Inteligencia Artificial para profesionales TIC

EDA e ingeniería de características

valgrai.eu

Índice

1. Análisis exploratorio de datos

Esencial. Los datos son nuestra fuente, nuestro principal entrada.

Análisis exploratorio de datos(EDA) es un paso crucial en cualquier proyecto de aprendizaje automático, ya que le ayuda a comprender el conjunto de datos y descubrir patrones, tendencias y relaciones subyacentes.

Pasos principales:

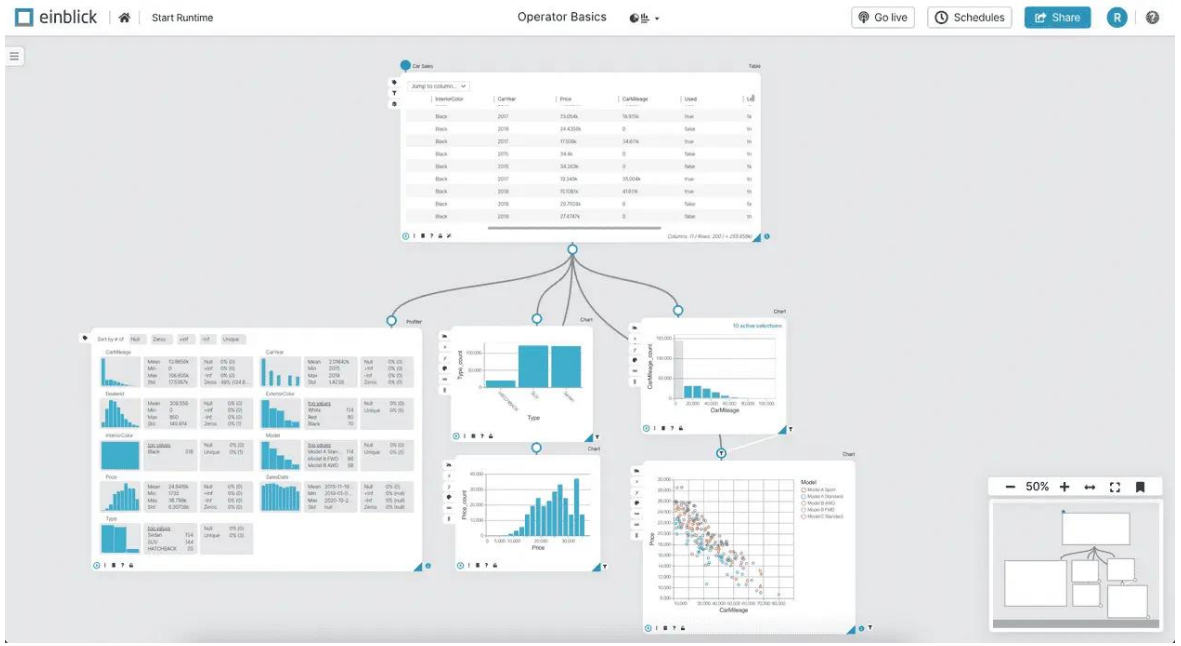
1. Importación de datos y exploración inicial
2. Limpieza de datos
3. Análisis de características
4. Correlación e interacción de características
5. Transformación de datos
6. Ingeniería de características
7. Reducción de dimensionalidad
8. Exploración visual

1. Análisis exploratorio de datos

Importación de datos y exploración inicial

Importación de datos y exploración inicial:

El objetivo es familiarizarse con el conjunto de datos y recopilar información esencial sobre su estructura y contenido, lo que le permite identificar posibles problemas, patrones y relaciones que deben abordarse o explorarse más a fondo.



1. Análisis exploratorio de datos

Importación de datos y exploración inicial

Pasos básicos en Python y Pandas (no te preocupes, se proporcionarán ejemplos específicos):

1. **Cargar el conjunto de datos:** Utilice las funciones adecuadas para cargar el conjunto de datos en el entorno de programación. Por ejemplo, en Python, puede usar pd.read_csv() para cargar archivos CSV.
2. **Examinar el conjunto de datos:** Observa las primeras filas del conjunto de datos usando data.head(). Esto le dará una idea de la estructura de datos, las características y sus valores.
3. **Conjunto de datos y dimensiones:** Compruebe el número de filas y columnas del conjunto de datos mediante data.shape. Esto le da una idea del tamaño y la complejidad del conjunto de datos.
4. **Nombres de columna:** Imprime los nombres de las columnas mediante data.columns. Esto te ayuda a comprender las características disponibles e identificar cualquier problema potencial con los nombres de columna, como espacios o caracteres especiales.
5. **Tipos de datos:** Examina los tipos de datos de cada columna mediante data.dtypes. Esto te ayuda a identificar si alguna columna tiene tipos de datos incorrectos que deben corregirse durante la fase de limpieza de datos.
6. **Resumen estadístico:** Utiliza data.describe() para generar estadísticas de resumen para las columnas numéricas, como la media, la desviación estándar, el mínimo, el máximo y los cuartiles. Esto proporciona información sobre la tendencia central, la dispersión y la distribución general de las características numéricas.
7. **Resúmenes de características categóricas:** Para características categóricas, usa data[column_name].value_counts() para obtener la distribución de frecuencias de cada categoría. Esto te ayuda a comprender el equilibrio de las categorías e identificar cualquier problema potencial, como el desequilibrio de clases o las categorías raras.
8. **Comprobar si faltan valores:** Utilice data.isnull().sum() para identificar el número de valores que faltan en cada columna. Esto te ayudará a determinar la estrategia adecuada para manejar los datos que faltan durante la fase de limpieza de datos.
9. **Identificar valores únicos:** Utiliza data[column_name].nunique() para encontrar el número de valores únicos en cada columna. Esto puede ayudarle a identificar si alguna columna tiene demasiados valores únicos o muy pocos, lo que indica posibles problemas que deben abordarse.
10. **Trazar visualizaciones iniciales:** Genera visualizaciones básicas, como histogramas, diagramas de barras o diagramas de dispersión, para obtener una comprensión inicial de las distribuciones de datos y las relaciones entre las entidades.

1. Análisis exploratorio de datos

Importación de datos y exploración inicial

Aunque veremos ejemplos concretos, las librerías clásicas en Python en esta etapa son:

- **Pandas**: Una potente biblioteca para la manipulación y el análisis de datos, que proporciona funciones esenciales para cargar, explorar y resumir conjuntos de datos. Las funciones clave incluyen andread_csv(), head(), shape, columns, dtypes, describe(), value_counts(), isnull(), nunique().
- **NumPy**: Una biblioteca para computación numérica en Python, que proporciona soporte para matrices, álgebra lineal y varias funciones matemáticas. Puede ser útil para manejar datos numéricos y realizar operaciones matemáticas durante la fase de exploración.
- **Matplotlib**: Una biblioteca fundamental para crear visualizaciones estáticas, animadas e interactivas en Python. Puede usarlo para crear visualizaciones básicas como histogramas, diagramas de barras o diagramas de dispersión para la exploración inicial.
- **Seaborn**: Una biblioteca de visualización de datos estadísticos de alto nivel construida sobre Matplotlib, que ofrece más opciones estéticas y visualizaciones complejas. Se puede utilizar para crear gráficos avanzados como diagramas de caja, diagramas de violín y diagramas de pares para explorar más a fondo los datos.

1. Análisis exploratorio de datos

Importación de datos y exploración inicial

Vale, pero cualquier biblioteca de alto nivel para facilitarnos aún más el trabajo:

- **Sweetviz:** Una biblioteca para crear visualizaciones automatizadas de alta calidad e informes resumidos de su conjunto de datos. Genera un informe HTML conciso e informativo, que incluye visualizaciones univariantes y bivariantes, análisis de valores faltantes y más.
- **Pandas Profiling:** Una biblioteca que genera informes de perfil a partir de un DataFrame de pandas, lo que proporciona una forma rápida de realizar una exploración inicial del conjunto de datos. El informe incluye estadísticas resumidas, análisis de valores faltantes, correlaciones y varias visualizaciones.
- **Dora:** Una biblioteca para optimizar el flujo de trabajo de exploración y preprocesamiento de datos, que proporciona funciones de alto nivel para la limpieza de datos, la selección de características y la visualización. Simplifica el proceso de exploración y preparación de datos para modelos de aprendizaje automático.
- **AutoViz:** Una biblioteca para generar visualizaciones automáticamente, dado un conjunto de datos. Produce rápidamente una variedad de gráficos para ayudarle a comprender las relaciones, los patrones y las tendencias de los datos.
- Y muchos más.

1. Análisis exploratorio de datos

Limpieza de datos

La vida es difícil. Y verás que los datos siempre serán **sucios y caóticos**. La mayor parte de su tiempo estará luchando con conjuntos de datos de entrada.

La **limpieza de datos** es esencial antes de intentar entrenar cualquier modelo. Los modelos son muy sensibles a los datos erróneos.

Este paso implica el control de los valores que faltan, la eliminación de entradas duplicadas y la corrección de datos incoherentes o erróneos. Puede usar métodos como la imputación, la eliminación o la interpolación para controlar los valores que faltan, según el contexto (Pandas, NumPy y otras bibliotecas).

Un buen consejo en cualquiera de estas etapas:

“Si decides manejar los valores que faltan de alguna manera, o eliminar duplicados o cualquier fila por cualquier motivo, o una columna (característica), o cualquier otra acción… micrómetro…. ¿Por qué? Echa un vistazo a las métricas del modelo antes y después de cualquier decisión. ¡¡¡Siempre!!!”

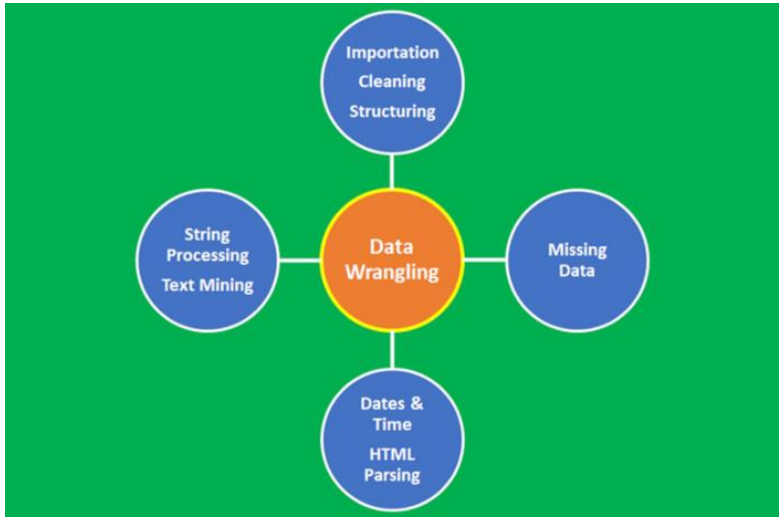
No hay decisiones arbitrarias, esto es ingeniería!

1. Análisis exploratorio de datos

Data wrangling

Data wrangling es el proceso de convertir los datos de su forma bruta a una forma ordenada lista para el análisis

Data wrangling es un paso importante en el preprocesamiento de datos e incluye varios procesos como la importación de datos, la limpieza de datos, la estructuración de datos, el procesamiento de cadenas, el análisis de HTML, el manejo de fechas y horas, el manejo de datos faltantes y la minería de texto



Web Scraping
Tools & libraries than can help you capture structured and unstructured data from URLs

1. Análisis exploratorio de datos

Análisis de características

Analiza cada variable (o característica) para comprender su distribución, tendencia central, dispersión y presencia de valores atípicos.

Traza histogramas, diagramas de caja o gráficos de violín para visualizar las distribuciones. En el caso de las variables categóricas, analice la distribución de frecuencias mediante diagramas de barras o gráficos circulares.

Tenemos muchas opciones en Python como se ha descrito anteriormente, y veremos algunos ejemplos prácticos.

1. Análisis exploratorio de datos

Correlación e interacción de características

Determinar las relaciones entre pares de entidades utilizando coeficientes de correlación (por ejemplo, Pearson, Spearman) y visualizarlas mediante mapas de calor o diagramas de dispersión.

Este paso le ayudará a identificar entidades altamente correlacionadas que podrían dar lugar a multicolinealidad en modelos lineales o redundancia innecesaria en otros modelos.

Multicolinealidad: Si dos o más características están altamente correlacionadas, pueden conducir a la multicolinealidad, lo que dificulta que el modelo distinga los efectos individuales de las características correlacionadas.

Complejidad del modelo: La inclusión de características altamente correlacionadas o que interactúan puede introducir una complejidad innecesaria en el modelo, lo que aumenta el riesgo de sobreajuste y dificulta la interpretación de las predicciones del modelo.

Características redundantes: Si no se comprueba la correlación y la interacción, es posible que se mantengan características redundantes en el conjunto de datos, lo que puede provocar un aumento de los costos de cálculo y tiempos de entrenamiento más largos sin proporcionar ningún valor adicional al rendimiento del modelo.

1. Análisis exploratorio de datos

Transformación de datos

En función de su análisis, puede aplicar transformaciones para **normalizar la distribución de las entidades o reducir el impacto de los valores atípicos**. Entre las transformaciones comunes se incluyen la transformación logarítmica, la transformación de raíz cuadrada y el escalado (por ejemplo, escalado mínimo-máximo o escalado estándar).

Razones para la transformación de datos:

Diferentes escalas: Las entidades del conjunto de datos pueden tener diferentes unidades y escalas, lo que puede hacer que algunos algoritmos de aprendizaje automático den más importancia a las entidades con magnitudes mayores.

Distribución: Muchos algoritmos de aprendizaje automático asumen que las entidades de entrada siguen una distribución específica (por ejemplo, la distribución gaussiana) o funcionan mejor cuando los datos son más simétricos.

Manejo de valores atípicos: Los valores atípicos pueden tener un impacto significativo en el rendimiento de ciertos algoritmos. Las técnicas de transformación de datos, como el escalado sólido o la transformación de registros, pueden ayudar a reducir la influencia de los valores atípicos en el modelo.

Codificación de características categóricas: Muchos algoritmos de aprendizaje automático requieren que las características categóricas se transformen en valores numéricos. Las técnicas de transformación de datos, como la codificación one-hot, la codificación de etiquetas o la codificación de destino, pueden ayudar a convertir características categóricas en un formato adecuado para el algoritmo.

2. Ingeniería de características

Uno de los pasos más críticos en EDA antes de entrenar nuestro modelo

La **ingeniería de características** es esencial y crítica para nuestro éxito en la creación de un modelo de aprendizaje automático eficaz.

Las decisiones que se toman durante la etapa de ingeniería de características suelen ser un **esfuerzo colaborativo** en el que participan científicos de datos, expertos en el dominio y otras partes interesadas.

Los científicos de datos aportan su experiencia en análisis de datos y aprendizaje automático, mientras que los expertos en el dominio proporcionan el contexto y la comprensión necesarios del dominio del problema específico. Juntos, pueden tomar decisiones informadas sobre estrategias de ingeniería de características que, en última instancia, conducirán a modelos de aprendizaje automático de mejor rendimiento y más interpretables.

2. Ingeniería de características

Un ejemplo

Un ejemplo de lo importante que es la colaboración en esta etapa:

Consideremos un ejemplo más simple del dominio del comercio electrónico, donde el objetivo es desarrollar un modelo de aprendizaje automático que prediga la pérdida de clientes en función de su historial de compras e interacciones con la tienda en línea.

Durante la etapa de ingeniería de características, un científico de datos y un experto en el dominio (por ejemplo, un especialista en marketing) colaboran para crear características significativas y tomar decisiones informadas. Así es como una mala decisión puede afectar al modelo:

Mala decisión:

El científico de datos, sin consultar al experto en el dominio, decide utilizar el número total de artículos comprados por un cliente como característica. Sin embargo, el número total de artículos no refleja necesariamente el compromiso del cliente con la tienda, ya que algunos clientes pueden realizar algunas compras de alto valor, mientras que otros pueden realizar muchas compras de bajo valor.

Como resultado, el modelo no logra capturar la verdadera relación entre el compromiso del cliente y la pérdida de clientes, lo que lleva a un rendimiento subóptimo y predicciones de pérdida de clientes inexactas. Si el equipo de marketing se basa en este modelo, es posible que se dirija a los clientes equivocados con sus esfuerzos de retención o que pierda a los clientes que en realidad corren un mayor riesgo de abandono.

2. Ingeniería de características

Un ejemplo

Buena decisión:

El experto en el dominio (especialista en marketing) sugiere que el valor medio de la compra (ingresos totales divididos por el número de transacciones) podría ser una característica más informativa para evaluar la participación del cliente, ya que captura tanto el valor monetario como la frecuencia de las compras. El científico de datos está de acuerdo y crea una nueva característica, "Average_Purchase_Value", basada en esta información.

Esta nueva función representa mejor el compromiso del cliente con la tienda y captura con mayor precisión la relación entre el comportamiento del cliente y la pérdida de clientes. Como resultado, el rendimiento del modelo de aprendizaje automático mejora y puede predecir con mayor precisión la pérdida de clientes. Con mejores predicciones, el equipo de marketing puede dirigirse de manera más efectiva a los clientes en riesgo de abandono con estrategias de retención, lo que en última instancia conduce a una mejor retención de clientes y un aumento de los ingresos para el negocio.

Una buena decisión durante la etapa de ingeniería de características, impulsada por una colaboración eficaz entre los científicos de datos y los expertos en el dominio, puede conducir a la creación de características relevantes e informativas que mejoren el rendimiento y la interpretabilidad del modelo de aprendizaje automático.

2. Ingeniería de características

Principales responsabilidades de los involucrados

Conocimiento del dominio: Comprender el dominio del problema específico y utilizar este conocimiento para crear características significativas y relevantes que puedan mejorar el rendimiento del modelo..

Data exploration: Investigar el conjunto de datos para identificar patrones, relaciones y posibles problemas que se pueden abordar mediante la ingeniería de características.

Creación de características: Diseñar nuevas características que capturen las relaciones subyacentes en los datos, a menudo mediante la combinación, transformación o agregación de características existentes.

Transformación de características: Aplicación de diversas técnicas, como la normalización, la estandarización o la codificación, para transformar las características existentes en un formato más adecuado para el algoritmo de aprendizaje automático elegido..

Selección de características: Identificación y eliminación de características irrelevantes, redundantes o ruidosas para reducir la dimensionalidad, mejorar el rendimiento del modelo y reducir el riesgo de sobreajuste.

Experimentación iterativa: Experimentar continuamente con diferentes técnicas de ingeniería de características, evaluar su impacto en el rendimiento del modelo y refinar el conjunto de características en función de los resultados.

Colaboración: Trabajar en estrecha colaboración con expertos en el dominio, científicos de datos y otras partes interesadas para recopilar información, validar suposiciones y garantizar que las características de ingeniería sean relevantes y útiles para el problema específico en cuestión..

2. Ingeniería de características

Algunas reflexiones finales

La importancia de la ingeniería de características es particularmente evidente en los datasets **estructurados**, donde las características de entrada a menudo están definidas por humanos y pueden no representar adecuadamente los patrones subyacentes en los datos. En estos casos, la ingeniería de características puede desempeñar un papel crucial en la mejora del rendimiento y la interpretabilidad del modelo.

Por el contrario, las técnicas de aprendizaje de representaciones, como el **aprendizaje profundo**, pueden aprender automáticamente las características o representaciones más relevantes de los datos **no estructurados**, lo que reduce la necesidad de ingeniería manual de características. Sin embargo, incluso en el aprendizaje profundo, la ingeniería de características puede proporcionar beneficios en ciertos casos, como cuando los datos disponibles son limitados, cuando existe la necesidad de mejorar la interpretabilidad del modelo o cuando el conocimiento del dominio se puede utilizar para crear características más informativas.

El término "**maldición de la dimensionalidad**" se refiere a los desafíos y problemas que surgen cuando se trabaja con datos de alta dimensión, particularmente en el aprendizaje automático y el análisis de datos. A medida que aumenta el número de características (dimensiones) de un conjunto de datos, la complejidad y los requisitos computacionales de los algoritmos crecen exponencialmente. Una buena selección de características y técnicas de reducción de dimensionalidad (PCA, por ejemplo) pueden proporcionar soluciones a este desafío.

2. Ingeniería de características

Algunas reflexiones finales

El **desequilibrio de datos** se produce cuando las clases de un conjunto de datos no se representan por igual, y una o más clases tienen significativamente menos instancias que otras. Este es un problema común en muchos problemas de aprendizaje automático, especialmente en las tareas de clasificación. Las consecuencias del desequilibrio de datos incluyen::

1. **Sesgo del modelo:** Los datos desequilibrados pueden dar lugar a modelos sesgados hacia la clase mayoritaria, lo que da lugar a un rendimiento deficiente de la clase minoritaria.

2. **Precisión engañosa:** Una alta precisión general puede ser engañosa, ya que el modelo puede estar simplemente prediciendo la clase mayoritaria la mayor parte del tiempo, descuidando la clase minoritaria.

3. **Sobreajuste:** Los conjuntos de datos desequilibrados pueden provocar un sobreajuste, ya que el modelo puede ajustar el ruido de la clase mayoritaria e ignorar la clase minoritaria infrarrepresentada.

2. Ingeniería de características

Algunas reflexiones finales

Supongamos que tenemos un conjunto de datos de pacientes, donde el objetivo es predecir si un paciente tiene COVID-19 en función de varias características, como la edad, el sexo, los síntomas y el historial médico. En este conjunto de datos, la mayoría de los pacientes no tienen COVID-19 (casos negativos), mientras que un porcentaje menor tiene COVID-19 (casos positivos).

Esto conduce a un conjunto de datos desequilibrado, siendo los casos positivos la clase minoritaria.

Para hacer frente al desequilibrio de datos en este conjunto de datos de pacientes con COVID-19, podemos aplicar las siguientes técnicas:

Métodos de remuestreo:

Sobremuestrear los casos positivos (pacientes con COVID-19) seleccionando aleatoriamente casos de la clase minoritaria con reemplazo hasta que la distribución de clases esté equilibrada.

Alternativamente, submuestree los casos negativos (pacientes que no tienen COVID-19) seleccionando aleatoriamente un subconjunto de casos de la clase mayoritaria para equilibrar la distribución de clases.

Generación de datos sintéticos:

Utilice SMOTE (Técnica de sobremuestreo de minorías sintéticas) para generar casos positivos sintéticos (pacientes con COVID-19) en función del espacio de características de los casos positivos existentes. Esto ayudará a equilibrar la distribución de clases y proporcionará más instancias de capacitación para la clase minoritaria.

2. Ingeniería de características

Algunas reflexiones finales

Funciones de pérdida ponderada:

Entrene un modelo de aprendizaje automático, como la regresión logística o las máquinas de vectores de soporte, utilizando una función de pérdida ponderada que asigne mayores ponderaciones a los casos positivos (pacientes con COVID-19). Esto hará que el modelo sea más sensible a la clase minoritaria y mejorará su desempeño en la predicción de casos de COVID-19.

Métodos de conjunto:

Utilice técnicas de ensamble como el bagging o el boosting con modelos base que sean capaces de manejar datos desequilibrados. Por ejemplo, puede usar árboles de decisión con penalizaciones ponderadas por clase o aprendizaje sensible a los costos como modelos base en un bosque aleatorio o un algoritmo de aumento de gradiente.

Evalúe el rendimiento del modelo utilizando las métricas adecuadas:

Dado que el conjunto de datos está desequilibrado, confiar únicamente en la precisión puede ser engañoso. Utilice métricas de evaluación adicionales como precisión, recuerdo, puntuación F1, etc.

Valencian Graduate School and Research Network of Artificial Intelligence

valgrai.eu

valgrai.eu

Valencian Graduate School and Research Network of Artificial Intelligence