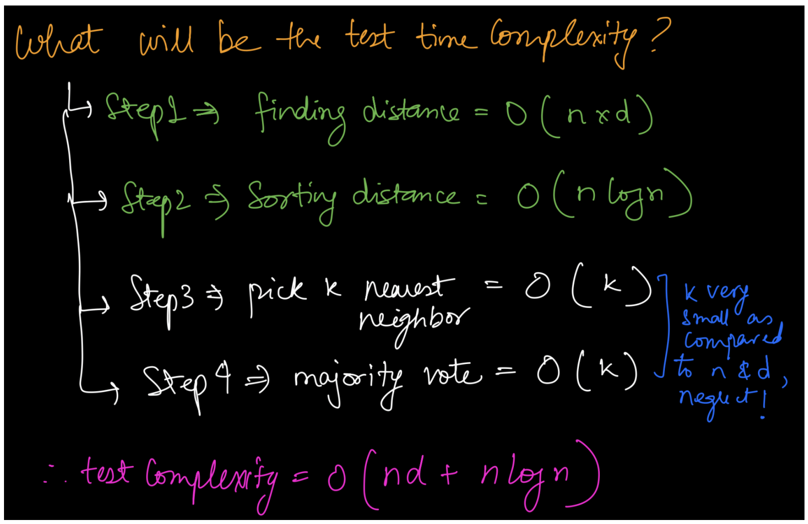
Train/Test time complexity of KNN

- Training -> $O(1)$ (since kNN just stores the data points)
- Test -> $O(n(d + logn))$
    - Distance calculation -> $O(nd)$
    - Sorting of distances -> $O(nlog(n))$
- n: Total number of points
- d: No. of features present in the dataset



Space Complexity of kNN:

1. the space complexity → $O(n \times d)$

kNN for Categorical features

1. Categorical variables are One Hot encoded → Cosine Similarity metric used

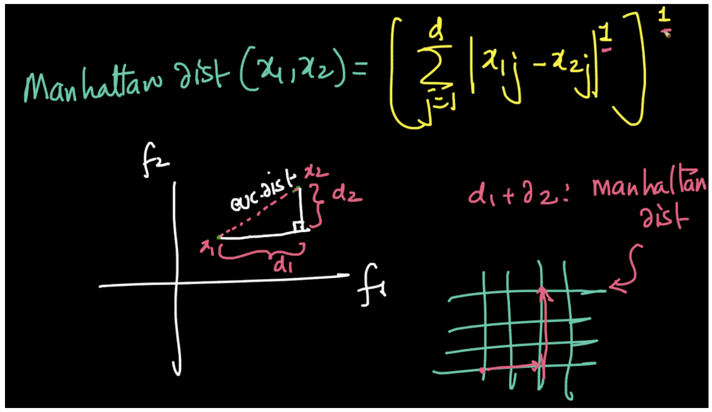
2. cosine similarity focuses on the direction of the vectors:
    - It can **effectively ignore irrelevant features** and make **kNN more robust for high-dimensional sparse data**

$$CosineSimilarity(x^{(1)}, x^{(2)}) = \frac{x^{(1)}.x^{(2)}}{||x^{(1)}||\,||x^{(2)}||}$$

Distance Metrics

1. Manhattan Distance:

$$d(x_q, x_i) = \sum_{j=1}^{d} |x_{qj} - x_{ij}|$$



- Time complexity -> O(n)

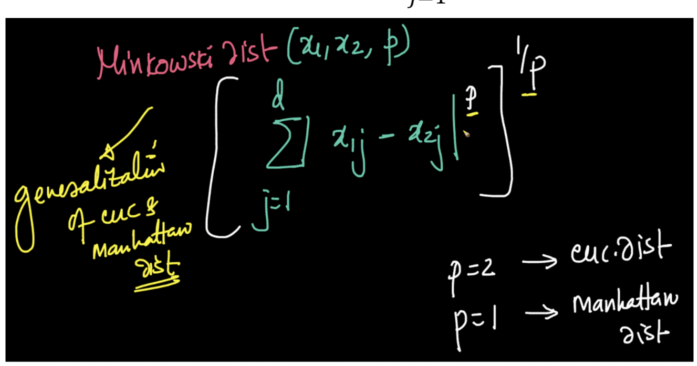
2. Euclidean Distance

$$||x_q - x_i|| = \sqrt{\sum_{j=1}^{d} (x_{qj} - x_{ij})^2}$$

- Suffers from the **curse of dimensionality**
- Time complexity -> O(n)

3. Minkowski Distance
   - Generalized version for pth degree

$$Minkowski(x_q, x_i) = [\sum_{j=1}^{d} |x_q - x_i|^p]^{\frac{1}{p}}$$



Which distance Metrics to use

1. Euclidean (Most Common) → useful when the dimension of data is small
2. Manhattan → useful when data represents maps
3. Cosine Similarity (Most Common) → useful when the dimension of data is large
4. Minkowski → useful when a custom distance metric is needed

Probabilistic Label

1. $P(y = a \mid x_i) = \frac{Count\ of\ a\ class\ label\ datapoints}{Count\ of\ total\ number\ of\ neighbors}$

Application of kNN

1. Google Image Search
    a. Local Sensitive Hashing(LSH) is used to fasten kNN by grouping images so now kNN runs on a subset of data
2. kNN Imputation
    a. Identify and locate missing values within the dataset.
    b. For each observation with missing values, calculate the distances to all other observations with complete data.
    c. Choose the "k" nearest neighbors based on a distance metric such as Euclidean distance.
    d. Impute missing values by averaging or using weighted averages of corresponding values from the nearest neighbors.