What is Discretization?

15 February 2024 06:35

Discretization or Binning in the context of machine learning and feature engineering is a process that involves converting continuous features or variables into discrete or categorical ones.

Person	Age	Income
A	23	30,000
B	49	80,000
C	35	45,000
D	60	95,000
E	18	15,000

cat cols

18-60

15-30

30-45

45-60

Person	Age Bin		Income Bin
A	0-30 years	1	Below 45,000
B	46+ years	3	45,000 and above
C	31-45 years	2	45,000 and above
D	46+ years	3	45,000 and above
E	0-30 years	1	Below 45,000

textual data → mLX

→ disorder

0-30

31-45

45+

& 45K

cat data → numbers

ordinal enc

OHE

0-30 - 1

31-45 - 2

46+ → 3

What → How

Why?

What → Discretization → Binning

How

Why

encoding var cat

cat → num

num → cat / discrete

feature encoding

Binning

encoding cat var

Why learn Discretization?

16 February 2024 08:26

1. Reduces Overfitting

By converting continuous variables into discrete bins, discretization effectively simplifies the feature space. This simplification means the model has fewer nuances to learn from the training data. While this might lead to a loss in detail or granularity, it also means there's less chance for the model to learn noise or overly complex patterns that don't generalize well to unseen data.

Discretization acts as a form of regularization, imposing a constraint on the model's complexity. By reducing the number of unique values a feature can take, it limits the model's ability to fit the training data too closely.

2. Handling non linear relationships

Linear models inherently assume a linear relationship between features and the target variable. Discretization allows these models to approximate non-linear relationships by fitting separate slopes to each bin, which can collectively approximate a non-linear trend.

3. Handling outliers

When you discretize the data, you categorize these continuous values into bins based on their range. An outlier's impact is diluted because it's grouped with other values in the same bin, reducing its ability to disproportionately influence the analysis. Essentially, within each bin, the data points are treated equivalently, regardless of their specific values.

4. Better interpretability

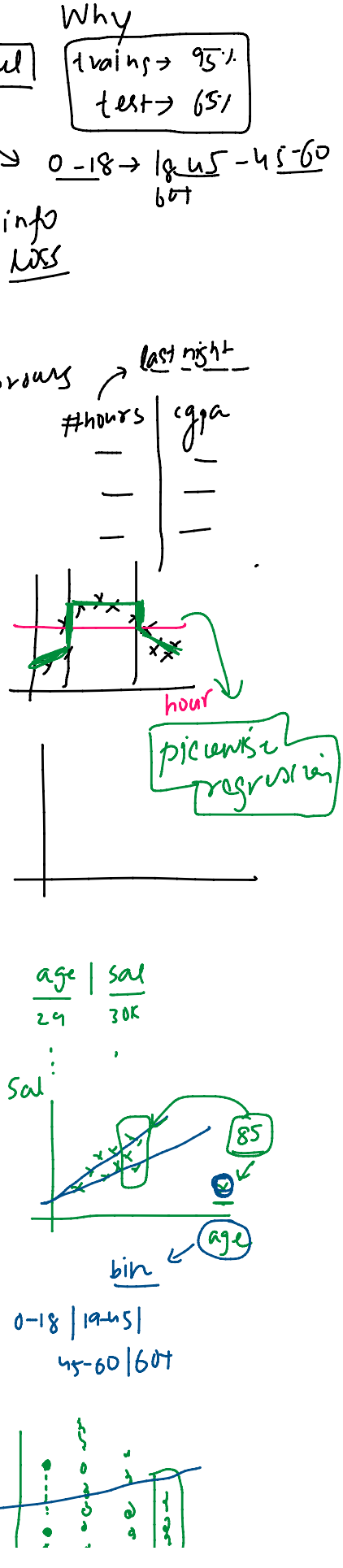
By grouping continuous data into bins, each bin can be treated as a distinct category with its own effect on the model's predictions. This categorical interpretation allows for straightforward explanations, such as "being in age group 30-40 increases the likelihood of buying a new car compared to age group 20-30," which is more intuitive than interpreting the effect of a one-year increase in age.

5. Model Compatibility

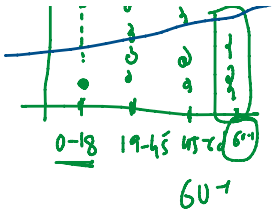
Discretization works particularly well with certain algorithms because it transforms continuous variables into discrete ones, which can align better with the way these algorithms process and interpret data. The effectiveness of discretization largely depends on the nature of the algorithm, the specific data being analysed, and the problem being solved.

Here's why discretization is favourable for some algorithms:

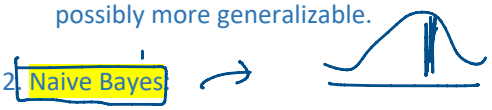
1. Decision Trees and Ensemble Methods:
   - Algorithms like decision trees (and by extension, ensemble methods like Random Forests and Gradient Boosting Machines) inherently split data into branches based on conditions. Discretization can make these splits more meaningful, especially if the continuous data does not have a clear linear relationship with the target variable. Pre-discretized features can lead to simpler trees that are easier to interpret and possibly more generalizable.



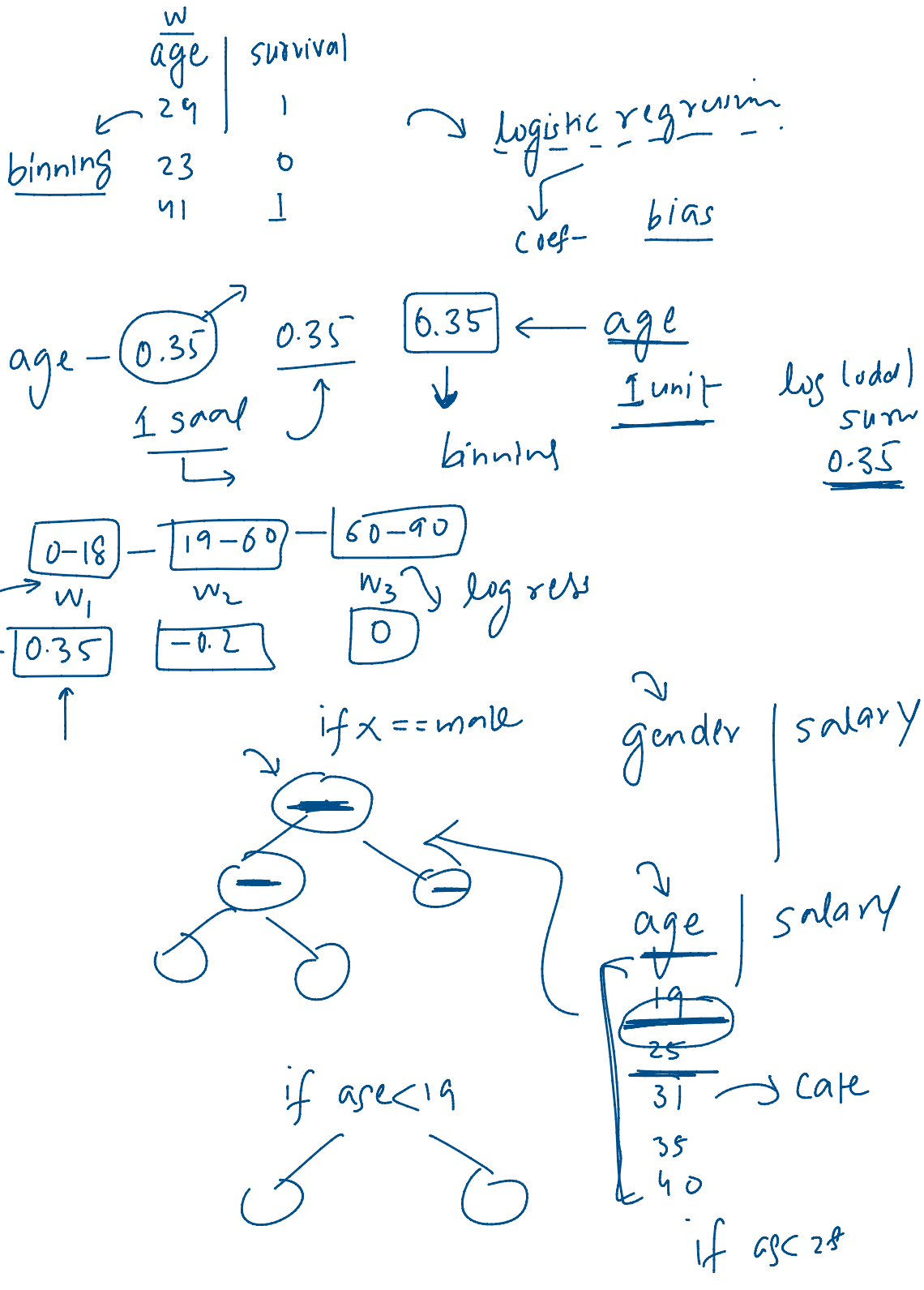
Pre-discretized features can lead to simpler trees that are easier to interpret and possibly more generalizable.



2. Naive Bayes



- Naive Bayes classifiers, particularly in their basic forms, assume that features are independent and often deal better with categorical data. Discretization can help when applying Naive Bayes to continuous data by fitting its assumption of category-based probabilities, potentially improving model performance and interpretability.



if GSC 28

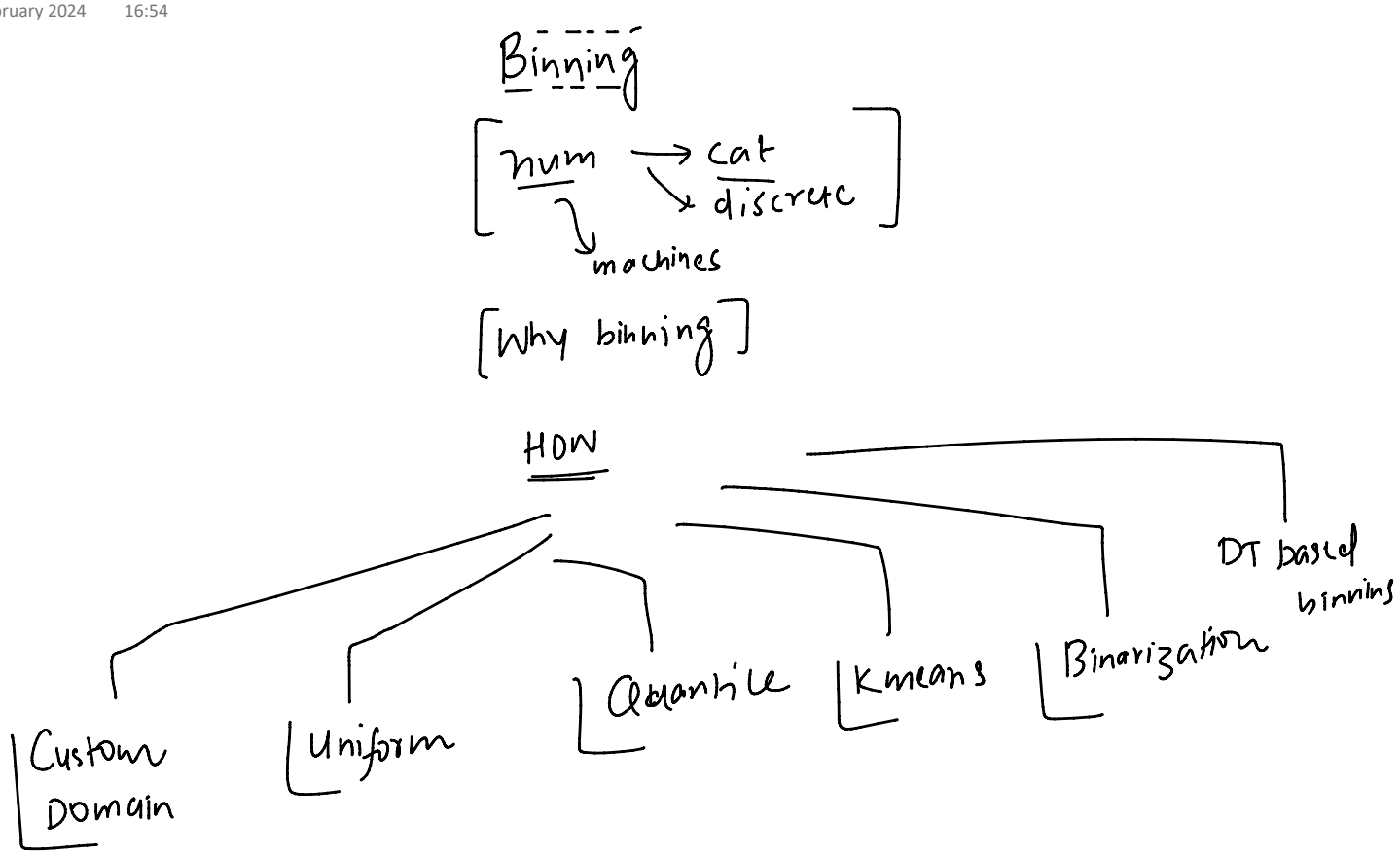
Disadvantages of Discretization

16 February 2024 17:31

1. Loss of information
2. Model Incompatibility
3. Difficulty in choosing bin size

Types of Discretization

16 February 2024 16:54



1. Custom Binning

15 February 2024 06:57

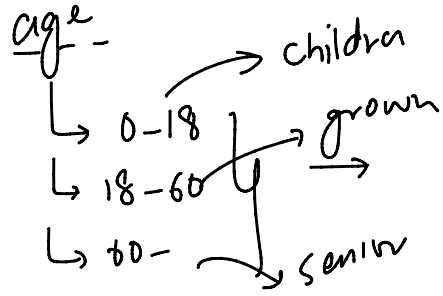
num → domain → bins
↳ cat/discrete

Custom binning, also known as domain binning, is a data pre-processing technique where the bins are defined based on domain knowledge, specific criteria, or predefined thresholds rather than through an automated or algorithmic process. This method allows for the creation of bins that have meaningful interpretations in the context of the specific problem domain or analysis goals.

bins → domain knowledge

Examples

1. Tax Slabs
2. Credit Score for Loan Eligibility
3. Healthcare - BMI Indexing → 33 | 60 | 85 | 85
4. Educational Grading System
5. Air Quality Reporting



2. Uniform Binning

15 February 2024 06:56

sklearn → Kbinsdiscretizer

uniform

↓

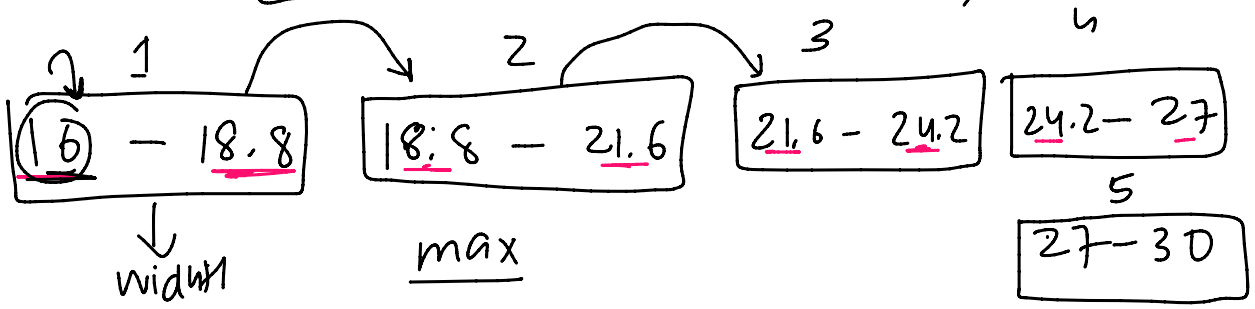
equal width binning

Day	Temperature (°C)
Monday	16
Tuesday	21
Wednesday	24
Thursday	18
Friday	30
Saturday	26
Sunday	22

→ # bins

binsize

bin width = (30 − 16) / 5 = 14 / 5 = 2.8



min → max

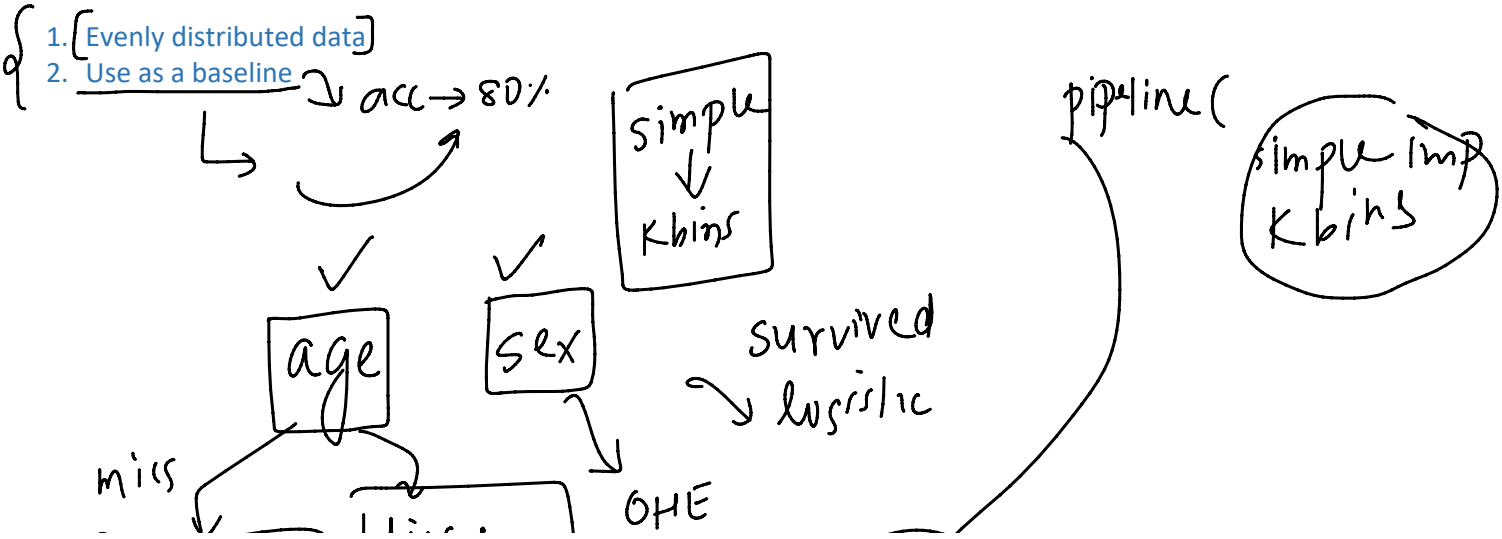
bins → 5 – 6 – 10

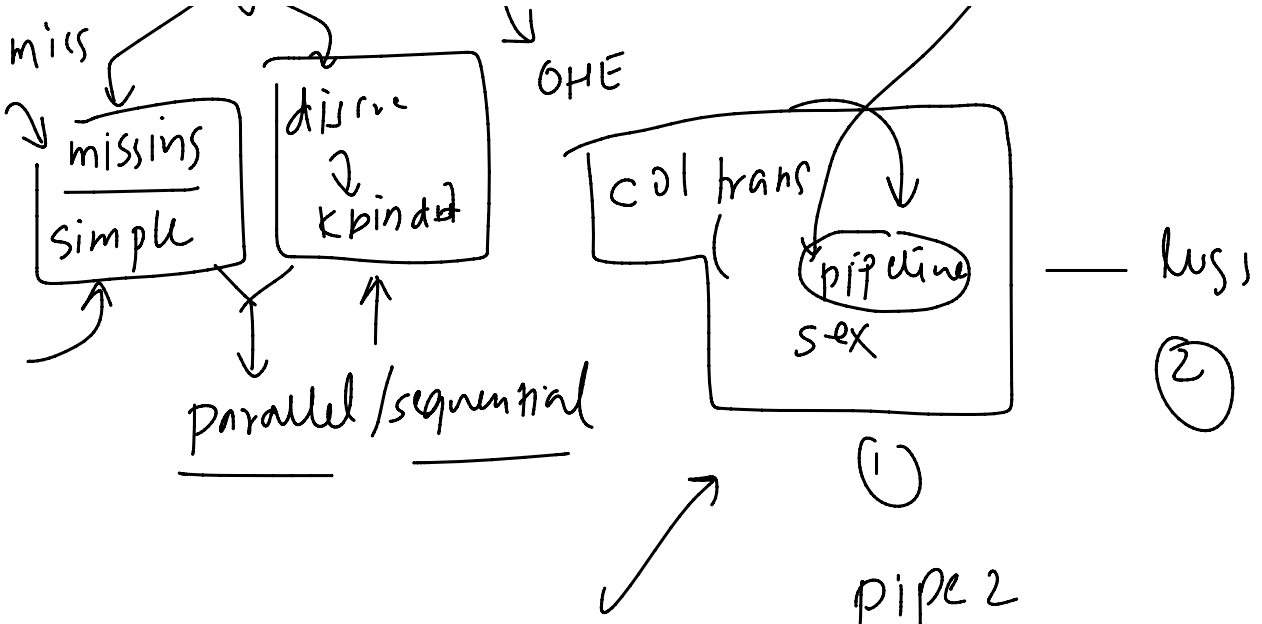
Advantages:

1. Simple
2. Uniform Coverage

When to use:

1. Evenly distributed data
2. Use as a baseline → acc → 80%





pipe 2
(col trans
quasilic rcy

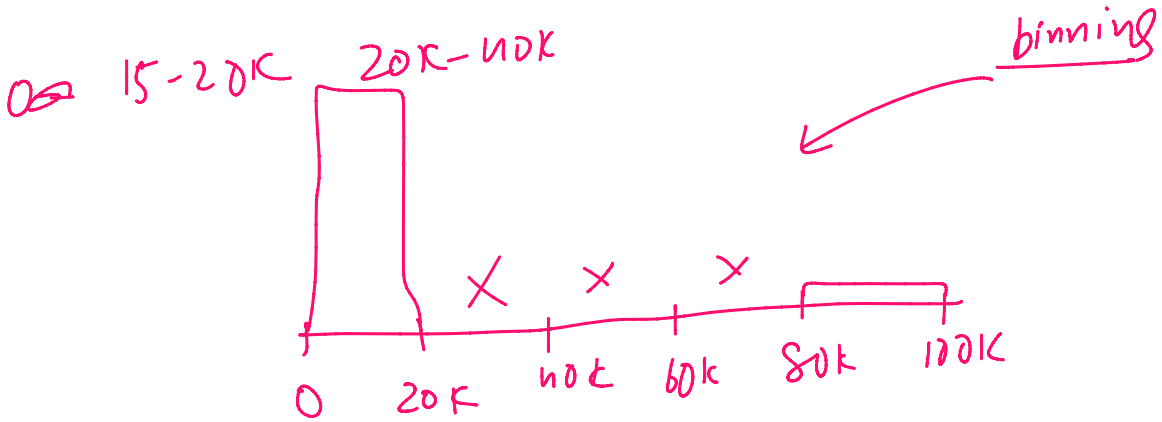
↗ output

32, 15, 16, 72, 81, 19, 21, 100000

#bins → 5

$$\text{bin width} = \frac{100000 - 15}{5} = 20000$$

model



3. Quantile Binning

15 February 2024 06:57

Quantile binning, also known as equal-frequency binning, is a method of binning continuous variables into categories with an equal number of data points. Unlike uniform binning, which divides the range of the data into intervals of equal size, quantile binning divides the data such that each bin has the same number of observations, regardless of the interval width. This approach is particularly useful for dealing with skewed data or when the aim is to normalize the distribution of the data for further analysis.

```
Data: 25, 30, 45, 22, 34, 28, 55, 43, 38, 31, 49, 27
```

```
Sorted Data: 22, 25, 27, 28, 30, 31, 34, 38, 43, 45, 49, 55
```

Advantages:

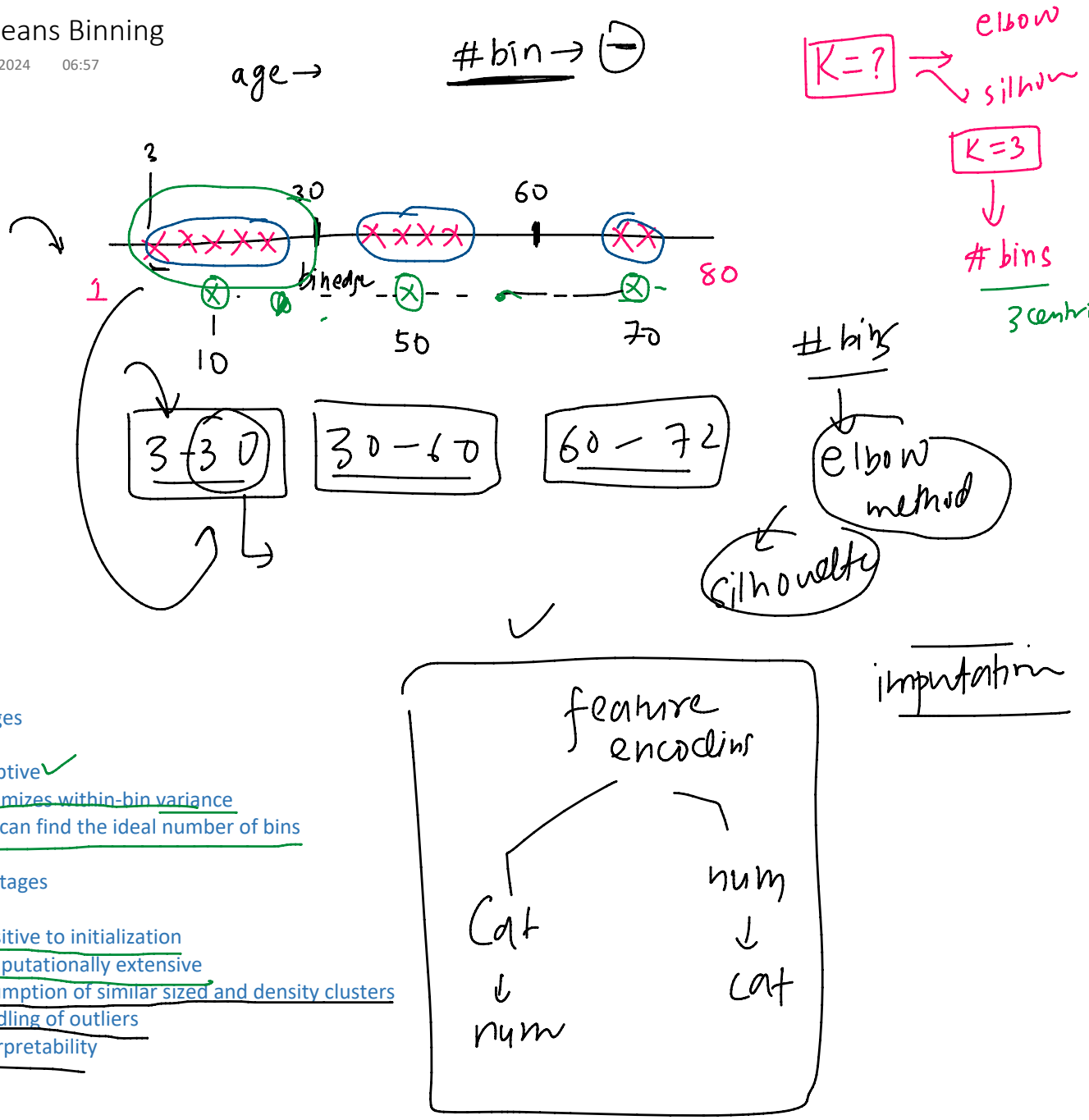
1. Mitigates the impact of outlier
2. Handles skewed distribution

Disadvantages

1. Difficulty in bins interpretation
2. True info about the data distribution is lost.
3. Finding number of bins is still a challenge
4. Computationally expensive

4. K-Means Binning

15 February 2024 06:57



Advantages

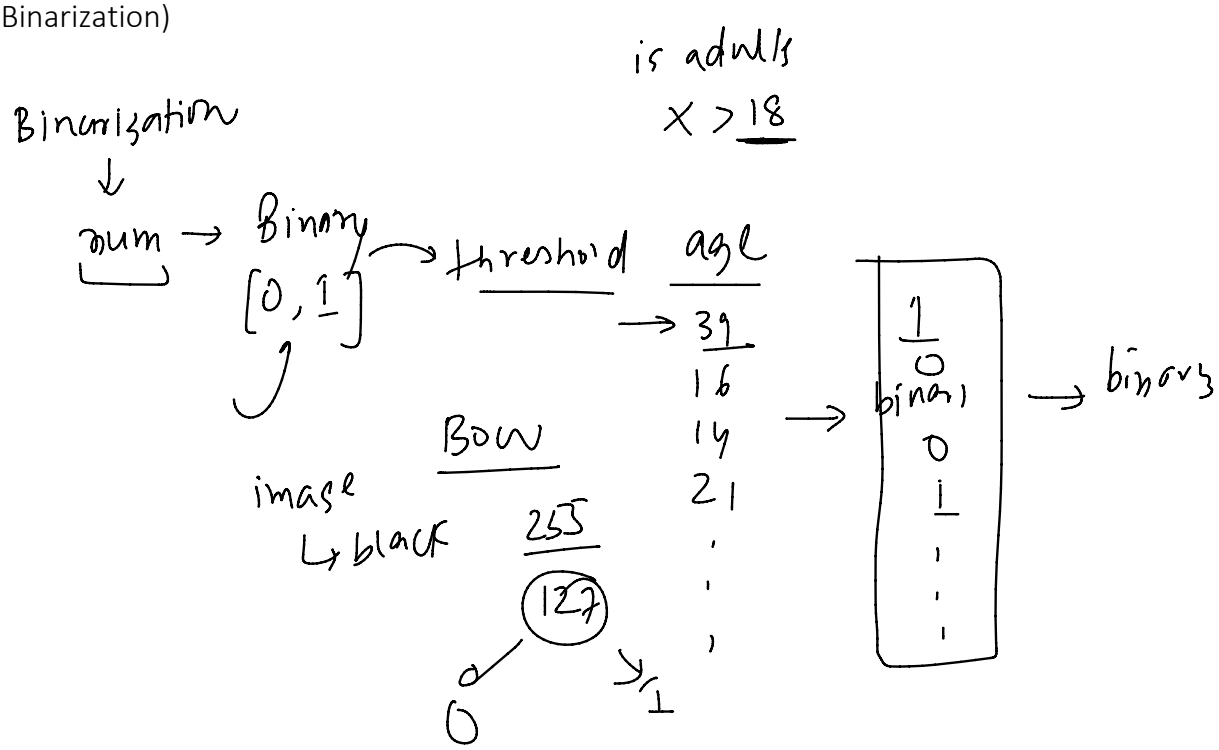
1. Adaptive
2. Minimizes within-bin variance
3. You can find the ideal number of bins

Disadvantages

1. Sensitive to initialization
2. Computationally extensive
3. Assumption of similar sized and density clusters
4. Handling of outliers
5. Interpretability

5. Threshold Binning (Binarization)

15 February 2024 06:57



6. Decision Tree Based Binning

15 February 2024 06:57