Actividad 2_02

Modelo Discriminante Gaussiano

Integrantes:

Cesar Eduardo Elías del Hoyo

José Luis Sandoval Pérez

Diego Emanuel Saucedo Ortega

Carlos Daniel Torres Macías

Universidad Autónoma de Aguascalientes

Aguascalientes, Ags, 02 de mayo, 2024

INTRODUCCIÓN

En el vasto campo del aprendizaje automático y la inteligencia artificial, la clasificación de datos es una tarea fundamental que permite a los sistemas automatizados tomar decisiones basadas en la información disponible. Entre las diversas técnicas de clasificación, el Modelo Discriminante Gaussiano (Gaussian Discriminant Analysis, GDA) destaca como un enfoque probado y eficaz para la separación de clases en conjuntos de datos con distribuciones gaussianas.

El Modelo Discriminante Gaussiano es un método supervisado que busca modelar la distribución de probabilidad de las diferentes clases en el conjunto de datos utilizando distribuciones gaussianas multivariadas. A través de este modelo, se puede calcular la probabilidad de que un punto de datos dado pertenezca a cada una de las clases definidas, lo que facilita su asignación a la clase más probable.

En esta investigación, exploraremos en detalle el Modelo Discriminante Gaussiano, su proceso de entrenamiento y su aplicación en la clasificación de datos.

Análisis del modelo

El Modelo Discriminante Gaussiano es una técnica de clasificación supervisada que asume que las características de cada clase siguen una distribución gaussiana (normal) y que estas distribuciones tienen la misma matriz de covarianza. Este modelo se basa en el teorema de Bayes para estimar la probabilidad de que un punto de datos pertenezca a una determinada clase.

El proceso de entrenamiento del Modelo Discriminante Gaussiano implica dos pasos principales: estimación de parámetros y clasificación.

1. **Estimación de parámetros:**
    - **Media y covarianza:** Se calculan las medias y matrices de covarianza de las características para cada clase.
    - **Probabilidad a priori:** Se calcula la probabilidad a priori de cada clase, es decir, la probabilidad de que una observación pertenezca a una clase específica antes de ver los datos.
2. **Clasificación:**
    - Dado un nuevo punto de datos, se calcula la probabilidad de que pertenezca a cada clase utilizando la función de densidad de probabilidad gaussiana multivariada.
    - Utilizando el teorema de Bayes, se calcula la probabilidad posterior de que el punto de datos pertenezca a cada clase.
    - El punto de datos se asigna a la clase con la probabilidad posterior más alta.

Funcionamiento del Modelo:

- **Hipótesis de distribución gaussiana:** El GDA asume que las clases tienen distribuciones gaussianas. Esta suposición puede ser restrictiva en algunos casos, especialmente si los datos no siguen una distribución normal.
- **Decisiones basadas en probabilidades:** El modelo clasifica los puntos de datos en función de las probabilidades de pertenencia a cada clase. Esto permite obtener una medida de confianza en la asignación de clases.

- **Sensibilidad a la matriz de covarianza:** El GDA supone que todas las clases comparten la misma matriz de covarianza. Si esta suposición no es válida, el modelo puede no funcionar bien.
- **Eficiencia computacional:** Debido a su simplicidad y suposiciones, el GDA puede ser computacionalmente eficiente y adecuado para conjuntos de datos de tamaño moderado.

Podemos decir que el Modelo Discriminante Gaussiano es una técnica clásica pero efectiva para la clasificación de datos que se basa en suposiciones bien definidas sobre la distribución de las características de las clases. Si estas suposiciones son válidas para el conjunto de datos en cuestión, el GDA puede proporcionar resultados precisos y eficientes en términos computacionales.

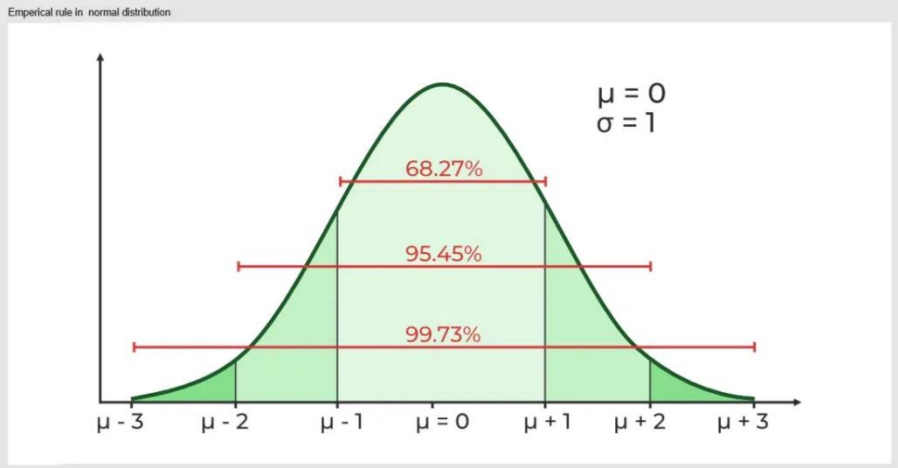
Implementación del modelo.

El análisis discriminante gaussiano a diferencia de una regresión lineal utiliza la distribución de los datos, de forma que aquello que se estima que un dato es más probable que pertenezca a una clase que otra.
Una distribución normal, la cual es utilizada para "agrupar" los datos requieren de una función distributiva normal que permita formar la campana de Gauss característica de la distribución.

$$f(x, \mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

"Se muestra la función de distribución Gaussiana"

Esta función permite que los datos se encuentren en una curva que indique la tendencia de los datos.



"Se muestra ejemplo de distribución normal (Campana de Gauss)"

Esta distribución se obtiene a partir de dos elementos del grupo de datos: la media y la varianza. Estos dos permiten conocer que tan dispersos están los datos; datos muy dispersos implican que sea más complicado encontrar una tendencia y, por ende, predecir para nuevas entradas.

La media se representa así:

$$\bar{x} = \frac{\sum \chi}{n}$$

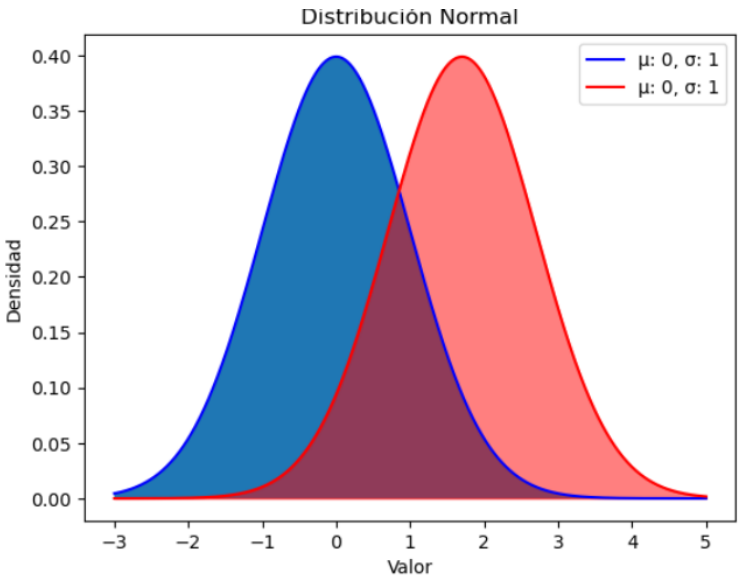
Se refiere a la sumatoria de todos los datos dividido entre el número de datos.

La varianza se determina de la siguiente manera:

$$Varianza = \sum_{i=1}^{n} \frac{(x_i - \bar{x})^2}{n}$$

En resumen, es la diferencia al cuadrado de cada dato y media de los datos, la suma total de diferencias dividido por la cantidad de datos.

Ahora, por cada característica de nuestro dataset, podremos obtener una distribución normal y su respectiva curva. El análisis exploratorio gaussiano evalúa para un dato a predecir qué tan probable es que pertenezca a un grupo de datos o a otro. Por ejemplo, en la siguiente imagen:

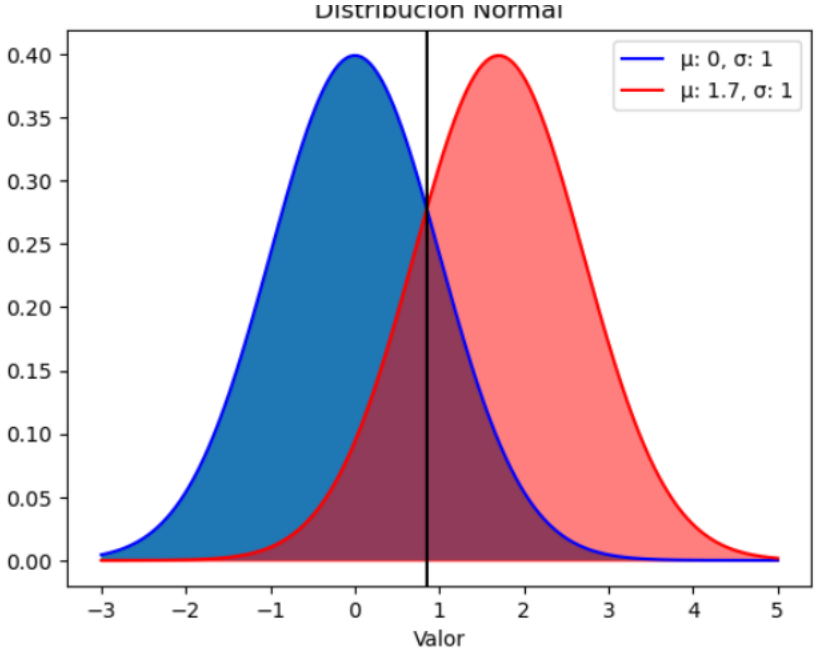


De dos clases distintas podemos observar una región azul y una roja, respectivamente. Si un dato se encuentra en estas regiones será evidente que la predicción tendrá un resultado, pero también es visible que dichas regiones tienen un gran espacio que comparte, es decir que comparten la probabilidad de que un dato en dicha región pertenezca a una clase u otra.

Para ello es necesario delimitar una recta de diferenciación, lo más sencillo es utilizar el punto donde cruzan ambas regiones como el límite entre cada clase. ¿Como determinarlo? Encontramos el punto medio entre las dos cúspides de la distribución que no es otro que la media. Tenemos que:

$$RD = \frac{\bar{x}_1 + \bar{x}_2}{2}$$

Siendo el resultado el siguiente:



Así, podemos discernir el resultado de los datos y poder dar como resultado la probabilidad de que pertenezca a esa clase.

Así el análisis discriminante gaussiano avanza con cada clase ya para cada característica. Para mejores resultados y evitar análisis innecesarios también es importante conocer la importancia en la decisión de cada una de las características, pues hay características que permiten discernir los datos con mayor facilidad.

Evaluación de modelo

```
8]: Y = df["BuenPagador"]
    X = df.drop(columns=["BuenPagador"])

    # separamos en muestra y entrenamiento
    X_train,X_test,Y_train,Y_test = train_test_split(X,Y,test_size=0.2)

    print("tamano x entrenamiento:",X_train.shape)
    print("tamano x prueba:",X_test.shape)
    print("tamano y entrenamiento:",Y_train.shape)
    print("tamano x prueba:",Y_test.shape)
```

```
tamano x entrenamiento: (2000, 5)
tamano x prueba: (500, 5)
tamano y entrenamiento: (2000,)
tamano x prueba: (500,)
```

Como se mencionó en la implementación tenemos la clase objetivo la decisión de si es o no un “buen pagador”, siendo esta 1 y 0. Separamos la muestra en datos de entrenamiento y de prueba.

A continuacion la implementacion del GaussianNB

```
]: modelo = LinearDiscriminantAnalysis(store_covariance=True)
   modelo.fit(X_train,Y_train)
```

```
]: LinearDiscriminantAnalysis
LinearDiscriminantAnalysis(store_covariance=True)
```

```
]: modelo.score(X_test,Y_test)
```

```
]: 0.846
```

El modelo permite una prediccion de clase y tambien una probabilidad

```
]: x_prueba = [[3,1,1,2,1]]
   print("Prediccion: ",modelo.predict(x_prueba))
   print("Probabilidad: ",modelo.predict_proba(x_prueba))
```

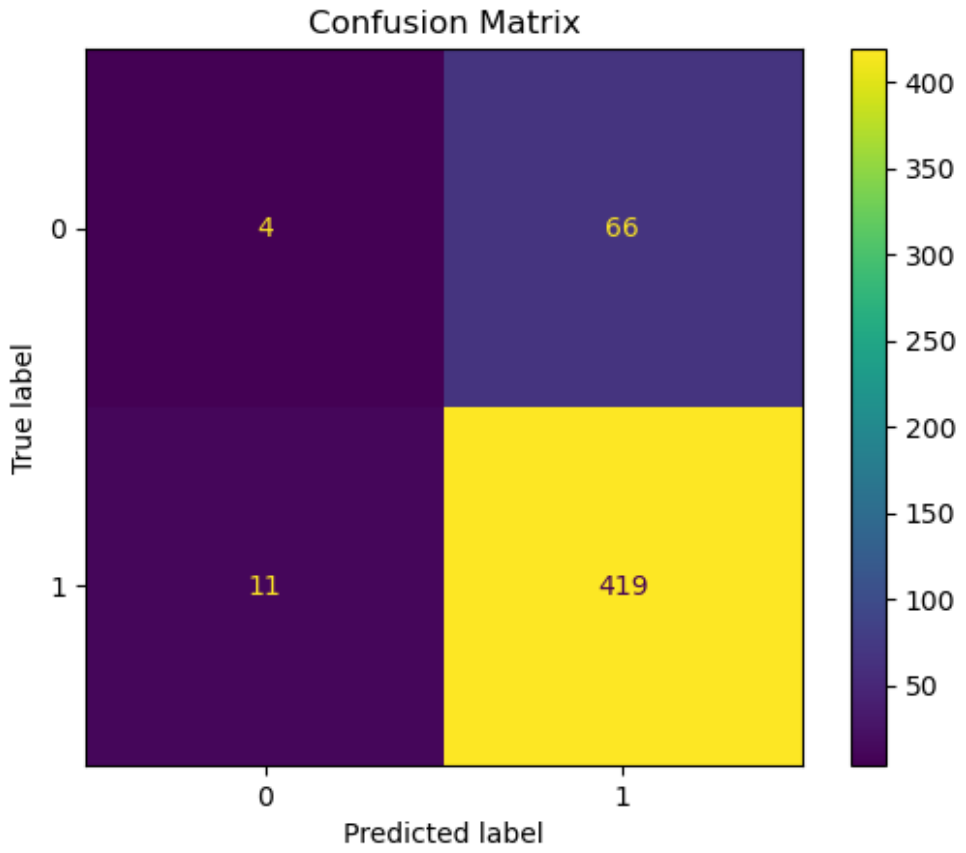
```
Prediccion:  ['No']
Probabilidad:  [[0.63917544 0.36082456]]
```

Una vez entrenado el modelo aplicamos el discriminante Gaussiano, obteniendo un score de .846 es decir 84.6%, este score nos dice que el discriminante Gaussiano es bastante eficiente para la clasificación y predicción.

De igual manera se obtuvo una probabilidad de la predicción, dándonos como resultados un 69.91% y 36.08% de predicción de acierto y de no acierto respectivamente.

Matriz de confusion

```
5]: from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
cm = confusion_matrix(Y_test,y_hat)
ConfusionMatrixDisplay(cm).plot()
plt.title("Confusion Matrix")
plt.show()
```



Ahora sacamos la matriz de confusión, la cual nos indica las predicciones exitosas y las no exitosas, siendo estas las verdaderas positivas [1][0] y verdaderas negativas [0][0], falsas positivas [1][0] y falsas negativas [1][1]

Podemos destacar 4 parámetros de evaluación:

- **Precisión**: La precisión mide la proporción de predicciones positivas correctas (verdaderos positivos) sobre todas las predicciones positivas realizadas por el modelo. Es una medida de la exactitud del modelo en la clasificación de muestras positivas.
- **Sensibilidad (Recall)**: La sensibilidad, también conocida como tasa de verdaderos positivos, mide la proporción de verdaderos positivos detectados por el modelo sobre todas las muestras positivas reales. Es una medida de la capacidad del modelo para detectar correctamente las muestras positivas.
- **Especificidad**: La especificidad mide la proporción de verdaderos negativos detectados por el modelo sobre todas las muestras negativas reales. Es una medida de la capacidad del modelo para detectar correctamente las muestras negativas.
- **Tasa de Error**: La tasa de error representa la proporción total de predicciones incorrectas realizadas por el modelo, el error en un problema de clasificación binaria se puede calcular como la proporción de predicciones incorrectas con respecto al total de predicciones. En otras palabras, es la suma de los falsos positivos y los falsos negativos dividida por el total de muestras.

Esto nos da los siguientes resultados respectivamente:

1. Precisión=TP/ TP+FP
2. Sensibilidad=TP/TP+FN
3. Especificidad=TN/TN+FP
4. Tasa de Error=FP+FN/TP+TN+FP+FN

Donde:

- TP: Verdaderos Positivos
- TN: Verdaderos Negativos

- FP: Falsos Positivos
- FN: Falsos Negativos

Utilizando los valores

- TP = 419
- TN = 4
- FP = 11
- FN = 66

Podemos calcular las métricas:

1. Precisión≈97.47%
2. Sensibilidad≈86.34% E
3. Especificidad≈26.67%
4. Tasa de Error=13.66%

Conclusión

La implementación de un modelo de distribución gaussiana proporciona una herramienta útil para modelar la distribución de los datos y hacer predicciones sobre nuevas muestras. Al utilizar la distribución gaussiana, podemos obtener parámetros que describen la media y la varianza de los datos para cada clase. Estos parámetros son valiosos para comprender la distribución subyacente de los datos y pueden ser utilizados para realizar inferencias y predicciones. Es importante evaluar el rendimiento del modelo utilizando métricas apropiadas y compararlo con otros enfoques para determinar su eficacia en la tarea específica. Además, es fundamental verificar los supuestos de la distribución gaussiana y realizar cualquier preprocesamiento necesario para garantizar resultados confiables y útiles.

BIBLIOGRAFIA:

GeeksforGeeks. (2024, 19 abril). *Python Normal Distribution in Statistics*. GeeksforGeeks. https://www.geeksforgeeks.org/python-normal-distribution-in-statistics/

sklearn.discriminant_analysis.LinearDiscriminantAnalysis. (s. f.). Scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html

Arnav, A. (2023, 6 mayo). Gaussian Naïve Bayes Classifier with Math behind it and the code. *Medium*. https://medium.com/@anupam.arnav/gaussian-na%C3%AFve-bayes-classifier-with-math-behind-it-and-the-code-fe51a338a854

Brownlee, J. (2019, 24 octubre). *Naive Bayes Classifier From Scratch in Python*. MachineLearningMastery.com. https://machinelearningmastery.com/naive-bayes-classifier-scratch-python/