Naïve Bayes Model

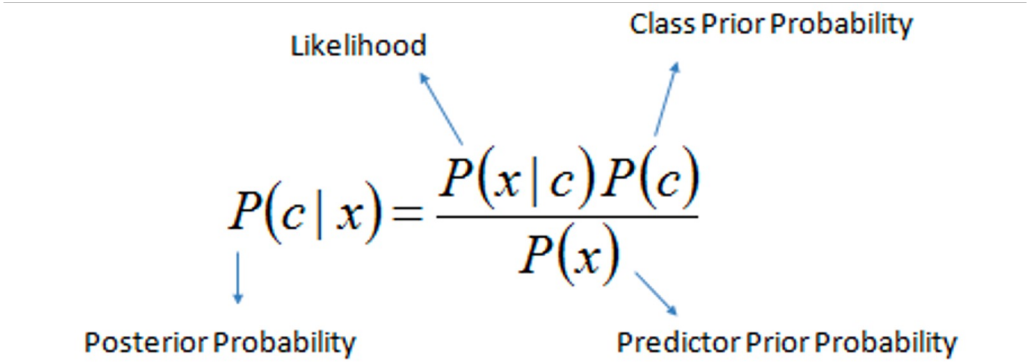
Naive Bayes Model

- **Naive Bayes** is a classification technique that is based on the *Bayes Theorem* with an assumption that all the features that predict the target value are independent of each other.
- It calculates the probability of each class and then picks the one with the highest probability.
- Bayes' Theorem describes the probability of an event, based on prior knowledge of conditions that might be related to that event.

What makes Naive Bayes as a “Naive” Algorithm?

- **Naive Bayes** classifier assumes that the features we use to predict the target are independent and do not affect each other.
- While in real-life data, features depend on each other in determining the target, but, this is ignored by the Naive Bayes classifier.
- Though the independence assumption is never correct in real-world data, but often works well in practice, so that it is called “**Naive**”.

Naive Bayes Formula



$$P(c \mid \mathrm{X}) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

P(c|x) is the posterior probability of class (c, target) given predictor (x, attributes).
P(c) is the prior probability of class.
P(x|c) is the likelihood which is the probability of predictor given class.
P(x) is the prior probability of predictor.

Types of Naive Bayes Classifiers

- **Gaussian**: It is used in classification and it assumes that features follow a normal distribution.
- **Multinomial**: It is used for discrete counts. Feature vectors represent the frequencies with which certain events have been generated by a multinomial distribution. For example, count how often each word occurs in the document. This is the event model typically used for document classification.
- **Bernoulli**: Like the multinomial model, this model is popular for document classification tasks, where binary term occurrence(i.e. a word occurs in a document or not) features are used rather than term frequencies(i.e. frequency of a word in the document).

Naive Bayes Algorithm for Categorical Data

1. Make sure the training data set D which contains tuples X along with m class labels $(C_1, C_2, \dots, C_m)$ is categorical data type. Each tuple with dimension n is declared as $X = (x_1, x_2, \dots, x_n)$ obtained from n attributes $A_1, A_2, \dots, A_n$
2. Calculate the prior probabilities of each class so that a class probability table is generated, where $P(C_i)$= the number of types in the class C_i divided by the total of all tuples in the training data set D.
3. Calculate the probability of each value on all n attributes $(A_1, A_2, \dots, A_n)$ for all m classes $(C_1, C_2, \dots, C_m)$ so that a number of n probability tables are generated.
4. If the probability is 0, perform Laplacian correction.
5. The learning outcomes are as many as $(n+1)$ tables, that is one class probability table and n tables containing the probability of each value on all existing attributes.
6. The learning outcomes can be used to classify an input tuple by finding the maximum probability of all existing classes.

$$P(X|C_i)P(C_i) = (\prod_{k=1}^{n} P(x_k|C_i)) \times P(C_i) = \big(P(x_1|C_i) \times \dots \times P(x_n|C_i)\big) \times P(C_i)$$

$$P(X|C_i)P(C_i) > P\big(X\big|C_j\big)P\big(C_j\big) \text{ for } i \leq j \leq m, j \neq i$$

Case 1: Online E-commerce Website Problem

- Suppose that we have the following historical data of an online e-commerce website (13 rows)
- Given a new data Time = morning, will it be yes or no?
- Using the **Naive Bayes Classifier**, the problem can be solved.

Time	Buy
morning	no
afternoon	yes
evening	yes
morning	yes
morning	yes
afternoon	yes
evening	no
evening	yes
morning	no
afternoon	no
afternoon	yes
afternoon	yes
morning	yes

(1) Naive Bayes Classifier - Steps

We first transform the historical data into a summary as follows:

Time	Buy
morning	no
afternoon	yes
evening	yes
morning	yes
morning	yes
afternoon	yes
evening	no
evening	yes
morning	no
afternoon	no
afternoon	yes
afternoon	yes
morning	yes

Time	yes	no	TOTAL
morning	3	2	5
afternoon	4	1	5
evening	2	1	3
TOTAL	9	4	13

(2) Naive Bayes Classifier - Steps

Time	yes	no	TOTAL
morning	3	2	5
afternoon	4	1	5
evening	2	1	3
TOTAL	9	4	13

$$P(C \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid C)P(C)}{P(\mathbf{X})} \propto P(\mathbf{X} \mid C)P(C)$$

New data (data 14) = morning
Class ??

- P(yes) = 9/13 = 0.6923
- P(no) = 4/13 = 0.3076
- P(morning|yes) = 3/9 = 0.333
- P(morning|no) = 2/4 = 0.5

- P(yes|morning) = P(morning|yes) * P(yes) = 0.333 * 0.6923 = 0.2305
- P(no|morning) = P(morning|no) * P(no) = 0.5 * 0.3076 = 0.1538

Therefore, the classifier will predict **YES**

Implementation in Python

Using the Sklearn library, we only need to provide the feature and the label.

```
# import the preprocessing and the Naive Bayes model from the SkLearn library
from sklearn import preprocessing
from sklearn.naive_bayes import GaussianNB


# assign the feature of the historical data (Time)
feature = ['morning', 'afternoon', 'evening', 'morning', 'morning', \
           'afternoon', 'evening', 'evening', 'morning', 'afternoon', \
           'afternoon', 'afternoon', 'morning']

# assign the label of the historical data (Buy)
label = ['no', 'yes', 'yes', 'yes', 'yes', 'yes', 'no', 'yes', 'no', 'no', \
         'yes', 'yes', 'yes']

# create a label encoder objects
feature_encoder = preprocessing.LabelEncoder()
label_encoder = preprocessing.LabelEncoder()

# transform the categorical feature and label into numeric
feature_encoded = feature_encoder.fit_transform(feature).reshape(-1, 1)
label_encoded = label_encoder.fit_transform(label)

# create the Naive Bayes classifier model
model = GaussianNB()

# train the model using the historical data
model.fit(feature_encoded, label_encoded)

# predict the value if given a new data 'morning'
prediction_encoded = model.predict([feature_encoder.transform(['morning'])])
prediction = label_encoder.inverse_transform(prediction_encoded)

print('Prediction = {}'.format(prediction[0]))
```

```
Prediction = yes
```

Case 2: Play Tennis Problem

- Suppose that we have 14 training data with 4 attributes/ features (Outlook, Temperature, Humidity, and Wind)
- We have 2 classes (Yes and No)
- What is the prediction from the naive bayes classifier if given a new data as follows (D15)?
  - Outlook = Sunny
  - Temperature = Cool
  - Humidity = High
  - Wind = Strong

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

(1) Naive Bayes Classifier - Steps

We first transform the historical data into a summary as follows:

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

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

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

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

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

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/14$

(2) Naive Bayes Classifier - Steps

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

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

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

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

$P(\text{Play=}Yes) = 9/14$ $P(\text{Play=}No) = 5/14$

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=*Sunny* | 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

$$P(C \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid C)P(C)}{P(\mathbf{X})} \propto P(\mathbf{X} \mid C)P(C)$$

New data

x' = (Outlook=Sunny, Temperature = Cool, Humidity = High, Wind=Strong)

P(Yes|x') = P(Sunny|Yes) * P(Cool|Yes) * P(High|Yes) * P(Strong|Yes) * P(Play=Yes) = 0.0053

P(No|x') = P(Sunny|No) * P(Cool|No) * P(High|No) * P(Strong|No) * P(Play=No) = 0.0206

Therefore, the prediction is **NO**

Naive Bayes Algorithm for Continuous Data

1. Make sure the training data set D which contains tuples X along with m class labels $(C_1, C_2, \dots, C_m)$ is continuous data type. Each tuple with dimension n is declared as $X = (x_1, x_2, \dots, x_n)$ obtained from n attributes $A_1, A_2, \dots, A_n$
2. Calculate the prior probabilities of each class so that a class probability table is generated, where $P(C_i)$= the number of types in the class C_i divided by the total of all tuples in the training data set D.
3. Calculate the mean and standard deviation of all n attributes $(A_1, A_2, \dots, A_n)$ for all m classes $(C_1, C_2, \dots, C_m)$ so that $m \times n$ mean values and $m \times n$ standard deviation values can be stored in a number of m tables.
4. The learning outcomes are as many as $(m + 1)$ tables, that is one class probability table and dan m tables containing the mean and standard deviation values for each attribute value.
5. The learning outcomes can be used to classify an input tuple by maximizing probability.

$$P(X|C_i)P(C_i) = \left(\prod_{k=1}^{n} P(x_k|C_i)\right) \times P(C_i) = \left(\prod_{k=1}^{n} \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{-\frac{(x_k-\mu_{ik})^2}{2{\sigma_{ik}}^2}}\right) \times P(C_i)$$

$$P(X|C_i)P(C_i) = \left(\frac{1}{\sigma_{i1}\sqrt{2\pi}} e^{-\frac{(x_k-\mu_{i1})^2}{2{\sigma_{i1}}^2}} \times \dots \times \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{-\frac{(x_k-\mu_{ik})^2}{2{\sigma_{ik}}^2}}\right) \times P(C_i)$$

where μ_{ik} and σ_{ik} are the mean and standard deviation of the values on the A_K attribute for C_i class

Case 3: Handphone Recommendation Problem

- Suppose that we have 14 training data with 3 attributes/ features
  - Battery → battery strength in hours
  - Camera → camera resolution in mega pixels
  - Price → handphone price in million rupiah
- We have 2 classes (Yes and No)
- What is the prediction from the naive bayes classifier if given a new data as follows (H15)?
  - Battery → 28 hours
  - Camera → 4 mega pixels
  - Price → 2 million rupiah

Handphone	Battery	Camera	Price	Recommended
H1	26	8	1,2	Yes
H2	27	13	15	Yes
H3	28	5	6	Yes
H4	25	2	5	No
H5	23	10	1	Yes
H6	20	7	3,5	Yes
H7	22	7	10	Yes
H8	24	8	2	Yes
H9	21	3	4	No
H10	16	13	0,8	Yes
H11	12	10	12	No
H12	14	5	5	No
H13	18	5	3	No
H14	15	3	14	No
H15	28	4	2	???

(1) Naive Bayes Classifier - Steps

We first calculate the mean and standard deviation of the values in each class and each attribute:

All data tuples in the recommended class = Yes

Handphone	Battery	Camera	Price	Recommended
H1	26	8	1,2	Yes
H2	27	13	15	Yes
H3	28	5	6	Yes
H5	23	10	1	Yes
H6	20	7	3,5	Yes
H7	22	7	10	Yes
H8	24	8	2	Yes
H10	16	13	0,8	Yes
Mean C1	23,2500	8,8750	6,8000	
SD C1	3,9551	2,9001	5,8052	

All data tuples in the recommended class = No

Handphone	Battery	Camera	Price	Recommended
H4	25	2	5	No
H9	21	3	4	No
H11	12	10	12	No
H12	14	5	5	No
H13	18	5	3	No
H14	15	3	14	No
Mean C2	17,5000	4,6667	7,1667	
SD C2	4,8477	2,8752	4,6224	

What is the prediction from the naive bayes classifier if given new data as follows (H15)?

- Battery → 28 hours
- Camera → 4 mega pixels
- Price → 2 million rupiah

$new\ data = x' = (battery = 28, camera = 4, price = 2)$

(2) Naive Bayes Classifier - Steps

Calculate the probability for each class using the following formula:

$$P(X|C_i)P(C_i) = \left(\prod_{k=1}^{n} \frac{1}{\sigma_{ik}\sqrt{2\pi}} e^{-\frac{(x_k-\mu_{ik})^2}{2\sigma_{ik}^2}}\right) \times P(C_i)$$

$$new\ data = x' = (battery = 28, camera = 4, price = 2)$$

$$P(X|recommended = Yes)P(recommended = Yes)$$
$$= \left(\prod_{k=1}^{n} \frac{1}{\sigma_{1k}\sqrt{2\pi}} e^{-\frac{(x_k-\mu_{1k})^2}{2\sigma_{1k}^2}}\right) \times P(recommended = Yes)$$
$$= \left(\frac{1}{3{,}9551\sqrt{2\pi}} e^{-\frac{(28-23{,}2500)^2}{2(3{,}9551)^2}}\right) \times \left(\frac{1}{2{,}9001\sqrt{2\pi}} e^{-\frac{(4-8{,}8750)^2}{2(2{,}9001)^2}}\right)$$
$$\times \left(\frac{1}{5{,}8052\sqrt{2\pi}} e^{-\frac{(2-6{,}8000)^2}{2(5{,}8052)^2}}\right) \times P(recommended = Yes)$$
$$= \frac{0{,}4862}{9{,}9139} \times \frac{0{,}2435}{7{,}2695} \times \frac{0{,}7105}{14{,}5513} \times \frac{8}{14}$$
$$= 0{,}0490 \times 0{,}0335 \times 0{,}0488 \times 0{,}5714$$
$$= 0{,}000046$$

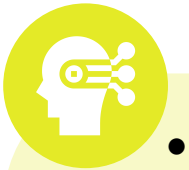
$$P(X|recommended = No)P(recommended = No)$$
$$= \left(\prod_{k=1}^{n} \frac{1}{\sigma_{1k}\sqrt{2\pi}} e^{-\frac{(x_k-\mu_{1k})^2}{2\sigma_{1k}^2}}\right) \times P(recommended = No)$$
$$= \left(\frac{1}{4{,}8477\sqrt{2\pi}} e^{-\frac{(28-17{,}5000)^2}{2(4{,}8477)^2}}\right) \times \left(\frac{1}{2{,}8752\sqrt{2\pi}} e^{-\frac{(4-4{,}6667)^2}{2(2{,}8752)^2}}\right)$$
$$\times \left(\frac{1}{4{,}6224\sqrt{2\pi}} e^{-\frac{(2-7{,}1667)^2}{2(4{,}6224)^2}}\right) \times P(recommended = No)$$
$$= \frac{0{,}0958}{12{,}1512} \times \frac{0{,}9735}{7{,}2069} \times \frac{0{,}5354}{11{,}5865} \times \frac{6}{14}$$
$$= 0{,}0079 \times 0{,}1351 \times 0{,}4621 \times 0{,}4286$$
$$= 0{,}000021$$

Select the class that has the highest probability. Based on the results above, the new data is predicted in the recommended class YES

When to use the Naive Bayes classifier?

- Naive Bayes classifier works better on small training dataset.
- The implementation of the model for numerical data becomes more complex and leads to missing information.
- Naive bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. This is something that not always occurs in real world cases.

Applications of Naive Bayes Algorithm



- **Real time Prediction**: Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
- **Multi class Prediction**: This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
- **Text classification/ Spam Filtering/ Sentiment Analysis**: Naive Bayes classifiers mostly used in text classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)
- **Recommendation System:** Naive Bayes Classifier and Collaborative Filtering together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not

Pros for Naive Bayes

- Requires a small amount of training data. So the training takes less time.
- Handles continuous and discrete data, and it is not sensitive to irrelevant features.
- Very simple, fast, and easy to implement.
- Can be used for both binary and multi-class classification problems.
- Highly scalable as it scales linearly with the number of predictor features and data points.
- When the Naive Bayes conditional independence assumption holds true, it will converge quicker than discriminative models like logistic regression.
- It perform well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

Cons for Naive Bayes

- The assumption of independent predictors/features. Naive Bayes implicitly assumes that all the attributes are mutually independent (completely independent) which is almost impossible to find in real-world data.
- If a categorical variable has a value/category that appears in the test dataset and is not observed in the training dataset, then the model will assign it a 0 (zero) probability and will not be able to make a prediction. This is what we called the “Zero Frequency problem“, and can be solved using smoothing techniques. One of the simplest smoothing techniques is called Laplace estimation.

Tips to Improve the Power of Naive Bayes Model

- If continuous features do not have normal distribution, we should use transformation or different methods to convert it in normal distribution.
- If test data set has zero frequency issue, apply smoothing techniques “Laplace Correction” to predict the class of test data set.
- Remove correlated features, as the highly correlated features are voted twice in the model and it can lead to over inflating importance.
- Naive Bayes classifiers has limited options for parameter tuning like alpha=1 for smoothing, fit_prior=[True|False] to learn class prior probabilities or not and some other options (look at detail here). I would recommend to focus on your pre-processing of data and the feature selection.
- You might think to apply some classifier combination technique like ensembling, bagging and boosting but these methods would not help. Actually, “ensembling, boosting, bagging” won’t help since their purpose is to reduce variance. Naive Bayes has no variance to minimize.

THANK YOU