

**IMPLEMENTASI ALGORITMA NAIVE BAYES UNTUK MELIHAT PENGARUH
MEROKOK PADA PENYAKIT DIABETES**



Oleh:

Niken Zalzabila	G.211.21.0009
Shafara Rahmaeda	G.211.21.0025
Doly Ilham Saputra Huta Julu	G.211.21.0038
Fanika Putri Tantisnio	G.211.21.0039
Madyantari Ipmas Ciptandini	G.211.21.0062

**FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI
UNIVERSITAS SEMARANG**

2023

BAB I

PENDAHULUAN

Algoritma Naive Bayes adalah algoritma klasifikasi probabilistik yang didasarkan pada teorema Bayes. Teorema Bayes adalah teorema matematika yang digunakan untuk menghitung probabilitas dari suatu kejadian, berdasarkan pengetahuan tentang kejadian lain yang terkait.

Dalam kasus algoritma Naive Bayes, kejadian lain yang terkait adalah fitur-fitur dari data yang sedang diklasifikasikan. Algoritma Naive Bayes mengasumsikan bahwa semua fitur independen satu sama lain. Ini berarti bahwa probabilitas dari suatu kelas tidak dipengaruhi oleh nilai dari fitur lainnya.

Asumsi ini dapat membuat algoritma Naive Bayes kurang akurat dalam kasus-kasus di mana fitur-fitur saling terkait. Namun, algoritma Naive Bayes tetap merupakan algoritma klasifikasi yang populer karena kemudahan implementasinya dan efisiensinya dalam hal waktu dan memori.

Teorema Bayes dinyatakan secara matematis dalam persamaan berikut:

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

Dimana $P(B) \neq 0$

- Pada dasarnya, kita mencoba mencari peluang kejadian A, apabila kejadian B bernilai benar. Kejadian B juga disebut sebagai bukti.
- $P(A)$ adalah apriori dari A (probabilitas sebelumnya, yaitu probabilitas peristiwa sebelum bukti terlihat). Bukti adalah nilai atribut dari instance yang tidak diketahui (peristiwa B).
- $P(A|B)$ adalah probabilitas posteriori dari B, yaitu probabilitas kejadian setelah bukti terlihat.

BAB II

PEMBAHASAN

Makalah ini bertujuan untuk mengetahui algoritma yang akan digunakan untuk mengklasifikasi penyakit kanker menggunakan algoritma Naive Bayes. Yang akan dilakukan pertama yaitu persiapan data dan selanjutnya dilakukannya modeling pada data.

🚦 Persiapan pada data

```
[ ] #import seluruh modul yang dibutuhkan
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing
```

- Pada script diatas, digunakan untuk mengimport modul yang akan digunakan pada dataset.

```
[ ] data = pd.read_csv('Cancer_Data.csv')
```

- Script diatas digunakan untuk membaca dataset yang akan digunakan.

#Melihat lima data teratas
data.head()

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	texture_worst	perimeter_worst
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	17.33	184.60
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	23.41	158.80
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	25.53	152.50
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	26.50	98.87
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	16.67	152.20

5 rows x 33 columns

- Untuk script diatas, digunakan untuk melihat 5 data teratas pada data.

```
#Melihat dimensi data
data.shape
```

(569, 31)

- Sedangkan script diatas, digunakan untuk melihat dimensi pada data.

```
#Cek type data setiap atribut
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 31 columns):
#   Column                      Non-Null Count  Dtype
---  -
0   diagnosis                   569 non-null    object
1   radius_mean                 569 non-null    float64
2   texture_mean                569 non-null    float64
3   perimeter_mean              569 non-null    float64
4   area_mean                   569 non-null    float64
5   smoothness_mean             569 non-null    float64
6   compactness_mean            569 non-null    float64
7   concavity_mean              569 non-null    float64
8   concave points_mean         569 non-null    float64
9   symmetry_mean               569 non-null    float64
10  fractal_dimension_mean      569 non-null    float64
11  radius_se                   569 non-null    float64
12  texture_se                  569 non-null    float64
13  perimeter_se                569 non-null    float64
14  area_se                     569 non-null    float64
15  smoothness_se               569 non-null    float64
16  compactness_se              569 non-null    float64
17  concavity_se                569 non-null    float64
18  concave points_se           569 non-null    float64
19  symmetry_se                 569 non-null    float64
20  fractal_dimension_se        569 non-null    float64
21  radius_worst                569 non-null    float64
22  texture_worst               569 non-null    float64
23  perimeter_worst             569 non-null    float64
24  area_worst                  569 non-null    float64
25  smoothness_worst            569 non-null    float64
26  compactness_worst           569 non-null    float64
27  concavity_worst             569 non-null    float64
28  concave points_worst        569 non-null    float64
29  symmetry_worst              569 non-null    float64
30  fractal_dimension_worst     569 non-null    float64
```

- Script diatas digunakan untuk mengetahui tipe apa saja yang ada di dalam data.

```
[ ] #Cek missing value
    data.isnull().sum()

diagnosis          0
radius_mean        0
texture_mean       0
perimeter_mean     0
area_mean          0
smoothness_mean    0
compactness_mean   0
concavity_mean     0
concave points_mean 0
symmetry_mean      0
fractal_dimension_mean 0
radius_se          0
texture_se         0
perimeter_se       0
area_se           0
smoothness_se      0
compactness_se     0
concavity_se       0
concave points_se  0
symmetry_se        0
fractal_dimension_se 0
radius_worst       0
texture_worst      0
perimeter_worst    0
area_worst         0
smoothness_worst   0
compactness_worst  0
concavity_worst    0
concave points_worst 0
symmetry_worst     0
fractal_dimension_worst 0
dtype: int64
```

- Script diatas digunakan untuk mengetahui apakah ada value/nilai yang hilang pada data.

🚦 Modeling pada data

```
[ ] from sklearn.metrics import confusion_matrix
    from sklearn.metrics import classification_report
    from sklearn.metrics import accuracy_score
    from sklearn import model_selection
    from sklearn.model_selection import GridSearchCV

    from sklearn.naive_bayes import GaussianNB

    target_st = ['B', 'M']
```

- Untuk script diatas digunakan untuk mengimport sklearn agar dapat memodeling data.

```
🔍 #Naive Bayes (Gaussian)

gnb = GaussianNB()
ntype = np.unique(targets)

gnb_grid = {
    'priors': [None, [0.1,]* len(ntype),],
    'var_smoothing': [ 1e-6, 1e-12]
}

gnb_gs = GridSearchCV(gnb, gnb_grid, n_jobs = -1)
gnb_fm = gnb_gs.fit(feature_train, target_train)

print("\nBest Parameters: ", gnb_gs.best_params_)
print("\nTraining Accuracy: ", gnb_gs.best_score_)

gnb_prdct = gnb_fm.predict(feature_test)
gnb_ac = accuracy_score(target_test, gnb_prdct)

print("Classification Report:\n", classification_report(target_test, gnb_prdct, target_names = target_st))
print("Confusion Matrix:\n", confusion_matrix(target_test, gnb_prdct))
print("\nAccuracy Score: ", gnb_ac)
```

```
👤 Best Parameters: {'priors': None, 'var_smoothing': 1e-06}

Training Accuracy: 0.9373101265822784
Classification Report:
              precision    recall  f1-score   support

         B            0.95      0.96      0.96         110
         M            0.93      0.92      0.93          61

   accuracy                0.95         171
  macro avg              0.94      0.94      0.94         171
weighted avg              0.95      0.95      0.95         171

Confusion Matrix:
[[106  4]
 [  5 56]]

Accuracy Score: 0.9473684210526315
```

- Script tersebut untuk menampilkan algoritma dan nilai akurasi pada data.

BAB III

PENUTUP

KESIMPULAN

Naive Bayes adalah salah satu algoritma yang sering digunakan dalam pengenalan pola, analisis teks, klasifikasi dokumen, dan banyak aplikasi lainnya.

Algoritma Naive Bayes mengandalkan asumsi dasar yang cukup sederhana, yaitu bahwa semua atribut (fitur) yang digunakan dalam klasifikasi adalah independen satu sama lain. Oleh karena itu, istilah "naive" digunakan, karena dalam dunia nyata, atribut seringkali tidak benar-benar independen. Namun, asumsi ini mempermudah perhitungan matematis dan sering kali menghasilkan hasil yang cukup baik, terutama dalam kasus klasifikasi teks.

Rumus dasar Naive Bayes adalah berdasarkan pada Teorema Bayes, yang menghubungkan probabilitas suatu peristiwa dengan probabilitas peristiwa lain yang berhubungan. Dalam konteks klasifikasi, Naive Bayes digunakan untuk menghitung probabilitas bahwa sebuah sampel data termasuk dalam suatu kelas tertentu.