# Naïve Bayes

**Aim**: To classify the cars based on their condition using naïve bayes classifier

**Naïve Bayes Classifier**:
Naive Bayes is a family of supervised machine learning algorithms based on Bayes' Theorem, which is used for classification tasks. The algorithm calculates the probability of a class label given a set of features, making predictions based on these probabilities. It is called "naive" because it assumes that the features are independent of each other, which simplifies the computations. It is widely used for tasks like text classification, spam filtering, and sentiment analysis. The algorithm is efficient, easy to implement, and particularly effective when dealing with large datasets or high-dimensional data.

**Algorithm**:

1. Load the dataset and assign column names.
2. Separate the target variable from features.
3. Encode categorical features using Label Encoding.
4. Split the dataset into training and testing sets.
5. Train a Naive Bayes classifier and evaluate using accuracy score.
6. Print the accuracy of the model.

**Code**:

```
import numpy as np
import pandas as pd
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

# Loading the dataset
df = pd.read_csv("car_evaluation.csv", header=None)

# Displaying dataset information
df.head()
df.shape
df.describe()
df.info()

# Assigning column names
col_names = ['buying', 'maint', 'doors', 'persons',
'lug_boot', 'safety', 'class']
```

```
df.columns = col_names

# Displaying value counts for each column
for col in df.columns:
    print(df[col].value_counts())

# Checking for missing values
df.isnull().sum()

# Separating target variable (y) and features (x)
y = df['class']
x = df.drop(columns=['class'])

# Encoding categorical features
encoder = LabelEncoder()
for col in x.columns:
    x[col] = encoder.fit_transform(x[col])

# Splitting the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.3, random_state=42)

# Training the Naive Bayes classifier
nb_model = GaussianNB()
nb_model.fit(x_train, y_train)

# Making predictions and evaluating the model
y_pred = nb_model.predict(x_test)
accuracy = accuracy_score(y_test, y_pred)

# Printing the accuracy
print(f"Accuracy of Naive Bayes Classifier: {accuracy}")
```

**Output**:

```
Accuracy of Naive Bayes Classifier: 0.6435452793834296
```