1. Dataset

Dataset yang digunakan untuk melakukan klasifikasi apakah dapat melakukan permainan golf atau tidak melalui atribut Outlook,Temp,Humidity,dan Windy dengan kelas Play Golf. Data yang digunakan 80% untuk data training dan 20% untuk data testing. Data didapatkan melalui sumber **https://www.kaggle.com/priy998/golf-play-dataset**.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Outlook | Temp | Humidity | Windy | Play Golf |
| Rainy | Hot | High | FALSE | No |
| Rainy | Hot | High | TRUE | No |
| Overcast | Hot | High | FALSE | Yes |
| Sunny | Mild | High | FALSE | Yes |
| Sunny | Cool | Normal | FALSE | Yes |
| Sunny | Cool | Normal | TRUE | No |
| Overcast | Cool | Normal | TRUE | Yes |
| Rainy | Mild | High | FALSE | No |
| Rainy | Cool | Normal | FALSE | Yes |
| Sunny | Mild | Normal | FALSE | Yes |
| Rainy | Mild | Normal | TRUE | Yes |
| Overcast | Mild | High | TRUE | Yes |
| Overcast | Hot | Normal | FALSE | Yes |
| Sunny | Mild | High | TRUE | No |

1. Klasifikasi Naïve Bayes

*Naïve Bayes* merupakan teknik probalistik klasifikasi yang sederhana berdasarkan pada penerapan teorema *Bayes* atau aturan *Bayes* dengan asumsi independesi (ketidak tergantungan) yang kuat (naif) atau dapat dikatakan dalam suatu fitur yang terdapat dalam kelas tidak terkait dengan adanya fitur lain (Anam, 2018). *Naïve Bayesian Classification* didasarkan pada teorema *Bayes* yang memiliki kemampuan klasifikasi serupa dengan *decision tree* dan *neural network*. *Naïve Bayesian Classification* terbukti memiliki akurasi dan kecepatan yang tinggi. Secara umum, perhitungan *Naïve Bayes* atau teorema Bayes yaitu probabilitas dari kemunculan atribut B pada kategori kelas P(A|B), dikali dengan probabilitas kelas P(A). Dari hasil kali tersebut kemudian dilakukan pembagian terhadap probabilitas kemunculan atribut P(B). Perhitungan *Naïve Bayes* dapat dituliskan seperti pada persamaan sebagai berikut:

𝑃(𝐻|𝑋)=

Dalam hal ini :

X = Data dengan class yang belum diketahui

H = Hipotesis data X merupakan suatu class spesifik

P(H|X) = Probabilitas hipotesis H berdasar kondisi X (posteriori probability)

P(H) = Probabilitas hipotesis H (prior probability)

P(X|H) = Probabilitas X berdasar kondisi pada hipotesis H

P(X) = Probabilitas dari X

1. Perhitungan Klasifikasi dengan Naïve Bayes

**P(Hi)**

P(‘Play Golf’ = ‘Yes’) = 7/11 = **0.636**

P(‘Play Golf’ = ‘No’) = 4/11 = **0.363**

**P(X|Hi)**

**Atribut Outlook dengan Kelas Yes dan No**

P(Outlook = ‘Rainy’ | Play Golf = ‘Yes’) = 2/7 = **0.285**

P(Outlook = ‘Overcast’ | Play Golf = ‘Yes’) = 2/7 **= 0.285**

P(Outlook = ‘Sunny’ | Play Golf = ‘Yes’) = 3/7 = **0.428**

P(Outlook = ‘Rainy’ | Play Golf = ‘No’) = 3/4 = **0.75**

P(Outlook = ‘Overcast’ | Play Golf = ‘No’) = 0/4 = **0**

P(Outlook = ‘Sunny’ | Play Golf = ‘No’) = 1/4 = **0.25**

**Atribut Temp dengan Kelas Yes dan No**

P(Temp = ‘Hot | Play Golf = ‘Yes’) = 1/7 = **0.142**

P(Temp = ‘Mild’ | Play Golf = ‘Yes’) = 3/7 = **0.428**

P(Temp = ‘Cool’ | Play Golf = ‘Yes’) = 3/7 = **0.428**

P(Temp = ‘Hot | Play Golf = ‘No’) = 2/4 = **0.5**

P(Temp = ‘Mild’ | Play Golf = ‘No’) = 1/4 = **0.25**

P(Temp = ‘Cool’ | Play Golf = ‘No’) = 1/4 = **0.25**

**Atribut Humidity dengan Kelas Yes dan No**

P(Humidity = ‘High’ | Play Golf = ‘Yes’) = 2/7 = **0.285**

P(Humidity = ‘Normal’ | Play Golf = ‘Yes’) = 5/7 = **0.714**

P(Humidity = ‘High’ | Play Golf = ‘No’) = 3/4 = **0.75**

P(Humidity = ‘Normal’ | Play Golf = ‘No’) = 1/4 = **0.25**

**Atribut Humidity dengan Kelas Yes dan No**

P(Windy = ‘False’ | Play Golf = ‘Yes’) = 5/7 = **0.714**

P(Windy = ‘True’ | Play Golf = ‘Yes’) = 2/7 = **0.285**

P(Windy = ‘False’ | Play Golf = ‘No’) = 2/4 = **0.5**

P(Windy = ‘True’ | Play Golf = ‘No’) = 2/4 = **0.5**

**Data Testing :**

**Overcast,Hot,Normal,FALSE**

P(X|Play Golf = “Yes”) = **0.285 \* 0.142 \* 0.714 \* 0.714 = 0,02063144412**

P(X|Play Golf = “No”) = **0 \* 0.5 \* 0.25 \* 0.5 = 0**

**Sunny,Mild,High,TRUE**

P(X|Play Golf = “Yes”) = **0.428 \* 0.428 \* 0.285 \* 0.285 = 0,0148791204**

P(X|Play Golf = “No”) = **0.25 \* 0.25 \* 0.75 \* 0.5 = 0.0234375**

**P(X|Hi)\*P(Hi)**

**Overcast,Hot,Normal,FALSE**

P(X|Play Golf = “Yes”) \* P(‘Play Golf’ = ‘Yes’) = **0.02063144412\*0.636 = 0,01312159846032**

P(X|Play Golf = “No”) \* P(‘Play Golf’ = ‘No) = **0\*0.363= 0**

**Sunny,Mild,High,TRUE**

P(X|Play Golf = “Yes”) \* P(‘Play Golf’ = ‘Yes) = **0,0148791204\*0.636 = 0,0094631205744**

P(X|Play Golf = “No”) \* P(‘Play Golf’ = ‘No) = **0.0234375 \* 0.363 = 0,0085078125**

**Jadi hasil dari data test**

**Overcast,Hot,Normal,FALSE adalah “Yes”**

**Sunny,Mild,High,TRUE adalah “Yes”**

1. Evaluasi

Untuk melakukan evaluasi pada metode klasifikasi *Naïve Bayes* menggunakan *Confusion Matrix*. Tujuan evaluasi untuk mengukur performa sistem yang dibangun. *Confusion Matrix* merupakan tabel yang digunakan untuk menentukan kinerja suatu model klasifikasi atau prediksi pada klasifikasi.

|  |  |  |  |
| --- | --- | --- | --- |
| *Classification* | *Predicted Class* | | |
|  | *C1* | *C2* | *C3* |
| *C1* |  |  |  |
| *C2* |  |  |  |
| *C3* |  |  |  |

Dari Tabel tersebut terdapat pengujian data dalam suatu proses pengujian yaitu*:*

1. Total True Positive Class (TTP), yaitu total hasi dari prediksi sistem yang positif dan sesui dengan target yang positif. Persamaan TTP dapat dilihat pada persamaan berikut:
2. Total True Negative Class (TTN), yaitu total hasil dari prediksi sistem yang negtif dan sesui dengan target yang negatife. Persamaan TTN dapat dilihat pada persamaan berikut:
3. Total False Negative (TFN), yaitu total hasil dari prediksi sistem yang positif namun hasil target yang peroleh negatif. Persamaan TFN dapat dilihat pada persamaan berikut:
4. Total False Positive (TFP), yaitu total hasi dari prediksi sistem yang negatif namun hasil target yang peroleh positif. Persamaan TFP dapat dilihat pada persamaan berikut:

Akurasi merupakan perbandingan antara data yang terklasifikasi benar dengan keseluruhan data. Persamaan Akurasi sebagai berikut:

Akurasi =

**DAFTAR PUSTAKA**

Anam, C. (2018). Perbandingan Kinerja Algoritma C4.5 dan Naive Bayes untuk Klasifikasi Penerima Beasiswa.

**LAMPIRAN SOURCE CODE**

import pandas as pd  
data = pd.read\_csv('golf-dataset.csv')  
def counter(data,colname,label,target):  
 temp = (data[colname] == label) & (data['Play Golf'] == target)  
 return len(data[temp])  
split\_data = [80]  
for i in split\_data:  
 prediksi = [] #menyimpan hasil prediksi nilai  
 probabilitas = {0:{},1:{}} #menyimpan nilai probabilitas  
 train\_len = int((i\*len(data))/100) #menghitung banyaknya nilai training  
 #split training dan testing  
 train\_X = data.iloc[:train\_len,:] #mengambil data sebanyak train\_len  
 test\_X = data.iloc[train\_len+1:,:-1] #mendapatkan nilai data test  
 test\_Y = data.iloc[train\_len+1:,-1] #mengambil nilai class  
 #data,colname,label,target  
 count\_yes = counter(train\_X,'Play Golf','Yes','Yes') #total class yes pada data  
 count\_no = counter(train\_X, 'Play Golf','No','No') #total class no pada data  
 prob\_yes = count\_yes/len(train\_X) #nilai probabilitas pada class yes  
 prob\_no = count\_no/len(train\_X) #nilai probabilitas pada class no  
  
 #training model  
 for j in train\_X.columns[:-1]:  
 # nilai probabilitas pada setiap atribut terhadap class  
 probabilitas[1][j] = {}  
 probabilitas[0][j] = {}  
 for k in train\_X[j].unique():  
 count\_k\_yes = counter(train\_X, j, k, 'Yes')  
 count\_k\_no = counter(train\_X, j, k, 'No')  
 # print(count\_k\_no,k)  
 probabilitas[1][j][k] = count\_k\_yes/count\_yes  
 probabilitas[0][j][k] = count\_k\_no/count\_no  
  
 #test model  
 for baris in range(0,len(test\_X)):  
 hasil\_yes = prob\_yes  
 hasil\_no = prob\_no  
 for kolom in test\_X.columns:  
 hasil\_yes \*= probabilitas[1][kolom][test\_X[kolom].iloc[baris]]  
 hasil\_no \*= probabilitas[0][kolom][test\_X[kolom].iloc[baris]]  
 if hasil\_yes > hasil\_no:  
 prediksi.append('Yes')  
 else:  
 prediksi.append('No')  
  
 #confusion matrix  
 tp, tn, fp, fn = 0, 0, 0, 0  
 for j in range(0,len(prediksi)):  
 if prediksi[j] == 'Yes':  
 if test\_Y.iloc[j] == 'Yes':  
 tp+=1  
 else:  
 fp+=1  
 else:  
 if test\_Y.iloc[j] == 'No':  
 tn+=1  
 else:  
 fn+=1  
 print('Akurasi dengan data training '+str(i)+ '% :' ,(tp+tn)/len(test\_Y)\*100)

