-: Data Preprocessing and features :-

Data preprocessing:

↳ Data: Text, Image, videos, audio etc.
(set of facts and statistics or symbols)

⟶ Data preprocessing is a process to convert raw data into ~~meg~~ meaningful data using different techniques.

~~adv~~:

dirty datas:
- ↳ Incomplete
- ↳ Noisy
- ↳ Inconsistent
- ↳ Duplicate

quality data:
- ↳ Accuracy
- ↳ Completeneness
- ↳ Consistency
- ↳ Believability
- ↳ Interpretability

Steps / Technique in Data prepaessing

i) data cleaning ✓
ii) Data integration
iii) Data Reduction
iv) Data transformation
v) Data Discretization

⇒ ① Data cleaning: fill in missing values, smooth out noise while identifying outliers, and correct inconsistencies in the data.

→ ⓘⓘ Data integration: merges data from multiple sources into a coherant data store, such as a data warehouse.

→ ⓘⓘⓘ Data Reduction: technique to reduce the data size by aggregating, eliminating redundant features, or clustering for instance

→ ⓘⓥ Data transformation: data are transformed or consolidated into forms appropriate for ML model training, such as normalization, may be applied where data are scaled to fall within a smaller range like 0.0 to 1.0. (feature scaling)

जब दो different features
के बीच Range में बहुत ज्यादा
difference है तो ML के लिए
से normalization करना होता है

(v) Data Discretization: technique transforms numeric data by mapping values to interval or concept labels.

techniques:
- Binning
- Histogram analysis
- cluster analysis
- Decision analysis
- Correlation analysis

- feature: is an attribute or property shared by all of the independent units on which analysis or prediction is to be done.

feature ~~Scaling~~: engineering:
- transformations that applied to the data before it is fed into some algo for some processing.
- Create feature/extract the feature from existing features by domain knowledge.
- it is an art

adv.: → improve accuracy ML algo model

major process of feature engineering:
- Brainstorming or testing feature
- Deciding what features to create
- Creating features
- Checking how the features work with your model
- Improving ~~your~~ feature if needed
- Go back to brainstorming/creating more features until the work done.

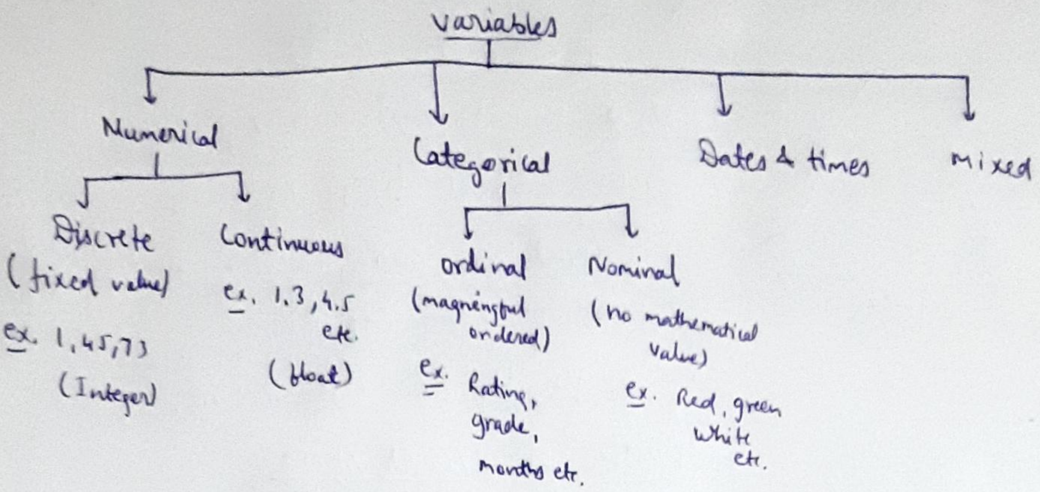
ex.

Train bording	Train Reach	Delayed/on	Time delay/on
10.00 AM	10.15 Am	Delay	00.15
5.00 pm	5.00 Am	On	00.00

Given

Create features

- **variable**: is any characteristic, number or quantity thatd can be measured or counted.



variables

Numerical — Discrete (fixed value) ex. 1,45,73 (Integer); Continuous ex. 1.3,4.5 etc. (float)

Categorical — ordinal (magningful ordered) ex. Rating, grade, months etc.; Nominal (no mathematical value) ex. Red, green white etc.

Dates & times

Mixed

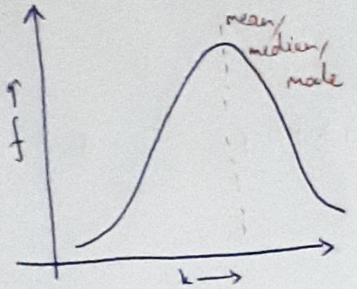
- **Missing value Handling**:
  - Ignore missing value row/Delete row
  - fill missing value manually (genrally ignoring becoz data too large)
  - Global constants
  - Measurment of central tendency (Mean, median & mode)
  - Measurment of central tendency for each class.
  - Most probable value (ML algo's) (linear Reg, decision tree, k-nearest etc.

- **Ignore missing values Row/ delete Row**:
  - it is genrally used if missing values are less than 5% of values. or depend on situation and data also.
  - missing values should be random.
  - This is use in extreame case and should only be used when there are many null values in coloum or row.

disadvantage: that may lose valuable information on that feature, as we have deleted it completly due to some null values.

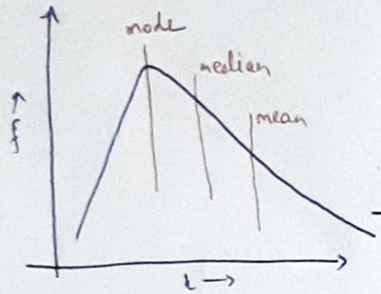
- Missing value imputation by mean, median:

if our distribution graph is



mean = median = mode

- This is normal distribution/gaussian distribution
- So, in normal distribution we can use ~~any imputation~~ mean, median, mode for missing value imputation.
- So, genrally we use mean.

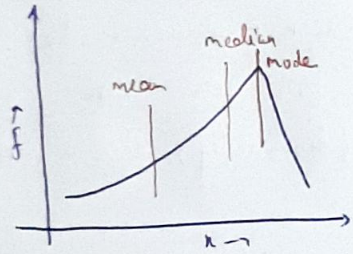


Positive skew/Right skew

- This is asymmetric distribution
- So, here we use median instead of mean.

→ mean > median > mode

Negative skew/ left skew



→ mean < median < mode

here, also we use median instead of mean.

- Incase if the arribute is categorical then replace the missing values of that categorical variasle with its mode.

- missing values by belonging to same class.
  - for, example, if classifying customers according to credit risk, we may replace the missing value with the mean income value for customers in the same ~~cred~~ credit risk category as that of the given tuple
  - if the data distribution for a given class is skewed, the median value is a better choice.

By Sklearn

⑥ Cleaning or Handle missing Data with SimpleImputer:

↳ sklearn class

↳ helpful in handle the missing data in predictive dataset.

↳ Replace missing values using a descriptive statistic (eg. mean, median, or mode) along each coloum, or using a constant value)

⇒ Class sklearn.impute.SimpleImputer (*, strategy='mean', missing-values=nan)

- missing_values: int, float, str, np.nan, defaullt=np.nan
- strategy: str, default = 'mean'
  - ↳ if 'mean', then replace missing values using the mean along each column. Can only be used with numeric data.
  - ↳ if 'median', then replace missing values using the median Can only be used with numeric data.
  - ↳ if 'most_frequent', then replace missing using the most.frequent value along each column. Can be used with numeric or numeric data.
  - ↳ if 'constant', then replace missing value with fill-value. Can be used with strings or numeric data.

→ for Numerical missing values imputation

- we use 'mean' strategy in ~~for~~ SimpleImputer function for numeric features in dataframe.
- we use transforme function for impute in whole dataset.
- Here, we can be also use 'median', 'mode' but we go for 'mean' becoz 'mean' very effective from those.

→ for categorical missing values imputation.

- Here, we use 'most_frequent' strategy in SimpleImputer.
- 'mean', 'media' not possible for Categorical missing values imputation.

- By Machine at learning algorithms
  - linear Regg.
  - Decision Tree
  - k-nearest Tree
  - multiple Imputation etc.

- linear Regg.: predict missing values by fitting a linear Regression model to non missing values, not well for non-linear datasets.

- Decision Tree: predict missing values based on the other variables in datasets.
  it can handle both numerical and categorical variables and it can capture non-linear relationships between variable.

- k-nearest Tree: predict missing values based on find k-nearest data points to missing value.
  This is more accurate.
  but, computationally expansive.

- multiple imputation: genrate multiple imputations of the missing value using probabilistic model and then combines then to create final datasets
  ↳ more accurate
  ↳ computational expansive.

⑩ One Hot Encoding:-

↳ various ML models do not work with categorical data and to fit this data into the ml model so, it need to be converted into numerical data.

↳ One approch is assign some numerical value to these lebel ex. male & female mapped to 0 & 1. but this can add bias in our model as it will start given higher preference to female parameter as 1>0. but ideally both are important.

↳ One Hot Encoding: ~~we use~~ Technique that we use to represent categorical variables as numerical values in ml ~~algo~~ model.

advantages:

- allow the use of categorical variables
- improve model performance

dis adv.:

- incresed dimensionally, make complex data.
- most observations have 0 value
- it can lead to overfitting.
-

↳ we can use pd.get_dummies() function from pandas to one_hot encode the categorical columns.

- Label & Order Encoding:

Label Encoding: apply on ordinal and nominal Categorical variables.

- preference order: 1st number then alphabetical order

Ordinal Encoding: apply on ordinal categorical data. variables.

- feature Scaling: is a method to scale numeric features in the same scale or range like (-1 to 1, 0 to 1).

  - This last step involved ~~inv~~ in data preprocessing and before ML model training.
  - it is also called as data normalization.
  - we can apply feature scalling on independent variables.
  - we fit feature scalling with train data and transform on train and test data.

- why feature scalling:

  - The scale of raw features is different according to its units.
  - ML algos can't understand features units, understand only numbers.
  - ex. if height 140 cm & 8.2 feet
    but in ML 140 > 8.2

- Which ML algo's Required feature Scaling?
  - Those algo's Calculate distance
    - K-nearest Neighbors
    - k-means
    - Support vector Analysis (~~PCA~~) (SVM)
    - principal Component Analysis (PCA)
    - Linear Discriminant Analysis

$$d^{E}(x,y) = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2}$$

- Gradient Decent Based algo's.
  - linear Regression
  - logistic Regresion
  - Neural network

$$\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i x_j^i$$

- Tree Based algo's not Required FS.
  - ↳ Decision tree
  - ↳ Random forest
  - ↳ XGBoost

- Types of feature scaling
  - ↳ Min max Scaler ✓ } mostly use
  - ↳ standard Scaler ✓
  - ↳ max Abs Scaler
  - etc. ..

⑥ <u>Standardization</u> & <u>Normalization</u>:

<u>Standardization</u>: rescale the feature such as mean(μ) = 0 and standard deviation (σ)=1.

$$Z(\text{z-score normalization}) = \frac{x-\mu}{\sigma}$$

↳ it use if data follow normal distribution (gaussian distribution)

<u>Normalization</u>: rescale the feature in fixed Range b/w 0 to 1.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

- Mostly standardization use for clustering analyses, principal Component Analysis (PLA).
- Normalization prefer for image processing because pixel intensity 0 to 255, neural network algorithms require data in scale 0-1, k-Nearest Neighbours.