

Mapa Mental: Métodos de Clasificación con Machine Learning



Universidad de Cundinamarca
Facultad de ingeniería
Machine Learning
Gestor: Edison Gustavo Cañón Varela
Francisco Vallejos Jaramillo
Joan moreno

Análisis de los modelos de clasificación:

Regresión Logística

¿Qué es la regresión logística?

La regresión logística es un modelo estadístico que se utiliza para predecir la probabilidad de que un evento ocurra en función de una o más variables independientes. A diferencia de la regresión lineal, que predice valores continuos, la regresión logística se utiliza para problemas de clasificación binaria, donde el resultado es categórico (por ejemplo, sí/no, verdadero/falso).

La regresión logística utiliza la función logística (o sigmoide) para transformar la salida de una combinación lineal de las variables independientes en un valor entre 0 y 1, lo que permite interpretar el resultado como una probabilidad. La fórmula general de la regresión logística es:

$$p(y=1|X) = 1 / (1 + e^{(-z)})$$

donde $p(y=1|X)$ es la probabilidad de que la variable dependiente y sea igual a 1 dado el vector de variables independientes X , e es la base del logaritmo natural, y z es una combinación lineal de las variables independientes.

La regresión logística se utiliza en una amplia variedad de aplicaciones, como la predicción de enfermedades, la evaluación de riesgos crediticios, el análisis de marketing y la clasificación de texto. Es un modelo relativamente simple y fácil de interpretar, lo que lo convierte en una opción popular para problemas de clasificación.

En resumen, la regresión logística es una técnica de modelado estadístico que permite predecir la probabilidad de un evento binario en función de variables independientes. Su capacidad para manejar problemas de clasificación la convierte en una herramienta valiosa en el análisis de datos y la toma de decisiones.

¿Por qué es importante la regresión logística?

La regresión logística es importante por varias razones:

1. ****Predicción de Probabilidades****: Permite predecir la probabilidad de que ocurra un evento, lo que es útil en muchas aplicaciones prácticas.
2. ****Interpretabilidad****: Los coeficientes del modelo son fáciles de interpretar, lo que ayuda a entender la relación entre las variables independientes y la variable dependiente.

3. Versatilidad: Se puede aplicar en una amplia gama de campos, como medicina, finanzas, marketing y ciencias sociales.
4. Manejo de Datos Categóricos: Puede manejar variables categóricas mediante la creación de variables ficticias (dummy variables).
5. Base para Modelos Más Complejos: La regresión logística es la base para modelos más complejos, como la regresión logística multinomial y la regresión de Poisson.
6. Evaluación de Riesgos: Se utiliza en la evaluación de riesgos, como en la predicción de enfermedades o el análisis de crédito.
7. Facilidad de Implementación: Es fácil de implementar y computacionalmente eficiente, lo que la hace adecuada para conjuntos de datos grandes.
8. Validación Cruzada: Permite la validación cruzada y la evaluación del rendimiento del modelo mediante métricas como la precisión, la sensibilidad y la especificidad.
9. Uso en Machine Learning: Es un modelo fundamental en el campo del aprendizaje automático y se utiliza como punto de partida para muchos algoritmos más avanzados.

¿Cuáles son las aplicaciones de la regresión logística?

La regresión logística tiene una amplia gama de aplicaciones en diversos campos. Algunas de las aplicaciones más comunes incluyen:

1. Medicina: Predicción de la probabilidad de enfermedades o condiciones médicas en función de factores de riesgo.
2. Finanzas: Evaluación del riesgo crediticio y predicción de la probabilidad de incumplimiento de pagos.
3. Marketing: Análisis de la efectividad de campañas publicitarias y predicción de la probabilidad de compra.
4. Ciencias Sociales: Estudio de la relación entre variables sociales y la probabilidad de ciertos comportamientos.
5. Análisis de Sentimiento: Clasificación de opiniones o sentimientos en texto (positivo/negativo).
6. Clasificación de Clientes: Segmentación de clientes en función de su probabilidad de respuesta a ofertas.

7. Detección de Fraude: Identificación de transacciones fraudulentas en sistemas financieros.

8. Análisis de Riesgos: Evaluación de riesgos en seguros y finanzas.

9. Predicción de Eventos: Predicción de eventos binarios, como la probabilidad de que un cliente abandone un servicio.

10. Investigación de Mercados: Análisis de encuestas y estudios de mercado para comprender el comportamiento del consumidor.

11. Análisis de Datos Genómicos: Predicción de la probabilidad de enfermedades genéticas en función de datos genómicos.

12. Análisis de Redes Sociales: Clasificación de usuarios o publicaciones en redes sociales.

Ejemplo de Regresión Logística

En este ejemplo simulamos una regresión logística con puntos clasificados y una curva sigmoide ajustada.

024681000.20.40.60.81

DatosCurva SigmoideGráfica de Regresión LogísticaXProbabilidad / Clase

referencias:



Regresión Logística

¿Qué es la regresión logística?

La regresión logística es un modelo estadístico que se utiliza para predecir la probabilidad de que un evento ocurra en función de una o más variables independientes. A diferencia de la regresión lineal, que predice valores continuos, la regresión logística se utiliza para problemas de clasificación binaria, donde el resultado es categórico (por ejemplo, sí/no, verdadero/falso).

La regresión logística utiliza la función logística (o sigmoide) para transformar la salida de una combinación lineal de las variables independientes en un valor entre 0 y 1, lo que permite interpretar el resultado como una probabilidad. La fórmula general de la regresión logística es:

$$p(y=1|X) = 1 / (1 + e^{-(z)})$$

donde $p(y=1|X)$ es la probabilidad de que la variable dependiente y sea igual a 1 dado el vector de variables independientes X, e es la base del logaritmo natural, y z es una combinación lineal de las variables independientes.

La regresión logística se utiliza en una amplia variedad de aplicaciones, como la predicción de enfermedades, la evaluación de riesgos crediticios, el análisis de marketing y la clasificación de texto. Es un modelo relativamente simple y fácil de interpretar, lo que lo convierte en una opción popular para problemas de clasificación.

En resumen, la regresión logística es una técnica de modelado estadístico que permite predecir la probabilidad de un evento binario en función de variables independientes. Su capacidad para manejar problemas de clasificación la convierte en una herramienta valiosa en el análisis de datos y la toma de decisiones.

¿Por qué es importante la regresión logística?

La regresión logística es importante por varias razones:

1. **Predicción de Probabilidades**: Permite predecir la probabilidad de que ocurra un evento, lo que es útil en muchas aplicaciones prácticas.
2. **Interpretabilidad**: Los coeficientes del modelo son fáciles de interpretar, lo que ayuda a entender la relación entre las variables independientes y la variable dependiente.
3. **Versatilidad**: Se puede aplicar en una amplia gama de campos, como medicina, finanzas, marketing y ciencias sociales.
4. **Manejo de Datos Categóricos**: Puede manejar variables categóricas mediante la creación de variables ficticias (dummy variables).
5. **Base para Modelos Más Complejos**: La regresión logística es la base para modelos más complejos, como la regresión logística multinomial y la regresión de Poisson.

127.0.0.1:5500/Templates/modelos_1.0.html

¿Cuáles son las aplicaciones de la regresión logística?

La regresión logística tiene una amplia gama de aplicaciones en diversos campos. Algunas de las aplicaciones más comunes incluyen:

1. **Medicina**: Predicción de la probabilidad de enfermedades o condiciones médicas en función de factores de riesgo.
2. **Finanzas**: Evaluación del riesgo crediticio y predicción de la probabilidad de incumplimiento de pagos.
3. **Marketing**: Análisis de la efectividad de campañas publicitarias y predicción de la probabilidad de compra.
4. **Ciencias Sociales**: Estudio de la relación entre variables sociales y la probabilidad de ciertos comportamientos.
5. **Análisis de Sentimiento**: Clasificación de opiniones o sentimientos en texto (positivo/negativo).
6. **Clasificación de Clientes**: Segmentación de clientes en función de su probabilidad de respuesta a ofertas.
7. **Detección de Fraude**: Identificación de transacciones fraudulentas en sistemas financieros.
8. **Análisis de Riesgos**: Evaluación de riesgos en seguros y finanzas.
9. **Predicción de Eventos**: Predicción de eventos binarios, como la probabilidad de que un cliente abandone un servicio.
11. **Investigación de Mercados**: Análisis de encuestas y estudios de mercado para comprender el comportamiento del consumidor.
11. **Análisis de Datos Genómicos**: Predicción de la probabilidad de enfermedades genéticas en función de datos genómicos.
12. **Análisis de Redes Sociales**: Clasificación de usuarios o publicaciones en redes sociales.

¿Cómo funciona el modelo de regresión logística?

El modelo de regresión logística funciona de la siguiente manera:

1. **Recopilación de Datos**: Se recopilan datos que contienen una variable dependiente binaria (la variable que se desea predecir) y una o más variables independientes (factores que influyen en la variable dependiente).
2. **Transformación de Datos**: Las variables independientes pueden ser continuas o categóricas. Si son categóricas, se convierten en variables ficticias (dummy variables) para su inclusión en el modelo.
3. **Modelo Logístico**: Se utiliza la función logística para modelar la relación entre las variables independientes y la variable dependiente. La función logística transforma la combinación lineal de las variables independientes en una probabilidad entre 0 y 1.

¿Cómo funciona el modelo de regresión logística?

El modelo de regresión logística funciona de la siguiente manera:

1. **Recopilación de Datos**: Se recopilan datos que contienen una variable dependiente binaria (la variable que se desea predecir) y una o más variables independientes (factores que influyen en la variable dependiente).
2. **Transformación de Datos**: Las variables independientes pueden ser continuas o categóricas. Si son categóricas, se convierten en variables ficticias (dummy variables) para su inclusión en el modelo.
3. **Modelo Logístico**: Se utiliza la función logística para modelar la relación entre las variables independientes y la variable dependiente. La función logística transforma la combinación lineal de las variables independientes en una probabilidad entre 0 y 1.

Ejemplo de Regresión Logística

En este ejemplo simulamos una regresión logística con puntos clasificados y una curva sigmoide ajustada.



K-Nearest Neighbors (KNN)

¿Qué es K-Nearest Neighbors (KNN)?

K-Nearest Neighbors (KNN) es un algoritmo de aprendizaje supervisado utilizado para clasificación y regresión. Funciona identificando los 'k' vecinos más cercanos a un punto de datos y tomando decisiones basadas en la mayoría de votos o el promedio de esos vecinos.

El algoritmo KNN se basa en la idea de que los puntos de datos similares tienden a estar cerca unos de otros en el espacio de características. Para clasificar un nuevo punto de datos, el algoritmo calcula la distancia entre ese punto y todos los puntos

de entrenamiento, selecciona los 'k' vecinos más cercanos y asigna la clase más común o el valor promedio entre ellos.

El valor de 'k' es un parámetro importante que afecta el rendimiento del modelo. Un valor pequeño de 'k' puede hacer que el modelo sea sensible al ruido, mientras que un valor grande puede suavizar demasiado la decisión.

KNN es fácil de entender e implementar, pero puede ser computacionalmente costoso, especialmente con grandes conjuntos de datos. Se utiliza en diversas aplicaciones, como reconocimiento de patrones, clasificación de imágenes y análisis de datos.

¿Por qué es importante KNN?

K-Nearest Neighbors (KNN) es importante por varias razones:

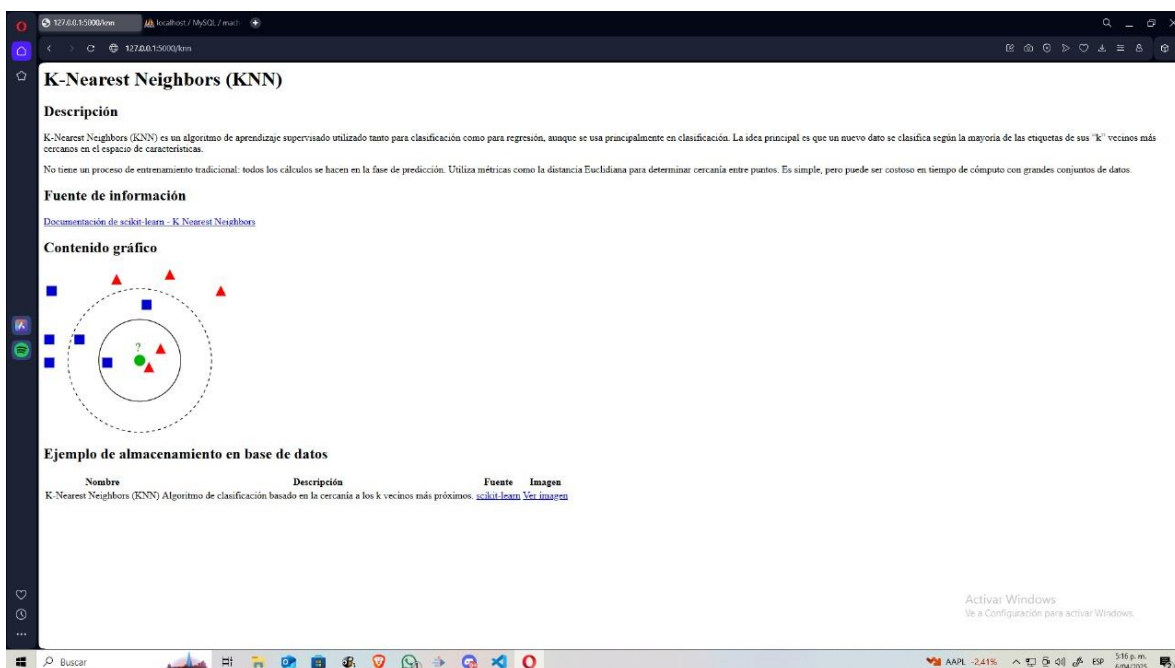
1. ****Simplicidad****: KNN es fácil de entender e implementar, lo que lo convierte en un buen punto de partida para problemas de clasificación y regresión.
2. ****No Paramétrico****: No asume ninguna distribución específica de los datos, lo que lo hace flexible y aplicable a una variedad de problemas.
3. ****Versatilidad****: Se puede utilizar tanto para clasificación como para regresión, lo que lo hace útil en diferentes contextos.
4. ****Adaptabilidad****: Puede adaptarse a diferentes tipos de datos y características sin necesidad de transformaciones complejas.
5. ****Efectividad en Datos Pequeños****: Funciona bien con conjuntos de datos pequeños y medianos, donde la distancia entre puntos es significativa.
6. ****Uso en Aprendizaje No Supervisado****: Puede ser utilizado como un método de agrupamiento no supervisado, ayudando a identificar patrones en los datos.
7. ****Base para Algoritmos Más Complejos****: KNN es la base para algoritmos más avanzados y técnicas de aprendizaje automático.

¿Cuáles son las aplicaciones de KNN?

K-Nearest Neighbors (KNN) tiene diversas aplicaciones en diferentes campos. Algunas de las aplicaciones más comunes incluyen:

1. ****Clasificación de Imágenes****: Identificación y clasificación de objetos en imágenes basándose en sus características.

2. **Reconocimiento Facial**: Identificación de rostros en imágenes o videos utilizando características faciales.
3. **Detección de Fraude**: Identificación de transacciones fraudulentas en sistemas financieros mediante la comparación con transacciones legítimas.
4. **Recomendación de Productos**: Sugerencia de productos a usuarios basándose en preferencias similares a las de otros usuarios.
5. **Análisis de Sentimiento**: Clasificación de opiniones o sentimientos en texto (positivo/negativo).
6. **Segmentación de Clientes**: Agrupación de clientes en función de características similares para personalización de marketing.
7. **Clasificación de Texto**: Asignación de categorías a documentos o correos electrónicos basándose en su contenido.



Árboles de Decisión

¿Qué son los Árboles de Decisión?

Los árboles de decisión son un algoritmo de aprendizaje supervisado utilizado para clasificación y regresión. Representan decisiones y sus posibles consecuencias en forma de un árbol, donde cada nodo interno representa una prueba en una característica, cada rama representa el resultado de la prueba y cada hoja representa una clase o valor de salida.

El proceso de construcción de un árbol de decisión implica dividir el conjunto de datos en subconjuntos más pequeños basándose en las características que mejor separan las clases. Esto se hace utilizando medidas como la ganancia de información o la impureza de Gini.

Los árboles de decisión son fáciles de interpretar y visualizar, lo que los hace populares en aplicaciones donde la interpretabilidad es importante. Sin embargo, pueden ser propensos al sobreajuste si no se controlan adecuadamente.

Los árboles de decisión se utilizan en una variedad de aplicaciones, como análisis de riesgos, diagnóstico médico, clasificación de clientes y análisis de datos. También son la base para algoritmos más complejos, como Random Forest y Gradient Boosting.

¿Por qué son importantes los Árboles de Decisión?

Los árboles de decisión son importantes por varias razones:

1. **Interpretabilidad**: Son fáciles de entender y visualizar, lo que permite a los usuarios interpretar fácilmente las decisiones del modelo.
2. **No Paramétricos**: No requieren suposiciones sobre la distribución de los datos, lo que los hace flexibles y aplicables a una variedad de problemas.

¿Cuáles son las aplicaciones de los Árboles de Decisión?

Los árboles de decisión tienen diversas aplicaciones en diferentes campos. Algunas de las aplicaciones más

← → ↻ 127.0.0.1:5500/MLFLASK%202.2/Flask-1/Templates/ArbolDecision.html 🔍 ☆ 🔄 📄

📱 WhatsApp 📰 Detail | News | Millo... 🌀 Naruto Shippuden... 📖 matematicas 📄 Download Freeletics... 🗨️ DISCORD 📱 App Inventor Tutor... 🎮 Dragon Ball Super 7... >> 📁 Todos

Árboles de Decisión

Descripción

Los árboles de decisión son modelos de clasificación supervisada que toman decisiones dividiendo los datos en subconjuntos basados en características. El modelo construye un árbol donde cada nodo representa una característica, cada rama una condición, y cada hoja una clase. Son fáciles de interpretar y visualizar, y pueden manejar tanto variables categóricas como numéricas. Sin embargo, son propensos al sobreajuste si no se podan adecuadamente.

Fuente de información

[Documentación de scikit-learn - Decision Trees](#)

Contenido gráfico

Ejemplo de almacenamiento en base de datos

Nombre	Descripción	Fuente	Imagen
Árboles de Decisión	Modelo de clasificación que divide los datos en ramas según decisiones basadas en características.	scikit-learn	Ver imagen

Support Vector Machine (SVM)

¿Qué es Support Vector Machine (SVM)?

Support Vector Machine (SVM) es un algoritmo de aprendizaje supervisado utilizado para clasificación y regresión. Funciona encontrando el hiperplano que mejor separa las clases en un espacio de características de alta dimensión.

El objetivo de SVM es maximizar el margen entre las clases, lo que significa encontrar el hiperplano que minimiza la distancia entre los puntos de datos más cercanos de cada clase (vectores de soporte). SVM puede manejar datos linealmente separables y no separables mediante el uso de núcleos (kernels) para transformar los datos a un espacio de características más alto.

SVM es eficaz en problemas de clasificación complejos y se utiliza en diversas aplicaciones, como reconocimiento de patrones, clasificación de texto y análisis de imágenes. Su capacidad para manejar grandes conjuntos de datos y su robustez frente al sobreajuste la convierten en una opción popular en el campo del aprendizaje automático.

¿Por qué es importante SVM?

Support Vector Machine (SVM) es importante por varias razones:

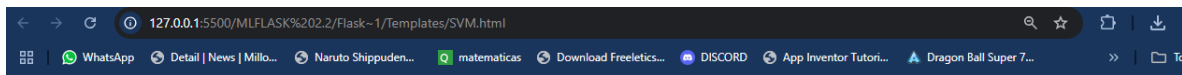
1. ****Eficiencia****: SVM es eficiente en términos computacionales, especialmente con grandes conjuntos de datos.
2. ****Manejo de Datos No Lineales****: Puede manejar datos no lineales mediante el uso de núcleos (kernels).
3. ****Robustez****: Es robusto frente al sobreajuste, lo que lo hace adecuado para problemas complejos.
4. ****Versatilidad****: Se puede utilizar tanto para clasificación como para regresión.

¿Cuáles son las aplicaciones de SVM?

Support Vector Machine (SVM) tiene diversas aplicaciones en diferentes campos. Algunas de las aplicaciones más comunes incluyen:

1. ****Clasificación de Imágenes****: Identificación y clasificación de objetos en imágenes basándose en sus características.

****Reconocimiento Facial****: Identificación de rostros en imágenes o videos utilizando características faciales.



Support Vector Machine (SVM)

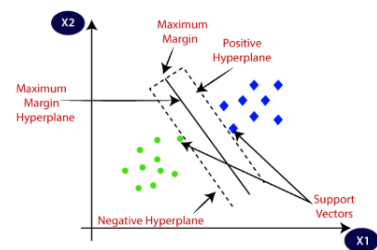
Descripción

Support Vector Machine (SVM) es un algoritmo supervisado que se utiliza para tareas de clasificación y regresión. Su objetivo principal es encontrar el hiperplano óptimo que mejor separa las clases en el espacio de características, maximizando el margen entre los datos de diferentes clases. SVM puede funcionar tanto con datos linealmente separables como no lineales, gracias al uso de funciones kernel como el radial basis function (RBF). Es efectivo en espacios de alta dimensión y en casos donde la cantidad de muestras es menor que el número de características.

Fuente de información

[Documentación de scikit-learn - Support Vector Machines](#)

Contenido gráfico



Ejemplo de almacenamiento en base de datos

Nombre	Descripción	Fuente	Imagen
Support Vector Machine (SVM)	Algoritmo que encuentra el hiperplano que mejor separa las clases, maximizando el margen.	scikit-learn	Ver imagen

Gradient Boosting (XGBoost, AdaBoost, etc.)

¿Qué es Gradient Boosting?

Gradient Boosting es un algoritmo de aprendizaje supervisado utilizado para clasificación y regresión. Se basa en la idea de construir un modelo fuerte combinando múltiples modelos débiles (generalmente árboles de decisión) de manera secuencial.

El proceso de Gradient Boosting implica entrenar un modelo débil, calcular el error y luego ajustar el siguiente modelo débil para corregir los errores del anterior. Este proceso se repite hasta que se alcanza un número predefinido de modelos o se cumple una condición de parada.

Gradient Boosting es conocido por su capacidad para manejar datos complejos y su eficacia en competencias de ciencia de datos. Existen variantes populares como XGBoost y AdaBoost, que ofrecen mejoras en términos de velocidad y rendimiento.

Gradient Boosting se utiliza en una variedad de aplicaciones, como análisis de riesgos, clasificación de clientes y predicción de ventas. Su capacidad para manejar grandes conjuntos de datos y su robustez frente al sobreajuste lo convierten en una opción popular en el campo del aprendizaje automático.

¿Porque es importante

Gradient Boosting es importante por varias razones:

1. **Eficiencia**: Gradient Boosting es eficiente en términos computacionales y puede manejar grandes conjuntos de datos.
2. **Manejo de Datos Complejos**: Puede manejar datos complejos y no lineales, lo que lo hace adecuado para problemas difíciles.
3. **Robustez**: Es robusto frente al sobreajuste, lo que lo hace adecuado para problemas complejos.
4. **Versatilidad**: Se puede utilizar tanto para clasificación como para regresión.

¿Cuales son las aplicaciones

Gradient Boosting tiene diversas aplicaciones en diferentes campos. Algunas de las aplicaciones más comunes incluyen:

1. **Clasificación de Imágenes**: Identificación y clasificación de objetos en imágenes basándose en sus características.

Resumen

En resumen, Gradient Boosting es un algoritmo de aprendizaje supervisado utilizado para clasificación y regresión. Se basa en la idea de construir un modelo fuerte combinando múltiples modelos débiles de manera secuencial. Su capacidad para manejar datos complejos y su robustez frente al sobreajuste lo convierten en una opción popular en el campo del aprendizaje automático.

127.0.0.1:5500/MLFLASK%202.2/Flask-1/Templates/XGBoost.html

WhatsAppDetail | News | Milla...Naruto Shippuden...matematicasDownload Freeletics...DISCORDApp Inventor Tutori...Dragon Ball Super 7...

Gradient Boosting (XGBoost, AdaBoost, etc.)

Descripción

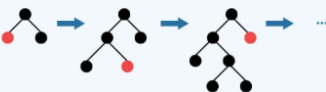
Gradient Boosting es una técnica de ensamblado que construye modelos predictivos de manera secuencial, donde cada nuevo modelo corrige los errores del anterior. Funciona sumando modelos débiles, comúnmente árboles de decisión, para formar un modelo fuerte. Entre sus variantes más conocidas están **AdaBoost**, que ajusta los pesos de las observaciones en función de los errores, y **XGBoost**, una implementación optimizada y paralelizable que ha ganado popularidad por su rendimiento en competencias de ciencia de datos.

Fuente de información

[documentación de scikit-learn - Gradient Boosting](#)
[documentación oficial de XGBoost](#)

Contenido gráfico

Light GBM



El diagrama ilustra el proceso de Gradient Boosting en Light GBM. Muestra una serie de árboles de decisión simples (modelos débiles) que se van sumando secuencialmente. Cada árbol corrige los errores del modelo anterior. La secuencia comienza con un árbol simple, luego se añade otro, y así sucesivamente, hasta que se forma un modelo complejo y fuerte. Las flechas azules indican el flujo de la construcción secuencial.

Ejemplo de almacenamiento en base de datos

Nombre	Descripción	Fuente	Imagen
Gradient Boosting	Técnica de ensamblado que combina modelos débiles de forma secuencial para mejorar la predicción.	scikit-learn , XGBoost Docs	Ver imagen

Naive Bayes

¿Qué es Naive Bayes?

Naive Bayes es un algoritmo de aprendizaje supervisado utilizado para clasificación. Se basa en el teorema de Bayes y asume que las características son independientes entre sí, lo que simplifica el cálculo de probabilidades.

El algoritmo Naive Bayes calcula la probabilidad de que un punto de datos pertenezca a una clase dada y selecciona la clase con la probabilidad más alta como la predicción final. A pesar de su suposición de independencia, Naive Bayes ha demostrado ser efectivo en una variedad de aplicaciones, especialmente en clasificación de texto y análisis de sentimientos.

Naive Bayes es rápido y eficiente, lo que lo hace adecuado para conjuntos de datos grandes. Sin embargo, su suposición de independencia puede no ser válida en todos los casos, lo que puede afectar su rendimiento en algunos problemas.

Naive Bayes se utiliza en una variedad de aplicaciones, como filtrado de spam, clasificación de documentos y análisis de sentimientos. Su capacidad para manejar grandes volúmenes de datos y su simplicidad lo convierten en una opción popular en el campo del aprendizaje automático.

¿Por qué es importante Naive Bayes?

Naive Bayes es importante por varias razones:

1. ****Simplicidad****: Es fácil de entender e implementar, lo que lo convierte en un buen punto de partida para problemas de clasificación.
2. ****Eficiencia****: Naive Bayes es rápido y eficiente, lo que lo hace adecuado para conjuntos de datos grandes.
3. ****Manejo de Datos Categóricos****: Puede manejar variables categóricas y continuas sin necesidad de transformaciones complejas.
4. ****Base para Algoritmos Más Complejos****: Naive Bayes es la base para algoritmos más avanzados y técnicas de aprendizaje automático.

¿Cuáles son las aplicaciones de Naive Bayes?

Naive Bayes tiene diversas aplicaciones en diferentes campos. Algunas de las aplicaciones más comunes incluyen:

1. **Filtrado de Spam**: Clasificación de correos electrónicos como spam o no spam basándose en su contenido.
2. **Clasificación de Texto**: Asignación de categorías a documentos o correos electrónicos basándose en su contenido.
3. **Análisis de Sentimiento**: Clasificación de opiniones o sentimientos en texto (positivo/negativo).
4. **Detección de Fraude**: Identificación de transacciones fraudulentas en sistemas financieros.
5. **Clasificación de Clientes**: Segmentación de clientes en función de su probabilidad de respuesta a ofertas.

Resumen

En resumen, Naive Bayes es un algoritmo de aprendizaje supervisado utilizado para clasificación. Se basa en el teorema de Bayes y asume que las características son independientes entre sí. Su capacidad para manejar grandes volúmenes de datos y su simplicidad lo convierten en una opción popular en el campo del aprendizaje automático.

127.0.0.1:5500/MLFLASK%2Ftask-17/templates/NaiveBayes.html

WhatsApp Detail | News | Millo... Naruto Shippuden... matematicas Download Freeletics... DISCORD App Inventor Tutori... Dragon Ball Super 7... >> | Todos los fav

Naive Bayes

Descripción

Naive Bayes es un conjunto de algoritmos de clasificación basados en el teorema de Bayes, con la suposición (ingenua) de que todas las características son independientes entre sí dado el valor de la clase. Es especialmente útil en tareas de clasificación de texto como filtrado de spam, análisis de sentimientos y clasificación de documentos. Es rápido, eficiente y funciona bien incluso con conjuntos de datos pequeños. Existen varias variantes como **Gaussian Naive Bayes** (para datos continuos), **Multinomial** (para datos discretos como conteos de palabras) y **Bernoulli** (para datos binarios).

Fuente de información

[Documentación de scikit-learn - Naive Bayes](#)

Contenido gráfico

Naive Bayes Classifier

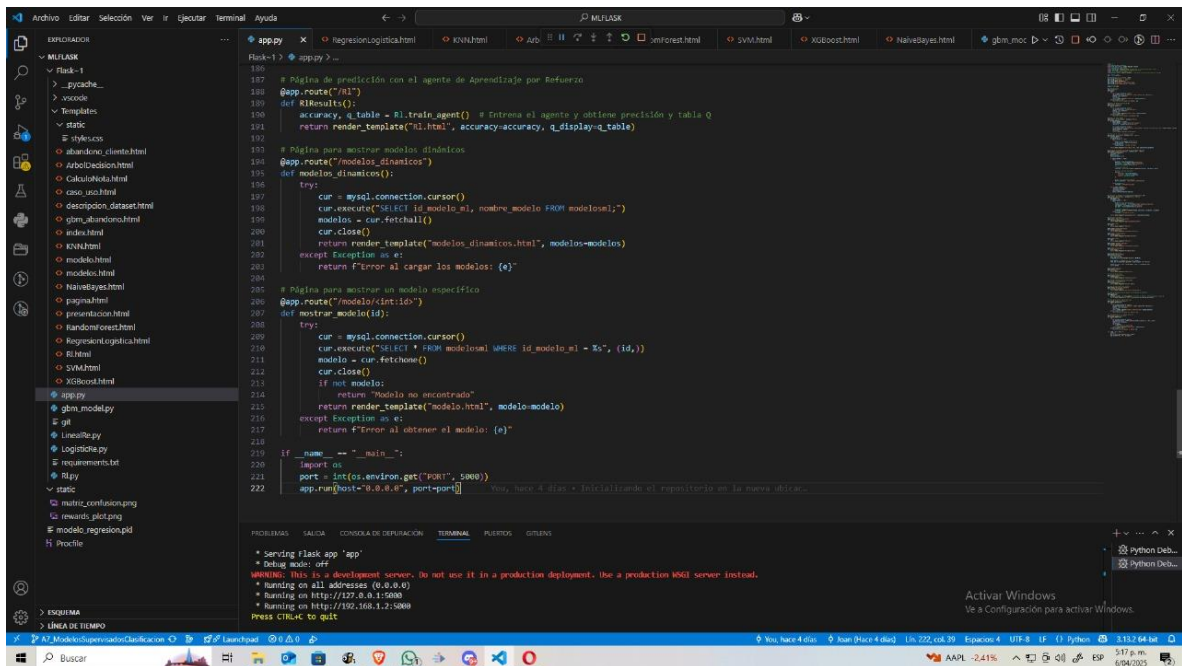
classifier

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

18
2023

Ejemplo de almacenamiento en base de datos

Nombre	Descripción	Fuente	Imagen
--------	-------------	--------	--------

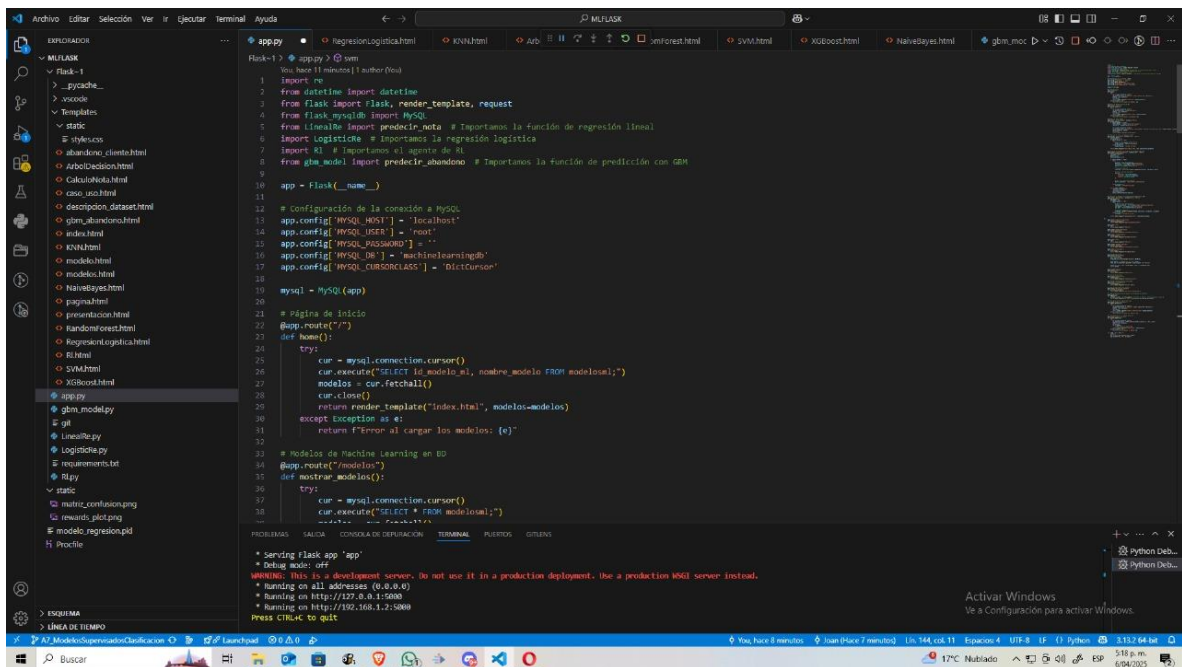


```
Flask-1.2 > app.py >
186
187 # Página de predicción con el agente de Aprendizaje por Refuerzo
188 @app.route("/rl")
189 def RLResult():
190     accuracy, q_table = RL.train_agent() # Entrena el agente y obtiene precisión y tabla Q
191     return render_template("rl.html", accuracy=accuracy, q_display=q_table)
192
193
194 # Página para mostrar modelos dinámicos
195 @app.route("/modelos_dinamicos")
196 def mostrar_modelos():
197     try:
198         cur = mysql.connection.cursor()
199         cur.execute("SELECT id_modelo_m1, nombre_modelo FROM modelos1;")
200         modelos = cur.fetchall()
201         cur.close()
202         return render_template("modelos_dinamicos.html", modelos=modelos)
203     except Exception as e:
204         return f"Error al cargar los modelos: {e}"
205
206 # Página para mostrar un modelo específico
207 @app.route("/modelo_dinamico")
208 def mostrar_modelo(id):
209     try:
210         cur = mysql.connection.cursor()
211         cur.execute("SELECT * FROM modelos1 WHERE id_modelo_m1 = %s", (id,))
212         modelo = cur.fetchone()
213         cur.close()
214         if not modelo:
215             return render_template("modelo.html", modelo=modelo)
216     except Exception as e:
217         return f"Error al obtener el modelo: {e}"
218
219 if __name__ == "__main__":
220     import os
221     port = int(os.environ.get("PORT", 5000))
222     app.run(host="0.0.0.0", port=port)
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS OTROS

- * Serving flask app 'app'
- * Debug mode: off
- WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
- * Running on all addresses (0.0.0.0)
- * Running on http://127.0.0.1:5000
- * Running on http://192.168.1.2:5000
- Press CTRL+C to quit

Activar Windows
Ve a Configuración para activar Windows.



```
Flask-1.2 > app.py > @ sm
You have 11 minutes 1 second left
1 import re
2 from datetime import datetime
3 from flask import Flask, render_template, request
4 from flask_sqlalchemy import SQLAlchemy
5 from lineal import prediccion_lineal # Importamos la función de regresión lineal
6 import LogisticRegression # Importamos la regresión logística
7 import RL # Importamos el agente de RL
8 from gbm_model import prediccion_abandono # Importamos la función de predicción con GBM
9
10 app = Flask(__name__)
11
12 # Configuración de la conexión a MySQL
13 app.config['MYSQL_HOST'] = 'localhost'
14 app.config['MYSQL_USER'] = 'root'
15 app.config['MYSQL_PASSWORD'] = ''
16 app.config['MYSQL_DB'] = 'machinelearningdb'
17 app.config['MYSQL_CURSORCLASS'] = 'DictCursor'
18
19 mysql = SQLAlchemy(app)
20
21 # Página de inicio
22 @app.route("/")
23 def home():
24     try:
25         cur = mysql.connection.cursor()
26         cur.execute("SELECT id_modelo_m1, nombre_modelo FROM modelos1;")
27         modelos = cur.fetchall()
28         cur.close()
29         return render_template("index.html", modelos=modelos)
30     except Exception as e:
31         return f"Error al cargar los modelos: {e}"
32
33 # Modelos de Machine Learning en BD
34 @app.route("/modelos")
35 def mostrar_modelos():
36     try:
37         cur = mysql.connection.cursor()
38         cur.execute("SELECT * FROM modelos1;")
39         modelos = cur.fetchall()
40         cur.close()
41         return render_template("modelos.html", modelos=modelos)
42     except Exception as e:
43         return f"Error al cargar los modelos: {e}"
44
45 if __name__ == "__main__":
46     import os
47     port = int(os.environ.get("PORT", 5000))
48     app.run(host="0.0.0.0", port=port)
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL PUERTOS OTROS

- * Serving flask app 'app'
- * Debug mode: off
- WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
- * Running on all addresses (0.0.0.0)
- * Running on http://127.0.0.1:5000
- * Running on http://192.168.1.2:5000
- Press CTRL+C to quit

Activar Windows
Ve a Configuración para activar Windows.

Branches

Overview Yours Active Stale All

Search branches...

Default

Branch	Updated	Check status	Behind	Ahead	Pull request
main	last week				

Your branches

Branch	Updated	Check status	Behind	Ahead	Pull request
A7_ModelosSupervisadosClasificacion	1 minute ago		4	3	
integracionlogistica_2aem	4 days ago		4	2	
actividad-1	2 months ago		1	0	

Active branches

Branch	Updated	Check status	Behind	Ahead	Pull request
A7_ModelosSupervisadosClasificacion	1 minute ago		4	3	
integracionlogistica_2aem	4 days ago		4	2	
actividad-1	2 months ago		1	0	

Activar Windows
Ve a Configuración para activar Windows.

Machine_learning

Compare & pull request

A7_ModelosSupervisadosClasificacion had recent pushes 2 minutes ago

main 4 Branches 0 Tags

Go to file Add file Code

Mercode Add files via upload e215161 last week 4 Comments

Plak-1 Add files via upload last week

1 Version 1 2 months ago

test Update test 2 months ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

Add a README

About

No description, website, or topics provided.

Activity

0 stars

1 watching

1 fork

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Languages

HTML 50.8% Python 45.3% CSS 3.9%

Suggested workflows

Based on your tech stack

Python Package using Anaconda

Create and test a Python package on multiple Python versions using Anaconda for package management.

Activar Windows
Ve a Configuración para activar Windows.

