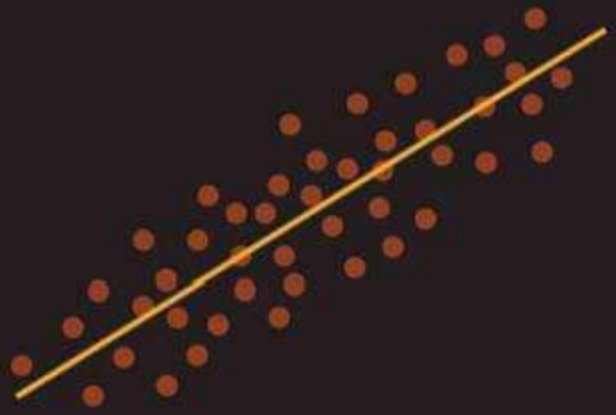


Linear Regression

GOOD

- Simple to implement and efficient to train.
- Overfitting can be reduced by regularization.
- Performs well when the dataset is linearly separable.

BAD

- Assumes that the data is independent which is rare in real life.
- Prone to noise and overfitting.
- Sensitive to outliers.

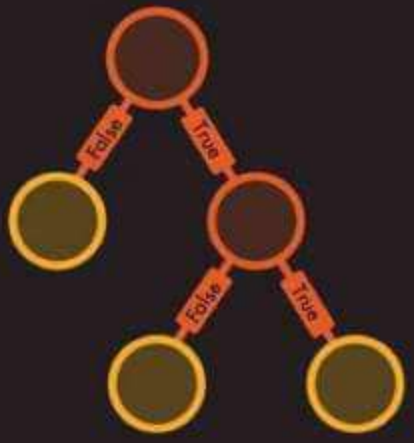
Logistic Regression

GOOD

- Less prone to over-fitting but it can overfit in high dimensional datasets.
- Efficient when the dataset has features that are linearly separable.
- Easy to implement and efficient to train.

BAD

- Should not be used when the number of observations are lesser than the number of features.
- Assumption of linearity which is rare in practise.
- Can only be used to predict discrete functions.



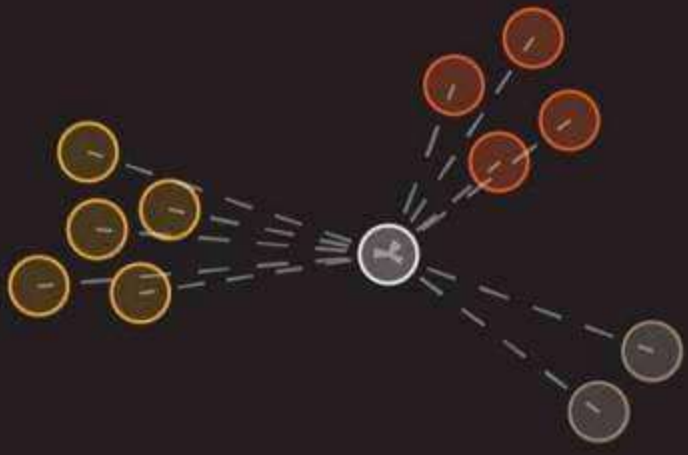
Decision Tree

GOOD

- Can solve non-linear problems.
- Can work on high-dimensional data with excellent accuracy.
- Easy to visualize and explain.

BAD

- Overfitting. Might be resolved by random forest.
- A small change in the data can lead to a large change in the structure of the optimal decision tree.
- Calculations can get very complex.



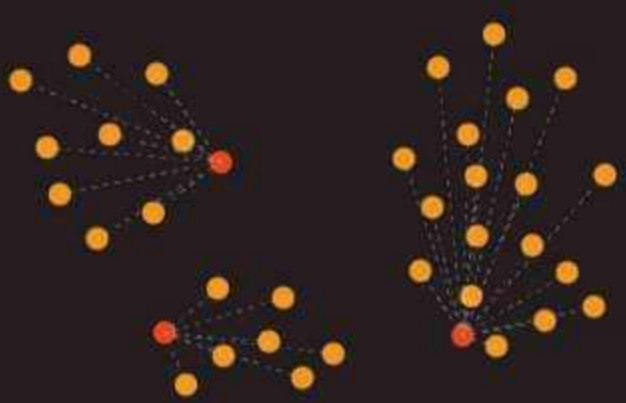
K Nearest Neighbour

GOOD

- Can make predictions without training.
- Time complexity is O(n).
- Can be used for both classification and regression.

BAD

- Does not work well with large dataset.
- Sensitive to noisy data, missing values and outliers.
- Need feature scaling.
- Choose the correct K value.



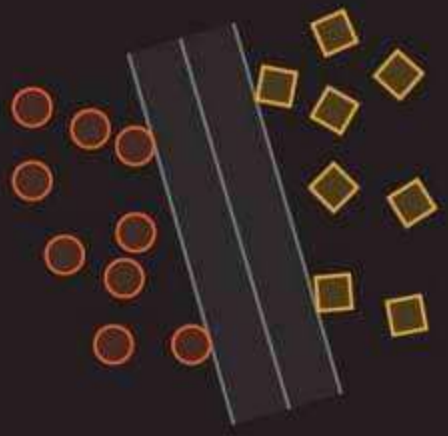
K Means

GOOD

- Simple to implement.
- Scales to large data sets.
- Guarantees convergence.
- Easily adapts to new examples.
- Generalizes to clusters of different shapes and sizes.

BAD

- Sensitive to the outliers.
- Choosing the k values manually is tough.
- Dependent on initial values.
- Scalability decreases when dimension increases.



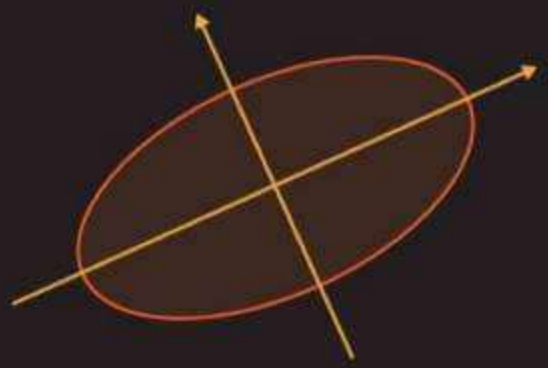
Support Vector Machine

GOOD

- Good at high dimensional data.
- Can work on small dataset.
- Can solve non-linear problems.

BAD

- Inefficient on large data.
- Requires picking the right kernal.



Principal Component Analysis

GOOD

- Reduce correlated features.
- Improve performance.
- Reduce overfitting.

BAD

- Principal components are less interpretable.
- Information loss.
- Must standardize databefore implementing PCA.

$$P(y|x) = \frac{P(x \mid y)P(y)}{P(x)}$$

Naive Bayes

GOOD

- Training period is less.
- Better suited for categorical inputs.
- Easy to implement.

BAD

- Assumes that all features are independent which is rarely happening in real life.
- Zero Frequency.
- Estimations can be wrong in some cases.

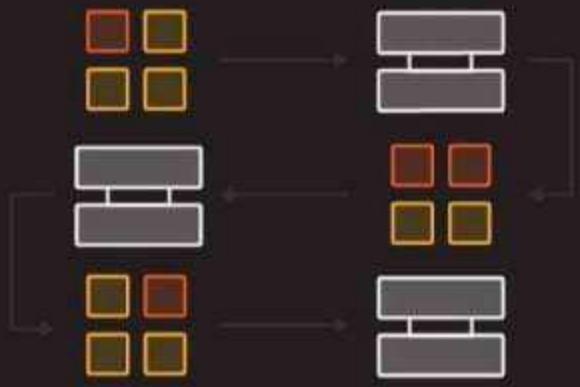
ANN

GOOD

- Have fault tolerance.
- Have the ability to learn and model non-linear and complex relationships.
- Can generalize on unsean data.

BAD

- Long training time.
- Non-guaranteed convergence.
- Black box. Hard to explain solution.
- Hardware dependence.
- Requires user's ability to translate the problem.



Adaboost

GOOD

- Relatively robust to overfitting.
- High accuracy.
- Easy to understand and to visualize.

BAD

- Sensitive to noise data.
- Affected by outliers.
- Not optimized for speed.