Naive Bayes

Conceito

O Naive Bayes é uma técnica de classificação baseada no teorema de Bayes que tem como premissa a suposição de independência ou não correlação entre as variáveis do conjunto de dados sendo utilizadas para resolver um determinado problema. O termo “Naive” que significa ingênuo, compõem seu nome justamente por adotar esta premissa, ou seja, este classificador assume que as características de uma variável não têm relação com quaisquer outras variáveis do mesmo conjunto de dados.

Sendo assim, ele considera que todas as variáveis são igualmente importantes para o resultado. Um veículo, por exemplo, pode ser considerado um carro se possuir 4 rodas, chassi, motor e um volante. Mesmo que essas características dependam umas das outras para formar o carro, todas estas características contribuem de forma independente para a probabilidade de que o veículo seja um carro e por isso, o algoritmo Naive Bayes é conhecido como Ingênuo.

Por sua simplicidade, apesar de ingênuo, ele é conhecido por ganhar de métodos mais sofisticados de classificação.

Por se tratar de um classificador, o algoritmo Naive Bayes se encaixa no grupo de algoritmos de aprendizado supervisionado. Ele é bastante útil em situações que precisamos lidar com grandes volumes de dados realizando uma classificação probabilística de observações e as caracterizando em classes pré-definidas.

O Teorema de Bayes

Criado pelo matemático inglês Thomas Bayes (1701-1761), o Teorema de Bayes fornece uma maneira de calcular a probabilidade posterior $P(c|x)$ a partir de $P(c)$, $P(x)$ e $P(x|c)$. A fórmula matemática pode ser visualizada através da Figura 1 – Cálculo de Probabilidade T. Bayes:

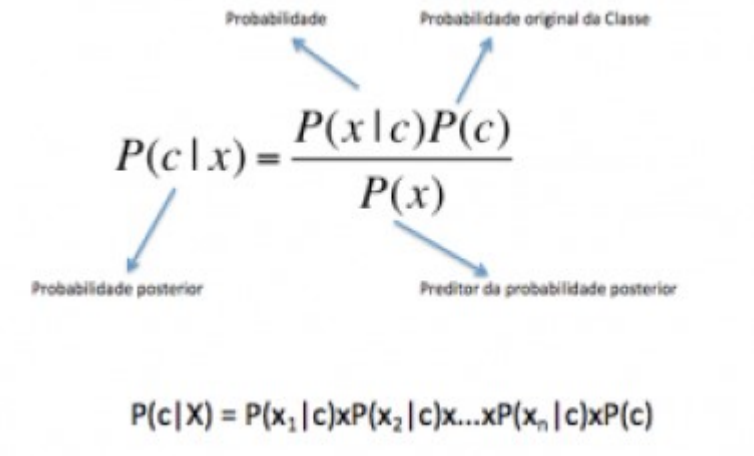

Figura 1 – Cálculo de Probabilidade T. Bayes

Onde,

$P(c|x)$: Probabilidade posterior da classe (c, alvo) dada preditor (x, atributos).

$P(c)$: Probabilidade original da classe.

$P(x|c)$: Probabilidade que representa a probabilidade de preditor dada a classe.

$P(x)$: Probabilidade original do preditor.

Como funciona

O funcionamento do Naive Bayes pode ser descrito em um conjunto de etapas baseadas em termos estatísticos, onde para realizar o cálculo da previsão, o algoritmo faz uso de uma tabela de probabilidade em que constam as frequências dos preditores em relação às variáveis de saída. Desta maneira, o cálculo final considera a maior probabilidade encontrada para produzir o resultado.

Para oferecer um melhor entendimento sobre seu funcionamento, abaixo temos um conjunto de dados de treinamento de clima com sua respectiva classe chamada de “Play”, na qual sugere as possibilidades de jogo de um determinado time de futebol. Nossa necessidade neste caso é classificar se vai haver ou não jogo com base na condição meteorológica. Aplicando as etapas lógicas do algoritmo Naive Bayes, temos:

1ª Etapa: Conversão do conjunto de dados em uma tabela de frequência;

2ª Etapa: Criação da tabela de probabilidades ao encontrar as probabilidades da variável tempo. Na Figura 2 – Etapas Utilizadas pelo Algoritmo Naive Bayes, é possível ver que a probabilidade do tempo Nublado = 0,29, Sol = 0,36 e Chuva = 0,36. Já a probabilidade de haver jogo é de 0,64;

TEMPO	"PLAY"
Sol	Não
Nublado	Sim
Chuva	Sim
Sol	Sim
Sol	Sim
Nublado	Sim
Chuva	Não
Chuva	Não
Sol	Sim
Chuva	Sim
Sol	Não
Nublado	Sim
Nublado	Sim
Chuva	Não

Tabela de frequência		
Clima	Não	Sim
Nublado	0	4
Sol	3	2
Chuva	2	3
Total	5	9

Tabela de probabilidade				
Clima	Não	Sim		
Nublado	0	4	=4/14	0,29
Sol	3	2	=5/14	0,36
Chuva	2	3	=5/14	0,36
Total	5	9		
	=5/14	=9/14		
	0,36	0,64		

Figura 2 – Etapas Utilizadas pelo Algoritmo Naive Bayes

3ª Etapa: Utilização da equação Bayesiana para realizar o cálculo da probabilidade posterior para cada classe. A classe que obtiver a maior probabilidade posterior será o resultado da previsão do nosso algoritmo.

Utilizando as etapas acima, é possível saber se haveria ou não jogo se o tempo estiver ensolarado, por exemplo. Utilizando a equação Bayesiana, temos:

$$P(Sim|Ensolarado) = (\frac{P(Ensolarado|Sim) * P(Sim)}{P(Ensolarado)})$$

Onde,

$P(Sim|Ensolarado)$

$P(Sim)$ = 9/14 = 0,64

$P(Ensolarado|Sim)$ = 3/9 = 0,33

$P(Ensolarado)$ = 5/14 = 0,36

Assim, nosso resultado poderia ser representado com $P(Sim|Ensolarado)$ = 0,33 * 0,64 / 0,36 = 0,60, que possui a maior probabilidade.

Tipos de Modelagem do Algoritmo Naive Bayes

Basicamente, existem 3 tipos de modelo Naive Bayes, são eles:

- Gaussian Naive Bayes: Utilizado na classificação assumindo uma distribuição normal. O cálculo da probabilidade é igual à densidade de probabilidade da distribuição normal. A média (μy) é o valor médio de xi, considerando as observações de classe y. E o desvio padrão (σy) é o desvio padrão da *feature* xi, considerando as observações da classe y. $P(xi|y)$ é dado por:

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} exp(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2})$$

- Bernoulli Naive Bayes: O modelo binomial é útil se os vetores são binários (0 ou 1). Uma aplicação seria de classificação de texto com um modelo de “saco de palavras” onde os uns e zeros seriam palavras que ocorrem no documento e palavras que não ocorrem no documento, respectivamente.

- Multinomial Naive Bayes: Esse algoritmo usa os dados em uma distribuição multinomial, que é uma generalização da distribuição binomial. Se, por exemplo, considerarmos um problema de classificação de texto. Podemos considerar tentativas de Bernoulli, onde ao invés de verificarmos a palavra que ocorre no documento, verificamos a contagem de vezes que a palavra apareceu no texto.

Campos de Aplicação

O Naive Bayes é um algoritmo bastante versátil. Abaixo, compartilho alguns campos de aplicação onde o mesmo pode ser utilizado:

- Previsões em tempo real: Naive Bayes é um classificador de aprendizagem rápida. Assim, pode ser usado para fazer previsões em tempo real;

- Classificador de *spam*: Ele analisa e-mails e tenta avaliar se ele é *spam* ou não (classes definidas) com base em suas informações e em estrutura. É um tipo de problema de processamento de linguagem natural;
- Classificadores de sentimento: Nesses casos, eles analisam os textos e tentam identificar a emoção expressada, geralmente entre opções específicas como “neutro”, “positivo” ou “negativo”;
- Área da Saúde: Como sistemas que determinam se alguém tem uma doença ou não;
- Sistemas de recomendação: Nesse caso, o objetivo é analisar certas pessoas e tentar sugerir algo que possa interessar a elas, seja conteúdo, seja produtos. É um tipo de abordagem de filtragem colaborativa.

Principais Vantagens e Desvantagens

Abaixo, compartilho as principais vantagens encontradas na adoção do algoritmo Naive Bayes, como também as desvantagens existentes nele:

Vantagens

- Possui bom desempenho na previsão de múltiplas classes;
- Lida melhor com variáveis de entrada categóricas do que com várias numéricas;
- Permite rápida e fácil previsão da classe de teste;
- Quando a suposição de independência ou não correlação entre as várias existe, o algoritmo gera um melhor desempenho e com menos dados de treinamento quando comparado com modelos concorrentes ao Naive Bayes.

Desvantagens

- A suposição de variáveis independentes pode ser uma limitação devido ser muito difícil ter um conjunto de dados em que não tenham variáveis com correlação positiva ou negativa entre elas;
- Em casos de variáveis categóricas, onde uma determinada categoria não foi observada nos dados de treinamento, o modelo atribui uma probabilidade zero não sendo capaz de gerar uma previsão. Este problema é conhecido como “*Zero Frequency_” e pode ser contornado com a técnica de alisamento chamada _Laplace Correction*;
- Este algoritmo pode não gerar boas estimativas, ou seja, pode haver probabilidades incorretas produzidas pelo algoritmo.

Aplicação Prática

Através do link abaixo, pode ser feito o download do(s) script(s) python criado(s) para exemplificar uma abordagem do uso prático do naive bayes como também o(s) respectivo(s) conjunto(s) de dados utilizados.

Script Python
Dataset 1
Dataset 2

```
## Naive Bayes

#### Carregando as bibliotecas
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix, accuracy_score
from yellowbrick.classifier import ConfusionMatrix

#### Carregando conjunto de dados
credito = pd.read_csv('credit.csv')
credito.shape

# Apresentando as 5 primeiras linhas
credito.head()

#### Alterando para um formato de matriz
previsores = credito.iloc[:,0:20].values
classe = credito.iloc[:,20].values

#### Transformação dos atributos categóricos em atributos numéricos
labelencoder1 = LabelEncoder()
previsores[:,0] = labelencoder1.fit_transform(previsores[:,0])

labelencoder2 = LabelEncoder()
previsores[:,2] = labelencoder2.fit_transform(previsores[:,2])

labelencoder3 = LabelEncoder()
previsores[:, 3] = labelencoder3.fit_transform(previsores[:, 3])

labelencoder4 = LabelEncoder()
previsores[:, 5] = labelencoder4.fit_transform(previsores[:, 5])

labelencoder5 = LabelEncoder()
previsores[:, 6] = labelencoder5.fit_transform(previsores[:, 6])

labelencoder6 = LabelEncoder()
previsores[:, 8] = labelencoder6.fit_transform(previsores[:, 8])

labelencoder7 = LabelEncoder()
previsores[:, 9] = labelencoder7.fit_transform(previsores[:, 9])

labelencoder8 = LabelEncoder()
previsores[:, 11] = labelencoder8.fit_transform(previsores[:, 11])
```

```
labelencoder9 = LabelEncoder()
previsores[:, 13] = labelencoder9.fit_transform(previsores[:, 13])

labelencoder10 = LabelEncoder()
previsores[:, 14] = labelencoder10.fit_transform(previsores[:, 14])

labelencoder11 = LabelEncoder()
previsores[:, 16] = labelencoder11.fit_transform(previsores[:, 16])

labelencoder12 = LabelEncoder()
previsores[:, 18] = labelencoder12.fit_transform(previsores[:, 18])

labelencoder13 = LabelEncoder()
previsores[:, 19] = labelencoder13.fit_transform(previsores[:, 19])

#### Segregando o conjunto de dados entre dados de treino e teste
X_treinamento, X_teste, y_treinamento, y_teste =
train_test_split(previsores,

classe,

test_size = 0.3,

random_state = 0)
X_teste

#### Criando o modelo
# naive_bayes = GaussianNB()
naive_bayes.fit(X_treinamento, y_treinamento)

#### Geração de previões
previsoes = naive_bayes.predict(X_teste)
previsoes

#### Matriz de confusão
confusao = confusion_matrix(y_teste, previsoes)
confusao

# Calculando a taxa de acerto e de erro
taxa_acerto = accuracy_score(y_teste, previsoes)
taxa_erro = 1 - taxa_acerto
taxa_acerto

# Gerando gráfico da matrix de confusão
v = ConfusionMatrix(GaussianNB())
v.fit(X_treinamento, y_treinamento)
```

```
v.score(X_teste, y_teste)
v.poof()

#### Previsão de novos registros

# Carregando novos dados
novo_credito = pd.read_csv('novo_credit.csv')
novo_credito.shape

#### Transformação dos atributos categóricos em atributos numéricos
novo_credito = novo_credito.iloc[:,0:20].values
novo_credito[:,0] = labelencoder1.transform(novo_credito[:,0])
novo_credito[:, 2] = labelencoder2.transform(novo_credito[:, 2])
novo_credito[:, 3] = labelencoder3.transform(novo_credito[:, 3])
novo_credito[:, 5] = labelencoder4.transform(novo_credito[:, 5])
novo_credito[:, 6] = labelencoder5.transform(novo_credito[:, 6])
novo_credito[:, 8] = labelencoder6.transform(novo_credito[:, 8])
novo_credito[:, 9] = labelencoder7.transform(novo_credito[:, 9])
novo_credito[:, 11] = labelencoder8.transform(novo_credito[:, 11])
novo_credito[:, 13] = labelencoder9.transform(novo_credito[:, 13])
novo_credito[:, 14] = labelencoder10.transform(novo_credito[:, 14])
novo_credito[:, 16] = labelencoder11.transform(novo_credito[:, 16])
novo_credito[:, 18] = labelencoder12.transform(novo_credito[:, 18])
novo_credito[:, 19] = labelencoder13.transform(novo_credito[:, 19])

#### Resultado da previsão
naive_bayes.predict(novo_credito)
```

Materiais complementares

- IA Expert Academy channel at Youtube: Naïve Bayes com Python
- ABRO, Abdul Ahad et al. Machine Learning Classifiers: A Brief Primer.
- LIU, Mujiexin et al. Naive Bayes clusterer. In: Data Science and Knowledge Engineering for Sensing Decision Support: Proceedings of the 13th International FLINS Conference (FLINS 2018). 2018. p. 637-644.
- WEBB, Geoffrey I.; KEOGH, Eamonn; MIIKKULAINEN, Risto. Naïve Bayes. Encyclopedia of machine learning, v. 15, p. 713-714, 2010.
- METSIS, Vangelis; ANDROUTSOPOULOS, Ion; PALIOURAS, Georgios. *spam* filtering with naive bayes-which naive bayes?. In: CEAS. 2006. p. 28-69.

Referências

Tópico anterior

- ABRO, Abdul Ahad et al. Machine Learning Classifiers: A Brief Primer.
- LIU, Mujiexin et al. Naive Bayes clusterer. In: Data Science and Knowledge Engineering for