

Naive Bayes

YURY PATRICIA BASTO FIGUEROA

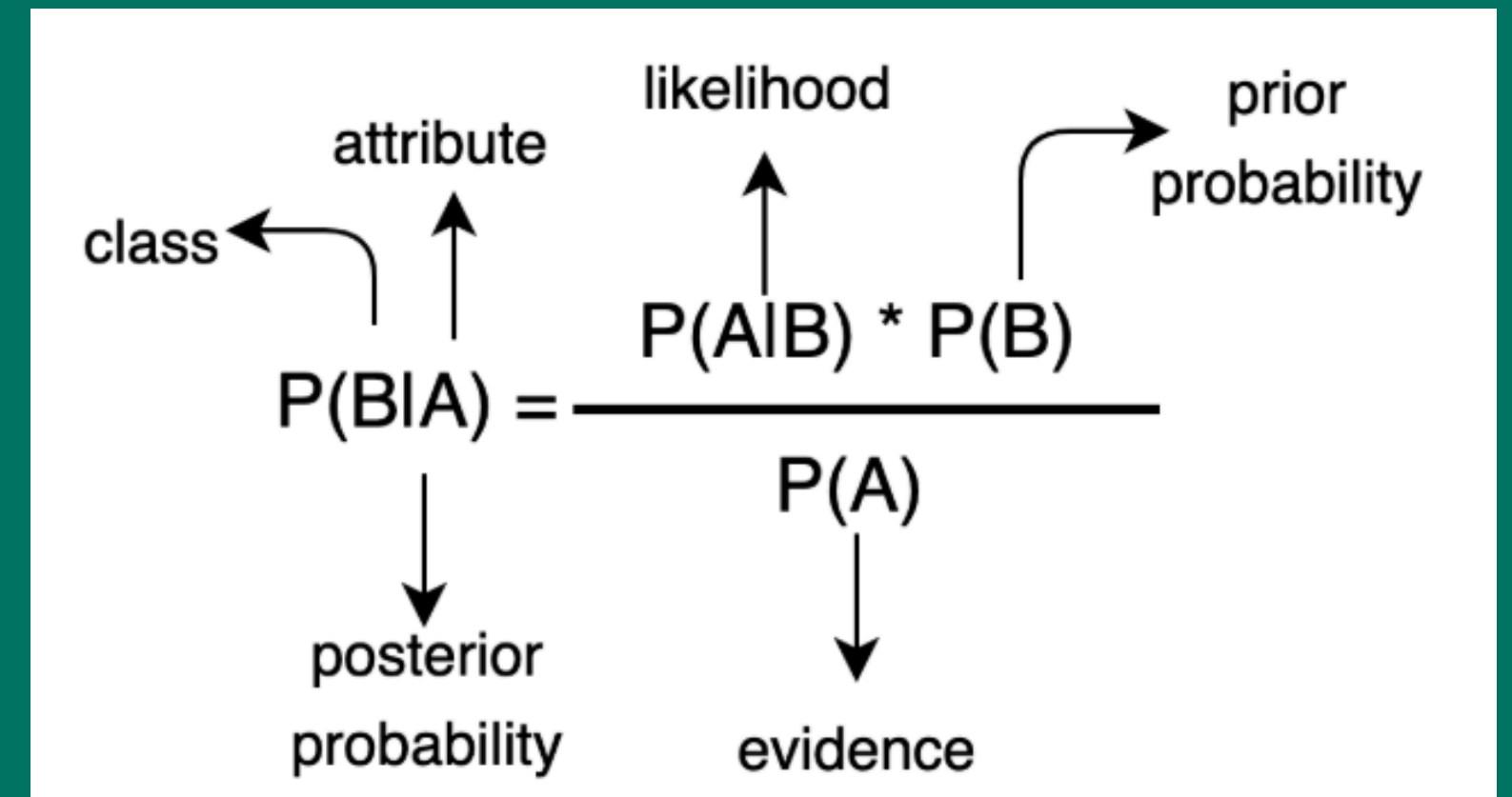
Naive Bayes Classifier

El algoritmo «Naive Bayes» es un clasificador probabilístico basado en el «Teorema de Bayes», el modelo fue creado por el matemático inglés, Thomas Bayes (1701 – 1761), para tratar de probar la existencia de Dios.



Teorema de Bayes

Este teorema se basa en calcular la probabilidad de que ocurra un cierto evento 'A', dado que ya ha ocurrido otro evento anterior 'B', lo que se denomina probabilidad condicional.



The diagram illustrates Bayes' Theorem with semantic labels for its components. The equation is $P(B|A) = \frac{P(A|B) * P(B)}{P(A)}$. Labels and arrows indicate the meaning of each term: 'class' points to $P(B|A)$; 'attribute' points to $P(A|B)$; 'likelihood' points to $P(A|B)$; 'prior probability' points to $P(B)$; 'posterior probability' points to $P(B|A)$; and 'evidence' points to $P(A)$.

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

Labels and arrows in the diagram:

- $P(B|A)$ is labeled as **posterior probability** (downward arrow) and **class** (leftward arrow).
- $P(A|B)$ is labeled as **likelihood** (upward arrow) and **attribute** (upward arrow).
- $P(B)$ is labeled as **prior probability** (upward arrow).
- $P(A)$ is labeled as **evidence** (downward arrow).

ell

EJEMPLO



El diagnóstico de enfermedades:

Sabiendo que:

- El resfriado provoca fiebre en el 50% de los casos;
- La probabilidad previa de que un paciente haya tenido un resfriado es $1/500000$
- La probabilidad previa de que un paciente haya tenido fiebre es $1/20$

¿Cuál es la probabilidad de que un paciente diagnosticado con fiebre haya estado resfriado?

$$P(A|B) = (P(B|A) \times P(A)) \div P(B) = (0,5 \times 1 \div 500000) \div (1 \div 20) = 0,0002$$



Es decir, la probabilidad de que el paciente haya estado resfriado previamente a la fiebre es del 0,02%.



¿Cómo funciona el algoritmo Naive Bayes?



$$P(S[?][?] | Sol) = (P(Sol | S[?][?]) \times P(S[?][?])) \div P(Sol)$$

$$P(Sol | S[?][?]) = 2 \div 7 = 0,28$$

$$P(S[?][?]) = 7 \div 10 = 0,70$$

$$P(Sol) = 3 \div 10 = 0,30$$

$$P(S[?][?] | Sol) = (0,28 \times 0,7) \div 0,3 = 0,65$$

En problemas reales, Naive Bayes utiliza un método similar para predecir la probabilidad de diferentes clases en función de varios atributos, clasificando según la probabilidad de clase más alta.

¿Cómo comprobar la multicolinealidad?



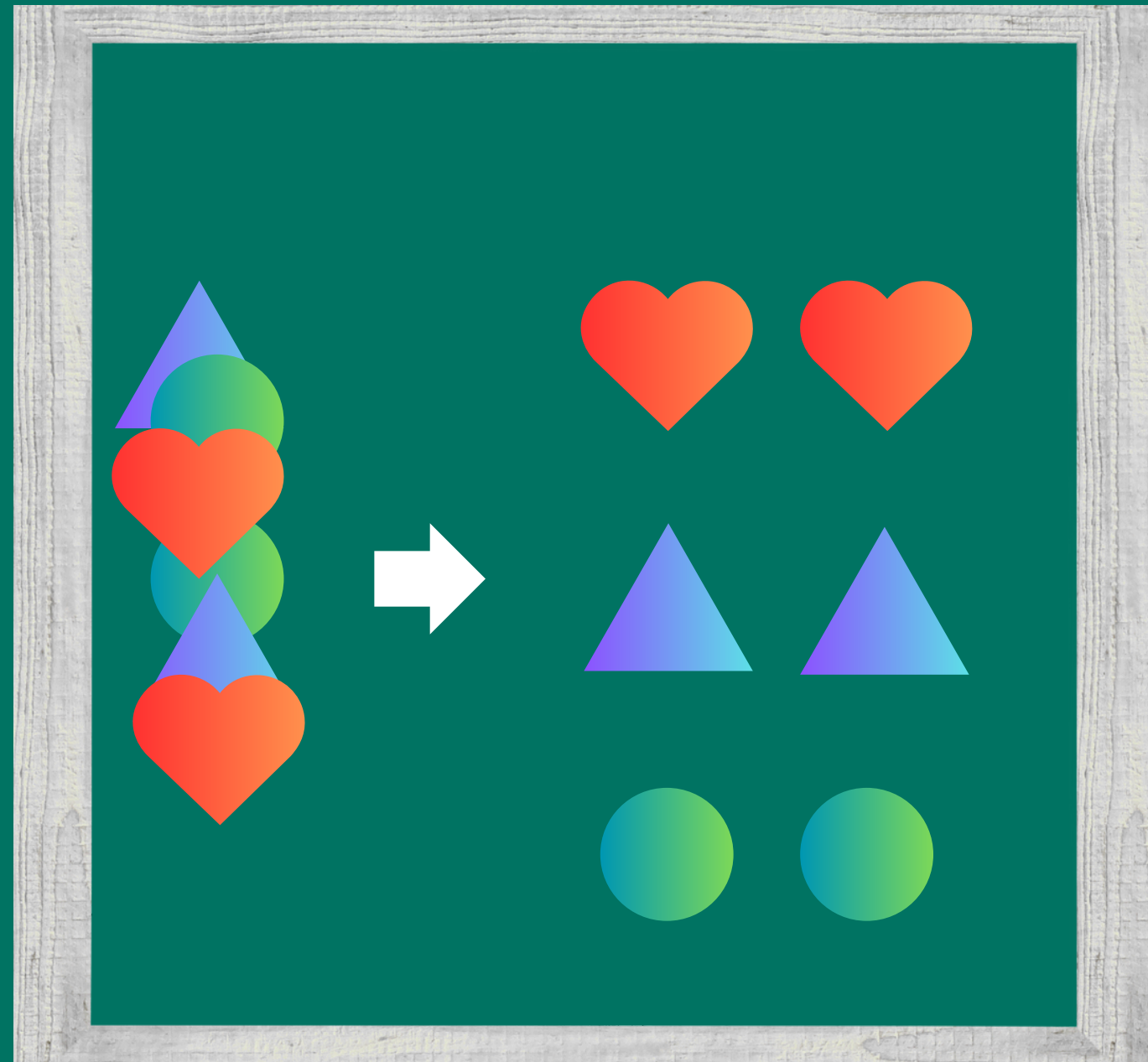
```
import pandas as pd  
df = pd.read_csv("data.csv")  
df.corr()
```

El siguiente código da como resultado la correlación de Pearson entre las columnas independientes y dependientes; podemos verificar la relación entre todas las columnas independientes con otra columna independiente para verificar la multicolinealidad.





¿Por qué el algoritmo se llama ingenuo?



el clasificador Naive Bayes asume que el conjunto de datos proporcionado al algoritmo es independiente y que las características independientes están separadas y no dependen de otros factores



Tipos de Bayes ingenuo



Naive Bayes
Classifier

Gaussian

Multinomial

Bernoulli

Bernoulli



```
from sklearn.datasets import  
make_classification  
from sklearn.naive_bayes import  
BernoulliNB  
from sklearn.model_selection import  
train_test_split  
nb_samples = 100  
X, Y =  
make_classification(n_samples=nb_sa  
mples, n_features=2, n_informative=2,  
n_redundant=0)  
X_train, X_test, Y_train, Y_test =  
train_test_split(X, Y, test_size=0.25)  
bnb = BernoulliNB(binarize=0.0)  
bnb.fit(X_train, Y_train)  
bnb.score(X_test, Y_test)
```

Este tipo de ingenuidad se utiliza principalmente en una columna de etiquetas categóricas binarias donde el enunciado del problema es predecir solo Sí o No. Por ejemplo, análisis de sentimientos con categorías positivas y negativas, un ID de orden específico presente o no en el texto

Bernoulli Bayes ingenuo

```
from sklearn.feature_extraction import DictVectorizer
from sklearn.naive_bayes import MultinomialNB

data = [
    {'parth1': 100, 'parth2': 50, 'parth3': 25, 'parth4': 100,
     'parth5': 20},
    {'parth1': 5, 'parth2': 5, 'parth3': 0, 'parth4': 10, 'parth5': 500,
     'parth6': 1}
]
dv = DictVectorizer(sparse=False)
X = dv.fit_transform(data)
Y = np.array([1, 0])
mnb = MultinomialNB()
mnb.fit(X, Y)

test_data = data = [
    {'parth1': 80, 'parth2': 20, 'parth3': 15, 'parth4': 70, 'parth5':
     10, 'parth6':
     1},
    {'parth1': 10, 'parth2': 5, 'parth3': 1, 'parth4': 8, 'parth5': 300,
     'parth6': 0}
]
mnb.predict(dv.fit_transform(test_data))
```

Este tipo de Bayes ingenuo se utiliza cuando los datos tienen una distribución multinomial. Se utiliza principalmente cuando existe un problema de clasificación de texto.

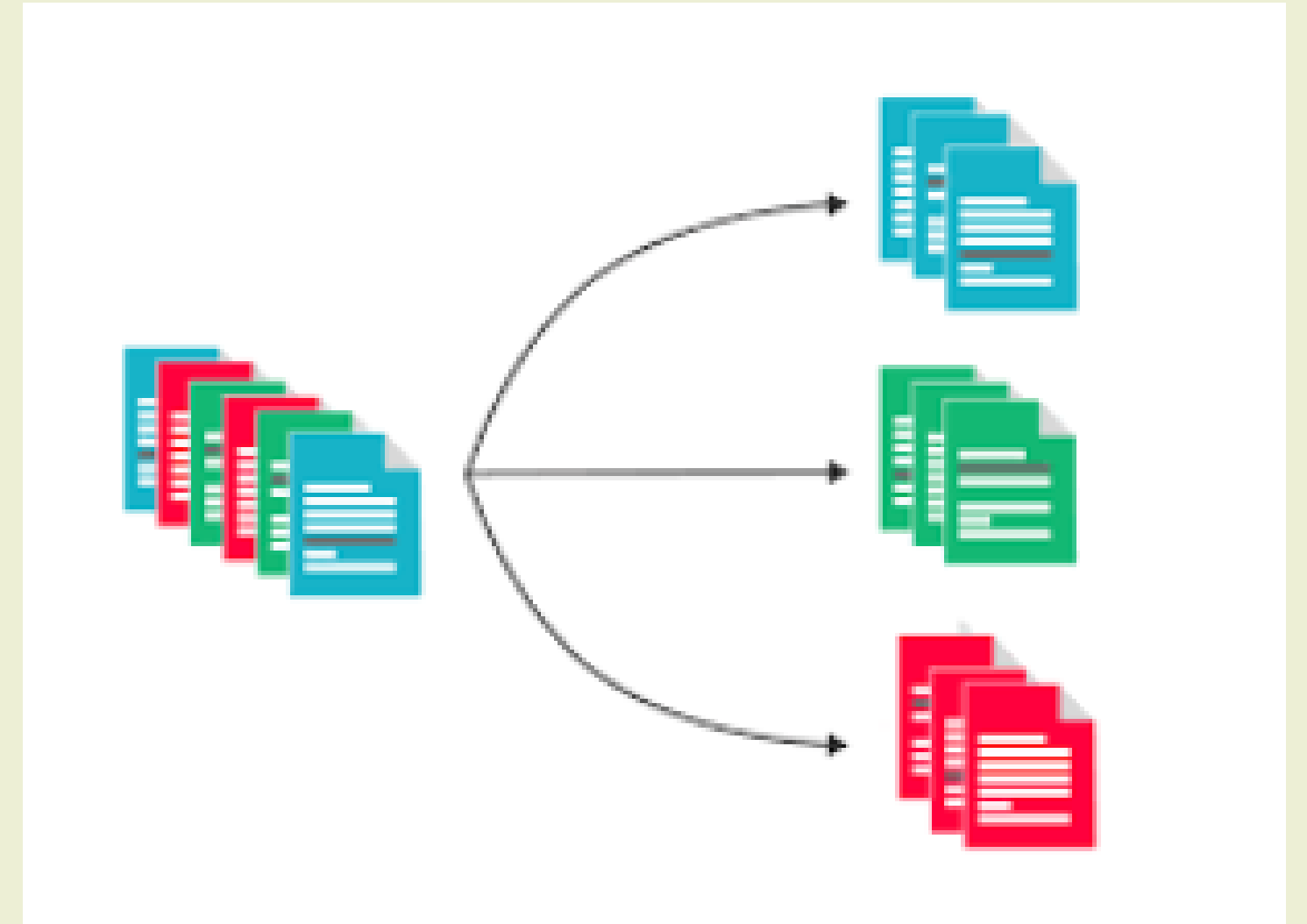
Bayes ingenuo gaussiano

```
from sklearn.datasets import  
make_classification  
from sklearn.naive_bayes import  
GaussianNB  
from sklearn.model_selection import  
train_test_split  
nb_samples = 100  
X, Y =  
make_classification(n_samples=nb_samples,  
n_features=2, n_informative=2,  
n_redundant=0)  
X_train, X_test, Y_train, Y_test =  
train_test_split(X, Y, test_size=0.25)  
gnb = GaussianNB(binarize=0.0)  
gnb.fit(X_train, Y_train)  
bngnb.score(X_test, Y_test)
```

Este tipo de ingenuidad se utiliza cuando las variables predictoras tienen valores continuos en lugar de discretos. En este caso, se asume que la distribución de los datos es gaussiana.

Aplicaciones

- Clasificación de texto
- Clasificación de Sentimientos
- . Sistemas de recomendación
- Predicciones en tiempo real



Ventajas

1. Algoritmos mas rápidos
2. Menos datos de entrenamiento
3. Rendimiento

Desventajas

1. Características independientes
2. Características independientes (Técnicas de suavizado de Laplace)

¿Cómo mejorar Naive Bayes?

- Eliminar características correlacionadas
 - Evitar multicolinealidad mejora el rendimiento.
- Ingeniería de características
 - Combinar y extraer rasgos útiles para mayor precisión.
- Conocimiento del dominio
 - Usar expertise acelera y mejora decisiones.
- Características probabilísticas
 - Resaltar rasgos que aportan mayor peso probabilístico.
- Transformada de Laplace
 - Soluciona categorías no vistas en el entrenamiento.
- Transformación de características
 - Aplicar Box-Cox o Yeo-Johnson para normalizar datos.

Conclusiones

- Se basa en probabilidades y el teorema de Bayes.
- Supone que las variables son independientes (de ahí lo “ingenuo”).
- Es un algoritmo simple, rápido y eficaz, ideal para clasificación.
- Tiene retos como el error de frecuencia cero y la multicolinealidad.
- Sus variantes (Bernoulli y Multinomial) lo hacen útil en texto y datos binarios.
- Con transformaciones adecuadas, puede mejorar su rendimiento.

Muchas gracias.



Any Questions?

Resource Page

