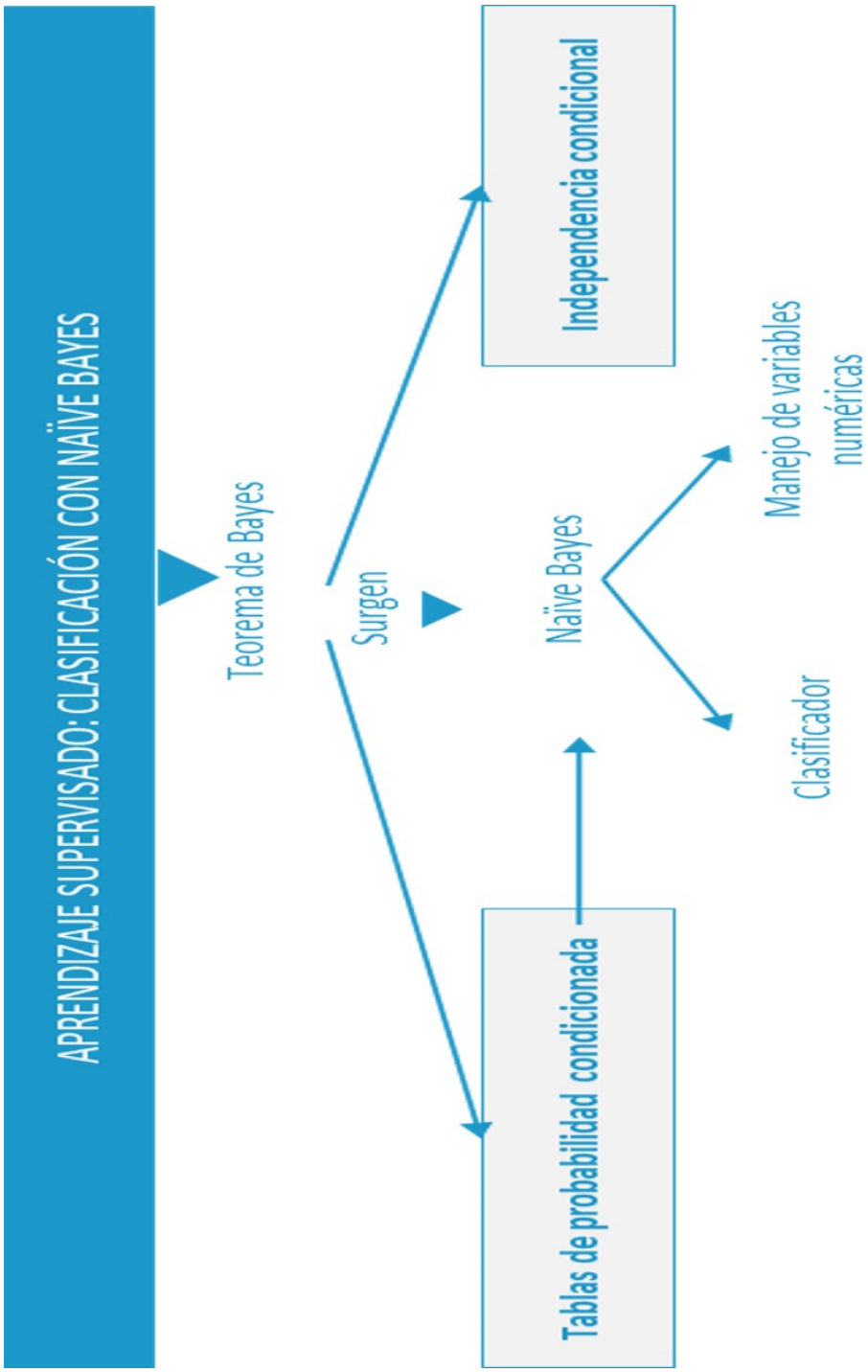
Técnicas de Aprendizaje Automático

Tema 8. Aprendizaje supervisado. Clasificación con Naïve Bayes

Índice

Esquema



8.1. Introducción

En este tema se va a introducir el clasificador Naïve Bayes, el cual está basado en el teorema de Bayes, que establece la relación entre probabilidad condicional y probabilidad *a priori*.

Es una técnica ampliamente utilizada en el campo del aprendizaje automático. Asume independencia entre las características o atributos de los datos, motivo por el cual se le denomina ingenuo. Aunque esta realidad no siempre sea así, resuelve una amplia gama de problemas y se utiliza en diversos ámbitos. Por ejemplo, en clasificación de texto, filtrado de spam, clasificación de documentos, detección de sentimientos y recomendación de productos. Su simplicidad y velocidad de procesamiento lo hacen fácil de implementar en entornos de grandes volúmenes de datos.

En este tema nos planteamos los siguientes **objetivos**:

- Describir la forma de calcular las tablas de probabilidad condicionada.
- Recordar el teorema de Bayes y su formulación.
- Aplicar el teorema de Bayes, asumiendo independencia condicional en diferentes conjuntos de datos.

8.2. Teorema de Bayes

El Teorema de Bayes es una proposición planteada por el filósofo inglés Thomas Bayes (1702-1761) en el año 1763 en su artículo *An Essay towards solving a Problem in the Doctrine of Chances* publicado en la revista Philosophical Transactions of the Royal Society of London. Naïve Bayes es un algoritmo simple pero sorprendentemente poderoso para el modelado predictivo.

Este teorema expresa la **probabilidad condicional** de un evento aleatorio A dado B, en términos de la distribución de probabilidad condicional del evento B dado A y la distribución de probabilidad marginal de solo A.

Figura 1. Reverendo Thomas Bayes. Fuente: Thomas Bayes, 2023.

El teorema de Bayes es bastante relevante porque **relaciona la probabilidad de dos eventos A y B utilizando la dependencia condicional de uno de ellos**. Es decir, relaciona la probabilidad de que ocurra el evento A y sabemos de antemano que ha ocurrido B, utilizando la probabilidad de que ocurra el evento B sabiendo que ha ocurrido A.

Ideas clave

El modelo se compone de dos tipos de probabilidades que se pueden calcular directamente a partir de los datos de entrenamiento:

- La probabilidad de cada clase.
- La probabilidad condicional para cada clase dado cada valor de x.

Una vez calculado, el modelo de probabilidad se puede utilizar para hacer predicciones de nuevos datos utilizando el teorema de Bayes. Cuando sus datos tienen valores reales, es común asumir una **distribución gaussiana** (curva de campana) para poder estimar fácilmente estas probabilidades.

Naïve Bayes es llamado ingenuo porque supone que cada variable de entrada es independiente. Esta es una suposición fuerte y poco realista para datos reales; sin embargo, la técnica es muy efectiva en una amplia gama de problemas complejos.

El teorema de Bayes **relaciona la comprensión de la probabilidad de aspectos causa-efecto dados los eventos dependientes observados**. Un evento dependiente es aquel cuyo resultado se ve afectado por el resultado de otro evento o serie de eventos. Los **eventos dependientes** son la base del modelado predictivo, puesto que se busca obtener la probabilidad de que ocurra un suceso teniendo en cuenta la existencia de una serie de eventos dependientes.

En el caso de dos eventos dependientes A y B, el teorema de Bayes describe su relación como en la ecuación (1):

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

Es decir, la probabilidad de A dado que ocurre B, es igual a la probabilidad de que ocurra B dado que ha ocurrido A, multiplicado por la probabilidad de que ocurra A, dividido por la probabilidad de que ocurra B.

Este teorema se puede **generalizar para más de dos eventos** de la siguiente forma: en primer lugar, se entiende por evento mutuamente excluyente cuando dos resultados diferentes de un evento no pueden ocurrir al mismo tiempo. En el caso, además, de que los eventos sean exhaustivos, por lo menos uno de ellos tiene que ocurrir.

De forma matemática, se puede definir el teorema de Bayes para n eventos: sea {A_1, A_2, …, A_i, …, A_n} un conjunto de sucesos mutuamente excluyentes y exhaustivos, y tales que la probabilidad de cada uno de ellos es distinta de cero. Sea B un suceso cualquiera del que se conocen las probabilidades condicionales $P(B|A_i)$. Entonces, la probabilidad $P(A_i|B)$ viene dada por la ecuación (2):

$$P(A_i|B) = \frac{P_{(B \vee A_i)} * P_{(A_i)}}{P_{(B)}} \quad (2)$$

Donde:

- $P(A_i)$ son las probabilidades *a priori*.
- $P(B|A_i)$ es la probabilidad de B en la hipótesis A_i.
- $P(A_i|B)$ son las probabilidades *a posteriori*.
- $P(B)$ es la verosimilitud marginal.

> Ejemplo 1. Probabilidad de que un mensaje sea spam.
>
> Supongamos que deseamos estimar la probabilidad de que un mensaje de correo electrónico sea spam si contiene la palabra «viagra». Debemos conocer la verosimilitud que sería la probabilidad de que la palabra viagra haya sido utilizada en mensajes spam previos, además debemos conocer la verosimilitud marginal que es la probabilidad de que la palabra viagra aparezca en un mensaje sea spam o normal. Además, es necesario conocer la probabilidad a priori de que un mensaje sea spam. Y aplicando

el teorema de Bayes se puede calcular la probabilidad a posteriori y si es mayor que 0,5 es más probable que el mensaje sea spam. En la ecuación (3) se muestra el cálculo que debería hacerse.

$$P\,(Spam\,|\,Viagra) = \frac{P(Viagra|Spam) * P(Spam)}{P(Viagra)} \quad (3)$$

Verosimilitud — Probabilidad a priori — Verosimilitud marginal

El **teorema de Bayes** relaciona la probabilidad de dos o más eventos utilizando la dependencia condicional de cada uno de ellos. Los eventos deben ser dependientes y mutuamente excluyentes.

8.3. Tablas de probabilidad condicionada

Para obtener cada uno de los componentes de las fórmulas anteriores es necesario construir una **tabla de frecuencias** que indica el número de veces que el evento aparece en cada una de las situaciones. En nuestro ejemplo de spam, es necesario calcular el número de veces que la palabra Viagra ha aparecido en los mensajes de spam. Esta tabla de frecuencias se utiliza posteriormente para calcular las tablas de verosimilitud o de probabilidad condicionada.

Ejemplo 2. Calcular la probabilidad de que un mensaje sea spam dado que el mensaje tiene la palabra viagra.

Siguiendo con el ejemplo de los mensajes de spam, en el caso hipotético de que tuviéramos la siguiente distribución histórica de 100 mensajes para la palabra Viagra. Un ejemplo de distribución sería el que podemos ver en la Tabla 1.

Frecuencia	Sí	No	Total
Spam	4	16	20
Ham	1	79	80
Total	5	95	100

Tabla 1. Ejemplo de distribución histórica de mensajes *spam* y *ham* para la palabra Viagra. Fuente: elaboración propia.

Obtendríamos la correspondiente tabla de verosimilitud (Tabla 2).

Verosimilitud	Sí	No	Total
Spam	4/20	16/20	20
Ham	1/80	79/80	80
Total	5/100	95/100	100

Tabla 2. Ejemplo de tabla de verosimilitud para mensajes *spam* y *ham* en función de la palabra Viagra. Fuente: elaboración propia.

Con estos datos, para calcular la probabilidad *a posteriori* de que un mensaje sea spam dado que nos ha llegado la palabra Viagra, tendríamos que hacer el cálculo de la ecuación (4):

$$P(Spam|Viagra) = [(4/20) * (20/100)] / (5/100) = 0.8 \quad (4)$$

Es decir, con los datos anteriores, la probabilidad de que un correo electrónico que contenga la palabra Viagra sea spam es del 0,8.

Ejemplo 3. Calcular la probabilidad de que un mensaje sea spam dado que el mensaje tiene las palabras *money*, *groceries* y *unsubscribe*.

Ahora supongamos que deseamos añadir a este cálculo otros términos más comunes que aparecen en los mensajes spam, como pueden ser *money*, *groceries* y *unsubscribe*.

En este caso, tendríamos la siguiente tabla de verosimilitud (Tabla 3):

	Viagra (W1)		Money (W2)		Groceries (W3)		Unsubscribe (W4)		
Verosimilitud	Sí	No	Sí	No	Sí	No	Sí	No	Tot.
Spam	4/20	16/20	10/20	10/20	0/20	20/20	12/20	8/20	20
Ham	1/80	79/80	14/80	66/80	8/80	72/80	23/80	57/80	80
Total	5/10	95/10	24/10	76/10	8/100	92/100	35/100	65/100	100

Tabla 3. Ejemplo de tabla de verosimilitud para los mensajes *spam* y *ham* en función de las palabras Viagra, *money*, *groceries* y *unsubscribe*. Fuente: elaboración propia.

De esta forma, si llega un nuevo mensaje que contiene las palabras Viagra y *unsubscribe*, pero no *money* ni *groceries*; utilizando el teorema de Bayes calcularíamos la ecuación (5):

$$P(Spam|W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4) = \frac{P(W_1|Spam)P(\neg W_2|Spam)P(\neg W_3|Spam)P(W_4|Spam) * P(Spam)}{P(W_1)P(\neg W_2)P(\neg W_3)P(W_4)}$$

El cálculo de la fórmula anterior es **computacionalmente costoso**, puesto que, a medida que se añaden nuevos términos, son necesarias grandes cantidades de memoria para almacenar todas las combinaciones. Para disminuir el coste computacional es necesario que se asuma independencia condicional.

8.4. Independencia condicional en el clasificador Naïve Bayes

Debido a que el cálculo riguroso de la fórmula del teorema de Bayes, como en el ejemplo anterior, es computacionalmente costoso, el clasificador Naïve Bayes se basa en una **modificación sencilla**. Básicamente, asume independencia condicional entre los eventos. Formalmente, **dos eventos son independientes** si el resultado del segundo evento no se ve afectado por el resultado del primer evento. Si A y B son eventos independientes, la probabilidad de que ambos eventos ocurran es el producto de las probabilidades de los eventos individuales.

Por otro lado, los **eventos dependientes son la base del modelado predictivo**, puesto que permiten predecir la presencia de un evento en función de otro. Por ejemplo, la presencia de nubes suele ser un evento predictivo de un día lluvioso, o la presencia de la palaba viagra en un correo electrónico suele ser un evento predictivo de spam.

No obstante, al no poder asumir dependencia condicional por el alto coste computacional, el clasificador Naïve Bayes asume **independencia condicional** entre los eventos condicionados al mismo valor de la clase. Este hecho es el que le ha dado el adjetivo de *Naïve* o clasificador ingenuo.

> Ejemplo 4. Calcular la probabilidad de que un mensaje sea spam asumiendo independencia condicional de las palabras Viagra, *money*, *groceries* y *unsubscribe*.
>
> En nuestro ejemplo anterior, asumiendo independencia condicional de las palabras para obtener la probabilidad de spam, tendríamos la ecuación (6):

$$P(Spam|W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4) = \frac{P(W_1|Spam)P(\neg W_2|Spam)P(\neg W_3|Spam)P(W_4|Spam) * P(Spam)}{P(W_1)P(\neg W_2)P(\neg W_3)P(W_4)}$$

= (4/20) * (10/20) * (20/20) * (12/20) * (20/100) = 0.012

Por otro lado, para obtener la probabilidad de *ham* utilizaríamos la ecuación (7):

$$P(Ham|W_1 \cap \neg W_2 \cap \neg W_3 \cap W_4) = \frac{P(W_1|Ham)P(\neg W_2|Ham)P(\neg W_3|Ham)P(W_4|Ham) * P(Ham)}{P(W_1)P(\neg W_2)P(\neg W_3)P(W_4))}$$

= (1/80) * (66/80) * (71/80) * (23/80) * (80/100) = 0.002

Como 0,012/0,002 = 6, se puede afirmar que es 6 veces más probable que el mensaje sea *spam* que *ham*.

Si queremos calcular la probabilidad de que el mensaje sea *spam*, sería igual a la verosimilitud de que el mensaje sea *spam* dividido por la verosimilitud de que sea *spam* o *ham*: 0,012/ (0,012 + 0,002) = 0,857

Análogamente, la probabilidad de ser *ham* es: 0,002/ (0,002 + 0,012) = 0,143

Por tanto, podemos estimar que, dadas las palabras del mensaje, hay una probabilidad de 0,857 de que sea *spam* y de 0,143 de que sea *ham*, y como son eventos mutuamente excluyentes suman 1.

8.5. Clasificador Naïve Bayes

Como hemos comentado previamente, el teorema de Bayes es la **base** para el clasificador Naïve Bayes. Este clasificador utiliza las probabilidades *a priori* de los eventos para estimar la **probabilidad de eventos futuros** por medio del teorema de Bayes.

Este clasificador **utiliza datos históricos o de entrenamiento** para calcular la probabilidad observada de cada evento en función de su vector de características.

La fórmula general del clasificador Naïve Bayes se puede definir de la siguiente manera: la probabilidad del nivel L de la clase C, dada la evidencia proporcionada por las variables de F1,… Fn, es igual al producto de las probabilidades de cada evidencia condicionada al nivel de la clase, la probabilidad *a priori* del nivel de la clase y un factor de escala 1/Z que convierte el resultado en probabilidad (8):

$$P(C_L|F_1,\ldots,F_n = \frac{1}{Z}p(C_L)\prod_{i=1}^{n} p(F_i|C_L)$$

Este clasificador se utiliza principalmente para **clasificar texto, para detección de intrusos en redes de computadores, diagnósticos médicos**, entre otros. Por ejemplo, se puede utilizar la frecuencia de las palabras de los correos electrónicos para identificar nuevos correos spam en el futuro.

Combinaciones desconocidas

Ejemplo de verosimilitud:

Supongamos que ahora recibimos un mensaje que contiene las palabras Viagra, *money*, *groceries* y *unsubscribe*.

La verosimilitud de *spam* es:

(4/20) * (10/20) * (0/20) * (12/20) * (20/100) = 0

Por otro lado, la verosimilitud de *ham* es:

(1/80) * (14/80) * (8/80) * (23/80) * (80/100) = 0.00005

La probabilidad de *spam* es:

0/(0 + 0.0099) = 0

Y la probabilidad de *ham* es:

0.00005 / (0 + 0.00005) = 1

Este problema sucede cuando un evento nunca ha ocurrido para una o más categorías de las clases. Por ejemplo, si nunca se ha visto el término *groceries* en un mensaje *spam* P(Spam | groceries) = 0.

La solución es añadir un pequeño número a todas las clases en la tabla, para asegurarse que no existe ninguna combinación con probabilidad de ocurrencia igual a 0, esto se conoce con el nombre de **estimador de Laplace** y la tabla de verosimilitud quedaría de la siguiente forma:

	Viagra (W1)		Money (W2)		Groceries (W3)		Unsubscribe (W4)		
Verosimilitud	Sí	No	Sí	No	Sí	No	Sí	No	Total
Spam	5/22	17/22	11/22	11/22	1/22	21/22	13/22	9/22	22
Ham	2/82	80/82	15/82	67/82	9/82	72/82	24/82	58/82	82
Total	7/104	97/104	26/104	78/104	10/104	93/104	37/104	67/104	104

Tabla 4. Tabla de verosimilitud de mensajes spam y ham utilizando el estimador de Laplace. Fuente: elaboración propia.

Por ejemplo, si usamos un valor de 1, la verosimilitud de *spam* y *ham* quedaría:

(5/22) * (11/22) * (1/22) * (13/22) * (22/104) = 0.00064

(2/82) * (15/82) * (9/82) * (24/82)*(82/104) = 0.00011

Y la probabilidad de que sea *spam* y *ham* es:

0.00064/(0.00064+0.00011)=0.85

0. 00011/(0.00064+0.00011)=0.15

La clasificación del mensaje es para la clase spam

8.6. Clasificador Naïve Bayes con variables numéricas

El rendimiento del Naïve Bayes y de otros algoritmos de aprendizaje automático puede verse afectado si las variables no tienen distribución de probabilidad normal. En muchas ocasiones, las variables de entrada pueden tener distribuciones muy sesgadas, como la distribución exponencial o las distribuciones multimodales. Esto presenta un desafío para modelar con Naïve Bayes y otros modelos de aprendizaje automático. Un enfoque es usar la transformada de la variable numérica para tener una distribución de probabilidad discreta donde a cada valor numérico se le asigna una etiqueta y las etiquetas tienen una relación ordinal. A esto se le denomina *binning* o transformación de discretización, y mejora el rendimiento del Naïve Bayes.

En el caso de Naïve Bayes al utilizar tablas de frecuencias para calcular las probabilidades, cada una de las variables utilizada debe de ser **categórica** y no se pueden utilizar de forma directa variables numéricas. **Discretizar las variables numéricas en N conjuntos, agrupamientos o *bins*** es un método ideal.

Transformaciones y discretización

La discretización es entonces el proceso de traducir una variable cuantitativa en un conjunto de dos o más categorías. Los valores de la variable son agrupados en contenedores discretos, a cada contenedor se le asigna un número entero único. De este modo se conserva la relación ordinal entre los contenedores. A menudo se le denomina *binning* o *k-bins*, donde k se refiere al número de grupos creados.

Se pueden usar métodos para agrupar los valores en k contenedores discretos. Se usan técnicas tales como:

- **Uniform:** cada contenedor tiene el mismo ancho e intervalo de valores.

- **Quantile:** cada contendor tiene la misma cantidad de valores, se divide por percentiles.
- **Clustered:** se identifican grupos y se le asignan instancias a cada grupo.

Es importante considerar el punto de corte óptimo para hacer cada uno de los agrupamientos. Una buena solución suele ser explorar los datos para observar los puntos de corte en la distribución de los datos.

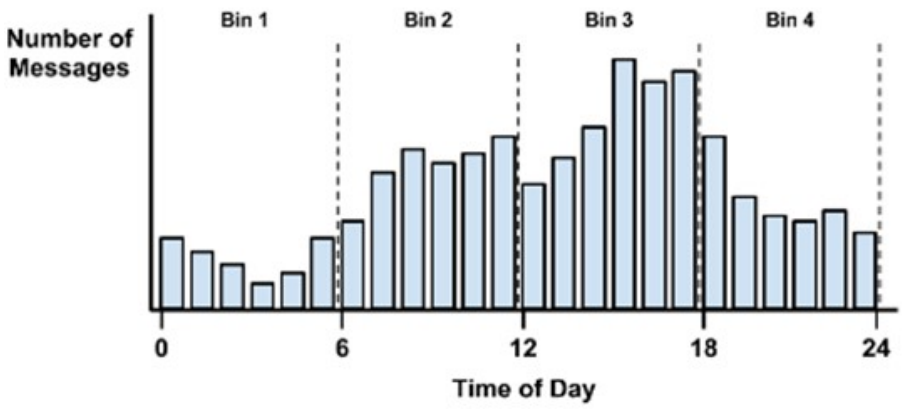


Figura 2. Discretización de las horas del día Fuente: Lantz, 2013.

En el histograma anterior se sugiere realizar una división en cuatro bins, 6 horas por cada bin.

La **discretización** siempre se traduce en una reducción de la información, ya que la granularidad inicial se reduce. Por tanto, es importante mantener un balance entre el número de *bins*, puesto que con muy pocos se pierde mucha información y con muchos el proceso es muy costoso.

La transformación de discretización está disponible en scikit learn a través de la clase *KBinDiscretizer*. Los argumentos que debemos pasar a la función son `strategy`, `n_bins` y `encode`.

Ideas clave

- **Strategy** hace referencia a la estrategia de división: uniforme, cuantil, kmedias.
- **N_bins** hace referencia a la cantidad de contenedores que deseo crear, aunque esto depende de la estrategia. Si es **uniforme**, es flexible. Si es **cuantil**, el número de n_bins debe ser menor que el número de observaciones o percentiles. Y **kmeans** debe usar un valor para el número de clúster que se puedan encontrar.
- **Encode** es la codificación ordinal. A cada valor numérico se le asignará un valor entero u ordinal, como en el ejemplo de discretización de las horas del día.

Vamos a verlo a través de un ejemplo práctico:

Ejemplo de discretización en Python

```
from numpy.random import randn

from sklearn.preprocessing import KBinsDiscretizer

from matplotlib import pyplot

#Generar 200 datos aleatorios

discretData=randn(200)

#Ver el histograma de los datos

pyplot.hist(discretData, bins=25)

pyplot.show()

#Pasar a un arreglo con filas y columnas

discretData = discretData.reshape(len(discretData),1)

#Transformación de discretización con KBinDiscretizer

bins=KBinsDiscretizer(n_bins=10, encode='ordinal', strategy='uniform')
```

```
dataTrans=bins.fit_transform(discretData)

dataTrans

pyplot.hist(dataTrans, bins=10)

pyplot.show()
```

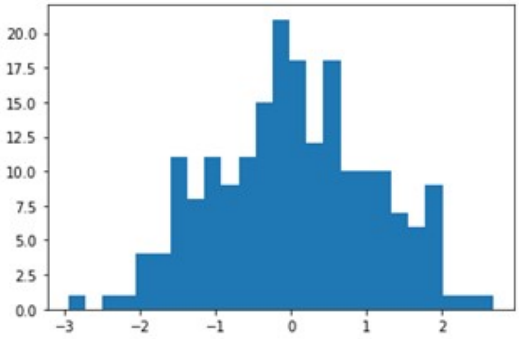


Figura 3. Histograma de datos brutos. Fuente: elaboración propia.

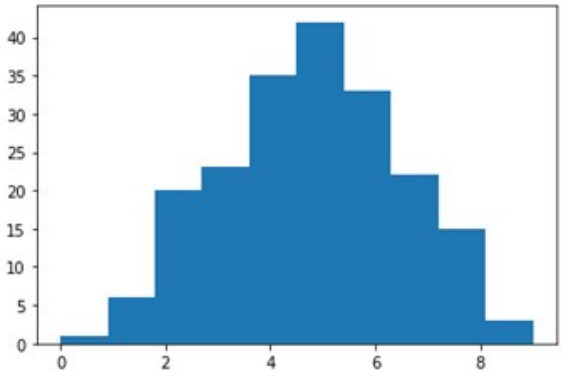


Figura 4. Histograma de datos discretizados. Fuente: elaboración propia.

Naïve Bayes en Python

En la librería de scikit learn existen varias implementaciones: la Gaussiana, la Multinomial y Bernoulli.

Gaussian Naïve Bayes implementa el algoritmo para la clasificación y asume que la distribución de probabilidad de las características es Gaussiana. La función de probabilidad que utiliza es la mostrada en la ecuación (9) (scikit learn.org, s.f.)

$$P_{(X_i|y)} = \frac{1}{\sqrt{2\pi\sigma_y^2}}\exp\left(-\frac{(X_i-\mu_y)^2}{2\sigma_y^2}\right)$$

Multinomial Naïve Bayes implementa el algoritmo Naïve Bayes para datos distribuidos multinomialmente; es muy utilizado en la clasificación de texto. Los datos se representan como conteos de vectores de palabras. La distribución está parametrizada por vectores para cada clase. En este algoritmo se calcula el número de veces que aparece una característica en la muestra para esa clase (scikit learn.org, s.f.).

BernoulliNB Naïve Bayes implementa los algoritmos ingenuos de clasificación y entrenamiento de Bayes para datos que se distribuyen de acuerdo con distribuciones multivariadas de Bernoulli; es decir, puede haber varias características, pero se supone que cada una es una variable de valor binario (Bernoulli, booleano). Por lo tanto, esta clase requiere que las muestras se representen como vectores de características con valores binarios (scikit-learn.org, s.f.).

Veamos un ejemplo en Python (scikit-learn.org, s.f.):

Ejemplo de Naïve Bayes en Python.

En este ejemplo se implementa un modelo utilizando el dataset de iris y se entrena un modelo usando la implementación Gaussian Naïve Bayes.

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
X, y = load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5, random_state=0)
gnb = GaussianNB()
y_pred = gnb.fit(X_train, y_train).predict(X_test)
print("Number of mislabeled points out of a total %d points : %d" % (X_test.shape[0],
                                                                     (y_test != y_pred).sum()))
```

```
Number of mislabeled points out of a total 75 points : 4
```

Figura 5. Implementación de Gaussian Naïve Bayes.

8.7. Cuaderno de ejercicios

- Con la siguiente tabla de verosimilitud, halla la probabilidad *a posteriori* de que un correo sea correo deseado si contiene la palabra *Unsubscribe*.

	Viagra(W_1)		Money (W_2)		Groceries (W_3)		Unsubscribe(W_4)		
Probabilidad	Si	No	Si	No	Si	No	Si	No	Total
spam	4/20	16/20	10/20	10/20	0/20	20/20	12/20	8/20	20
ham	1/80	79/80	14/80	66//80	8/80	72/80	23/80	57/80	80
Total	5/10	95/100	24/100	76/100	8/100	92/100	35/100	65/100	100

Figura 6. Tabla de verosimilitud de mensajes *spam* y *ham* en la que se analizan cuatro palabras: viagra, *money*, *groceries* y *unsubscribe*. Fuente: elaboración propia.

SOLUCIÓN

```
# P( ham |'Unsubscribe' ) = P('Unsubscribe'|ham) * P(ham) /
P('Unsubscribe')

P_ham = ((23/80)*(80/100)) / (35/100)

print('probabilidad posterior P (ham|unsubscribe): '+str(P_ham))
```

```
probabilidad posterior P (ham|unsubscribe): 0.6571428571428571
```

- Crea un modelo Naïve Bayes con el dataset de *wine* que se encuentra en la librería de sklearn. El modelo debe predecir la clase de vino. Analiza si es un modelo de regresión o de clasificación y aplica las mejores métricas para evaluar el modelo.

```
from sklearn.datasets import load_wine

data = load_wine()

y = data.target
```

```
X = data.data

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20,
random_state = 0)



from sklearn.naive_bayes import GaussianNB

classifier = GaussianNB()

classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)



from sklearn.metrics import confusion_matrix,accuracy_score

cm = confusion_matrix(y_test, y_pred)

ac = accuracy_score(y_test,y_pred)



from sklearn.metrics import precision_recall_fscore_support



precision_recall_fscore_support(y_test, y_pred, average=None,

                                labels=[0, 1, 2])
```

- Supongamos que tienes un conjunto de datos que contienen información sobre correos electrónicos y su clasificación en «spam» o «no spam». Utilizando el algoritmo de clasificador ingenuo Naïve Bayes, implementa un modelo que pueda clasificar nuevos correos electrónicos como «spam» o «no spam». En el preprocesamiento tendrás que eliminar palabras comunes y realizar técnicas de lematización. Describe uno a uno los pasos a seguir:

SOLUCIÓN

- Preprocesar los datos. Limpiarlos, normalizarlos, eliminar signos de puntuación, palabras vacías y realizar técnicas de reducción de palabras con lematización.
- Construir un modelo. Calcula las probabilidades condicionales para cada palabra en los correos electrónicos, considerando tanto la probabilidad de que aparezca en correos de spam como en correos no spam.
- Realiza la clasificación. Utiliza el modelo para clasificar nuevos correos electrónicos en «spam» o «no spam» basándose en la probabilidad condicional de cada palabra en el correo.
- Evalúa el rendimiento del modelo utilizando la matriz de confusión, porque es un problema de clasificación.
- Supongamos que tienes un conjunto de datos que contienen información sobre signos, síntomas de un paciente y si tiene o no diabetes. Utilizando el algoritmo de clasificador ingenuo Naïve Bayes, implementa un modelo que pueda clasificar nuevos pacientes con diabetes o sin. Describe uno a uno los pasos a seguir.

SOLUCIÓN

- Preprocesar los datos. Limpiarlos, normalizar los datos para que todas las variables estén en rangos similares.
- Construir un modelo. Calcula las probabilidades condicionales para cada síntoma en

cada uno de los posibles diagnósticos.

- Realiza la clasificación. Utiliza el modelo para clasificar nuevos pacientes.
- Evalúa el rendimiento del modelo utilizando la matriz de confusión, porque es un problema de clasificación.
- Con el conjunto de datos de diabetes de la librería de sklearn implementa el algoritmo en Python. Utiliza las métricas necesarias para evaluar el problema, identificando si la variable target es categórica o numérica.

SOLUCIÓN

```
from sklearn.datasets import load_diabetes

data = load_diabetes()

X = data.data

y = data.target

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.20,
random_state = 0)



classifier = GaussianNB()

classifier.fit(X_train, y_train)

y_pred = classifier.predict(X_test)

# métricas de regresion

from sklearn.metrics import explained_variance_score

from sklearn.metrics import max_error
```

```
from sklearn.metrics import mean_absolute_error

from sklearn.metrics import mean_squared_error



explained_variance_score(y_test, y_pred)
```

8.8. Referencias bibliográficas

Lantz, B. (2013). *Machine Learning with R*. Packt Publishing.

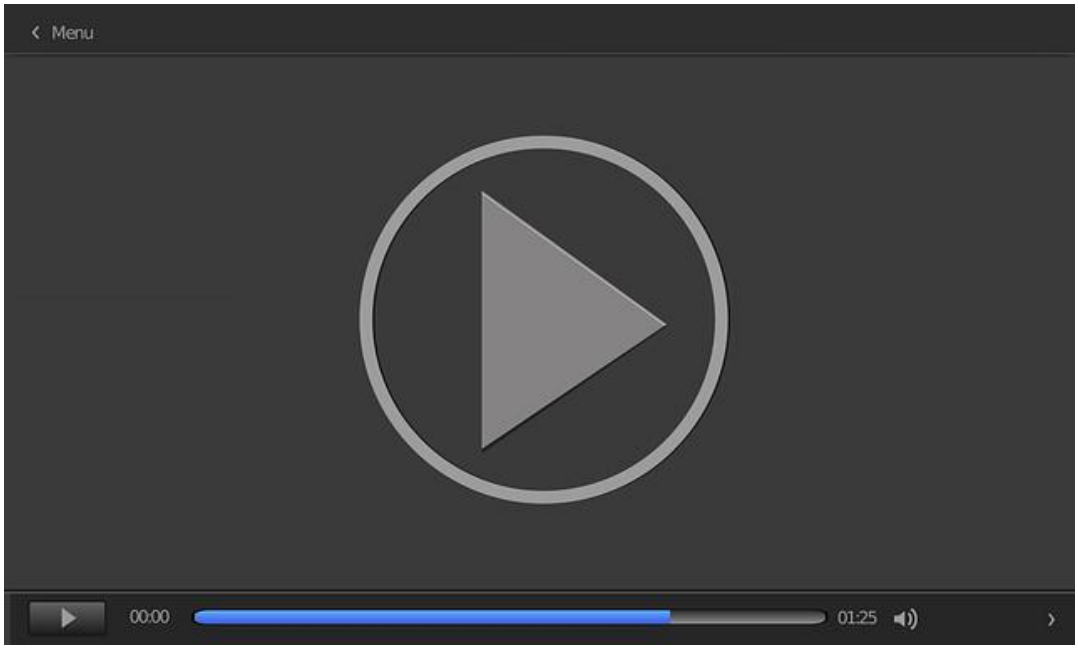
Pérez López, C. y Santin González, D. (2007). *Minería de datos: herramientas y técnicas prácticas de aprendizaje automático*. Ediciones Paraninfo.

scikit-learn.org. (s.f.). *Naïve Bayes*. https://scikit-learn.org/stable/modules/naive_bayes.html

Wikipedia. (2023). *Thomas Bayes*. https://es.wikipedia.org/wiki/Thomas_Bayes

How to Implement Naive Bayes from Scratch with Python

AssemblyAI. (2022, septiembre 17). *How to Implement Naive Bayes from scratch with Python* [Vídeo]. YouTube. https://www.youtube.com/watch?v=TLInuAorxqE



Accede al vídeo:

https://www.youtube.com/embed/TLInuAorxqE

Ejemplo de creación de un modelo de Naïve Bayes desde cero utilizando el lenguaje de programación Python.

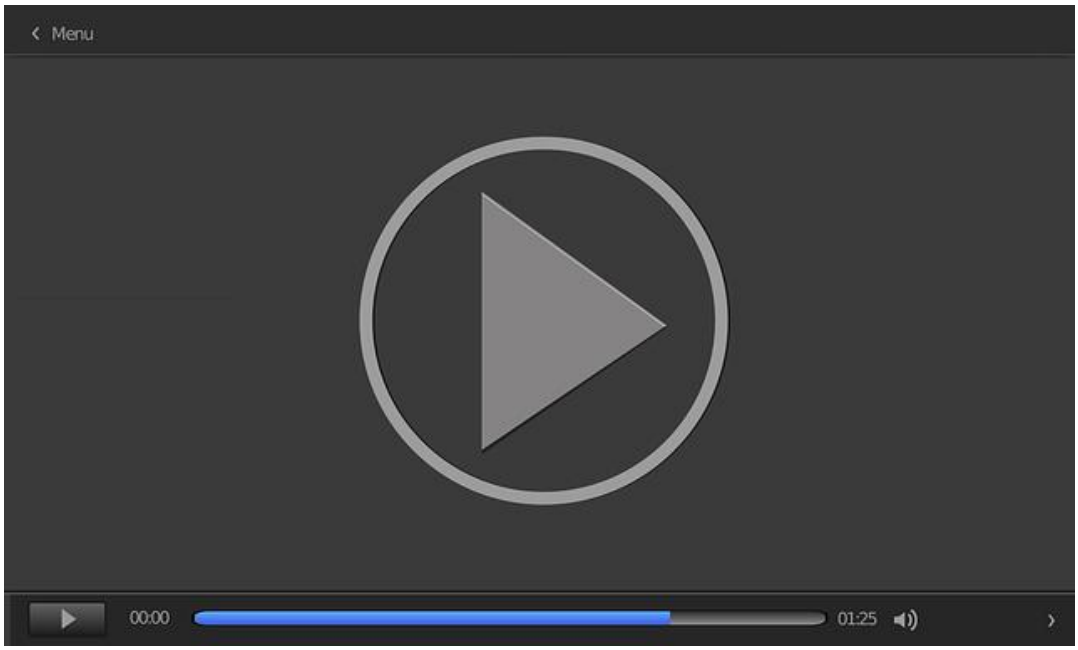
Doing Naïve Bayes Classification: aplicado al libro 50 sombras de Grey

Cherny, L. (2015). *Doing Naive Classification*. Jupyter nbviewer. https://nbviewer.org/github/arnicas/NLP-in-Python/blob/master/4.%20Naive%20Bayes%20Classification.ipynb?imm_mid=0cd660&cmp=em-data-na-na-newsltr_20150225

Ejemplo de un clasificador Naïve Bayes sobre el texto del libro *50 sombras de Grey* con el objetivo de clasificar el contenido de las páginas.

Naïve Bayes – caso aplicado a préstamos P2P

DavidBU, L. (2022, diciembre 11). *60) Naive Bayes-Caso aplicado prestamos P2P* [Vídeo]. YouTube. https://www.youtube.com/watch?v=20RX5gk3wUE



Accede al vídeo:

https://www.youtube.com/embed/20RX5gk3wUE

Un ejemplo práctico de cómo resolver un problema utilizando Naïve Bayes para entrenar un modelo con datos pertenecientes a préstamos, para predecir si un cliente pagará o no su préstamo.

Test

1. ¿Cuál de las siguientes afirmaciones es verdadera sobre el clasificador ingenuo Naïve Bayes?

A. Es un algoritmo de aprendizaje supervisado.

B. Es un algoritmo de aprendizaje no supervisado.

C. Es un algoritmo de aprendizaje por refuerzo.

D. No se utiliza para problemas de clasificación.

2. ¿Qué suposición se hace sobre la independencia de las variables en Naïve Bayes?

A. Las variables son dependientes entre sí.

B. Las variables son independientes entre sí.

C. No se hace ninguna suposición con respecto a las variables.

D. La dependencia o independencia de las variables no es de interés para el algoritmo.

3. Si dos eventos son exhaustivos:

A. Deben ocurrir los dos.

B. Al menos uno de los dos debe ocurrir.

C. Si uno ocurre el otro no puede ocurrir.

D. Ninguna de las anteriores describe la exhaustividad.

4. Dos eventos son mutuamente excluyentes:

A. Deben ocurrir los dos.

B. Al menos uno de los dos debe ocurrir.

C. Si uno ocurre el otro no puede ocurrir.

D. Ninguna de las anteriores describe la exhaustividad.

5. Las tablas de frecuencias:

A. Indican el número de veces que el evento aparece en cada una de las situaciones.

B. Sirven para medir el éxito del modelo.

C. Son las mismas tablas de verosimilitud.

D. Son la base para la construcción del modelo Naïve Bayes.

6. Los eventos dependientes:

A. Permiten estimar la presencia de un evento en función del otro.

B. Siempre ocurren a la vez.

C. Implica que la existencia de uno puede conllevar la existencia del otro.

D. Ninguna de las anteriores describe los eventos dependientes.

7. ¿Cuáles de las siguientes afirmaciones son ciertas sobre el clasificador de Naïve Bayes?

A. Utiliza datos históricos para obtener la probabilidad observada de cada evento en función de su vector de características.

B. Asume independencia condicional entre los eventos.

C. A y B son ciertas.

D. El cálculo riguroso del teorema de Bayes es computacionalmente costoso.

8. Cuando existen combinaciones desconocidas en los datos de entrada:

A. Las probabilidades *a posteriori* obtenidas pueden no tener sentido.

B. El teorema de Bayes utiliza el estimador de Laplace.

C. Se eliminan estas combinaciones de los datos de entrada.

D. El algoritmo es capaz de ignorarlas.

9. La discretización de variables:

A. Es una técnica que se aplica para utilizar el clasificador Naïve Bayes con variables numéricas.

B. Es ideal cuando hay grandes cantidades de datos.

C. Funciona mejor cuando hay pocos datos.

D. A y B son correctas.

10. ¿Cuál de las siguientes afirmaciones describe mejor el proceso de discretización de variables en el análisis de datos?

A. Es un método utilizado para convertir variables numéricas en variables categóricas.

B. Es un proceso utilizado para eliminar datos atípicos en un conjunto de datos.

C. Es una técnica que se aplica para reducir la dimensionalidad de un conjunto de datos.

D. Es un enfoque utilizado para dividir una variable continua en intervalos o categorías discretas.