Introduction to Machine Learning (Part-II)

Sachin Tripathi

IIT(ISM), Dhanbad

Naïve Bayes

- A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task.
- Bayes theorem

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.
  - **P(A|B) is Posterior probability**: Probability of hypothesis A on the observed event B.
  - **P(B|A) is Likelihood probability**: Probability of the evidence given that the probability of a hypothesis is true.
  - **P(A) is Prior Probability**: Probability of hypothesis before observing the evidence.
  - **P(B) is Marginal Probability**: Probability of Evidence.

Probability Basics

- Prior, conditional and joint probability for random variables
  - Prior probability: $P(x)$
  - Conditional probability: $P(x_1 \mid x_2), P(x_2 \mid x_1)$
  - Joint probability: $\mathbf{x} = (x_1, x_2), P(\mathbf{x}) = P(x_1, x_2)$
  - Relationship: $P(x_1, x_2) = P(x_2 \mid x_1)P(x_1) = P(x_1 \mid x_2)P(x_2)$
  - Independence: $P(x_2 \mid x_1) = P(x_2), P(x_1 \mid x_2) = P(x_1), P(x_1, x_2) = P(x_1)P(x_2)$
- Bayesi

$$P(c \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid c)P(c)}{P(\mathbf{x})}$$

$$Posterior = \frac{Likelihood \times Prior}{Evidence}$$

Discriminative

Generative

Probabilistic Classification Principle

- **M**aximum **A P**osterior (**MAP**) classification rule
  - For an input ***x***, find the largest one from L probabilities output by a discriminative probabilistic classifier $P(c_1 \mid \mathbf{x}), \ldots, P(c_L \mid \mathbf{x})$.
  - Assign ***x*** to label *c** if $P(c^* \mid \mathbf{x})$ is the largest.
- Generative classification with the MAP rule
  - Apply Bayesian rule to convert them into posterior probabilities

$$P(c_i \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid c_i)P(c_i)}{P(\mathbf{x})} \propto P(\mathbf{x} \mid c_i)P(c_i) \quad \text{for } i = 1, 2, \cdots, L$$

Common factor for all L probabilities

  - Then apply the MAP rule to assign a label

Naïve Bayes

- The Naïve Bayes classifier is a popular supervised machine learning algorithm used for classification tasks such as text classification.
- It models the distribution of inputs for a given class

Naïve Bayes

- Bayes classification

$$P(c|\mathbf{x}) \propto P(\mathbf{x}|c)P(c) = P(x_1,\cdots,x_n|c)P(c) \quad \text{for } c = c_1,...,c_L.$$

Difficulty: learning the joint probability $P(x_1,\cdots,x_n|c)$ is often infeasible!

- Naïve Bayes classification
  - Assume all input features are class conditionally independent!

$$P(x_1,x_2,\cdots,x_n|c) = \underline{P(x_1|x_2,\cdots,x_n,c)}P(x_2,\cdots,x_n|c)$$
$$= \underline{P(x_1|c)}P(x_2,\cdots,x_n|c)$$
$$= P(x_1|c)P(x_2|c)\cdots P(x_n|c)$$

Applying the independence assumption

Naïve Classification

- Apply the MAP classification rule: assign $\mathbf{x}' = (a_1, a_2, \cdots, a_n)$ to c* if

$$[P(a_1 \mid c^*) \cdots P(a_n \mid c^*)]P(c^*) > [P(a_1 \mid c) \cdots P(a_n \mid c)]P(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

estimate of $P(a_1, \cdots, a_n \mid c^*)$

esitmate of $P(a_1, \cdots, a_n \mid c)$

Naïve Bayes Algorithm

- Algorithm: Discrete-Valued Features
  - Learning Phase: Given a training set S of F features and L classes,

For each target value of c_i $(c_i = c_1, \cdots, c_L)$

$\hat{P}(c_i) \leftarrow$ estimate $P(c_i)$ with examples in S;

For every feature value x_{jk} of each feature x_j $(j = 1, \cdots, F; k = 1, \cdots, N_j)$

$\hat{P}(x_j = x_{jk} \mid c_i) \leftarrow$ estimate $P(x_{jk} \mid c_i)$ with examples in S;

Output: $F * L$ conditional probabilistic (generative) models

  - Test Phase: Given an unknown instance $\mathbf{x}' = (a_1', \cdots, a_n')$

"Look up tables" to assign the label c^* to $\mathbf{X}'$ if

$$[\hat{P}(a_1' \mid c^*) \cdots \hat{P}(a_n' \mid c^*)]\hat{P}(c^*) > [\hat{P}(a_1' \mid c_i) \cdots \hat{P}(a_n' \mid c_i)]\hat{P}(c_i), \quad c_i \neq c^*, c_i = c_1, \cdots, c_L$$

Working of Naïve Bayes' Classifier

➢ Example

- Suppose we have a dataset of **weather conditions** and corresponding target variable "**Play**". So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions. So to solve this problem, we need to follow the below steps:
  - Convert the given dataset into frequency tables.
  - Generate Likelihood table by finding the probabilities of given features.
  - Now, use Bayes theorem to calculate the posterior probability.
- **Problem**: If the weather is sunny, then the Player should play or not?
- **Solution**: To solve this, first consider the dataset:

Training Dataset

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Learning Phase

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

Humidity	Play=*Yes*	Play=N*o*
High	3/9	4/5
Normal	6/9	1/5

Temperature	Play=*Ye s*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

P(Play=*Yes*) = 9/14

P(Play=*No*) = 5/14

Test Phase

- Example
  - Given a new instance, predict its label

    $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
  - Look up tables achieved in the learning phrase

P(Outlook=*Sunny*|Play=*Yes*) = 2/9
P(Temperature=*Cool*|Play=*Yes*) = 3/9
P(Huminity=*High*|Play=*Yes*) = 3/9
P(Wind=*Strong*|Play=*Yes*) = 3/9
P(Play=*Yes*) = 9/14

P(Outlook=S*unny*|Play=*No*) = 3/5
P(Temperature=*Cool*|Play==*No*) = 1/5
P(Huminity=*High*|Play=*No*) = 4/5
P(Wind=*Strong*|Play=*No*) = 3/5
P(Play=*No*) = 5/14

  - Decision making with the MAP rule

$P(Yes|\mathbf{x}') \approx [P(Sunny|Yes)P(Cool|Yes)P(High|Yes)P(Strong|Yes)]P(Play=Yes) = 0.0053$

$P(No|\mathbf{x}') \approx [P(Sunny|No)\ P(Cool|No)P(High|No)P(Strong|No)]P(Play=No) = 0.0206$

Given the fact $P(Yes|\mathbf{x}') < P(No|\mathbf{x}')$, we label $\mathbf{x}'$ to be "*No*".

Zero conditional probability

- If no example contains the feature value
  - In this circumstance, we face a zero conditional probability problem during test

$$\hat{P}(x_1 \mid c_i) \cdots \hat{P}(a_{jk} \mid c_i) \cdots \hat{P}(x_n \mid c_i) = 0 \qquad \text{for } x_j = a_{jk},\ \hat{P}(a_{jk} \mid c_i) = 0$$

  - For a remedy, class conditional probabilities re-estimated with

$$\hat{P}(a_{jk} \mid c_i) = \frac{n_c + mp}{n + m} \qquad \text{(m-estimate)}$$

n_c : number of training examples for which $x_j = a_{jk}$ and $c = c_i$

n : number of training examples for which $c = c_i$

p : prior estimate (usually, $p = 1/t$ for t possible values of x_j)

m : weight to prior (number of "virtual" examples, $m \geq 1$)

Zero conditional probability

- Example: P(outlook=overcast|no)=0 in the play-tennis dataset
  - Adding m "virtual" examples (m: tunable but up to 1% of #training examples)
    - In this dataset, # of training examples for the "no" class is 5.
    - Assume that we add $m=1$ "virtual" example in our m-estimate treatment.
  - The "outlook" feature can takes only 3 values. So $p=1/3$.
  - Re-estimate P(outlook|no) with the m-estimate

$$P(\text{overcast}|\text{no}) = \frac{0+1*\left(\frac{1}{3}\right)}{5+1} = \frac{1}{18}$$

$$P(\text{sunny}|\text{no}) = \frac{3+1*\left(\frac{1}{3}\right)}{5+1} = \frac{5}{9} \qquad P(\text{rain}|\text{no}) = \frac{2+1*\left(\frac{1}{3}\right)}{5+1} = \frac{7}{18}$$

Advantages of Naïve Bayes Classifier

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.
- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for **text classification problems**.

Disadvantage of Naïve Bayes Classifier

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

Applications of Naïve Bayes Classifier

- It is used for **Credit Scoring**.
- It is used in **medical data classification**.
- It can be used in **real-time predictions** because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as **Spam filtering** and **Sentiment analysis**.

K-Nearest Neighbour

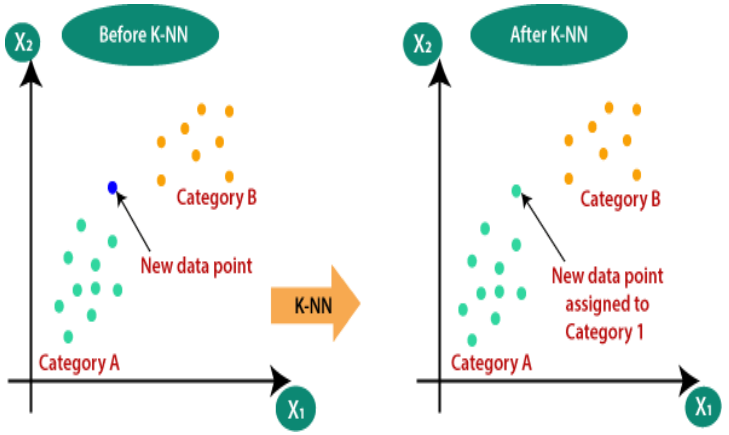
- K-Nearest Neighbour is Supervised Learning technique.
- It assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
- It stores all the available data and classifies a new data point based on the similarity.
- It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data, then it
  - classifies that data into a category that is much similar to the new data.

- It does not exactly generate a discriminative function(used for creating decision boundaries)during training as does algorithms like Logistics regression or Support Vector Machine.

- It rather predicts classes using a distance measurement method and a voting system of the nearest/closest neighbor/data points. Hence its name — ‘K-nearest Neighbour’. This is also why it cannot exactly be classified as either a ‘Discriminative model’ or a ‘Generative model’.

Why K-NN Algorithm

- Suppose there are two categories, i.e., Category A and Category B, New data point x1 arrives, so this data point will lie in which of these categories.



- To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.

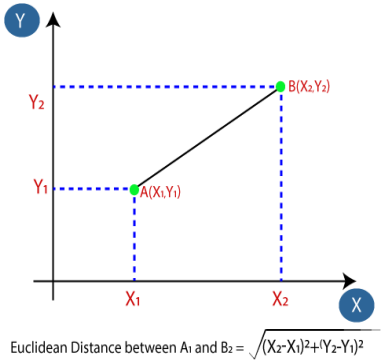
Working principle of K-NN

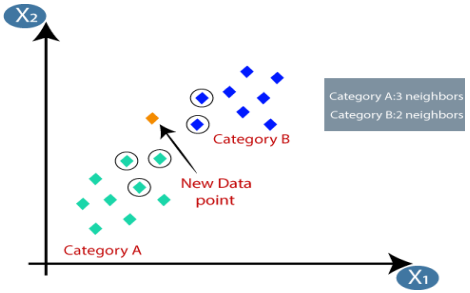
- The K-NN working can be explained on the basis of the below algorithm:
  - **Step-1:** Select the number K of the neighbors
  - **Step-2:** Calculate the Euclidean distance of **K number of neighbors**
  - **Step-3:** Take the K nearest neighbors as per the calculated Euclidean distance.
  - **Step-4:** Among these k neighbors, count the number of the data points in each category.
  - **Step-5:** Assign the new data points to that category for which the number of the neighbor is maximum.
  - **Step-6:** Our model is ready.

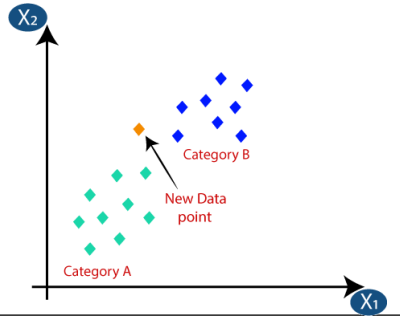
Example

- Suppose we have a new data point and we need to put it in the required category.
- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A and two nearest neighbors in category B.

- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.







How to select the value of K in the K-NN Algorithm?

- Below are some points to remember while selecting the value of K in the K-NN algorithm:
  - There is no particular way to determine the best value for "K", so we need to try some values to find the best out of them. The most preferred value for K is 5.
  - A very low value for K such as K=1 or K=2, can be noisy and lead to the effects of outliers in the model.
  - Large values for K are good, but it may find some difficulties.

Advantages of KNN

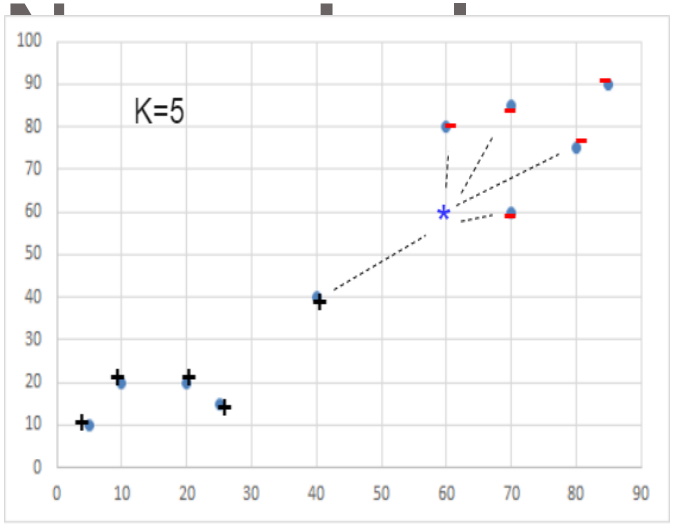
- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantage of KNN

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

Example- I (Numerical Data)

	X	Y	Class	distance
1	5	10	+	74.33
2	10	20	+	64.03
3	20	20	+	56.60
4	25	15	+	57.01
5	40	40	+	28.28
6	60	80	-	20.0
7	70	60	-	10.0
8	70	85	-	26.93
9	80	75	-	25.0
10	85	90	-	39.05
Test Distance =	60	60	?	$\sqrt{(x_i - x_t)^2 + (y_i - y_t)^2}$



Example II (Categorical data)

	Age	Income	Student	Credit rating	But computer
1	<=30	High	No	Fair	No
2	<=30	High	No	Excellent	No
3	30-40	High	No	Fair	Yes
4	>40	Low	Yes	Fair	Yes
5	>40	Medium	No	Fair	Yes
6	>40	Low	Yes	Excellent	No
7	30-40	Low	Yes	Excellent	Yes
8	<=30	Medium	No	Fair	No
9	<=30	Low	Yes	Fair	Yes
10	>40	Medium	Yes	Fair	Yes
11	<=30	Medium	Yes	Excellent	Yes
12	30-40	Medium	No	Excellent	Yes
13	30-40	High	Yes	Fair	Yes
14	>40	Medium	No	Excellent	No
Test sample	<=30	Medium	No	Excellent	?

Example on categorical data

- Similarity $(a_i, b_i) = \frac{1}{m}\sum_{i=1}^{m} \partial(a_i, b_i)$

Where, m is number of features, $\partial(a_i, b_i)$ is 1 if $a_i = b_i$ *otherwise zero*

Illustration

	Age	Income	Student	Credit rating	But computer	Similarity
1	<=30	High	No	Fair	No	0.5
2	<=30	High	No	Excellent	No	0.75
3	30-40	High	No	Fair	Yes	0.25
4	>40	Low	Yes	Fair	Yes	0
5	>40	Medium	No	Fair	Yes	0.5
6	>40	Low	Yes	Excellent	No	0.25
7	30-40	Low	Yes	Excellent	Yes	0.25
8	<=30	Medium	No	Fair	No	0.75
9	<=30	Low	Yes	Fair	Yes	0.25
10	>40	Medium	Yes	Fair	Yes	0.25
11	<=30	Medium	Yes	Excellent	Yes	0.75
12	30-40	Medium	No	Excellent	Yes	0.75
13	30-40	High	Yes	Fair	Yes	0
14	>40	Medium	No	Excellent	No	0.75
Test sample	<=30	Medium	No	Excellent	?	

- If K=5 Yes=2, No= 3 , Decision =No

Tutorial-I

Consider a set of five training examples given as ((xi, yi), ci) values, where xi and yi are the two attribute values (positive integers) and ci is the binary class label: {((1, 1), −1), ((1, 7), +1), ((3, 3), +1), ((5, 4), −1), ((2, 5), −1)}.
Classify a test example at coordinates (3, 6) using a k-NN classifier with k = 3 and Manhattan distance defined by d((u, v), (p, q)) = |u − p| + |v − q|. Your answer should be either +1 or -1.

Individual point distance calculation from test sample.

X_i	Y_i	C_i	|u-p|+|v-q|	K=3
1	1	-1	2+5=7	
1	7	+1	2+1=3	3rd nearest
3	3	+1	0+3=3	2nd nearest
5	4	-1	2+2=4	
2	5	-1	1+1=2	1st nearest

Writing 3 nearest neighbor obtained

As number of (+) Label > (-) Label of nearest neighbors , so classified as (+)

THANK YOU