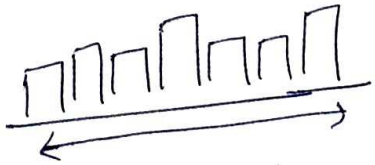
Each interval contain 10% of total observations

intervals:

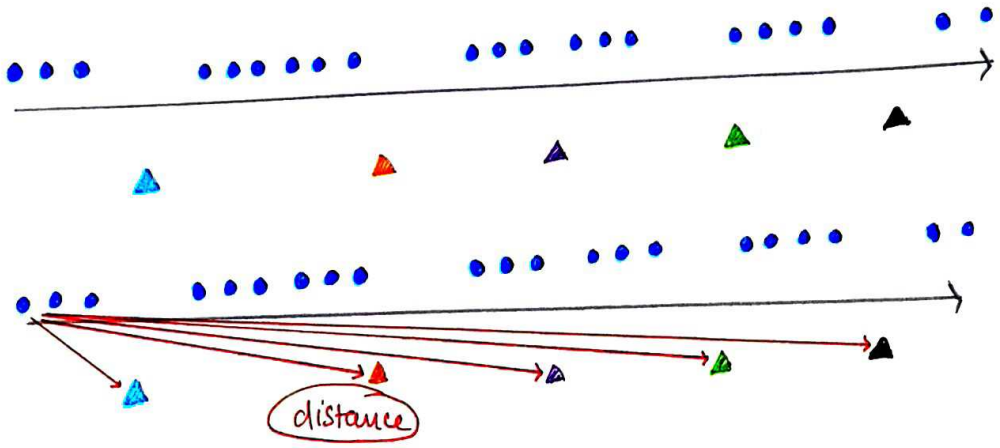
0-16	16-20	20-22
10% data	20% data	30% data

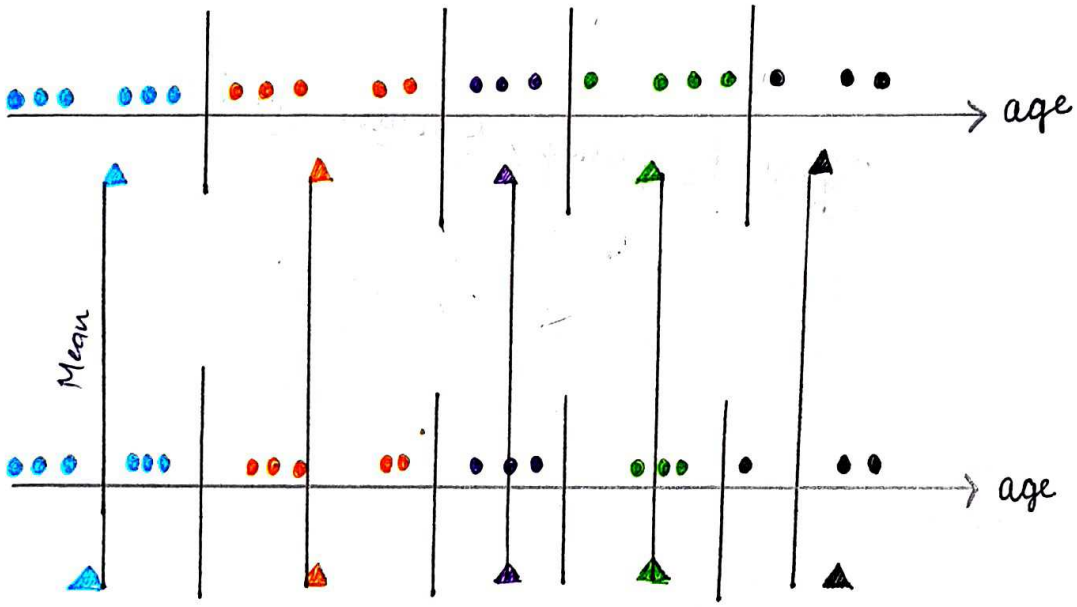
→ Handle outlier

→ value spread Uniform



KMeans Binning





Encoding the discretized variable

Sk learn

↳ KBinsDiscretizer( )

- bins=?
- Strategy
  - uniform
  - quantile
  - kmeans
- encoding
  - ordinal
  - OHE

Custom / Domain Based Binning

[0-18] → kids
[18-60] → adult
[60-80] → Old

} → Pandas

Mixed Data

Mixed → Cabin | Data

Data
B5
D41
B6

→

Categorical	Numerical
B	5
D	41
B	6

Data
7
3
1
A
C
4
D

→

Num	Data (Categorical)
7	NA
3	NA
1	NA
NA	A
NA	C
4	NA
NA	D