→ Sequential / Probabilistic Algorithms

- Naive-Bayes - single decision
- Decision Tree - multiple decisions in single tree
- Random Forest - multiple decisions in multiple trees

Naive-Bayes Clasiifier using GaussianNB

- single decision based on probability
- performs Probabilistic Classification
- calculates Probability / certainity
- training is very fast, just requireing considering each attribute in each class separately
- test is straight forward, just looking up tables or calculating conditional probabilities with normal distributions
- a popular generative model, being a performance competitive to most of the state-of-the-art classifiers even in the presence of violating independence assumption
- based on Bayes Theorem
  - `P(A|B)` = `P(B|A).P(A) / P(B)`
- assumes
  - classes are mutually exclusive and exhaustive
  - attributes are independent given the class
- called `Naive` classifier because of these assumptions
  - empirically proven to be useful
  - scales very well

Advantages of Naive-Bayes Classifier

- one of the fast and easy ML algorithms for classification
- can be used for binary as well as Multi-class classifications
- performs better in multi-class classification as compared to other algorithms

Disadvantages of Naive-Bayes Classifier

- Naive-Bayes Classifier assumes that all classes are mutually exclusive and exhaustive, so it cannot learn the relationship between features

Types of Naive-Bayes Classifier

- Gaussian classifier
  - assumes that features follow a normal distribution
  - as it follows normal distribution, predictors can take continuous values instead of discrete values
- Multinomial classifier
  - used when data is multinomial distributed
  - works on the frequency of words, so it is mostly used to classify documents
- Bernoulli classifier
  - similar to Multinomial classfier, but it considers the presence of words as boolean instead of frequency of words

Conditional Probability

- `A`: observation Event

- `B` : condition, which is occuring
- Conditional Probability, `P(A|B)` = Probability of A when event B occurs
- Conditional Probability, `P(A|B)` = `P(A ∩ B)` / `P(B)`
- `P(A|B)` : Posterior , Probability of hypothesis A when we have occurred an evidence B
- `P(B|A)` : Likelihood / Evidence
- `P(A)` : Prior Probability
- `P(B)` : Marginal Probability
- `P(A∩B)` : Joint probability of A & B
- `P(A|B)` = `P(B|A) P(A)` / `P(B)` ←Bayes Theorem

Bayes Theorem Derivation

- We have, `P(A|B)` = `P(A∩B)` / `P(B)`
    - so, `P(A∩B)` = `P(A|B) P(B)`
- also , `P(B|A)` = `P(B∩A)` / `P(A)`
    - so, `P(B∩A)` = `P(B|A) P(A)`
- Since, `P(A∩B)` = `P(B∩A)`
- So , `P(A|B) P(B)` = `P(B|A) P(A)`
- thus `P(A|B)` = `P(B|A) P(A)` / `P(B)` ←Bayes Theorem

importing libs

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

Assigning features and label variables

```
### First Feature
weather=['Sunny','Sunny','Overcast','Rainy','Rainy','Rainy','Overcast','Sunny','Sunny', 'Rainy','Sunny','Overcast','Overcast','Ra
```

```
### Second Feature
temp=['Hot','Hot','Hot','Mild','Cool','Cool','Cool','Mild','Cool','Mild','Mild','Mild','Hot','Mild']
```

```
### Label or target varible
play=['No','No','Yes','Yes','Yes','No','Yes','No','Yes','Yes','Yes','Yes','Yes','No']
```

preprocessing

Label Encoding

```
from sklearn.preprocessing import LabelEncoder
```

```
la = LabelEncoder()
# creating label encoder
```

encoding predictors

```
w_encode = la.fit_transform(weather)
w_encode
# overcast: 0       # Rainy : 1     # sunny: 2
```

```
array([2, 2, 0, 1, 1, 1, 0, 2, 2, 1, 2, 0, 0, 1], dtype=int64)
```

```
t_encode = la.fit_transform(temp)
t_encode
# Cool : 0      # Hot : 1       # Mild : 2
```

```
array([1, 1, 1, 2, 0, 0, 0, 2, 0, 2, 2, 2, 1, 2], dtype=int64)
```

encoding target

```
p_encode = la.fit_transform(play)
p_encode
# No : 0        # Yes : 1
```

```
array([0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0], dtype=int64)
```

Combining Predictors: zip()

```
features = list(zip(w_encode, t_encode))
features[:5]
```

```
[(2, 1), (2, 1), (0, 1), (1, 2), (1, 0)]
```

Training

```
from sklearn.naive_bayes import GaussianNB
```

```
model = GaussianNB()
# creating model
```

```
model.fit(features, p_encode)
# Training model
```

```
▾ GaussianNB
GaussianNB()
```

Prediction

```
predicted = model.predict([[0, 2]])
predicted
```

```
array([1], dtype=int64)
```

```
predicted = model.predict([[2, 0]])
predicted
```

```
array([0], dtype=int64)
```

Evaluation

Test data needs to be provided for evaluation

confusion_matrix

```
from sklearn.metrics import confusion_matrix
```

```
# confusion_matrix(y_test, y_pred)
# Test data needs to be provided.
```

classification_report

```
from sklearn.metrics import classification_report
```

```
# print(classification_report(y_test, y_pred))
# Test data needs to be provided.
```

accuracy_score

```
from sklearn.metrics import accuracy_score
```

```
# accuracy_score(y_test, y_pred)
# Test data needs to be provided.
```

precision_score

```
from sklearn.metrics import precision_score
```

```
# precision_score(y_test, y_pred)
```

recall_score

```
from sklearn.metrics import recall_score
```

```
# recall_score(y_test, y_pred)
```

Interview Questions:

---

1. What is the Naive-Bayes Algorithm?
2. How does Naive-Bayes Algorithm work?
3. What are the different applications of Naive-Bayes Algorithm?
4. What is the formula given by Bayes Theorem?
5. What is Posterior Probability & Prior Probability in Bayes Theorem?
6. Define Likelihood and Evidence in Bayes Theorem.
7. What is Bernoulli's Distribution in Naive-Bayes?
8. What is the best dataset scenario for the Naive-Bayes Classifier?
9. Is Naive-Bayes discriminative or generative classifier? [it is generative]
10. How does Naive-Bayes Algorithm treat numerical & categorical values? [categorical : Bernoulli distribution, continuous: Gaussian distribution, Discrete: Multinomial distribution]

```

```