|  |  |
| --- | --- |
| **Nombre:** | Luis de la Garza González |
| **Matrícula:** | al03101869 |
| **Nombre del curso:** | Machine Learning. |
| **Trabajo:** | Actividad Fase I.- Modelos matemáticos para el aprendizaje en territorio. |
| **Nombre del profesor:** | Mtro. Igor García Atutxa |
| **Fecha:** | 05 de octubre de 2025 |

**Contenido**

[**Objetivo** 3](#_Toc205912285)

[**Instrucciones:** 3](#_Toc205912286)

[**Rúbrica:** 5](#_Toc205912287)

[**Desarrollo** 7](#_Toc205912288)

[**Conclusiones** 9](#_Toc205912289)

[**Liga al código en Github** 10](#_Toc205912290)

# **Objetivo**

Aplicar …

# **Instrucciones:**

**1. Revisa:**

* Las operaciones con escalares, vectores y matrices.
* Los diferentes tipos de normas (Manhattan y euclidiana) y sus diferentes usos.
* La descomposición matricial en valores singulares y su interpretación geométrica.
* Las características del espacio de probabilidad, las variables aleatorias y los diferentes tipos de distribuciones de probabilidad.
* El teorema de Bayes y su aplicación para solucionar problemas probabilísticos.
* Los métodos de optimización y sus diferentes implementaciones.

**2. Lee con detenimiento la siguiente situación:**

En una planta industrial se está considerando la introducción de un vehículo inteligente que sea capaz de trasladarse de forma autónoma a través de las instalaciones y de transportar los productos desde el área de almacenamiento central hasta las diferentes zonas de manufactura. La empresa tiene tres áreas de manufactura y cada producto se encuentra empacado en un contenedor específico que identifica el área a la que pertenece.

**3. Aplicando el pensamiento lógico y analítico, modela la situación planteada anteriormente** y determina, a partir de tus conocimientos actuales, cuáles serían los componentes de aprendizaje automático que consideras necesarios incluir para implementar la solución.

**4.** Considerando la incorporación de los recursos disponibles de una plataforma en la nube (por ejemplo, IBM Cloud) y conociendo que el vehículo puede conectarse a esta de alguna forma. ¿Cuáles serían las nuevas características que podrías agregarle a tu solución para mejorar aún más la propuesta inicial?

**5.** Se conoce que un vehículo como el descrito puede tener una o varias cámaras incorporadas, cuya resolución de captura es de 1920 x 1080 píxeles.

Si la máxima calidad con la que se pueden transmitir las imágenes a través de la red está limitada a la tercera parte de la resolución original, desarrolla un programa que sea capaz de realizar esta reducción con la menor pérdida posible. Considera el lenguaje de programación Python y la aplicación de la descomposición matricial en valores singulares.

**6.** Realiza un análisis a partir de los conocimientos matemáticos recordados durante el módulo y determina cuáles de estos se necesitan poner en práctica para solucionar una problemática como la planteada en el punto 2.

**7.** Elabora un informe digital con tus respuestas y con el código utilizado para resolver el punto 5.

Entrega el reporte en el espacio indicado por la plataforma institucional.

# **Rúbrica:**

| **Nivel de desempeño** | | | | |
| --- | --- | --- | --- | --- |
| **Criterios de evaluación** | **Altamente competente**  100% – 86% | **Competente**  85% – 70%**)** | **Aún sin desarrollar la competencia**  69% – 0% | **%** |
| 1. Modelado de la situación y propuesta de solución | **20 – 18** | **17 – 15** | **14 – 0** | **20** |
| Modela correctamente la situación planteada, identifica de forma clara y detallada los elementos de aprendizaje automático necesarios para la solución, y los justifica adecuadamente en el contexto del problema. | Modela la situación y sugiere los elementos de aprendizaje automático, pero con justificaciones limitadas o generales sobre su elección. | No modela la situación de forma adecuada o no sugiere elementos relevantes de aprendizaje automático para la solución. |
| 2. Incorporación de características con plataforma en la nube | **20 – 18** | **17 – 15** | **14 – 0** | **20** |
| Propone características innovadoras y bien fundamentadas al incorporar una plataforma en la nube, mejorando significativamente la solución planteada. | Sugiere características adicionales con la plataforma en la nube, pero con impacto limitado o con fundamentos insuficientes. | No sugiere características nuevas relevantes o no considera el uso de la plataforma en la nube para mejorar la solución. |
| 3. Programa de compresión de imágenes en Python | **20 – 18** | **17 – 15** | **14 – 0** | **20** |
| Implementa un programa funcional que reduce la resolución de las imágenes utilizando descomposición matricial en valores singulares (SVD), con código bien documentado y resultados reproducibles. | Construye un programa funcional de compresión de imágenes, pero presenta errores menores en la implementación o en la documentación del código. | No implementa correctamente el programa de compresión de imágenes o no utiliza SVD como técnica principal. |
| 4. Análisis de conocimientos matemáticos necesarios | **20 – 18** | **17 – 15** | **14 – 0** | **20** |
| Realiza un análisis exhaustivo e identifica correctamente los conocimientos matemáticos necesarios (álgebra lineal, probabilidades, cálculo) para abordar la problemática, justificando su relevancia en cada etapa del proceso. | Identifica los conocimientos matemáticos necesarios, pero con explicaciones limitadas o con omisiones en algunos conceptos clave. | No identifica correctamente los conocimientos matemáticos necesarios o no justifica su aplicación en la problemática planteada. |
| 5. Informe y código entregable | **20 – 18** | **17 – 15** | **14 – 0** | **20** |
| Elabora un informe claro y bien estructurado que incluye evidencia completa del análisis y los resultados, con el código correctamente integrado y explicaciones detalladas de su funcionamiento. | Elabora un informe estructurado, pero con detalles incompletos o sin una integración clara del código. | No presenta un informe estructurado o no incluye evidencias completas del análisis y los resultados, o el código no es funcional o explicado adecuadamente. |

# **Desarrollo**

**1. Revisa:**

* Las operaciones con escalares, vectores y matrices.
* Los diferentes tipos de normas (Manhattan y euclidiana) y sus diferentes usos.
* La descomposición matricial en valores singulares y su interpretación geométrica.
* Las características del espacio de probabilidad, las variables aleatorias y los diferentes tipos de distribuciones de probabilidad.
* El teorema de Bayes y su aplicación para solucionar problemas probabilísticos.
* Los métodos de optimización y sus diferentes implementaciones.

**2. Lee con detenimiento la siguiente situación:**

En una planta industrial se está considerando la introducción de un vehículo inteligente que sea capaz de trasladarse de forma autónoma a través de las instalaciones y de transportar los productos desde el área de almacenamiento central hasta las diferentes zonas de manufactura. La empresa tiene tres áreas de manufactura y cada producto se encuentra empacado en un contenedor específico que identifica el área a la que pertenece.

**3. Aplicando el pensamiento lógico y analítico, modela la situación planteada anteriormente** y determina, a partir de tus conocimientos actuales, cuáles serían los componentes de aprendizaje automático que consideras necesarios incluir para implementar la solución.

¡Excelente reto, Luis! Esta situación se presta muy bien para aplicar un enfoque lógico y analítico que integre modelos matemáticos, visión computacional y aprendizaje automático en un contexto industrial. Vamos a descomponer el problema y modelarlo paso a paso.

Modelado lógico de la situación

Objetivo general

Diseñar un sistema de navegación autónoma para un vehículo inteligente que transporte productos desde el área de almacenamiento central hacia tres zonas de manufactura, identificando correctamente el destino de cada contenedor.

Componentes del sistema

1. Vehículo autónomo con capacidad de navegación y manipulación de carga.

2. Contenedores etiquetados con información del destino (zona de manufactura).

3. Mapa de la planta industrial con rutas, obstáculos y zonas definidas.

4. Sistema de control y toma de decisiones basado en datos sensoriales y etiquetas.

Análisis lógico del flujo de operación

[Inicio en almacenamiento central]

↓

[Identificación del contenedor y destino]

↓

[Planificación de ruta óptima]

↓

[Navegación autónoma evitando obstáculos]

↓

[Entrega en zona de manufactura correspondiente]

↓

[Retorno o nueva asignación]

```

---

🤖 Componentes de aprendizaje automático necesarios

Aquí se pueden integrar varios módulos de ML, cada uno con una función específica:

1. Clasificación de contenedores

- Objetivo: Identificar el destino del contenedor (Zona A, B o C).

- Modelo sugerido: Clasificador supervisado (e.g., CNN si se usan imágenes, o Random Forest si se usan códigos QR o RFID).

- Datos requeridos: Etiquetas visuales, códigos, colores, formas del contenedor.

2. Localización y mapeo (SLAM)

- Objetivo: Crear y actualizar un mapa de la planta en tiempo real.

- Modelo sugerido: Algoritmos de SLAM con redes neuronales para detección de obstáculos y reconocimiento de entorno.

- Datos requeridos: Sensores LIDAR, cámaras, GPS interno, odometría.

3. Planificación de rutas

- Objetivo: Determinar la ruta más eficiente y segura hacia el destino.

- Modelo sugerido: Algoritmos de búsqueda (A\*, Dijkstra) combinados con aprendizaje por refuerzo para adaptarse a cambios dinámicos.

- Datos requeridos: Mapa, ubicación actual, destino, obstáculos.

4. Detección y evasión de obstáculos

- Objetivo: Evitar colisiones durante el trayecto.

- Modelo sugerido: Redes neuronales convolucionales (CNN) para visión computacional o modelos de segmentación semántica.

- Datos requeridos: Imágenes en tiempo real, sensores de proximidad.

5. Optimización y aprendizaje continuo

- Objetivo: Mejorar el rendimiento del sistema con el tiempo.

- Modelo sugerido: Aprendizaje por refuerzo o aprendizaje federado si hay múltiples vehículos.

- Datos requeridos: Historial de rutas, tiempos de entrega, errores, condiciones del entorno.

Ejemplo de integración práctica

Supongamos que el vehículo detecta un contenedor con una etiqueta visual que indica "Zona B". El sistema:

1. Usa visión computacional para clasificar el destino.

2. Consulta el mapa y planifica la ruta más corta.

3. Se desplaza evitando obstáculos detectados por sensores.

4. Entrega el contenedor en la zona B.

5. Registra la operación para mejorar futuras decisiones.

Consideraciones sociales y técnicas

Dado tu interés en la equidad y el impacto territorial, podrías también considerar:

- Accesibilidad del sistema para operarios humanos.

- Capacitación del personal en el uso del vehículo.

- Evaluación del impacto en la eficiencia y condiciones laborales.

¿Te gustaría que modeláramos esto en Python con pseudocódigo o diagramas de flujo? También podríamos explorar cómo documentarlo en Jupyter para una presentación académica o técnica.

**4.** Considerando la incorporación de los recursos disponibles de una plataforma en la nube (por ejemplo, IBM Cloud) y conociendo que el vehículo puede conectarse a esta de alguna forma. ¿Cuáles serían las nuevas características que podrías agregarle a tu solución para mejorar aún más la propuesta inicial?

**¡Gran giro, Luis! Incorporar una plataforma en la nube como IBM Cloud abre un abanico de posibilidades para escalar, optimizar y hacer más inteligente el sistema. Vamos a explorar cómo esta integración puede transformar la propuesta inicial en una solución más robusta, colaborativa y adaptable.**

**---**

**## ☁️ Ventajas clave de usar IBM Cloud**

**- \*\*Escalabilidad\*\*: Procesamiento y almacenamiento sin límites físicos.**

**- \*\*Interoperabilidad\*\*: Integración con servicios de IA, IoT, bases de datos y APIs.**

**- \*\*Seguridad\*\*: Gestión de identidades, cifrado y monitoreo.**

**- \*\*Analítica avanzada\*\*: Capacidad de procesar grandes volúmenes de datos en tiempo real.**

**---**

**## 🚀 Nuevas características que podrías agregar**

**### 1. 📡 \*\*Monitoreo remoto y en tiempo real\*\***

**- \*\*Función\*\*: Visualizar la ubicación, estado y desempeño del vehículo desde cualquier dispositivo.**

**- \*\*Implementación\*\*: Usar IBM Watson IoT Platform para recibir datos de sensores y mostrar dashboards en IBM Cloud.**

**- \*\*Beneficio\*\*: Mejora la supervisión operativa y permite intervención rápida ante fallos.**

**---**

**### 2. 🧠 \*\*Entrenamiento y actualización dinámica de modelos\*\***

**- \*\*Función\*\*: Reentrenar modelos de clasificación, navegación y planificación con nuevos datos.**

**- \*\*Implementación\*\*: Usar IBM Watson Machine Learning para automatizar pipelines de entrenamiento.**

**- \*\*Beneficio\*\*: El sistema se adapta a cambios en el entorno físico o en los patrones de operación.**

**---**

**### 3. 📊 \*\*Análisis predictivo y mantenimiento preventivo\*\***

**- \*\*Función\*\*: Predecir fallos mecánicos o cuellos de botella logísticos.**

**- \*\*Implementación\*\*: Integrar IBM Cloud Pak for Data para analizar registros de operación y detectar anomalías.**

**- \*\*Beneficio\*\*: Reduce tiempos muertos y mejora la eficiencia operativa.**

**---**

**### 4. 🔄 \*\*Optimización colaborativa multi-vehículo\*\***

**- \*\*Función\*\*: Coordinar varios vehículos inteligentes en la planta.**

**- \*\*Implementación\*\*: Usar servicios de mensajería como IBM Event Streams (Kafka) para compartir rutas y estados.**

**- \*\*Beneficio\*\*: Evita congestión, mejora la distribución de tareas y permite decisiones colectivas.**

**---**

**### 5. 🧾 \*\*Registro histórico y trazabilidad\*\***

**- \*\*Función\*\*: Mantener un historial completo de entregas, rutas y decisiones tomadas.**

**- \*\*Implementación\*\*: Almacenar en IBM Cloudant (NoSQL) o IBM Db2.**

**- \*\*Beneficio\*\*: Facilita auditorías, análisis de desempeño y mejora continua.**

**---**

**### 6. 🌐 \*\*Interfaz web para usuarios humanos\*\***

**- \*\*Función\*\*: Permitir a operarios asignar tareas, consultar estados y modificar parámetros.**

**- \*\*Implementación\*\*: Crear una app web con IBM App Connect + Node-RED + servicios de backend.**

**- \*\*Beneficio\*\*: Mejora la interacción humano-máquina y democratiza el uso del sistema.**

**---**

**### 7. 🔐 \*\*Gestión de identidad y acceso\*\***

**- \*\*Función\*\*: Controlar quién puede modificar, supervisar o acceder a los datos.**

**- \*\*Implementación\*\*: Usar IBM Identity and Access Management (IAM).**

**- \*\*Beneficio\*\*: Asegura la integridad del sistema y protege datos sensibles.**

**---**

**## 🧭 Ejemplo de arquitectura en IBM Cloud**

**```plaintext**

**[Vehículo inteligente]**

**↓**

**[IoT Gateway] → [IBM Watson IoT Platform]**

**↓**

**[IBM Cloudant / Db2] ← [IBM Watson Machine Learning]**

**↓**

**[Dashboard web] ← [IBM App Connect + Node-RED]**

**↓**

**[Administradores / Operarios]**

**```**

**---**

**## 🌱 Consideraciones éticas y sociales**

**Dado tu enfoque en equidad y participación comunitaria, podrías:**

**- Diseñar interfaces inclusivas para operarios con distintos niveles de alfabetización digital.**

**- Evaluar el impacto en los roles laborales y proponer estrategias de reentrenamiento.**

**- Usar los datos para mejorar la distribución de tareas y evitar sobrecarga en zonas específicas.**

**¿Te gustaría que armemos un prototipo conceptual en Jupyter Notebook simulando la arquitectura y flujos de datos? También podríamos bosquejar un diagrama de flujo o mapa de componentes para presentarlo en un entorno académico o profesional.**

**5.** Se conoce que un vehículo como el descrito puede tener una o varias cámaras incorporadas, cuya resolución de captura es de 1920 x 1080 píxeles.

Si la máxima calidad con la que se pueden transmitir las imágenes a través de la red está limitada a la tercera parte de la resolución original, desarrolla un programa que sea capaz de realizar esta reducción con la menor pérdida posible. Considera el lenguaje de programación Python y la aplicación de la descomposición matricial en valores singulares.

**Perfecto, Luis. Esta es una excelente oportunidad para aplicar \*\*descomposición en valores singulares (SVD)\*\* como técnica de compresión de imágenes, manteniendo la mayor calidad posible dentro de la restricción de resolución.**

**---**

**## 🎯 Objetivo**

**Reducir una imagen de resolución \*\*1920×1080\*\* a \*\*640×360\*\* (un tercio en cada dimensión), utilizando \*\*SVD\*\* para preservar la mayor cantidad de información visual.**

**---**

**## 🧠 ¿Por qué SVD?**

**La SVD permite descomponer una imagen en componentes que capturan su estructura esencial. Al conservar solo los valores singulares más significativos, se puede reconstruir una versión comprimida de la imagen con \*\*mínima pérdida perceptual\*\*.**

**---**

**## 🧮 Fundamento matemático**

**Para una matriz de imagen \( A \in \mathbb{R}^{m \times n} \), la SVD es:**

**\[**

**A = U \Sigma V^T**

**\]**

**Donde:**

**- \( U \) y \( V \) son matrices ortogonales.**

**- \( \Sigma \) es una matriz diagonal con los \*\*valores singulares\*\*.**

**- Al conservar solo los primeros \( k \) valores singulares, obtenemos una \*\*aproximación rank-k\*\* de la imagen.**

**---**

**## 🐍 Código en Python**

**Aquí tienes un programa completo que:**

**1. Carga una imagen.**

**2. Convierte a escala de grises.**

**3. Reduce la resolución.**

**4. Aplica SVD para compresión.**

**5. Reconstruye la imagen comprimida.**

**```python**

**import numpy as np**

**import cv2**

**import matplotlib.pyplot as plt**

**def svd\_compression(img, k):**

**"""Aplica SVD a una imagen y reconstruye con los primeros k valores singulares."""**

**U, S, VT = np.linalg.svd(img, full\_matrices=False)**

**S\_k = np.diag(S[:k])**

**U\_k = U[:, :k]**

**VT\_k = VT[:k, :]**

**return np.dot(U\_k, np.dot(S\_k, VT\_k))**

**# Cargar imagen y convertir a escala de grises**

**img\_color = cv2.imread('vehiculo.jpg') # Reemplaza con tu imagen**

**img\_gray = cv2.cvtColor(img\_color, cv2.COLOR\_BGR2GRAY)**

**# Redimensionar a 640x360**

**img\_resized = cv2.resize(img\_gray, (640, 360), interpolation=cv2.INTER\_AREA)**

**# Determinar número óptimo de valores singulares (por ejemplo, 50)**

**k = 50**

**img\_svd = svd\_compression(img\_resized, k)**

**# Mostrar resultados**

**plt.figure(figsize=(12, 6))**

**plt.subplot(1, 2, 1)**

**plt.title("Imagen original reducida")**

**plt.imshow(img\_resized, cmap='gray')**

**plt.axis('off')**

**plt.subplot(1, 2, 2)**

**plt.title(f"Imagen comprimida con SVD (k={k})")**

**plt.imshow(img\_svd, cmap='gray')**

**plt.axis('off')**

**plt.show()**

**```**

**---**

**## 📌 Notas importantes**

**- Puedes ajustar el valor de `k` para balancear entre calidad y compresión. Valores entre \*\*30 y 100\*\* suelen funcionar bien.**

**- Si deseas aplicar esto a imágenes en color, puedes aplicar SVD \*\*por canal (R, G, B)\*\*.**

**- Para transmisión, puedes serializar la imagen comprimida usando formatos como JPEG o PNG después de la reconstrucción.**

**---**

**## 🧩 Extensiones posibles**

**- Integrar este módulo en un pipeline de transmisión en tiempo real usando \*\*OpenCV + MQTT o WebSockets\*\*.**

**- Comparar la compresión por SVD con otros métodos como JPEG estándar.**

**- Evaluar métricas de calidad como \*\*PSNR\*\* o \*\*SSIM\*\* para validar la pérdida.**

**¿Quieres que lo adaptemos para imágenes en color o que lo empaquetemos como módulo para usar en una arquitectura con IBM Cloud?**

**6.** Realiza un análisis a partir de los conocimientos matemáticos recordados durante el módulo y determina cuáles de estos se necesitan poner en práctica para solucionar una problemática como la planteada en el punto 2.

¡Vamos allá, Luis! La problemática del punto 2 —diseñar un sistema para que un vehículo inteligente identifique contenedores y entregue productos en áreas específicas de manufactura— requiere una integración de varios conocimientos matemáticos que seguramente has trabajado durante el módulo. Aquí te presento un análisis estructurado que conecta teoría con aplicación práctica.

---

## 🧠 Conocimientos matemáticos necesarios

### 1. 🔍 \*\*Álgebra lineal\*\*

- \*\*Representación matricial de imágenes\*\*: Las imágenes de los contenedores pueden modelarse como matrices de píxeles.

- \*\*Transformaciones lineales\*\*: Para procesar imágenes (rotación, escalado, filtrado).

- \*\*Descomposición en valores singulares (SVD)\*\*: Para compresión o extracción de características visuales.

📌 \*Aplicación\*: Procesamiento de imágenes captadas por el vehículo para identificar el área de destino del contenedor.

---

### 2. 🧠 \*\*Modelado y resolución de problemas\*\*

- \*\*Formulación matemática del entorno\*\*: Representar la planta como un grafo o red de caminos.

- \*\*Variables de decisión\*\*: Qué ruta tomar, qué contenedor transportar, cuándo hacerlo.

- \*\*Restricciones\*\*: Capacidad del vehículo, tiempos de entrega, obstáculos físicos.

📌 \*Aplicación\*: Permite estructurar el problema logístico como un modelo matemático optimizable.

---

### 3. 📈 \*\*Optimización\*\*

- \*\*Algoritmos de rutas óptimas\*\*: Dijkstra, A\*, Bellman-Ford para encontrar trayectorias eficientes.

- \*\*Programación lineal o entera\*\*: Para asignar tareas y minimizar tiempos o distancias.

- \*\*Heurísticas\*\*: Cuando el problema es muy complejo para soluciones exactas.

📌 \*Aplicación\*: El vehículo decide la mejor ruta para entregar cada contenedor.

---

### 4. 📊 \*\*Estadística y análisis de datos\*\*

- \*\*Distribución de entregas\*\*: Análisis de frecuencia por zona, horarios pico.

- \*\*Predicción de demanda\*\*: Modelos probabilísticos para anticipar necesidades por área.

- \*\*Evaluación de desempeño\*\*: Métricas como tiempo promedio de entrega, tasa de error.

📌 \*Aplicación\*: Mejora continua del sistema mediante análisis de datos históricos.

---

### 5. 🧮 \*\*Lógica matemática y sistemas de decisión\*\*

- \*\*Condicionales y reglas de inferencia\*\*: Para que el vehículo actúe ante situaciones imprevistas.

- \*\*Algoritmos de control\*\*: Para ajustar velocidad, dirección y comportamiento.

- \*\*Sistemas expertos\*\*: Reglas basadas en conocimiento para tomar decisiones autónomas.

📌 \*Aplicación\*: Asegura que el vehículo opere de forma segura y coherente.

---

### 6. 🧭 \*\*Geometría y trigonometría\*\*

- \*\*Ubicación y orientación\*\*: Cálculo de ángulos, distancias y trayectorias.

- \*\*Mapeo del entorno\*\*: Representación espacial de la planta y obstáculos.

📌 \*Aplicación\*: Navegación precisa dentro de la planta industrial.

---

## 🧩 Síntesis interdisciplinaria

Este tipo de problemática no solo requiere conocimientos técnicos, sino también sensibilidad al contexto:

- \*\*Educativo\*\*: Puede ser una excelente oportunidad para enseñar cómo las matemáticas se aplican en entornos reales.

- \*\*Territorial\*\*: La solución puede adaptarse a plantas con distintas configuraciones físicas.

- \*\*Social\*\*: Implica considerar cómo esta automatización afecta a los trabajadores y cómo se puede integrar de forma justa.

---

## 🧠 Reflexión final

Resolver esta problemática implica activar una red de saberes matemáticos que van desde la abstracción (modelado) hasta la acción (optimización y decisión). Es un ejemplo perfecto de cómo la matemática aplicada puede transformar procesos industriales de forma inteligente y contextualizada.

¿Te gustaría que armemos un esquema visual que conecte cada área matemática con su aplicación en esta problemática? También podríamos construir un modelo simplificado en Python para simular rutas y decisiones.

**7.** Elabora un informe digital con tus respuestas y con el código utilizado para resolver el punto 5.

Entrega el reporte en el espacio indicado por la plataforma institucional.

# **Conclusiones**

Métodos de optimización numérica y aprendizaje automático

1. Fundamento tecnológico

- Los métodos de optimización numérica son necesarios para el funcionamiento de computadoras y dispositivos electrónicos, ya que permiten el desempeño eficiente de cálculos complejos.

2. Aplicación en el aprendizaje automático

- Se basan en el descenso de gradiente, que es un concepto esencial en el aprendizaje automático (ML) y se utiliza para reducir funciones de costo o pérdida con el fin de predecir mejor los resultados.

3. Importancia del cálculo diferencial

- Conocer cosas como gradientes, derivadas, máximos y mínimos de alguna función es crucial para el correcto funcionamiento de estas técnicas.

4. Metáfora del descenso de gradiente

- Si no tienes cuidado al bajar al valle con una linterna iluminando el camino frente a ti, así es cada paso que necesitamos dar en cada iteración del algoritmo.

5. Optimización con restricciones

- Cuando hay restricciones en nuestros problemas, los multiplicadores de Lagrange nos ayudan a encontrar las soluciones óptimas.

6. Uso de herramientas computacionales

- Esto facilita los cálculos, pero se espera que el usuario tenga cierto juicio y conocimiento para usarlo correctamente (a diferencia de un sistema completamente automático).

7. Tres pilares fundamentales

- Los datos y modelos, utilizados en el aprendizaje, son los ingredientes básicos de todo sistema de aprendizaje automático. Un modelo no es bueno o malo según qué tan bien se desempeñe con datos previamente vistos; un buen modelo proporciona resultados refinados para que el nuevo conjunto de datos no visto, pero que sin embargo tiene alguna apuesta ofrecida, también haga una predicción.

8. Transformación de datos

- Los datos pueden estar en muchas formas, pero después de realizar las transformaciones adecuadas, todos se convierten en una forma vectorial, que es aceptada por estos modelos para aplicar lógica matemática y hacer algunas comparaciones.

9. El predictor como modelo matemático

- En el corazón del aprendizaje automático está esta idea: aprender un buen predictor. En términos generales, una buena función o modelo probabilístico que describa correctamente los datos.

10. Ajuste del modelo

- Escenarios:

1) Algunos de los parámetros del modelo no se pueden estimar,

2) Solo un conjunto de estimaciones de parámetros del modelo o

3) Todos los conjuntos del dominio tienen estimaciones de parámetros.

- Sobreajuste: el modelo se ajusta tan bien a los datos conocidos que pierde su capacidad de generalización.

- Subajuste: el modelo no aprende el patrón de los datos.

- Buen ajuste: el modelo encuentra el punto medio entre precisión y generalización.

11. Criterios de selección

- Una razón importante para los criterios de ajuste del modelo es que el predictor final requerirá trabajar de manera óptima con nuevos datos.

# **Liga al código en Github**

<https://github.com/luisgg121/ML-Fase-I.git>