INSTITUTO SUPERIOR TECNOLÓGICO QUITO

Formamos tu PROPÓSITO DE VIDA

GUÍA GENERAL DE

MACHINE E-LEARNING 1

AUTOR: ELVIS PACHACAMA

ISBN: 978-9942-562-18-0

ITQ

WWW.ITQ.EDU.EC

INVESTIGACIÓN

DOXA EDITION

INSTITUTO SUPERIOR TECNOLÓGICO QUITO
Excelencia en Educación Superior

GUÍA GENERAL DE MACHINE E-LEARNING 1

AUTOR: ELVIS PACHACAMA

EDICIÓN: PRIMERA

AÑO: 2025

TRABAJO EN EDICIÓN:

DOXA EDITION

EQUIPO EDITORIAL:

MSc. CAMILA DÍAZ DÍAZ

MSc. KEYERMAN TOAPANTA CISNEROS

ISBN: 978-9942-562-18-0

ISBN: 978-9942-562-18-0
9 789942 562180

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

CONTENIDO

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

ÍNDICE DE FIGURAS

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

ÍNDICE DE TABLAS

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

GUÍA GENERAL DE MACHINE E-LEARNING 1

1. DESCRIPCIÓN DE LA ASIGNATURA

La asignatura de Machine E-learning ofrece una inmersión profunda en el estudio y aplicación de algoritmos que permiten a las computadoras aprender de datos y realizar predicciones y decisiones sin ser explícitamente programadas para cada tarea. Este curso se centra en proporcionar una comprensión integral de las técnicas y herramientas clave utilizadas en el campo del aprendizaje automático.

A lo largo de la asignatura, se explorarán definiciones esenciales como aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo. Además, se abordarán técnicas específicas como regresión lineal, árboles de decisión. Máquinas de soporte vectorial, redes neuronales y métodos de ensemble. Los estudiantes utilizaran herramientas prácticas como Python, scikit-learn, TensorFlow y Keras para implementar y experimentar con estos algoritmos en situaciones reales.

La asignatura de Machine E-learning se integra estratégicamente con el perfil de egreso al proporcionar competencias esenciales para el análisis y la interpretación de grandes volúmenes de datos, habilidades altamente demandadas en el mercado laboral actual. Además, fomenta la capacidad de adaptación y aprendizaje continuo, características cruciales para el desarrollo profesional en un entorno tecnológico en constante cambio.

2. BIBLIOGRAFÍA

2.1. Básica

- Benítez Iglésias, R. (2014). Inteligencia artificial avanzada: (ed.). Editorial UOC.

El texto aborda los fundamentos, técnicas y aplicaciones de la inteligencia artificial avanzada, con énfasis en algoritmos modernos que buscan emular capacidades cognitivas humanas como el razonamiento, el aprendizaje, la percepción y la adaptación en entornos complejos. Más allá de una simple introducción, el autor se adentra en modelos simbólicos y conexionistas, y en enfoques contemporáneos que permiten a las máquinas tomar decisiones inteligentes.

- Bosch Rué, A. Casas Roma, J. & Lozano Bagén, T. (2019). *Deep learning: principios y fundamentos:* (ed.). Editorial UOC.

El libro proporciona una visión general y didáctica del campo del **Deep Learning**, una rama avanzada del aprendizaje automático que ha revolucionado múltiples áreas del conocimiento como

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

el procesamiento de imágenes, el reconocimiento de voz, el análisis de texto y la automatización inteligente. El texto se centra en los fundamentos matemáticos, las arquitecturas neuronales más utilizadas y los principales desafíos actuales en el desarrollo de modelos de aprendizaje profundo.

2.2. Complementaria

- (Restrepo Leal, Viloria Porto, & Robles Algarin, 2021).

Este libro ofrece una introducción didáctica al fascinante mundo de las redes neuronales artificiales, destacando su evolución histórica, sus fundamentos matemáticos y sus aplicaciones prácticas. Su enfoque combina rigor académico con una clara orientación pedagógica, lo cual lo hace especialmente útil para quienes se inician en el estudio de la inteligencia artificial desde una perspectiva formal y aplicada.

- (Bobadilla, 2020)

El libro Machine Learning y Deep Learning: Usando Python, Scikit y Keras, de Bobadilla (2020), constituye una referencia fundamental para la formación de estudiantes y profesionales que desean adquirir competencias en el desarrollo de sistemas inteligentes basados en datos. Este texto destaca por su enfoque práctico y progresivo, ya que introduce los conceptos esenciales del aprendizaje automático y el aprendizaje profundo utilizando herramientas ampliamente reconocidas en la industria, como Python, Scikit-learn y Keras. Además, el autor presenta explicaciones claras acompañadas de ejemplos aplicados, lo que facilita la comprensión de algoritmos, el procesamiento de datos y la construcción de modelos predictivos. Su contenido permite al lector desarrollar habilidades tanto teóricas como prácticas, fortaleciendo su capacidad para diseñar, entrenar y evaluar modelos de Machine Learning. Por esta razón, este libro se convierte en un recurso académico pertinente y actualizado para apoyar el proceso de enseñanza-aprendizaje en asignaturas relacionadas con inteligencia artificial y ciencia de datos.

- (Bosch Rué, 2019)

El libro Deep learning: principios y fundamentos, de Bosch Rué, Casas Roma y Lozano Bagén (2019), constituye una referencia académica esencial para la comprensión del aprendizaje profundo, ya que presenta de manera estructurada los conceptos fundamentales de las redes neuronales artificiales y su aplicación en el campo de la inteligencia artificial. Esta obra introduce al lector desde los principios básicos del aprendizaje automático hasta el funcionamiento interno de las redes neuronales, incluyendo su estructura, entrenamiento y optimización, lo que permite construir una base sólida para el desarrollo de modelos inteligentes.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Su enfoque descriptivo y progresivo facilita que los estudiantes comprendan no solo la implementación técnica, sino también los fundamentos teóricos que sustentan el Deep Learning, permitiendo interpretar cómo los algoritmos aprenden a partir de los datos y cómo pueden aplicarse en problemas reales. Además, el libro forma parte de la colección tecnológica de Editorial UOC y aborda temas como el preprocesamiento de datos, la evaluación de modelos y las arquitecturas neuronales, aspectos esenciales para la formación en Machine Learning y ciencia de datos.

Por esta razón, este libro se convierte en un recurso académico pertinente y actualizado para apoyar el proceso de enseñanza-aprendizaje en la asignatura de Machine Learning, ya que fortalece las competencias del estudiante en el análisis, diseño e implementación de modelos basados en redes neuronales, contribuyendo al desarrollo de habilidades necesarias para la creación de soluciones inteligentes en entornos reales.

- (Benítez Iglésias, 2014)

El libro Inteligencia artificial avanzada, de Benítez Iglésias (2014), constituye un recurso académico fundamental para la comprensión de los principios, técnicas y aplicaciones de la inteligencia artificial en contextos modernos. Esta obra presenta de manera estructurada los conceptos que permiten entender cómo los sistemas informáticos pueden simular procesos cognitivos humanos, como el aprendizaje, el razonamiento y la toma de decisiones. Su contenido proporciona una base sólida sobre algoritmos inteligentes, modelos computacionales y técnicas avanzadas que son esenciales para el desarrollo de soluciones basadas en Machine Learning y Deep Learning. Además, el libro contribuye a fortalecer el pensamiento analítico del estudiante, permitiéndole comprender cómo los sistemas inteligentes pueden aplicarse en el procesamiento de datos, la automatización y la resolución de problemas complejos. Por esta razón, esta obra se convierte en una referencia pertinente para apoyar el proceso de enseñanza-aprendizaje en la asignatura de Machine Learning, facilitando la formación de profesionales capaces de diseñar e implementar sistemas inteligentes en entornos reales.

- (Degli-Esposti, 2023)

El libro La ética de la inteligencia artificial, de Degli-Esposti (2023), constituye un recurso académico fundamental para comprender las implicaciones sociales, morales y humanas del uso de la inteligencia artificial en la sociedad actual. Esta obra permite analizar cómo la inteligencia artificial, basada en el uso de grandes volúmenes de datos, influye en distintos aspectos de la vida cotidiana y plantea desafíos relacionados con la privacidad, la equidad y la toma de decisiones automatizadas. Además, el texto aborda temas relevantes como los sesgos algorítmicos, la transparencia y la responsabilidad en el desarrollo de sistemas inteligentes, destacando la

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

importancia de diseñar tecnologías orientadas al bienestar humano y al respeto de los valores éticos. Por esta razón, este libro resulta especialmente pertinente para la formación en Machine Learning, ya que no solo fortalece el conocimiento técnico, sino que también promueve una visión crítica y responsable en el desarrollo de soluciones basadas en inteligencia artificial, contribuyendo a la formación de profesionales capaces de crear tecnologías seguras, confiables y socialmente responsables.

3. COMPETENCIAS GENÉRICAS Y ESPECÍFICAS

Valores y habilidades blandas: respeto, inteligencia emocional

Valores y habilidades blandas: amor, comunicación asertiva y escucha activa.

Valores y habilidades blandas: compromiso social, adaptabilidad.

Valores y habilidades blandas: lealtad, liderazgo.

Valores y habilidades blandas: optimismo, planificación y gestión del tiempo.

Valores y habilidades blandas: verdad y horades, proactividad.

Valores y habilidades blandas: orgullo nacional, pensamiento crítico.

Valores y habilidades blandas: cultura, creatividad.

Valores y habilidades blandas: tolerancia, flexibilidad.

Valores y habilidades blandas: gratitud, resiliencia.

Valores y habilidades blandas: solidaridad, trabajo en equipo.

4. OBJETIVO GENERAL

El objetivo de la asignatura de Machine E-Learning es proporcionar a los estudiantes la capacidad de analizar y comprender los principios y fundamentos del aprendizaje automático. A través de una exploración critica de conceptos clave y enfoques como el aprendizaje supervisado, no supervisado y por refuerzo, así como la aplicación de métodos y técnicas relacionadas como la regresión, los árboles de decisión y las redes neuronales, los estudiantes desarrollaran habilidades para analizar y resolver desafíos en diversos ámbitos e industrias.

Este conocimiento les permitirá aplicar herramientas específicas como Python, scikit-learn, tensorFlow y Keras de manera efectiva en contexto prácticos y situaciones reales. Al hacerlo,

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

contribuirán al desarrollo de un perfil profesional sólido, capaz de abordar y solucionar problemas complejos en áreas como la ciencia de datos, la inteligencia artificial, la salud, las finanzas y el marketing entre otras.

5. FORMACIÓN CIUDADANA, VALORES Y HABILIDADES BLANDAS

Valores y habilidades blandas: respeto, inteligencia emocional

Valores y habilidades blandas: amor, comunicación asertiva y escucha activa.

Valores y habilidades blandas: compromiso social, adaptabilidad.

Valores y habilidades blandas: lealtad, liderazgo.

Valores y habilidades blandas: optimismo, planificación y gestión del tiempo.

Valores y habilidades blandas: verdad y horades, proactividad.

Valores y habilidades blandas: orgullo nacional, pensamiento crítico.

Valores y habilidades blandas: cultura, creatividad.

Valores y habilidades blandas: tolerancia, flexibilidad.

Valores y habilidades blandas: gratitud, resiliencia.

Valores y habilidades blandas: solidaridad, trabajo en equipo.

6. NORMAS DE CLASE

En relación con las normas de clase es importante destacar que la evaluación de los componentes de gestión académica se compone de tres notas sumativas, cada una puntuación máxima de 60/60, así como un proyecto práctico, como evaluación formativa que se valora con 3.40/3.40, lo que da un total de 10/10 para la calificación del módulo. los parciales se califican en una escala de hasta 6.60 puntos, representando cada uno el 2.22 de la calificación total de6.6 puntos. para presentarse al proyecto final, el estudiante debe haber obtenido al menos 4.50 puntos sumando las tres primeras notas.

En caso de no alcanzar este mínimo en el proyecto, se otorga una oportunidad de recuperación dentro de las 48 horas laborables siguientes, según el calendario académico oficial. la nota mínima acumulada requerida para aprobar la asignatura es 7/10, y es esencial mantener al menos un 70% de asistencia a las clases.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Los docentes deben informar a los estudiantes sobre sus notas individuales antes de registrarlas en el sistema, y se espera que los alumnos confirmen su aceptación y conformidad con estas calificaciones. Además, los docentes deben entregar un reporte de notas y asistencia a través del sga y notificado a la coordinación de carrera y registrar las calificaciones en el sistema en un plazo máximo de 5 días posteriores a la recepción del proyecto final

7. SISTEMA DE EVALUACIÓN

INSTRUMENTO	PORCENTAJE	PUNTAJE
PARCIAL 1	22.00	2.2
PARCIAL 2	22.00	2.2
PARCIAL 3	22.0	2.2
EVALUACIÓN PROYECTO FINAL	34.00	3.4
TOTAL	**100.00**	**10.00**

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

8. UNIDADES

UNIDAD 1: INTRODUCCIÓN AL LENGUJAE DE PROGRAMACIÓN PYTHON

Temas y Subtemas

Tema 1: INTRODUCCIÓN A LA ASIGNATURA

- Socialización del PEA
- Normas Generales

Tema 2: INTRODUCCIÓN A PYTHON

1. Conceptos generales
2. Mi primer programa en Python

Tema 3: PREPARACIÓN DEL ENTORNO DE TRABAJO .

1. Instalación de Anaconda

Tema 4: TIPOS DE DATOS EN PYTHON.

1. Tipos de datos primitivos
2. Tipos de datos objetos

Tema 5: ESTRUCTURAS DE CONTROL Y ESTRUCTURAS DE REPETICIÓN.

1. Sentencia If
2. Sentencia For
3. Sentencia While
4. Funciones

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Introducción a Python

En primer lugar, Python es un lenguaje de programación de alto nivel, interpretado y orientado a objetos, diseñado para ser fácil de leer y escribir. Su simplicidad sintáctica y la amplia variedad de bibliotecas disponibles lo han convertido en uno de los lenguajes más populares para la programación moderna, especialmente en el ámbito del análisis de datos, inteligencia artificial y desarrollo web.

Entre sus características más destacadas se encuentran:

- Sintaxis sencilla y clara, que facilita su aprendizaje y reduce el tiempo de desarrollo.
- Código legible y estructurado, lo que hace ideal para la colaboración de proyectos grandes.
- Amplia comunidad de desarrolladores y una extensa documentación disponible en línea.
- Multiparadigma, permitiendo la programación estructurada, orientada a objetos y funcional.
- Compatible con múltiples plataformas, lo que significa que un mismo código ejecutarse en distintos sistemas operativos sin modificaciones.

Python es un lenguaje preferido en el campo del aprendizaje automático debido a la existencia de bibliotecas como Scikit-learn, TensorFlow, Keras, Pandas entre otras, que facilitan la manipulación de datos y la implementación de algoritmos de Machine E-Learning.

De la misma manera en Python el tipado es dinámico, esto quiere decir que se puede manejar diferentes tipos de datos sin que tengan que especificarse. Por ejemplo, el número de hijos de una persona es un dato que no tiene decimales, la estatura de una persona es un número con decimales, y el nombre de una persona es una cadena de caracteres (Muñoz Guerrero & Trejos Buriticá, 2021).

Tabla 1
Datos en Python

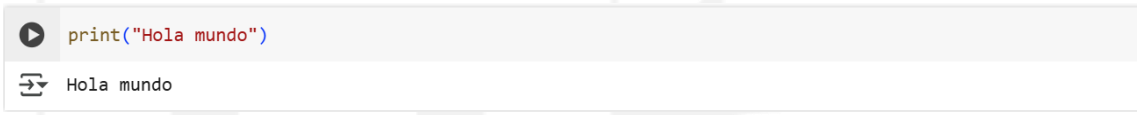
Nombre Variable	**Valor**
Nombre de la persona	“Norma”
Número de hijos	2
Estatura	1.70

Nota. La tabla muestra los datos que se manejan en Python. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

En muchos lenguajes de programación antes de utilizar un dato hay que especificar sus características. En Python no es necesario hacerlo. El lenguaje de programación es un lenguaje multiparadigma y multiplataforma. Que sea multiparadigma significa que con este lenguaje se pueden construir programas con enfoques diferentes, a saber: enfoque funcional (se privilegia el procesamiento), enfoque imperativo (se privilegia el almacenamiento en la memoria principal) y enfoque orientado a objetos (se privilegia la unión entre atributos, que son las características de las cosas u objetos, y los métodos, que son los usos que pueden tener dichas cosas u objetos). Que el lenguaje Python se multiplataforma significa que se puede utilizar tanto en Windows como en Macintosh y en Linux y en cualquier otro sistema operativo (Muñoz Guerrero & Trejos Buriticá, 2021).

Figura 1.
Código en Python



Nota. La imagen nos muestra el código de Python y la salida en pantalla el mensaje es Hola mundo. *Fuente*: El investigador.

Instalación y Configuración del Entrono de Trabajo.

Para trabajar con Python en Machine Learning, se recomienda el uso de Anaconda, este software es una distribución que incluye Python, herramientas de desarrollo y bibliotecas especializadas para análisis de datos e inteligencia artificial.

A diferencia de cualquier IDE de desarrollo Anaconda me permite configurar entorno de desarrollo virtuales independiente de la computadora que se esté usando.

Requisitos mínimos para la instalación de Anaconda.

Antes de proceder con la instalación, es importante asegurarse de que el sistema cumple con los siguientes requisitos mínimos.

- Sistema Operativo: Windows 8.1 o superior, macOS 10.12 o superior, Linux (Ubuntu, Fedora, Debian).
- Procesador: 64 bits, arquitectura x86 o ARM (para macOS con Apple Silicon se recomienda utilizar Rosetta 2).
- Memoria RAM: mínimo 4 GB de RAM (Se recomienda 8 GB o más para un mejor rendimiento en proyectos de Machine Learning).

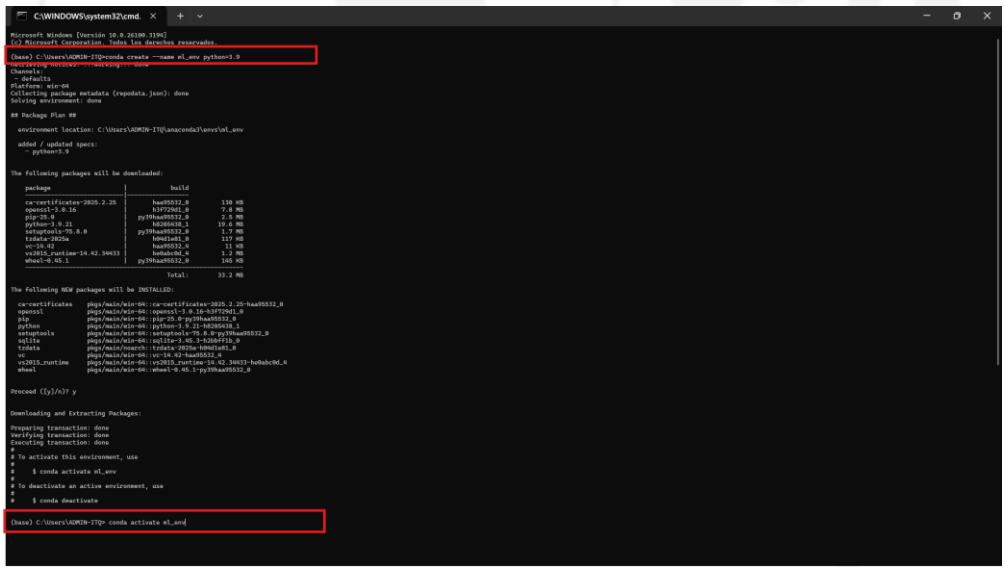
(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

- Espacio en Disco: Al menos 3 GB de espacio libre en el disco duro para la instalación básica (se recomienda más si se instalara paquetes adicionales).
- Conexión a Internet: Requerida para descargar la instalación y paquetes adicionales.

Pasos para instalar Anaconda

1. Descarga Anaconda: Ir al sitio oficial de Anaconda en https://www.anaconda.com/ y descargar la versión correspondiente a tu sistema operativo.
2. Instalar Anaconda
    a. En Windows ejecutar el instalador .exe y seguir las instrucciones del asistente de instalación.
    b. En macOS y Linux, abrir el archivo descargado y seguir las instrucciones.
3. Configurar el entorno:
    a. Abrir Anaconda Navigator o ejecutar conda en la terminal para asegurarse de que la instalación fue exitosa.
4. Crear un entorno virtual: este paso es opcional pero recomendado para proyectos específicos.

Figura 2.
Crear Entorno Virtual



Nota. La imagen muestra la configuración de un ambiente de trabajo en el cual tiene un nombre y una versión de Python, y también se le activa ese ambiente para poder trabajar en él. *Fuente*: El investigador.

5. Instalar bibliotecas necesarias:

Figura 3.
Instalar bibliotecas

```
(ml_env) C:\Users\ADMIN-ITQ> conda install numpy pandas matplotlib scikit-learn jupyterla
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: failed

PackagesNotFoundError: The following packages are not available from current channels:

  - jupyterla

Current channels:

  - defaults

To search for alternate channels that may provide the conda package you're
looking for, navigate to

    https://anaconda.org

and use the search bar at the top of the page.
```

Nota. La imagen muestra cómo se instala bibliotecas para poder utilizar las herramientas en Python. *Fuente*: El investigador.

Tipos de Datos en Python

¿Qué es una variable? Es una de las preguntas que más nos hacemos pues aquí la explicación una variable es un espacio de memoria en donde se puede guardar un dato. En una variable se puede guardar un número entero o decimal, o una cadena de caracteres. Las variables se identifican con un nombre, algo muy importante en la declaración de este nombre es que no debe contener caracteres especiales, no deben tener espacios en blanco entre nombre y una buena práctica debe hacer referencia a lo que se está guardando.

En general existen dos tipos de variables las que guardan números y las que guardan o almacenan texto. Las variables que almacenan números pueden guardar dos tipos los tipos entero o int (1, 2,100,250) y también pueden guardar número de tipo decimal (1.5) flotante. Las variables que almacenan texto pueden guardar cadenas de texto (string), que son conjunto de caracteres que van encerrados entre comillas dobles ejemplo (“Norma”).

Para que Python reconozca que tipo de dato está guardando se debe realizar como se muestra

Figura 4.
Tipos de Variables

```
# Variable de tipo entero
edad=10
# Variable de tipo flotante
peso=58.8
#Variable de tipo String
nombre="Elvis"
```


(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Nota. La imagen nos muestra los tipos de variables que se puede usar en Python. *Fuente*: El investigador.

Algo muy particular en Python es que si se usa la misma variable para almacenar otro valor esto es posible ya que la variable toma el valor en el cual se ha cambiado, como se muestra en la **Figura *5*** .

Figura 5.
Misma Variable.

Nota. La imagen nos muestra el código donde asignamos un valor a una variable numero1 y liego a esa variable se le asigna otro valor. *Fuente*: El investigador.

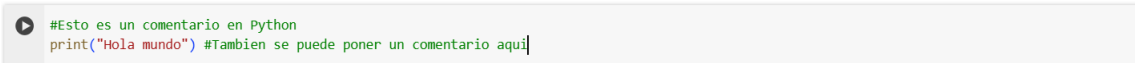
Sintaxis y estructura básica de un programa en Python

Python es un lenguaje de programación que se caracteriza por su simplicidad y legibilidad. Su sintaxis clara y concisa permite a los desarrolladores escribir código de manera eficiente y fácil de entender. A continuación, se presentan los elementos esenciales de la sintaxis de Python y la estructura básica de un programa.

Comentarios en Python

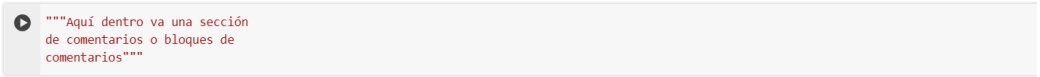
Los comentarios son líneas de texto dentro del código que no se ejecutan. Se utilizan para explicar el código y mejorar su legibilidad. Cabe recalcar que los comentarios son líneas que se encuentran en el código pero que no se van a compilar.

Figura 6.
Comentarios en Python



Nota: La imagen muestra cómo se pone el comentario en Python. *Fuente*: El investigador.

Figura 7.
Bloques de Comentarios



Nota. La imagen muestra cómo poner comentarios en un bloque y esta encerrados entre 3 pares de comillas dobles. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Indentación: A diferencia de otros lenguajes de programación, Python utiliza la indentación para definir bloques de código en lugar de las comunes pares de llaves {}.

Figura 8.
Indentación en Python

```
if True:
  print("Está es una línea indentada")
  print("Todo lo que esta dentro de la indentación pertenece al bloque if")
print("Esta línea no pertenece al bloque de código if")
```

Nota. La imagen nos muestra cómo se debe indentar el código para que este dentro de una sección de bloque de código. *Fuente*: El investigador.

Si la indentación es incorrecta Python no mostrara un mensaje de error de sintaxis.

Estructura básica de un script en Python

Un programa en Python generalmente sigue la siguiente estructura:

Figura 9.
Script en Python

```
#importamos bibliotecas necesaria
import math
#definir funciones
def suma(a,b):
  return a+b

# código principal
if __name__ == "__main__":
  resultado = suma(5,3)
  print (f"El resultado de la suma es : {resultado}")
```

Nota. La imagen muestra el código de una estructura básica de un programa básico en Python *Fuente*: El investigador.

1. Importación de bibliotecas, se utilizan para acceder a funcionalidades externas
2. Definición de funciones, sección de código que permite organizar la programación en bloques reutilizables.
3. Bloque principal, Se ejecuta cuando el programa es ejecutado directamente.

Variables y tipos de datos en Python

Como Python es un lenguaje de alto nivel, también ofrece varios tipos de datos básicos que se utilizan el desarrollo de aplicaciones.

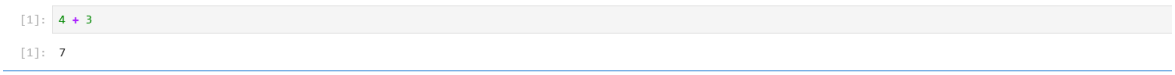
(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

De forma muy genérica, al ejecutarse un programa en Python, simplemente se realizan operaciones sobre objetos,

Estos dos términos son fundamentales:

- **Objetos:** cualquier tipo de datos (números, caracteres o datos más complejos)
- **Operaciones:** como manipulamos estos datos

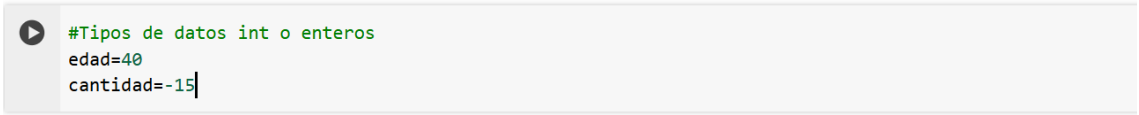
Figura 10.
Operaciones



Nota. La imagen muestra la operación entre objetos. *Fuente*: El investigador.

Tipos de datos entero (int): Estos tipos de datos se representan números enteros, positivos y negativos.

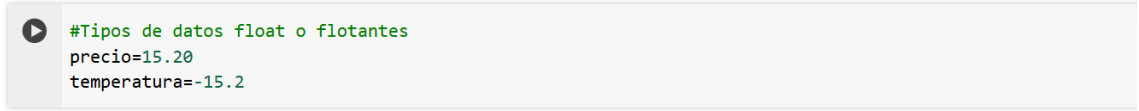
Figura 11.
Datos de tipo int



Nota. La imagen muestra cómo se declara una variable de tipo entero. *Fuente*: El investigador.

Tipos de datos Flotantes (float): Este tipo de datos representa a números decimales tanto positivos y negativos.

Figura 12.
Tipos de datos float



Nota. La imagen muestra los tipos de datos flotantes. *Fuente*: El investigador.

Tipos de datos cadenas de texto (str): Este tipo de datos representan secuencias de caracteres de tipo texto. No nos olvidemos que para declarar un variable de tipo String el valor de la variable puede estar en comillas simples o comillas dobles.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 13
Tipos de datos String

```
#Tipos de datos String o texto
mensaje="Hola desde Python"
nombre='Elvis'
```

Nota. La imagen muestra los tipos de datos String. *Fuente*: El investigador.

Tipos de datos booleanos: Este tipo de datos devuelven dos opciones True o False:

Figura 14.
Datos booleanos

```
#Tipos de datos booleanos
es_par= True
aprobado=False
```

Nota. La imagen muestra los tipos de datos booleanos en Python. *Fuente*: El investigador.

A diferencia de otros lenguajes de programación en Python todas las variables son objetos o hacen referencias a objetos, las variables y los objetos se almacena en diferentes zonas de memoria las variables siempre hacen referencia a objetos y nunca a otra variable.

A diferencia los objetos si pueden referenciar a otros objetos. Ejemplo las listas. También hay que recordar que las variables no tienen tipo, las variables apuntan a objetos que sí lo tienen, dado que Python es un lenguaje de tipado dinámico, la misma variable puede apuntar, en momentos diferentes de la ejecución del programa, a los objetos de diferente tipo.

Figura 15.
Tipos de referencia de objetos

```
a = 3
print(a)
print(type(a))

a = 'Pablo García'
print(a)
print(type(a))

a = 4.5
print(a)
print(type(a))
```

```
3
<class 'int'>
Pablo García
<class 'str'>
4.5
<class 'float'>
```

Nota. La imagen muestra como se referencia las variables a un objeto y como un objeto puede referenciar a otro objeto. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

No olvidar que cuando un objeto deja de estar referenciado, este se elimina automáticamente a este proceso se le denomina Garbage collection, y ese espacio de memoria queda libre para después ser ocupado por otro objeto.

Cuando creamos una variable el nombre debe seguir ciertas reglas que no están escritas, pero es muy importante para un programador.

- No se puede poner números delante del nombre de las variables.
- Por convención, evitar CamelCase, en Python es mejor usar snake_case el uso de "_" para separar palabras.
- El lenguaje diferencia entre mayúsculas y minúsculas.
- Deben ser descriptivos.
- Hay palabras o métodos reservados que no se pueden usar como nombres de variables o métodos.

Figura 16.
Error asignar un método a una variable

Nota. La imagen muestra un error al asignar el nombre de una variable con una palabra reservada de python como pow. *Fuente*: El investigador.

Al igual que otros lenguajes de programación en Python podemos convertir un tipo de objeto a otro tipo de objeto.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 17.
Conversión de tipo de objetos

```
a = int(2.8)    # a será 2
b = int("3")    # b será 3
c = float(1)    # c será 1.0
d = float("3")  # d será 3.0
e = str(2)      # e será '2'
f = str(3.85)    # f será '3.0'
g = bool("a")   # g será True
h = bool("")    # h será False
i = bool(3)     # i será True
j = bool(0)     # j será False
k = bool(None)

print(a)
print(type(a))
print(b)
print(type(b))
print(c)
print(type(c))
print(d)
print(type(d))
print(e)
print(type(e))
print(f)
print(type(f))
print(g)
print(type(g))
print(h)
print(type(h))
print(i)
print(type(i))
print(j)
print(type(j))
print(k)
```

```
2
<class 'int'>
3
<class 'int'>
1.0
<class 'float'>
3.0
```

Nota. La imagen muestra como convertir un tipo de objeto a otro tipo de objeto y la salida de pantalla. *Fuente*: El investigador.

Tipos de datos Listas: Este tipo de datos son colecciones ordenadas y mutables de elementos.

Figura 18.
Listas en Python

```
#Tipos de datos listas
numeros=[1 ,2, 3, 4, 5]
frutas = ["manzana", "banana", "naranja"]
```

Nota. La imagen nos muestra cómo se declara y representa listas en Python. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Tuplas (tuple): A diferencia de las listas las tuplas son colecciones de datos inmutables.

Figura 19.
Tuplas en Python

```
#Tipos de datos tuplas
coordenadas=(10.2,5.6)
colores=("rojo", "verde", "amarillo")
```

Nota. La imagen muestra cómo crear una Tupla en Python. *Fuente*: El investigador.

Diccionarios: Este tipo de dato a diferencia de los otros son colecciones en pares eso quiere decir clave-valor.

Figura 20.
Diccionarios en Python

```
#Tipos de datos diccionarios
persona={"nombre":"Norma", "edad":30, "ciudad":"Quito"}
```

Nota. La imagen muestra la creación de diccionarios en Python donde las variables están conformado de pares clave-valor. *Fuente*: El investigador.

Conversión de Tipos de Datos

Python permite convertir datos de un tipo a otro tipo de datos utilizando funciones específicas.

Figura 21.
Convertir datos.

```
#Convesión de datos a otro tipo
# Conversión de string a int
edad_str="30"
edad= int(edad_str)
#Convertir de string a float
precio=float("15.25")
#Convetir un int a string
texto=str(100)
#Convertimos un int a booleano
es_par=bool(1)
```

Nota. La imagen muestra el código para convertir un tipo de dato a otro tipo. *Fuente*: El investigador.

Las variables y tipos de datos son fundamentales en Python ya que permiten almacenar y manipular información de manera eficiente. Conocer como declararlas, utilizarlas es esencial para el desarrollo de programas más complejos, incluyendo aplicaciones de Machine Learning.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Operadores y Expresiones en Python

Los operadores en Python permiten realizar diversas operaciones matemáticas y lógicas sobre los datos. Se combinan con variables y valores para forma expresiones que producen resultados específicos.

Operadores Matemáticos

Los operadores aritméticos se utilizan para realizar operaciones matemáticas básicas.

Tabla 2.
Operadores Aritméticos

Operador	Descripción	Ejemplo
+	Suma	5+2=7
-	Resta	5-7=-2
*	Multiplicación	7*3=21
/	División	9/2=4.5
//	División entera	9//2=4
%	Modulo (resto)	10%3
**	Potencia	2**3=8

Nota. La tabla nos muestra los distintos operadores matemáticos que se utilizan en Python. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 22.
Operadores matemáticos

```
x = 3
y = 2

print('x + y = ', x + y)
print('x - y = ', x - y)
print('x * y = ', x * y)
print('x / y = ', x / y)
print('x // y = ', x // y)
print('x % y = ', x % y)
print('x ** y = ', x ** y)
```

```
x + y =  5
x - y =  1
x * y =  6
x / y =  1.5
x // y =  1
x % y =  1
x ** y =  9
```

Nota. La imagen muestra los tipos de operadores aritméticos en Python. *Fuente*: El investigador.

Operadores de Comparación.

Estos operadores lo que haces es comparar entre dos valore y lo que te devuelve es un valor booleano sea True o False.

Tabla 3.
Operadores de Comparación

Operador	Descripción	Ejemplo
==	Igualdad	3 == 3 True
¡=	Diferente	3 != 5 True
>	Mayor que	7 > 3 True
<	Menor que	3 < 7 True
>=	Mayor o igual que	6 >= 6 True
<=	Menor o igual que	2 <= 4 True

Nota. La tabla representa los operadores de comparación dentro de Python con sus respectivos ejemplos.

Fuente: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 23.
Operadores de comparación

```
x = 10
y = 12

print('x > y  es ', x > y)
print('x < y  es ', x < y)
print('x == y es ', x == y)
print('x != y es ', x != y)
print('x >= y es ', x >= y)
print('x <= y es ', x <= y)
```

```
x > y  es  False
x < y  es  True
x == y es  False
x != y es  True
x >= y es  False
x <= y es  True
```

Nota. La imagen muestra el código como se utilizan los operadores de comparación en Python. *Fuente*: El investigador.

Operadores Lógicos

Estos operadores nos permiten combinar operaciones o condiciones lógicas.

Tabla 4.
Operadores Lógicos

Operador	Descripción	Ejemplo
and	Devuelve True si ambas condiciones son verdaderas	(5>2) and (10>5) True
or	Devuelve True si al menos una de las condiciones es verdadera	(5>2) or (10<5) True
not	Invierte el resultado booleano	Nota(5 > 2)

Nota. La tabla representa todos los operadores lógicos que se utilizan en Python. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 24.
Operadores lógicos

```
[10]: x = True
      y = False

      print('x and y es :', x and y)
      print('x or y es  :', x or y)
      print('x xor y es  :', x ^ y)
      print('not x es   :', not y)

      x and y es : False
      x or y es  : True
      x xor y es  : True
      not x es   : True
```

Nota. La imagen muestra el código de cómo se utiliza los operadores lógicos en Python. *Fuente*: El investigador.

Operadores de Asignación

Estos operadores asignan valores o variables y permiten modificar su valor existente.

Tabla 5.
Operadores de Asignación

Operador	Ejemplo	Equivalente a
=	x=5	x=5
+=	x+=3	x=x+3
-=	x-=2	x=x-2
*=	x*=4	x=x*4
/=	x/=2	x=x/2
%=	x%=3	x=x%3
**=	x **=2	x=x**2
&=		And y asignación
|=		Or y asignación
^=		Xor y asignación

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

>>=		Desplazamiento Derecha y asignación
//=	x//=3	x=x//3
<<=		Desplazamiento izquierda y asignación

Nota. La tabla nos muestra todos los operadores de asignación que se pueden usar en Python. *Fuente*: El investigador.

Figura 25.
Operadores de asignación

```
a = 5
a *= 3   # a = a * 3
a += 1   # No existe a++, ni ++a,a--,--a
print(a)

b = 6
b -= 2   # b = b - 2
print(b)
```

```
16
4
```

Nota. La imagen muestra como realizar las operaciones de asignación en Python. *Fuente*: El investigador.

Los operadores y expresiones en Python permiten realizar cálculos, comparaciones y evaluaciones lógicas dentro de los programas. Su dominio es esencial para desarrollar scripts más avanzados y aplicar técnicas de Machine Learning de manera efectiva.

Operadores de Identidad

Los operadores de identidad en Python son estructuras fundamentales que permiten determinar si dos variables hacen referencia exactamente al mismo objeto en la memoria, en lugar de simplemente comparar si sus valores son iguales. Estos operadores, representados por las palabras clave is y is not, permiten al programador verificar la identidad real de los objetos, lo cual es especialmente útil cuando se trabaja con estructuras de datos, objetos complejos o valores nulos como None. A diferencia de los operadores de comparación, que evalúan la igualdad de contenido, los operadores de identidad analizan la ubicación del objeto en la memoria, proporcionando un nivel más preciso de verificación. Este concepto es importante en el desarrollo de aplicaciones en Python, particularmente en el ámbito del Machine Learning, donde es necesario controlar referencias a objetos, validar estados de variables y optimizar el uso de recursos. Comprender el uso correcto de

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

estos operadores permite desarrollar programas más eficientes, seguros y coherentes en el manejo de datos y estructuras internas del sistema.

Figura 26.
Operadores de identidad

Operador	Desc
a is b	True, si ambos operadores son una referencia al mismo objeto
a is not b	True, si ambos operadores *no* son una referencia al mismo objeto

Nota. La imagen muestra cual es la salida de pantalla cuando se utiliza los operadores de identidad. Fuente propia. *Fuente*: El investigador.

Figura 27.
Operadores de identidad

```
a = 4444
b = a
print(a is b)
print(a is not b)
```

```
True
False
```

Nota. La imagen muestra cómo usar los operadores de identidad en Python. *Fuente*: El investigador.

Operadores de pertenencia

Los operadores de pertenencia en Python son elementos que permiten verificar si un valor específico se encuentra dentro de una secuencia o colección de datos, como listas, tuplas, cadenas de texto, conjuntos o diccionarios. Estos operadores están representados por las palabras clave in y not in, y su función principal es evaluar la presencia o ausencia de un elemento dentro de una estructura de datos determinada, devolviendo como resultado un valor booleano, es decir, True o False. Su uso es especialmente relevante cuando se requiere validar datos, filtrar información o tomar decisiones dentro de un programa, lo cual es común en el desarrollo de aplicaciones y en el análisis de datos en Machine Learning. Por ejemplo, permiten comprobar si una característica, etiqueta o valor existe dentro de un conjunto de datos antes de procesarlo. Comprender estos operadores facilita el manejo eficiente de colecciones y contribuye a la creación de programas más claros, organizados y funcionales en Python.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 28.
Operadores de pertenencia

Operador	Desc
a in b	True, si *a* se encuentra en la secuencia *b*
a not in b	True, si *a* no se encuentra en la secuencia *b*

Nota. La imagen nos muestra los operadores de pertenencia que se utiliza en Python. *Fuente*: El investigador.

Figura 29.
Implementación operadores de pertenencia

```
x = 'Hola Mundo'
y = {1:'a',2:'b'}

print('H' in x)
print('hola' not in x)

print(1 in y)
print('a' in y)
```

```
True
True
True
False
```

Nota. La imagen muestra como implementar el código de operadores de pertenencia en Python. *Fuente*: El investigador.

Entrada de valores

La entrada de valores por teclado en Python es un mecanismo que permite al programa interactuar directamente con el usuario, facilitando el ingreso de datos durante su ejecución. Esta funcionalidad se realiza mediante la función input(), la cual permite capturar información ingresada desde el teclado y almacenarla en una variable para su posterior procesamiento. Los datos ingresados son interpretados inicialmente como texto, por lo que, en caso de requerir valores numéricos, es necesario realizar una conversión utilizando funciones como int() o float(). Esta característica es fundamental en el desarrollo de aplicaciones interactivas, ya que permite personalizar el comportamiento del programa en función de la información proporcionada por el usuario. En el contexto del Machine Learning, la entrada de datos puede utilizarse para definir parámetros, seleccionar opciones o ingresar valores que serán utilizados en el análisis y procesamiento de la información.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 30.
Ingreso datos por teclado

```
valor = input("Inserte valor:")
print(valor)
print(type(valor))
```

```
Inserte valor: hola
hola
<class 'str'>
```

```
grados_c = int(input("Conversión de grados a fahrenheit, inserte un valor: "))

print(f"Grados F: {1.8 * (grados_c) + 32}")
```

```
Conversión de grados a fahrenheit, inserte un valor:  15
Grados F: 59.0
```

Nota. La imagen muestra la implementación en código como ingresar datos por teclado en Python tanto tipo String, como objetos de tipo numérico. *Fuente*: El investigador.

Tipos de datos compuestos

Listas

Las listas en Python son estructuras de datos compuestas que permiten almacenar múltiples elementos dentro de una sola variable, organizados de forma secuencial y ordenada. Estas colecciones se caracterizan por ser dinámicas y modificables, lo que significa que es posible agregar, eliminar o modificar sus elementos durante la ejecución del programa. Las listas pueden contener diferentes tipos de datos, como números, cadenas de texto o incluso otros objetos, lo que las convierte en una herramienta flexible para representar conjuntos de información. Su uso es fundamental en el desarrollo de aplicaciones y en el análisis de datos, especialmente en Machine Learning, donde se emplean para almacenar conjuntos de datos, resultados intermedios o características de los modelos. Comprender el funcionamiento de las listas permite gestionar la información de manera eficiente y facilita la implementación de algoritmos y soluciones basadas en el procesamiento de datos.

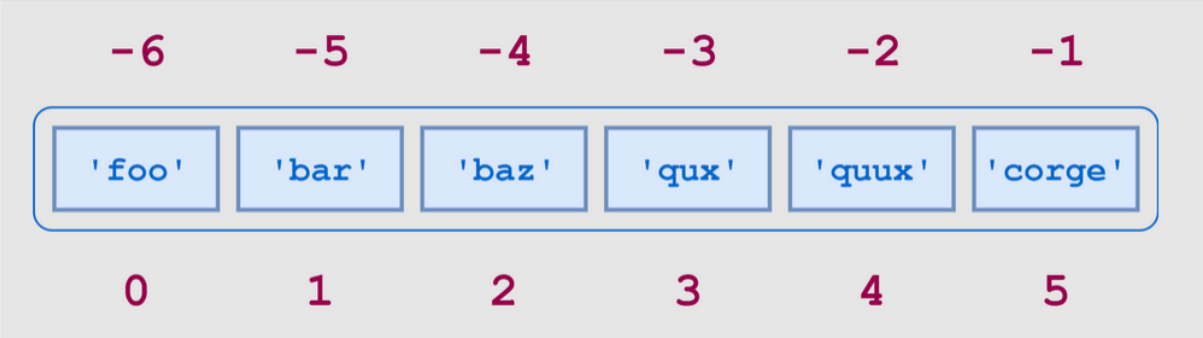
Algunas de las características de las listas son:

- Una colección de objetos
- Mutables
- Tipos arbitrarios heterogéneos
- Puede contener duplicados
- No tiene tamaño fijo. Pueden contener tantos elementos como quepan en memoria.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

- Los elementos van ordenados por posición
- Los índices van de 0 a n-1, donde n es el número de la lista
- Son un tipo de Secuencia, al igual que los string: por lo tanto, el orden es importante
- Soportan anidamiento
- Son una implementación de tipo abstracto de datos quiere decir un Array dinámico.

Figura 31.
Ejemplo de una lista



Nota. La imagen muestra cómo se crea una lista y el índice de cada uno de los elementos de la lista. *Fuente*: El investigador.

Operaciones con listas

Creación de listas

La creación de listas en Python es un proceso que permite definir estructuras de datos compuestas capaces de almacenar múltiples elementos dentro de una sola variable, organizados de forma ordenada y accesible mediante índices. Estas listas se crean utilizando corchetes [], dentro de los cuales se pueden incluir diferentes tipos de datos, como números, cadenas de texto o incluso otras listas, lo que proporciona una gran flexibilidad en la representación de la información. Una de sus principales características es que son estructuras dinámicas, lo que significa que pueden modificarse durante la ejecución del programa, permitiendo agregar, eliminar o actualizar elementos según las necesidades del sistema. La creación de listas es fundamental en el desarrollo de programas y aplicaciones orientadas al análisis de datos, ya que facilita el almacenamiento y la manipulación de conjuntos de información. En el ámbito del Machine Learning, las listas se utilizan para gestionar datos de entrada, resultados intermedios y estructuras necesarias para el entrenamiento y evaluación de modelos predictivos.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 32
Creación de listas

```
letras = ['a', 'b', 'c', 'd']
palabras = 'Hola mundo como estas'.split()
numeros = list(range(7))

print(letras)
print(palabras)
print(numeros)
print(type(numeros))
```

```
['a', 'b', 'c', 'd']
['Hola', 'mundo', 'como', 'estas']
[0, 1, 2, 3, 4, 5, 6]
<class 'list'>
```

```
# # Pueden contener elementos arbitrarios / heterogeneos
mezcla = [1, 3.4, 'a', None, False]
print(mezcla)
print(len(mezcla)) # len me da el tamaño de la lista número de elementos
```

```
[1, 3.4, 'a', None, False]
5
```

```
# # Pueden incluso contener objetos más "complejos"
lista_con_funcion = [1, 2, len, pow]
print(lista_con_funcion)
```

```
[1, 2, <built-in function len>, <function pow at 0x0000013EEAB2CCA0>]
```

```
# # Pueden contener duplicados
lista_con_duplicados = [1, 2, 3, 3, 3, 4]
print(lista_con_duplicados)
```

```
[1, 2, 3, 3, 3, 4]
```

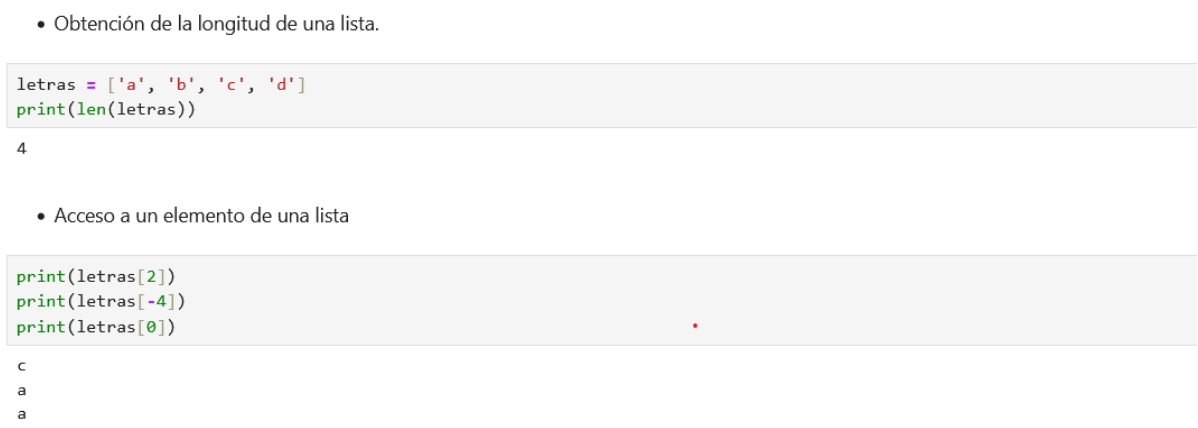
Nota. En el código vemos la implementación como crear una lista en Python. *Fuente*: El investigador.

Obtención de la longitud de una lista y acceso a un elemento de una lista

La obtención de la longitud de una lista y el acceso a sus elementos son operaciones fundamentales en Python que permiten gestionar y manipular colecciones de datos de manera eficiente. La longitud de una lista se determina mediante la función len(), la cual devuelve el número total de elementos almacenados en la estructura, facilitando el control y la validación de la información contenida. Por otro lado, el acceso a un elemento específico se realiza utilizando su índice, el cual representa la posición que ocupa dentro de la lista, comenzando desde el valor cero. Esta característica permite recuperar, analizar o modificar datos de forma precisa según su ubicación. Estas operaciones son esenciales en el desarrollo de programas, especialmente en aplicaciones relacionadas con Machine Learning, donde es necesario recorrer conjuntos de datos, verificar su tamaño y acceder a características específicas para su procesamiento y análisis.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 33.
longitud y acceso a una lista



Nota. La imagen muestra la implementación del código para obtener la longitud de una lista y acceder a un elemento de una lista. *Fuente*: El investigador.

Slincing

El slicing en Python es una técnica que permite extraer una porción o subconjunto de elementos de una lista u otra estructura de datos secuencial, utilizando un rango específico de índices. Este proceso se realiza mediante el uso de corchetes [] junto con dos puntos :, donde se define el índice inicial y el índice final del segmento que se desea obtener. El slicing facilita el acceso parcial a la información sin necesidad de modificar la estructura original, lo que resulta especialmente útil cuando se requiere analizar, dividir o procesar subconjuntos de datos. Esta funcionalidad es ampliamente utilizada en el desarrollo de programas y en el análisis de datos, ya que permite manipular colecciones de manera flexible y eficiente. En el contexto del Machine Learning, el slicing es fundamental para separar conjuntos de entrenamiento y prueba, seleccionar características específicas o trabajar con segmentos determinados de un conjunto de datos.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 34.
Slincing

Nota. La imagen muestra la implementación del concepto de slincing en Python que permite obtener una fracción de una lista. *Fuente*: El investigador.

Añadir un elemento al final de una lista

Añadir un elemento al final de una lista en Python es una operación fundamental que permite ampliar dinámicamente una colección de datos durante la ejecución de un programa. Esta acción se realiza mediante el método append(), el cual permite incorporar un nuevo elemento al final de la lista sin alterar el orden de los elementos existentes. Esta característica es especialmente útil cuando se requiere almacenar datos que se generan en tiempo real, como resultados de cálculos, entradas del usuario o valores obtenidos de un proceso de análisis. La capacidad de agregar elementos de forma progresiva convierte a las listas en estructuras flexibles y adaptables a diferentes necesidades de programación. En el contexto del Machine Learning, esta operación es importante para construir conjuntos de datos, almacenar predicciones o registrar resultados intermedios durante el entrenamiento y evaluación de modelos, facilitando una gestión eficiente de la información.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 35.
Añadir un elemento al final de la lista

Nota. La imagen muestra la implementación el código para añadir un elemento al final de la lista. *Fuente*: El investigador.

Estructuras de Control Condicionales y Bucles

Una sentencia condicional es un esquema de instrucciones que permite elegir uno de entre dos caminos lógicos o uno entre varias opciones lógicos. Estas sentencias condicionales sirven para validar datos, es decir, para verificar que se cumplan determinadas condicionales que sean favorables a los programas que construyamos.

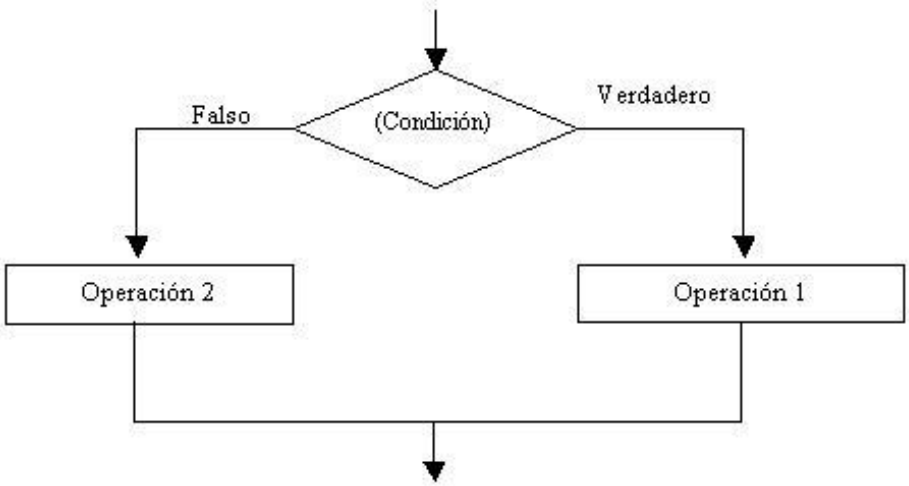
Las estructuras de control permiten modificar el flujo de ejecución de un programa. En Python proporciona sentencias de control condicional (if, elif,else) y blucles (for, while) para repetir acciones según una condición.

Condición if

La condición if permite ejecutar bloques de código según ciertas condiciones.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 36.
Diagrama de flujo if



Nota. La imagen muestra la representación del condicional if. *Fuente*: El investigador.

Estructura básica del if

Figura 37.
Estructura if

```
#Estructura básica if
edad = 18
if edad >=18:
  print ("Eres mayor de edad. ")
```

Nota. La imagen representa el código de la sentencia de control if que verifica si la edad es mayor o igual a 18. *Fuente*: El investigador.

Uso de if, elif y else

Como se conoce el condicional if tiene dos caminos el verdadero y falso en esta sección de código se explica la estructura de las condicionales.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 38.
Código if anidados

```
# uso de la sentencia if, elif y else
calificaciones=7
if calificaciones >=9:
  print("Aprobaste cpon excelencia.")
# caso contrario
elif calificaciones >=70:
  print("Aprobaste...")
#caso contrario
else:
  print("Reprobaste")
```

Nota. La imagen muestra el código de la implementación de un if anidado. *Fuente*: El investigador.

Condiciones combinadas con operadores lógicos

Figura 39.
if operadores lógicos

```
# declaramos una variable de tipo int
edad = 25
# declaramos una variable de tipo string ciudad
ciudad = "Quito"
#realizamos un condicional para verificar si es mayor de eda y vive en Quito
if edad >= 18 and ciudad == "Quito":
  print("Eres mayor de edad y vives en Quito")
```

Nota. La imagen muestra el código de un if con operadores lógicos. *Fuente*: El investigador.

Sentencias Repetitivas

Una sentencia repetitiva es una instrucción que permite que un conjunto de instrucciones se repita la cantidad de veces que se determinó. En Python las dos sentencias repetitivas o iterativas más usadas son la instrucción for y la instrucción while.

Bucle for

Se usa para iterar sobre secuencias como listas, tuplas y cadenas.

Figura 40.
Ciclo for

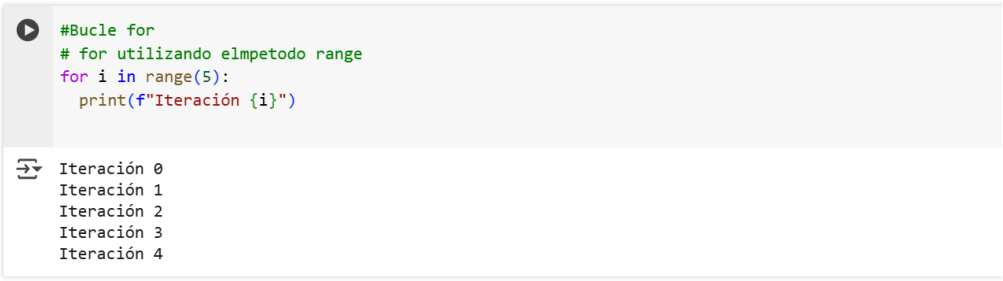
```
#Bucle for
#Creamos una tupla
frutas = ["manzana", "banana", "naranja"]
for fruta in frutas:
  print (f"Me gusta la {fruta}.")
```

Nota. La imagen muestra el código del ciclo for recorriendo una tupla. *Fuente*: El investigador.

Ahora veamos las posibilidades que ofrece el ciclo for de Python cuando se trabaja con rangos de valores específicos definidos por el programador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 41.
range for

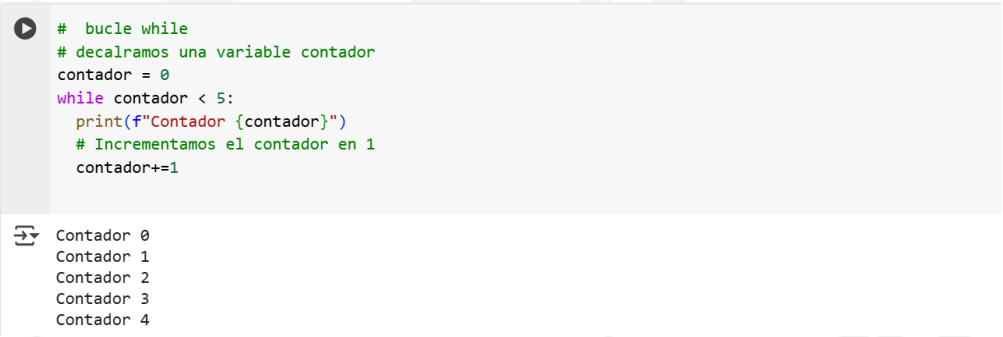


Nota. La imagen muestra el uso de la función range donde se itera el for en un rango determinado por el programador. *Fuente*: El investigador.

Ciclo while

El otro tipo de bucle se implementa a partir del uso del ciclo repetitivo while, esta instrucción permite repetir un conjunto de instrucciones mientras una condición sea **verdadera.**

Figura 42.
Ciclo while



Nota. La imagen muestra el uso del ciclo while en Python y la salida en pantalla. *Fuente*: El investigador.

Las estructuras de control son esenciales para la lógica de programación, Los condicionales permiten tomar decisiones, y los bucles facilitan la ejecución repetitiva del código.

Funciones en Python

Sabes ¿Qué es una función? Te explico una función es un conjunto breve de instrucciones que permiten alcanzar fácilmente un pequeño objetivo. Las funciones son la base para resolver los tres grandes problemas de la programación.

1. La reutilización del código.
2. La simplificación del objetivo.
3. La facilidad para realizar las pruebas en frio.

Con las funciones se hace efectiva la estrategia “Divide y vencerás” que es una manera de hacer que los programas, por complejos que parezcan, sean más fáciles de entender y sobre todo más fáciles de corregir.

A demás las funciones permiten organizar el código en bloques reutilizables, lo que mejora la legibilidad y facilita el mantenimiento del código. Además, Python permite agrupar funciones en módulos para modularizar el desarrollo de programas más complejos.

Las funciones en Python se definen con la palabra **def,** seguida del nombre de la función y paréntesis () el cual pueden contener parámetros.

Figura 43.
Método con Python

```
#  ejeplo de una función
def saludar ():
  print ("Hola, bienvenido al curso de Machine Learning")
saludar()
```

```
Hola, bienvenido al curso de Machine Learning
```

Nota. La imagen muestra el código de la implementación de un método en Python. *Fuente*: El investigador.

Funciones con Parámetros

Las funciones pueden recibir valores cono argumentos para procesar información. Cuando una función retorna un conjunto de valores, lo hace como dice su nombre, devolviendo todo el conjunto de valores para que sean procesados en la función que la invoca.

Los generadore ofrecen la posibilidad de extraer un conjunto de valores para cada valor lo retorna de forma independiente, dejando el proceso en la función en espera, mientras se realiza alguna operación con dicho valor especifico. Esto permite resolver algo que parecía imposible hasta el momento como es trabajar con un conjunto de datos infinitos (Muñoz Guerrero & Trejos Buriticá, 2021).

Las funciones y módulos en Python permiten escribir código más modular, reutilizable y eficiente.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Autoevaluación 1

1. **¿Qué es Python y cuáles son sus principales características?**

    a. Un lenguaje compilado, de tipado fuerte y sintaxis compleja.

    b. Un lenguaje interpretado, de tipado dinámico y con sintaxis sencilla.

    c. Un software para edición de texto.

    d. Un lenguaje de bajo nivel exclusivo para desarrollo de sistemas operativos

2. **¿Cuál de las siguientes opciones NO es una ventaja de Python?**

    a. Gran cantidad de bibliotecas disponibles.

    b. Código difícil de leer y escribir.

    c. Portabilidad entre diferentes sistemas operativos.

    d. Sintaxis sencilla y clara

3. **¿Cuál de los comandos utilizados para instalar una biblioteca en Python con Anaconda?**

    a. pip install nombre-biblioteca

    b. conda install nombre_biblioteca

    c. python install nombre_biblioteca

    d. install nombre_biblioteca

4. **¿Cuál de los siguientes es un tipo de dato en Python?**

    a. Texto

    b. Número real

    c. Booleano

    d. Todas las anteriores

5. **¿Cuál de las siguientes expresiones devuelve True en Python?**

    a. 10 == “10”

    b. 5 >= 3 and 10 < 20

    c. “hola” + 5

    d. None or False

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

6. **¿Cómo se define una función en Python?**

    a. function mi_funcion():

    b. def mi_funcion():

    c. define mi_funcion():

    d. créate mi_funcion():

7. **¿Cuál de los siguientes operadores se utiliza para la división entera en Python?**

    a. /.

    b. //.

    c. %.

    d. **.

8. **¿Qué estructura de control permite repetir un bloque de código mientras se cumpla una condición?**

    a. If-else.

    b. For.

    c. While.

    d. Switch-case.

9. **¿Qué palabra clave se utiliza para importar un módulo en Python?**

    a. import.

    b. Include.

    c. Require.

    d. Use.

10. **¿Cuál de las siguientes afirmaciones sobre las funciones en Python es correcto?**

    a. Todas las funciones en Python deben devolver un valor obligatorio.

    b. Se pueden definir funciones con parámetros opcionales.

    c. Python no permite definir funciones dentro de otras funciones.

    d. Las funciones en Python no pueden recibir argumentos.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Resumen de la Unidad 1

En esta unidad los estudiantes han adquirido una comprensión fundamental del lenguaje de programación Python, desde sus conceptos básicos hasta su aplicación en la estructuración de programas eficientes.

Se comenzó con una introducción al lenguaje, destacando su simplicidad, legibilidad y versatilidad en el desarrollo de aplicaciones, especialmente en el campo de Machine Learning. Se abordo la instalación del entorno de trabajo recomendando Anaconda como herramienta clave para la gestión de bibliotecas y entornos virtuales.

Luego se exploró la sintaxis y estructura de los programas en Python, destacando la importancia de la indentación y el uso de comentarios para mejorar la claridad del código. Se revisaron los tipos de datos, variables explicando su uso y conversión en distintos escenarios prácticos.

También se analizaron los operadores y expresiones, incluyendo aritméticos, lógicos y de comparación, fundamentales para la construcción de estructuras de control. En este sentido, se estudiaron detalladamente los condicionales if, elif, else y los bucles for, while, mostrando su importancia en la automatización de tareas y tomas de decisiones dentro de un programa.

Finalmente, se abordó el concepto de funciones y módulos, enseñando a los estudiantes a definir funciones reutilizables y organizar código en módulos separados para mejorar su mantenimiento y escalabilidad.

Esta unidad ha sentado las bases necesarias para el desarrollo de habilidades en programación estructurada preparando a los estudiantes para aplicar estos conocimientos en problemas más complejos dentro del contexto de Machine Learning.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

UNIDAD 2 APRENDIZA AUTOMÁTICO

Temas y Subtemas

Introducción al Aprendizaje Automático

- Definición y origén del aprendizaje automático.
- Diferencia entre inteligencia artificial, aprendizaje automático y aprendizaje profundo.
- Importancia y aplicaciones del aprendizaje automático en la actualidad.

Tema 2 Tipos de Aprendizaje Automático

1. Conceptos y caracteristicas
2. Ejemplos de algoritmos supervisados: regresión lineal, regresión logistica

Tema 3 Herramientas y Bibliotecas para Machine Learning

1. Introducción a los entornos de desarrolo para MAchine Learning
2. Python como lenguaje principal para ML
3. Bibliotecas esenciales para el analis de datos

El aprendizaje automático o Machine Learning es una rama de la inteligencia artificial que permite a las computadoras aprender de los datos sin ser programadas explícitamente. A través de algoritmos y modelos matemáticos, los sistemas pueden identificar patrones, realizar predicciones y mejorar su desempeño con la experiencia.

El objetivo principal del aprendizaje automático es desarrollar modelos que permitan la automatización del análisis de datos, facilitando la toma de decisiones en distintos campos como la salud, las finanzas, el marketing y la industria. (Bagnato, 2020).

El Machine Learning -traducido al español como Aprendizaje Automático- es un subcampo de la Inteligencia Artificial que busca resolver el “cómo construir programas de computadora que mejoran automáticamente adquiriendo experiencia”. Esta definición indica que el programa que se crea con ML no necesita que el programador indique explícitamente las reglas que debe seguir para

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

lograr su tarea si no que esta mejora automáticamente. Grandes volúmenes de datos están surgiendo de diversas fuentes en los últimos años y el Aprendizaje Automático relacionado al campo estadístico consiste en extraer y reconocer patrones y tendencias para comprender qué nos dicen los datos. Para ello, se vale de algoritmos que pueden procesar Gygas y/o Terabytes y obtener información útil (Bagnato, 2020).

Diagrama de Venn

Drew Conway creó un simpático diagrama de Venn en el que inerrelaciona diversos campos. Aquí copio su versión al español traducida por mi:

Figura 44.
Diagrama de Venn

Nota. En esta aproximación al Machine Learning. Podemos ver que hay una intersección entre conocimientos de matemáticas y estadística con habilidades de hackeo del programador. *Fuente*: El investigador.

Una mención distintiva merece las RNAs ya que son algoritmos que utilizan un comportamiento similar a las neuronas humanas y su capacidad de sinopsis para la obtención de resultados, interrelacionándose diversas capas de neuronas para darle mayor poder. Aunque estos códigos existen desde hace más de 70 años, en la última década han evolucionado notoriamente (en paralelo a la mayor capacidad tecnológica de procesamiento, memoria RAM y disco, la nube, etc.) y están logrando impresionantes resultados para analizar textos y síntesis de voz, traducción de

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

idiomas, procesamiento de lenguaje natural, visión artificial, análisis de riesgo, clasificación y predicción y la creación de motores de recomendación.

Diferencias entre inteligencia Artificial, Aprendizaje Automático y Aprendizaje Profundo

La Inteligencia Artificial es un campo de la información que busca desarrollar sistemas capaces de realizar tareas que normalmente requieren inteligencia humana, como el reconocimiento de voz, la toma de decisiones y la resolución de problemas. La IA abraca múltiples enfoques incluyendo:

- Sistemas expertos, que utilizan reglas programadas para tomar decisiones.
- Lógica difusa, que maneja incertidumbre en la toma de decisiones.
- Algoritmos evolutivos, que se inspiran en la evolución biológica para optimizar soluciones.

Ejemplos de IA incluye asistentes virtuales como Siri o Alexa, chatbots, reconocimiento facial y sistemas de automatización empresarial.

Aprendizaje Automático (Machine Elearning)

EL aprendizaje automático ML es un subconjunto de la IA que permite a las computadoras aprender patrones a partir de datos sin ser programadas explícitamente. Se basa en algoritmos que ajustan sus parámetros a medida que procesan información, mejorando su desempeño con la experiencia.

Tipos de algoritmos en ML:

- Regresión Lineal y logística, para hacer predicciones numéricas o de clasificación.
- Árboles de decisión y Random Forest, utilizados en problemas de clasificación y regresión.
- Máquinas de soporte vectorial SVM, este tipo de algoritmos se utilizan en clasificación de datos complejos.

Aplicaciones incluyen detección de fraudes, predicción de ventas, diagnósticos médicos y análisis de sentimientos en redes sociales.

Aprendizaje Profundo (Deep Learning, DL)

El aprendizaje profundo es un subconjunto del ML que utiliza rede neuronales profundas con múltiples capas para procesar datos de manera jerárquica. Este enfoque es particularmente útil para el reconocimiento de patrones se son complejos:

Las características principales de este algoritmo son:

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

- Se base en redes neuronales artificiales
- Incluye redes neuronales convolucionales para visión artificial y redes neuronales recurrentes.
- Requiere grandes cantidades de datos y recursos computacionales avanzados.

Algunos ejemplos incluyen Google Translate, reconocimiento facial en redes sociales y asistentes virtuales.

Importancia y aplicaciones del Aprendizaje Automático en la Actualidad

El aprendizaje automático ha adquirido una relevancia sin precedentes en la era digital debido a su capacidad para analizar grandes volúmenes de datos y extraer patrones útiles para la toma de decisiones. Algunas de sus principales ventajas incluyen.

- Automatización de procesos: Reduce la intervención humana en tareas repetitivas, aumentando la eficiencia y reduciendo costos.
- Mejora en la toma de decisiones: Optimiza la experiencia del usuario en plataformas digitales mediante recomendaciones personalizadas.
- Avances en la investigación científica, facilita el descubrimiento de nuevos tratamientos en la medicina y en la identificación de patrones en la biología y otras ciencias.
- Seguridad y detección de fraudes, se utiliza ampliamente en la banca y en la ciberseguridad para identificar actividades sospechosas y prevenir fraudes.

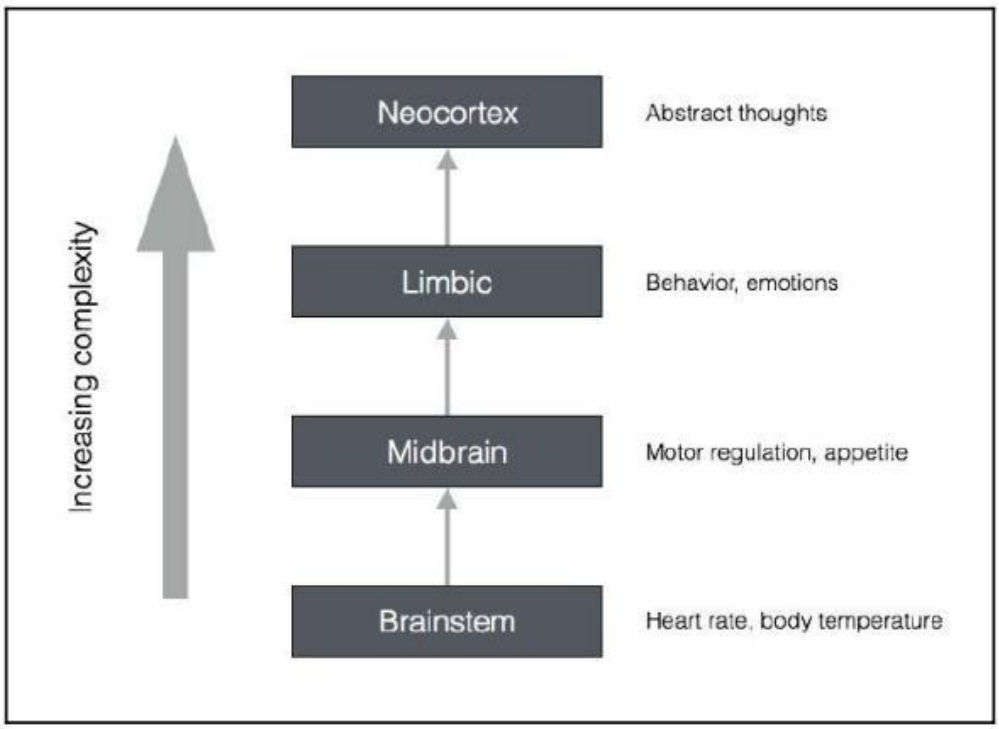
¿Por qué necesitamos estudiar IA?

La IA tiene la capacidad de impactar cada aspecto de nuestras vidas. El campo de la IA trata de comprender los patrones y comportamientos de las entidades. Con IA, queremos construir sistemas inteligentes y comprender también el concepto de inteligencia. Los sistemas inteligentes que construimos son muy útiles para comprender cómo un sistema inteligente como nuestro cerebro construye otro sistema inteligente.

Echemos un vistazo a cómo nuestro cerebro procesa la información:

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 45.
Procesos del cerebro



Nota. La imagen representa los procesos para procesar la información. *Fuente*: El investigador.

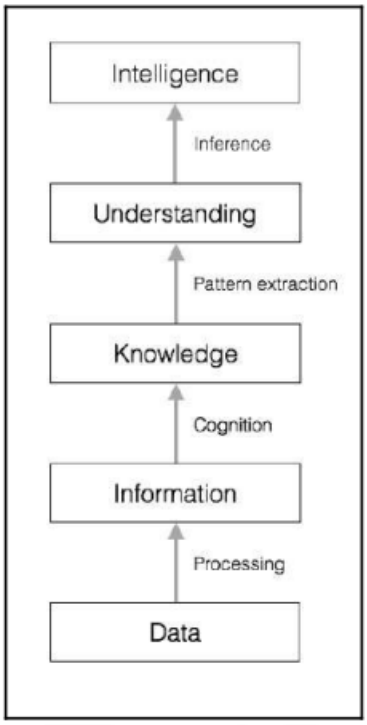
En comparación con otros campos, como las Matemáticas o la Física, que han existido durante siglos, la IA está relativamente en su infancia. Durante las últimas dos décadas, la IA ha producido algunos productos espectaculares, como automóviles autónomos y robots inteligentes que pueden caminar. Según la dirección en la que nos dirigimos, es bastante obvio que lograr la inteligencia tendrá un gran impacto en nuestras vidas en los próximos años.

No podemos dejar de preguntarnos cómo el cerebro humano se las arregla para hacer tanto con tanta facilidad sin esfuerzo. Podemos reconocer objetos, comprender idiomas, aprender cosas nuevas y realizar muchas tareas más sofisticadas con nuestro cerebro. ¿Cómo hace esto el cerebro humano? Cuando intentes hacer esto con una máquina, ¡verás que se queda muy atrás! Por ejemplo, cuando tratamos de buscar cosas como vida extraterrestre o viajes en el tiempo, no sabemos si esas cosas existen. Lo bueno del santo grial de la IA es que sabemos que existe. ¡Nuestro cerebro es el santo grial! Es un ejemplo espectacular de un sistema inteligente. Todo lo que tenemos que hacer es imitar su funcionalidad para crear un sistema inteligente que pueda hacer algo similar, posiblemente incluso más.

Veamos cómo los datos sin procesar se convierten en sabiduría a través de varios niveles de procesamiento:

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 46.
Como se obtiene inteligencia



Nota. La imagen muestra como es cerebro convierta los datos en inteligencia. *Fuente*: El investigador.

Una de las principales razones por las que queremos estudiar IA es para automatizar muchas cosas. Vivimos en un mundo donde:

- Trabajamos con enormes e insuperables cantidades de datos. El cerebro humano no puede realizar un seguimiento de tantos datos.
- Los datos se originan de múltiples fuentes simultáneamente.
- Los datos son desorganizados y caóticos.
- El conocimiento derivado de estos datos debe actualizarse constantemente porque los datos mismos siguen cambiando.
- La detección y la actuación tienen que ocurrir en tiempo real con alta precisión.

Aunque el cerebro humano es excelente para analizar las cosas que nos rodean, no puede seguir el ritmo de las condiciones anteriores. Por lo tanto, necesitamos diseñar y desarrollar máquinas inteligentes que puedan hacer esto. Necesitamos sistemas de IA que puedan:

- Manejar grandes cantidades de datos de manera eficiente. Con la llegada de la computación en la nube, ahora podemos almacenar grandes cantidades de datos.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

- Ingiera datos simultáneamente desde múltiples fuentes sin ningún retraso. Indexe y organice los datos de una manera que nos permita obtener información.
- Aprenda de nuevos datos y actualícese constantemente utilizando los algoritmos de aprendizaje adecuados.
- Piense y responda a situaciones basadas en las condiciones en tiempo real.

Las técnicas de IA se están utilizando activamente para hacer que las máquinas existentes sean más inteligentes, de modo que puedan ejecutarse de manera más rápida y eficiente.

Clasificación y regresión Uso del aprendizaje supervisado

Aprendizaje supervisado vs. no supervisado

Una de las formas más comunes de incorporar inteligencia artificial a una máquina es mediante el aprendizaje automático. El mundo del aprendizaje automático se divide, a grandes rasgos, en aprendizaje supervisado y no supervisado. Existen otras divisiones, pero las analizaremos más adelante.

Aprendizaje Supervisado

El aprendizaje supervisado se refiere al proceso de construir un modelo de aprendizaje automático basado en datos de entrenamiento etiquetados. Por ejemplo, supongamos que queremos construir un sistema para predecir automáticamente los ingresos de una persona, basándose en diversos parámetros como la edad, la educación, la ubicación, etc. Para ello, necesitamos crear una base de datos de personas con todos los detalles necesarios y etiquetarla. De esta forma, le indicamos a nuestro algoritmo qué parámetros corresponden a qué ingresos. Con base en esta asignación, el algoritmo aprenderá a calcular los ingresos de una persona utilizando los parámetros que se le proporcionen.

Aprendizaje No Supervisado

El aprendizaje no supervisado se refiere al proceso de construir un modelo de aprendizaje automático sin depender de datos de entrenamiento etiquetados. En cierto sentido, es lo opuesto a lo que acabamos de comentar en el párrafo anterior. Dado que no hay etiquetas disponibles, es necesario extraer información basándose únicamente en los datos proporcionados. Por ejemplo, supongamos que queremos crear un sistema en el que debemos separar un conjunto de puntos de datos en varios grupos. El problema radica en que no sabemos exactamente cuáles deberían ser los criterios de separación. Por lo tanto, un algoritmo de aprendizaje no supervisado debe separar el conjunto de datos en varios grupos de la mejor manera posible.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

¿Qué es la clasificación?

El proceso de clasificación es una de estas técnicas, en la que clasificamos los datos en un número determinado de clases. Durante la clasificación, organizamos los datos en un número fijo de categorías para que se puedan utilizar de la forma más eficaz y eficiente.

En el aprendizaje automático, la clasificación resuelve el problema de identificar la categoría a la que pertenece un nuevo punto de datos. Construimos el modelo de clasificación basándonos en el conjunto de datos de entrenamiento que contiene los puntos de datos y las etiquetas correspondientes. Por ejemplo, supongamos que queremos comprobar si la imagen dada contiene el rostro de una persona. Construiríamos un conjunto de datos de entrenamiento que contenga clases correspondientes a estas dos clases: con rostro y sin rostro. Después, entrenaríamos el modelo con base en las muestras de entrenamiento disponibles. Este modelo entrenado se utiliza para la inferencia.

Un buen sistema de clasificación facilita la búsqueda y recuperación de datos. Esto se utiliza ampliamente en reconocimiento facial, identificación de spam, motores de recomendación, etc. Los algoritmos de clasificación de datos establecerán los criterios adecuados para separar los datos dados en el número de clases especificado.

Necesitamos proporcionar un número suficiente de muestras para que el algoritmo pueda generalizar dichos criterios. Si el número de muestras es insuficiente, el algoritmo se sobreajustará a los datos de entrenamiento. Esto significa que no funcionará bien con datos desconocidos porque ajustó demasiado el modelo para que no se ajustara a los patrones observados en los datos de entrenamiento. Este es un problema muy común en el mundo del aprendizaje automático. Es importante considerar este factor al crear diversos modelos de aprendizaje automático.

Regresión Lineal en Python

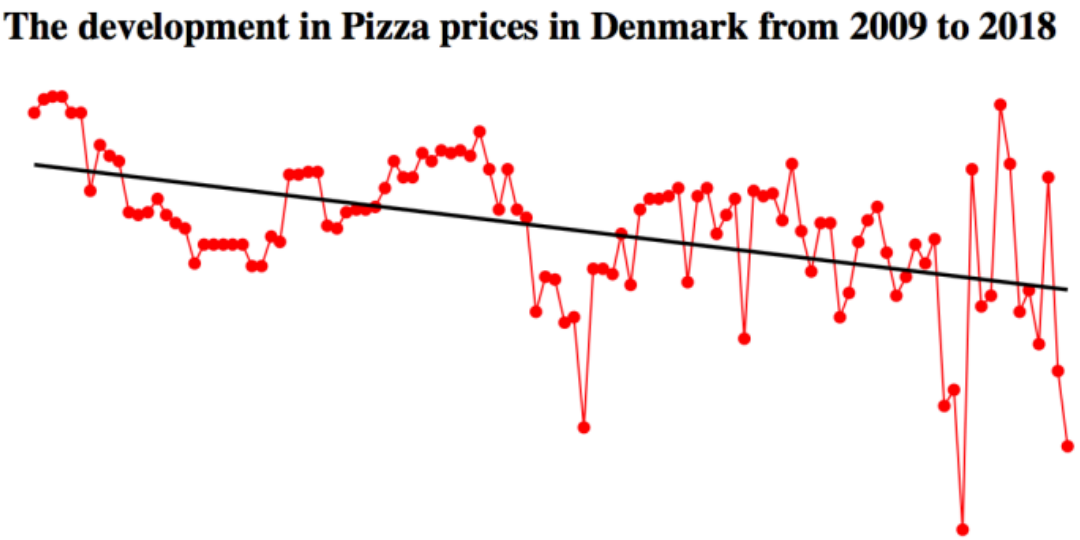
La regresión lineal es un algoritmo de aprendizaje supervisado que se utiliza en Machine Learning y en estadística. En su versión más sencilla, lo que haremos es “dibujar una recta” que nos indicará la tendencia de un conjunto de datos continuos (si fueran discretos, utilizaríamos Regresión Logística). En estadísticas, regresión lineal es una aproximación para modelar la relación entre una variable escalar dependiente “y” y una o más variables explicativas nombradas con “X”. Recordemos rápidamente la fórmula de la recta:

$$y = mX + b$$

Donde Y es el resultado, X es la variable, m la pendiente (o coeficiente) de la recta y b la constante o también conocida como el “punto de corte con el eje Y” en la gráfica (cuando X=0)

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 47.
Precio de la Pizza 2009 al 2018



Nota. Aquí vemos un ejemplo donde vemos datos recabados sobre los precios de las pizzas en Dinamarca (los puntos en rojo) y la línea negra es la tendencia. Esa es la línea de regresión que buscamos que el algoritmo aprenda y calcule solo. *Fuente*: traída https://www.kaggle.com/code/fernandobordi/fb-aprendizaje-autom-tico-regresi-n

¿Cómo funciona el algoritmo de regresión lineal en Machine Learning?

Recordemos que los algoritmos de Machine Learning Supervisados (Bagnato, 2020), aprenden por sí mismos y -en este caso- a obtener automáticamente esa "recta" que buscamos con la tendencia de predicción. Para hacerlo se mide el error con respecto a los puntos de entrada y el valor "Y" de salida real. El algoritmo deberá minimizar el coste de una función de error cuadrático y esos coeficientes corresponderán con la recta óptima. Hay diversos métodos para conseguir minimizar el coste. Lo más común es utilizar una versión vectorial y la llamada Ecuación Normal que nos dará un resultado directo.

NOTA: cuando hablo de "recta" es en el caso particular de regresión lineal simple. Si hubiera más variables, hay que generalizar el término.

Ejercicio Práctico.

En este ejemplo cargaremos un archivo .csv de entrada obtenido por webscraping que contiene diversas URLs a artículos sobre Machine Learning de algunos sitios muy importantes como Techcrunch o KDnuggets y como características de entrada -las columnas- tendremos:

- Title: Titulo del Artículo
- url: ruta al artículo

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

• Word count: la cantidad de palabras del artículo,

• # of Links: los enlaces externos que contiene,

• # of comments: cantidad de comentarios,

• # Images video: suma de imágenes (o videos),

• Elapsed days: la cantidad de días transcurridos (al momento de crear el archivo)

• # Shares: nuestra columna de salida que será la "cantidad de veces que se compartió el artículo".

A partir de las características de un artículo de machine learning intentaremos predecir, cuantas veces será compartido en Redes Sociales. Haremos una primera predicción de regresión lineal simple con una sola variable predictora- para poder graficar en 2 dimensiones (ejes X e Y) y luego un ejemplo de regresión Lineal Múltiple, en la que utilizaremos 3 dimensiones (X,Y,Z) y predicciones.

Requerimientos

Para realizar este ejercicio, crearemos una Jupyter notebook con código Python y la librería ScikitLearn muy utilizada en Data Science. Recomendamos utilizar la suite de Anaconda. Podrás descargar los archivos de entrada csv o visualizar la notebook online.

Figura 48.
Importamos bibliotecas

```
# Importamos las biblitecas necesarias
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
%matplotlib inline
from mpl_toolkits.mplot3d import Axes3D
from matplotlib import cm
plt.rcParams['figure.figsize']=[16,9]
plt.style.use('ggplot')
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, r2_score
```

Nota. La imagen muestra el código para importar todas las bibliotecas necesarias, que vamos a utilizar para la predicción del ejercicio. *Fuente*: El investigador.

Figura 49.
Lectura del archivo csv

```
# Leeamos el archiv csv y lo cargamos como un dataset de Pandas. Y vemos us tamaño
data = pd.read_csv("articulos_ml.csv")
#Vemos cuantas dimensiones y registros que contiene
data.shape
```

Nota. La imagen muestra el código para leer el archivo csv y mostrar cual es la dimensión de los registros del archivo. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 50.
Mostrando datos

```
#
data.head()
```

	Title	url	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
0	What is Machine Learning and how do we use it ...	https://blog.signals.network/what-is-machine-l...	1888	1	2.0	2	34	200000
1	10 Companies Using Machine Learning in Cool Ways	NaN	1742	9	NaN	9	5	25000
2	How Artificial Intelligence Is Revolutionizing...	NaN	962	6	0.0	1	10	42000
3	Dbrain and the Blockchain of Artificial Intell...	NaN	1221	3	NaN	2	68	200000
4	Nasa finds entire solar system filled with eig...	NaN	2039	1	104.0	4	131	200000

Podemos ver que algunos campos estan Nan(nulos) por ejemplo algunas utls o en comentarios.

Nota. La imagen muestra los primeros 10 registros del archivo que fue leído. *Fuente*: El investigador.

Figura 51.
Mostramos la estadística.

En nuestro caso la columna Shares será nuestra salida, es decir nuestro valor "Y", el valor que queremos predecir

```
# Ahora veamos algunas estadísticas de nuestros datos
data.describe()
```

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	161.000000	161.000000	129.000000	161.000000	161.000000	161.000000
mean	1808.260870	9.739130	8.782946	3.670807	98.124224	27948.347826
std	1141.919385	47.271625	13.142822	3.418290	114.337535	43408.006839
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	990.000000	3.000000	2.000000	1.000000	31.000000	2800.000000
50%	1674.000000	5.000000	6.000000	3.000000	62.000000	16458.000000
75%	2369.000000	7.000000	12.000000	5.000000	124.000000	35691.000000
max	8401.000000	600.000000	104.000000	22.000000	1002.000000	350000.000000

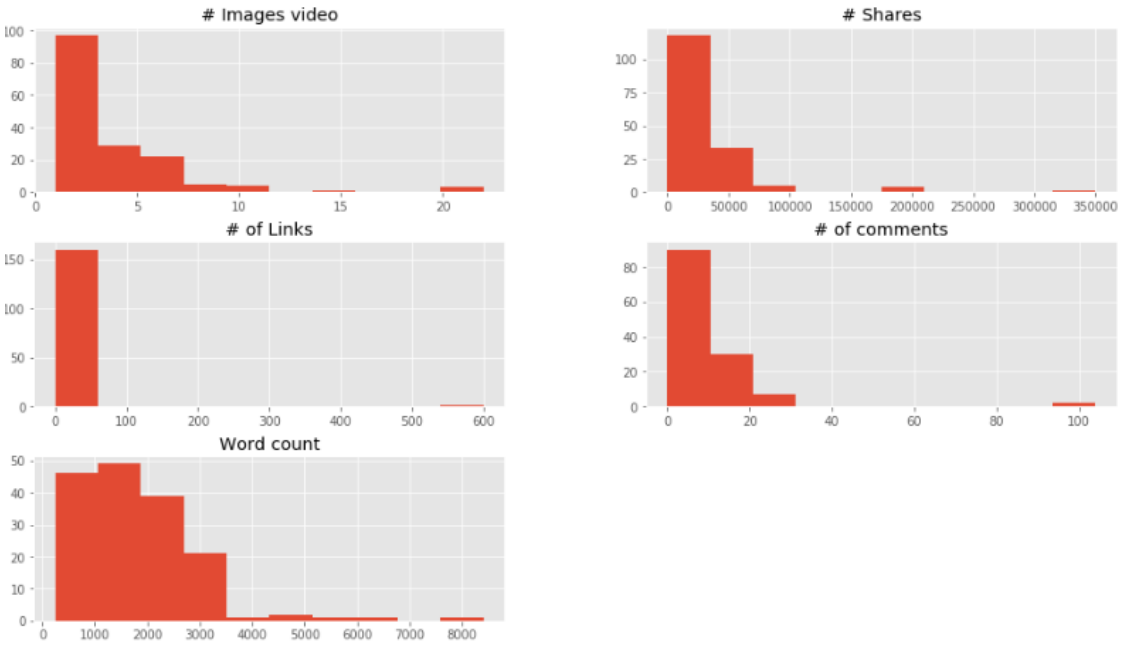
Nota. La imagen muestra datos estadísticos del archivo que se subió dándonos la media, el valor mínimo entre otros valores. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 52

Datos de entrada

```
# Visualizamos rápidamente las caraterísticas de entrada
data.drop(['Title','url', 'Elapsed days'],1).hist()
plt.show()
```



Nota. La imagen muestra las gráficas de los datos de entrada. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 53.
Visualización de datos.

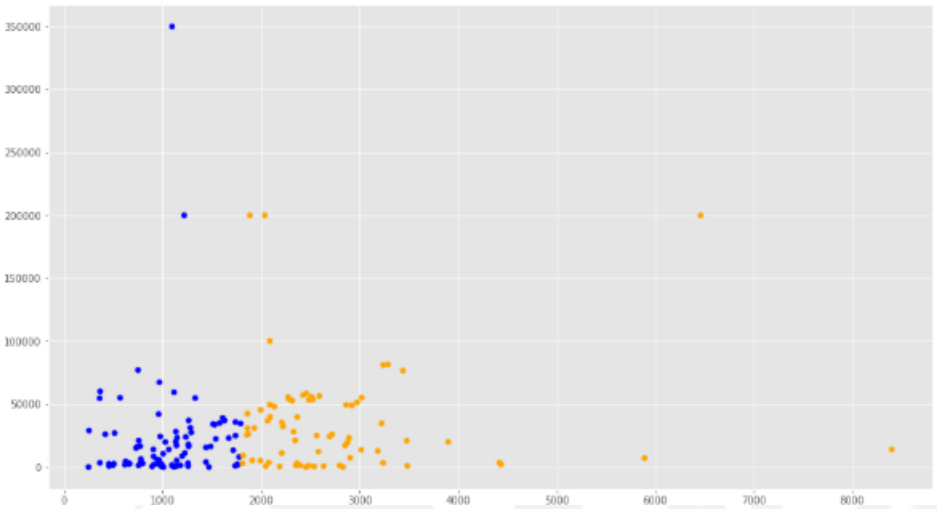
Visualizamos Cantidad de Palabras vs. Compartidos

```
#vamos a Visualizar los datos de entrada
colores=['orange','blue']
tamanios=[30,60]

f1 = data['Word count'].values
f2 = data['# Shares'].values

# Vamos a pintar en 2 colores los puntos por debajo de la media de Cantidad de Palabras
asignar=[]
for index, row in data.iterrows():
    if(row['Word count']>1808):
        asignar.append(colores[0])
    else:
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```



Nota. La imagen muestra una imagen de dispersión entre la cantidad de palabras vs número de veces compartido. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

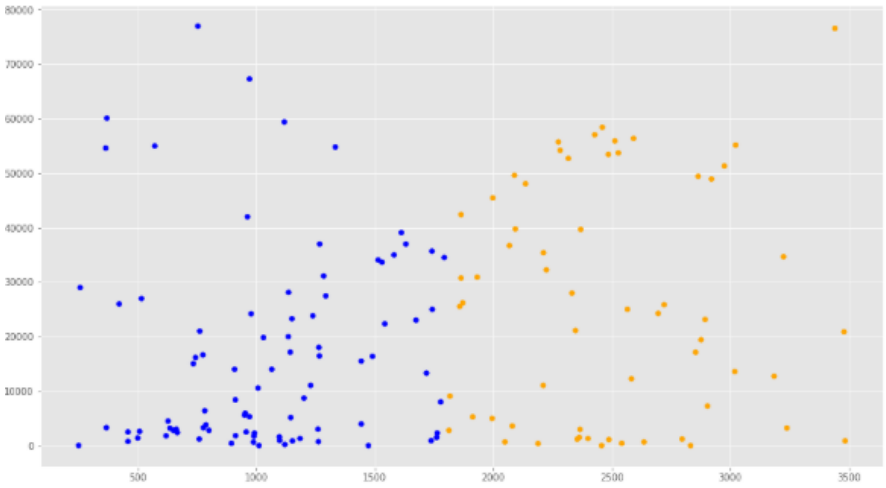
Figura 54.
Visualización de datos concentrados.

```
# Vamos a RECORTAR los datos en la zona donde se concentran más los puntos
# esto es en el eje X: entre 0 y 3.500
# y en el eje Y: entre 0 y 80.000
filtered_data = data[(data['Word count'] <= 3500) & (data['# Shares'] <= 80000)]

f1 = filtered_data['Word count'].values
f2 = filtered_data['# Shares'].values

# Vamos a pintar en colores los puntos por debajo y por encima de la media de Cantidad de Palabras
asignar=[]
for index, row in filtered_data.iterrows():
    if(row['Word count']>1808):
        asignar.append(colores[0])
    else:
        asignar.append(colores[1])

plt.scatter(f1, f2, c=asignar, s=tamanios[0])
plt.show()
```



Nota. La imagen muestra los datos en la zona donde más se concentra los datos. *Fuente*: El investigador.

Figura 55.
Datos estadísticos.

```
# Veamos como cambian los valores una vez filtrados
filtered_data.describe()
```

	Word count	# of Links	# of comments	# Images video	Elapsed days	# Shares
count	148.000000	148.000000	121.000000	148.000000	148.000000	148.000000
mean	1640.209459	5.743243	7.256198	3.331081	91.554054	20545.648649
std	821.975365	6.064418	6.346297	2.706476	91.143923	19933.865031
min	250.000000	0.000000	0.000000	1.000000	1.000000	0.000000
25%	971.000000	3.000000	2.000000	1.000000	28.750000	2750.000000
50%	1536.000000	5.000000	6.000000	3.000000	60.000000	15836.000000
75%	2335.750000	7.000000	11.000000	4.000000	110.500000	34177.500000
max	3485.000000	49.000000	30.000000	22.000000	349.000000	77000.000000

Nota. La imagen muestra datos estadísticos solo de los datos donde se encuentran más puntos. *Fuente*: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 56.
Regresión simple.

Regresión Lineal Simple (1 variable)

Vamos a intentar primero una Regresión Lineal con 1 sóla variable

```
# Asignamos nuestra variable de entrada X para entrenamiento y las etiquetas Y.
dataX =filtered_data[["Word count"]]
X_train = np.array(dataX)
y_train = filtered_data['# Shares'].values
```

Nota: Asignamos la variable de entrada X para el entrenamiento y las etiquetas Y. Fuente propia

Figura 57.
Creación objeto linear

```
# Creamos el objeto de Regresión Linear
regr = linear_model.LinearRegression()

# Entrenamos nuestro modelo
regr.fit(X_train, y_train)

# Hacemos las predicciones que en definitiva una línea (en este caso, al ser 2D)
y_pred = regr.predict(X_train)

# Veamos los coeficienetes obtenidos, En nuestro caso, serán la Tangente
print('Coefficients: \n', regr.coef_)
# Este es el valor donde corta el eje Y (en X=0)
print('Independent term: \n', regr.intercept_)
# Error Cuadrado Medio
print("Mean squared error: %.2f" % mean_squared_error(y_train, y_pred))
# Puntaje de Varianza. El mejor puntaje es un 1.0
print('Variance score: %.2f' % r2_score(y_train, y_pred))
```

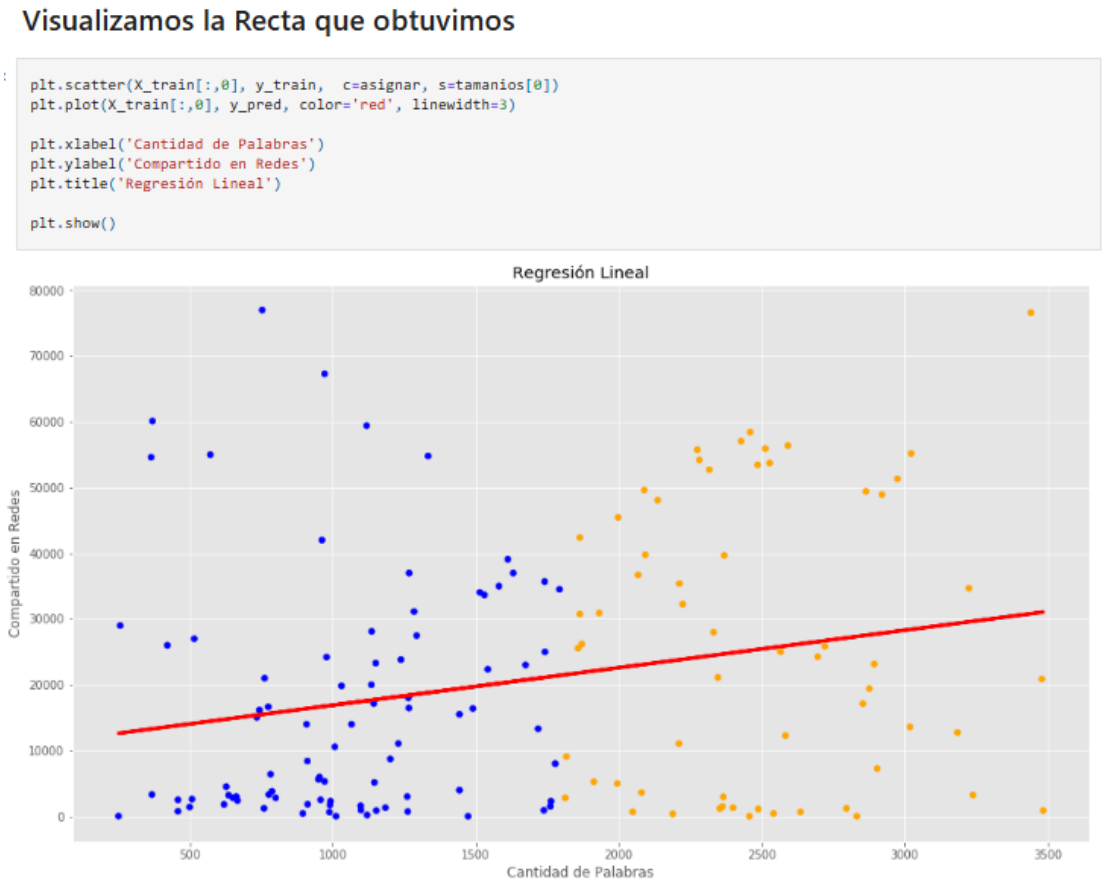
```
Coefficients:
 [5.69765366]
Independent term:
 11200.303223074163
Mean squared error: 372888728.34
Variance score: 0.06
```

Nota. La imagen muestra el código donde se crea el objeto de la regresión linear realizado la predicción.

Fuente: El investigador.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Figura 58.
Visualización de la regresión linear.



Nota. La imagen muestra la predicción y la línea de la regresión lineal. *Fuente*: El investigador.

Figura 59.
Predicción.

Nota. La imagen muestra la predicción de un artículo si tiene 2000 palabras. En esta predicción no dice que se va a compartir 22595 veces. *Fuente*: El investigador.

Regresión logística.

Utilizaremos algoritmos de Machine Learning en Python para resolver un problema de Regresión Logística. A partir de un conjunto de datos de entrada (características), nuestra salida será discreta (y no continua) por eso utilizamos Regresión Logística (y no Regresión Lineal). La Regresión Logística es un Algoritmo Supervisado[63] y se utiliza para clasificación. Vamos a clasificar problemas

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

con dos posibles estados “SI/NO”: binario o un número finito de “etiquetas” o “clases”: múltiple. Algunos Ejemplos de Regresión Logística son:

- Clasificar si el correo que llega es Spam o No es Spam.
- Dados unos resultados clínicos de un tumor clasificar en “Benigno” o “Maligno”.
- El texto de un artículo a analizar es: Entretenimiento, Deportes, Política ó Ciencia.
- A partir de historial bancario conceder un crédito o no. Confiaremos en la implementación del paquete Scikit-learn de Python para ponerlo en práctica.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Autoevaluación 2

1. **¿Cuál de las siguientes opciones describe correctamente el aprendizaje supervisado?**
    a. El modelo se entrena con datos sin etiquetas.
    b. El modelo interactúa con un entorno y aprende de recompensas.
    c. El modelo aprende a partir de datos etiquetados.
    d. El modelo genera sus propias etiquetas.
2. **¿Cuál de estos algoritmos es un ejemplo de aprendizaje no supervisado?**
    a. Regresión logística.
    b. Árboles de decisión.
    c. K-Means
    d. Regresión Lineal.
3. **El aprendizaje por refuerzo, ¿Qué elemento representa la estrategia que sigue el agente para actuar?**
    a. Recompensa
    b. Política
    c. Entorno
    d. Estado
4. **¿Cuál es la función de activación usada comúnmente en la regresión logística?**
    a. reLu
    b. Tanh
    c. Sigmoide
    d. Softmax
5. **¿Qué tipo de datos requiere el aprendizaje supervisado?**
    a. Datos no estructurados.
    b. Datos con etiquetas.
    c. Datos con ruido.
    d. Datos sin etiquetas.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

6. **¿Cuál es el objetivo principal del aprendizaje no supervisado?**

    a. Clasificar correos como spam o ham

    b. Identificar grupos o patrones ocultos en los datos.

    c. Aprender de la retroalimentación del entorno.

    d. Predecir valores numéricos con exactitud.

7. **¿Qué algoritmos de aprendizaje automático busca minimizar la distancia entre puntos de datos y su centroide?**

    a. SNM

    b. K-Means

    c. Regresión logística.

    d. Árboles de decisión.

8. **¿Cuál de las siguientes afirmaciones sobre el aprendizaje por refuerzo es correcta?**

    a. No utiliza recompensas ni penalizaciones.

    b. Utiliza datos totalmente estructurados y etiquetados.

    c. El agente aprende de las retroalimentaciones del entrono para maximizar recompensas.

    d. Solo se utiliza en análisis de sentimientos.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Resumen de la Unidad 2

La Unidad 2 ha proporcionado una visión amplia y profunda sobre los fundamentos del aprendizaje automático, permitiendo al estudiante identificar sus tipos, estructuras matemáticas y aplicaciones prácticas en diversos campos.

Se inició con la definición de aprendizaje automático como una disciplina de la inteligencia artificial que permite a las máquinas aprender de los datos para realizar tareas sin ser programadas explícitamente. Se diferenciaron los conceptos de inteligencia artificial, aprendizaje automático y aprendizaje profundo, comprendiendo sus relaciones jerárquicas y campos de aplicación.

Posteriormente, se estudiaron los tres tipos principales de aprendizaje automático: supervisado, no supervisado y por refuerzo. En el aprendizaje supervisado, se revisaron modelos como regresión lineal y regresión logística, destacando sus fórmulas y funciones de activación como la sigmoide. En el aprendizaje no supervisado, se exploraron técnicas de agrupamiento como K-Means y reducción de dimensionalidad como PCA. En el aprendizaje por refuerzo, se analizaron sus componentes clave, como la función de valor y la ecuación de Bellman.

Asimismo, se detallaron las aplicaciones actuales del aprendizaje automático en sectores como salud, finanzas, comercio electrónico, educación, ciberseguridad y vehículos autónomos. Estas aplicaciones demuestran su utilidad para optimizar procesos, predecir comportamientos y personalizar servicios.

Finalmente, se reforzó la importancia de dominar las herramientas y bibliotecas como Scikit-learn, TensorFlow y Keras, y se destacó la necesidad del preprocesamiento de datos, la selección de modelos y la evaluación basada en métricas precisas.

Esta unidad ha sentado una base teórica sólida para comprender cómo se entrenan, implementan y evalúan los sistemas inteligentes actuales, abriendo camino a las siguientes fases del aprendizaje automático con enfoques prácticos y modelos más complejos.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

UNIDAD 3 ÁRBOLES DE DECISIÓN

Temas y Subtemas

Tema 1 Árboles de decisión

Tema 2 Random Forest

Árboles de Decisión.

En este capítulo describiremos en qué consisten y cómo funcionan los árboles de decisión utilizados en Aprendizaje Automático y nos centraremos en un divertido ejemplo en Python en el que analizaremos a los cantantes y bandas que lograron un puesto número uno en las listas de Billboard Hot 100 e intentaremos predecir quién será el próximo Ed Sheeran a fuerza de Inteligencia Artificial. Realizaremos Gráficas que nos ayudarán a visualizar los datos de entrada y un grafo para interpretar el árbol que crearemos con el paquete Scikit-Learn.

¿Qué es un árbol de decisión?

Los árboles de decisión son representaciones gráficas de posibles soluciones a una decisión basadas en ciertas condiciones, es uno de los algoritmos de aprendizaje supervisado más utilizados en machine learning y pueden realizar tareas de clasificación o regresión (acrónimo del inglés CART). La comprensión de su funcionamiento suele ser simple y a la vez muy potente. Utilizamos mentalmente estructuras de árbol de decisión constantemente en nuestra vida diaria sin darnos cuenta: ¿Llueve? ⇒ lleva paraguas. ¿Soleado? ⇒ lleva gafas de sol. ¿estoy cansado? ⇒ toma café. Son Decisiones del tipo IF THIS, THEN THAT Los árboles de decisión tienen un primer nodo llamado

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

raíz (root) y luego se descomponen el resto de los atributos de entrada en dos ramas (podrían ser más, pero no nos meteremos en eso ahora) planteando una condición que puede ser cierta o falsa. Se bifurca cada nodo en 2 y vuelven a subdividirse hasta llegar a las hojas que son los nodos finales y que equivalen a respuestas a la solución: Si/No, Comprar/Vender, o lo que sea que estemos clasificando. Otro ejemplo son los populares juegos de adivinanza:

1. ¿Animal ó vegetal? -Animal
2. ¿Tiene cuatro patas? -Si
3. ¿Hace guau? -Si
4. -Es un perro!

¿Qué necesidad hay de usar el Algoritmo de árbol?

Supongamos que tenemos atributos como Género con valores "hombre o mujer" y edad en rangos: "menor de 18 o mayor de 18" para tomar una decisión. Podríamos crear un árbol en el que dividamos primero por género y luego subdividir por edad. O podría ser al revés: primero por edad y luego por género. ¡El algoritmo es quien analizando los datos y las salidas -por eso es supervisado! decidirá la mejor forma de hacer las divisiones (splits) entre nodos. Tendrá en cuenta de qué manera lograr una predicción (clasificación o regresión) con mayor probabilidad de acierto. Parece sencillo, no? Pensemos que, si tenemos 10 atributos de entrada cada uno con 2 o más valores posibles, las combinaciones para decidir el mejor árbol serían cientos o miles… Esto ya no es un trabajo para hacer artesanalmente. Y ahí es donde este algoritmo cobra importancia, pues él nos devolverá el árbol óptimo para la toma de decisión más acertada desde un punto de vista probabilístico.

¿Cómo funciona un árbol de decisiones?

Para obtener el árbol óptimo y valorar cada subdivisión entre todos los árboles posibles y conseguir el nodo raíz y los subsiguientes, el algoritmo deberá medir de alguna manera las predicciones logradas y valorarlas para comparar de entre todas y obtener la mejor. Para medir y valorar, utiliza diversas funciones, siendo las más conocidas y usadas los "Indice gini" y "Ganancia de información" que utiliza la denominada "entropía". La división de nodos continuará hasta que lleguemos a la profundidad máxima posible del árbol o se limiten los nodos a una cantidad mínima de muestras en cada hoja. A continuación, describiremos muy brevemente cada una de las estrategias nombradas:

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Índice Gini:

Se utiliza para atributos con valores continuos (precio de una casa). Esta función de coste mide el "grado de impureza" de los nodos, es decir, cuán desordenados o mezclados quedan los nodos una vez divididos. Deberemos minimizar ese GINI index.

Ganancia de información.

Se utiliza para atributos categóricos (como en hombre/mujer). Este criterio intenta estimar la información que aporta cada atributo basado en la "teoría de la información". Para medir la aleatoriedad de incertidumbre de un valor aleatorio de una variable "X" se define la Entropia[88]. Al obtener la medida de entropía de cada atributo, podemos calcular la ganancia de información del árbol. Deberemos maximizar esa ganancia

Random Forest, el poder del Ensamble

Luego del algoritmo de árbol de Decisión, tu próximo paso es el de estudiar Random Forest. Comprende qué es y cómo funciona con un ejemplo práctico en Python. Random Forest es un tipo de Ensamble en Machine Learning en donde combinaremos diversos árboles -ya veremos cómo y con qué características- y la salida de cada uno se contará como "un voto" y la opción más votada será la respuesta del Bosque Aleatorio. Random Forest, al igual que el árbol e decisión, es un modelo de aprendizaje supervisado para clasificación (aunque también puede usarse para problemas de regresión).

¿Cómo surge Random Forest?

Uno de los problemas que aparecía con la creación de un árbol de decisión es que, si le damos la profundidad suficiente, el árbol tiende a "memorizar" las soluciones en vez de generalizar el aprendizaje. Es decir, a padecer de overfitting. La solución para evitar esto es la de crear muchos árboles y que trabajen en conjunto. Veamos cómo. ¿Cómo funciona Random Forest? Random Forest funciona así:

• Seleccionamos k features (columnas) de las m totales (siendo k menor a m) y creamos un árbol de decisión con esas k características.

• Creamos n árboles variando siempre la cantidad de k features y también podríamos variar la cantidad de muestras que pasamos a esos árboles (esto es conocido como "bootstrap sample")

• Tomamos cada uno de los n árboles y le pedimos que hagan una misma clasificación. Guardamos el resultado de cada árbol obteniendo n salidas.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

• Calculamos los votos obtenidos para cada "clase" seleccionada y consideraremos a la más votada como la clasificación final de nuestro "bosque".

¿Por qué es aleatorio?

Contamos con una doble aleatoriedad: tanto en la selección del valor k de características para cada árbol como en la cantidad de muestras que usaremos para entrenar cada árbol creado.

Es curioso que para este algoritmo la aleatoriedad sea tan importante y de hecho es lo que lo "hace bueno", pues le brinda flexibilidad suficiente como para poder obtener gran variedad de árboles y de muestras que, en su conjunto aparentemente caótico, producen una salida concreta. Darwin estaría orgulloso

Ventajas y Desventajas del uso de Random Forest

Vemos algunas de sus ventajas son:

• funciona bien -aún- sin ajuste de hiperparámetros

• funciona bien para problemas de clasificación y también de regresión.

• al utilizar múltiples árboles se reduce considerablemente el riesgo de overfiting.

• se mantiene estable con nuevas muestras puesto que al utilizar cientos de árboles sigue prevaleciendo el promedio de sus votaciones.

Y sus desventajas:

• en algunos datos de entrada "particulares" random forest también puede caer en overfitting

• es mucho más "costo" de crear y ejecutar que "un sólo árbol" de decisión.

• Puede requerir muchísimo tiempo de entrenamiento

• OJO! Random Forest no funciona bien con datasets pequeños.

• Es muy difícil poder interpretar los ¿cientos? de árboles creados en el bosque, si quisiéramos comprender y explicar a un cliente su comportamiento.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Autoevaluación 3

1. **¿Qué representa un árbol de decisión en el contexto del aprendizaje automático?**

   a. Un modelo lineal para regresión
   b. Un gráfico de barras
   c. Una representación gráfica de decisiones basada en condiciones
   d. Un algoritmo de agrupamiento no supervisado

2. **¿Cómo se llama el primer nodo de un árbol de decisión?**

   a. Hoja.
   b. Raíz.
   c. Rama.
   d. Nodo.

3. **¿Cuál es la finalidad principal del algoritmo CART?**

   a. Crear árboles de decisión para clasificación y regresión.
   b. Generar gráficos de dispersión.
   c. Predecir datos sin supervisión.
   d. Reducir dimensionalidad

4. **¿Qué criterio se utiliza comúnmente para dividir nodos en un árbol de un árbol de decisión cuando se trata de variables continuas?**

   a. Ganancia de información.
   b. Entropía inversa.
   c. Índice Gini.
   d. Promedio de clases

5. **¿Cuál de los siguientes ejemplos representa una decisión binaria en un árbol de decisión?**

   a. Clasificación entre varios géneros musicales.
   b. Predecir el precio exacto de una casa.
   c. Determinar si un correo es spam o no spam.
   d. Estimar el número de visitas a una web.

6. **¿Qué problema resuelve el algoritmo Random Forest respecto a un único árbol de decisión?**

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

a. Aumenta el número de variables a analizar.

b. Reduce el tiempo de cómputo.

c. Evita el overfitting combinando múltiples árboles.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC

Resumen de la Unidad 3

Los árboles de decisión son estructuras gráficas en forma de árbol utilizadas en el aprendizaje automático supervisado. Permiten tomar decisiones a partir de condiciones lógicas del tipo “si esto, entonces aquello”. Cada árbol comienza en un nodo raíz, se ramifica a través de decisiones binarias y termina en hojas que representan resultados finales. Estos modelos pueden usarse tanto para clasificación como para regresión.

El algoritmo analiza los datos y determina cómo dividir los nodos para lograr la mayor precisión posible en sus predicciones. Para ello utiliza funciones como el índice Gini, que mide la impureza en nodos con variables continuas, y la ganancia de información, basada en la entropía, para atributos categóricos. Cuanto menor sea la impureza o mayor la ganancia de información, mejor será la división.

Un problema común en los árboles de decisión es el sobreajuste (overfitting), que ocurre cuando el modelo se adapta demasiado a los datos de entrenamiento. Para resolverlo, se utilizan técnicas de ensamble como Random Forest.

Random Forest es un conjunto de múltiples árboles de decisión que trabajan en paralelo. Cada árbol se entrena con una muestra aleatoria (con reemplazo) del conjunto de datos, y cada división del árbol utiliza un subconjunto aleatorio de características. El resultado final se decide mediante una votación mayoritaria entre todos los árboles, lo que hace al modelo más robusto y preciso.

Entre las ventajas de Random Forest están su resistencia al overfitting, su capacidad de manejar tanto problemas de clasificación como de regresión, y su rendimiento incluso sin ajuste de hiperparámetros. Entre las desventajas, destaca su alta demanda computacional y la dificultad para interpretar los resultados debido a la complejidad del modelo.

En resumen, tanto los árboles de decisión como Random Forest son herramientas poderosas para la toma de decisiones basada en datos, siendo fáciles de implementar y altamente efectivas en una gran variedad de problemas del mundo real.

(+593) 96 356 1961 admisiones@itq.edu.ec Antonio de Ulloa N28-30 y Diego de Atienza (Esq). WWW.ITQ.EDU.EC