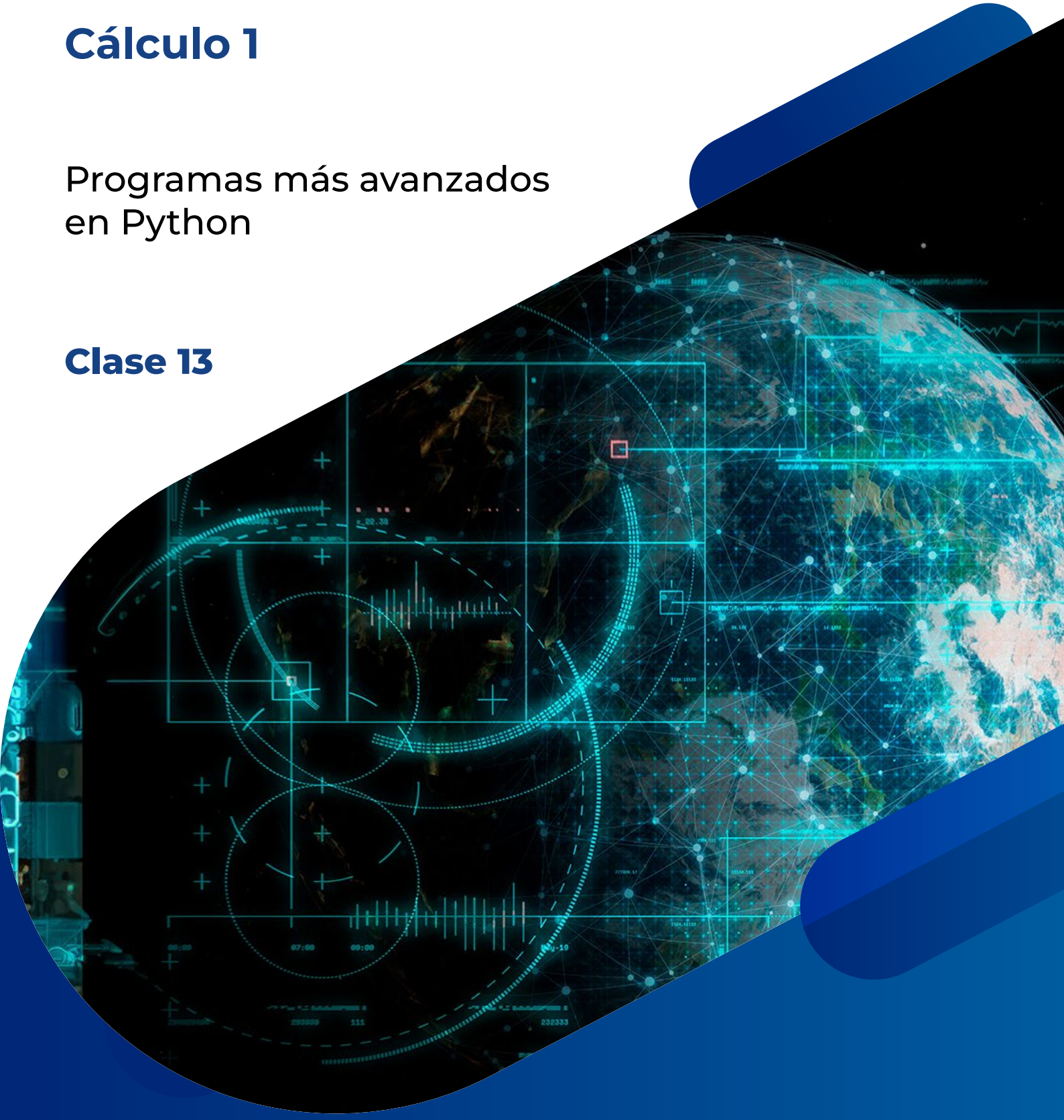
Cálculo 1

Programas más avanzados en Python

Clase 13



Ingeniería en ciberseguridad

La excelencia no se improvisa

Clase 13. Programas más avanzados en Python

Python ha ganado gran popularidad debido a su facilidad de uso y su capacidad para escalar hacia aplicaciones más complejas. Desde simples scripts de automatización hasta sistemas de aprendizaje automático y aplicaciones web de gran escala, Python proporciona las herramientas necesarias para abordar una amplia gama de problemas. Su ecosistema, rico en bibliotecas como NumPy, Pandas, TensorFlow, Flask y Django, permite a los desarrolladores implementar soluciones avanzadas de manera eficiente. La comunidad activa de Python también asegura la constante mejora del lenguaje y sus herramientas, proporcionando soporte y recursos valiosos para el aprendizaje continuo y la resolución de problemas.

A continuación, se presentan ejemplos de programas avanzados en Python en diversas áreas, ilustrando cómo este lenguaje puede utilizarse para resolver problemas complejos.

Ejemplo 1. Análisis de Datos con Pandas

Figura 57*:*

```python
import pandas as pd

# Cargar un conjunto de datos
data = pd.read_csv('datos.csv')

# Limpieza de datos
data = data.dropna()
data['fecha'] = pd.to_datetime(data['fecha'])

# Análisis de datos
resumen = data.describe()

# Agrupación y agregación
ventas_por_año = data.groupby(data['fecha'].dt.year)['ventas'].sum()

print(resumen)
print(ventas_por_año)
```

Nota. Este programa carga un conjunto de datos, realiza limpieza de datos, convierte una columna a formato de fecha, genera un resumen estadístico y agrupa las ventas por año.

Ejemplo 2. Desarrollo Web con Flask

Figura 58

```python
from flask import Flask, render_template, request

app = Flask(__name__)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/saludar', methods=['POST'])
def saludar():
    nombre = request.form['nombre']
    return f'Hola, {nombre}!'

if __name__ == '__main__':
    app.run(debug=True)
```

Nota. Este programa utiliza Flask para crear una aplicación web simple con una ruta de inicio y una ruta que procesa un formulario para saludar al usuario por su nombre.

Ejemplo 3. Aprendizaje Automático con Scikit-Learn

Figura 59

```python
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Cargar datos
iris = load_iris()
X, y = iris.data, iris.target

# Dividir datos en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=4

# Entrenar un modelo
modelo = RandomForestClassifier(n_estimators=100)
modelo.fit(X_train, y_train)

# Hacer predicciones
predicciones = modelo.predict(X_test)

# Evaluar el modelo
precision = accuracy_score(y_test, predicciones)
print(f'Precisión del modelo: {precision:.2f}')
```

Nota. Este programa carga el conjunto de datos de iris, divide los datos en conjuntos de entrenamiento y prueba, entrena un modelo de bosque aleatorio y evalúa su precisión.

13.1 Herramientas de cálculo con Python para modelaje de funciones aplicado a negocios: análisis marginal

En esta sección, exploraremos cómo utilizar Python para analizar puntos críticos en funciones matemáticas aplicadas a los negocios. Nos centraremos en la identificación de máximos y mínimos relativos, así como en cómo estos conceptos se aplican en el análisis marginal de funciones de costo, ingreso y lucro. Estos conocimientos son esenciales para tomar decisiones estratégicas en un negocio.

Puntos críticos y máximos/mínimos relativos

Un punto crítico de una función se encuentra cuando su primera derivada respecto a x se iguala a

cero. Matemáticamente, esto se expresa como: $f'(x)=0$.

El valor de x que satisface esta ecuación es el punto crítico. Este punto puede ser un máximo, un mínimo o un punto de inflexión.

Análisis de máximos y mínimos relativos

Criterio de la primera derivada: Para determinar si un punto crítico es un máximo o un mínimo relativo, analizamos el signo de la primera derivada en intervalos cercanos a ese punto:

- Si $f'(x)$ es positiva en un intervalo y negativa en otro, el punto crítico es un máximo relativo.
- Si $f'(x)$ es negativa en un intervalo y positiva en otro, el punto crítico es un mínimo relativo.

Criterio de la segunda derivada: El criterio de la segunda derivada proporciona un método más directo para clasificar puntos críticos:

- Si la segunda derivada $f''(x)$ evaluada en el punto crítico es menor que cero, el punto es un máximo relativo.
- Si $f''(x)$ evaluada en el punto crítico es mayor que cero, el punto es un mínimo relativo.

Aplicación en negocios: análisis marginal

El análisis marginal se refiere a la evaluación de la tasa de cambio instantánea en funciones de negocio, tales como el costo, el ingreso y el lucro.

Funciones marginales

- **Costo marginal**: El costo marginal se define como la derivada de la función de costo total $C(x)$: $C'(x)$. Representa la tasa instantánea de cambio del costo total con respecto al número de productos producidos.
- **Ingreso marginal**: El ingreso marginal es la derivada de la función de ingreso total $R(x)$: $R'(x)$. Indica la tasa instantánea de cambio del ingreso total por cada unidad adicional vendida.
- **Lucro marginal**: El lucro marginal es la derivada de la función de lucro total $P(x)$, que es la

diferencia entre ingreso y costo: $P'(x)=R'(x)-C'(x)$

Ejemplo práctico con Python

Problema:
Una compañía produce tanques de gasolina para autos. El costo total de producir x tanques está dado por la función C(x). Queremos encontrar el costo marginal y el costo exacto de producir un tanque adicional en un nivel de producción de 500 tanques por semana.

Solución:
Definimos la función de costo total C(x) y calculamos su derivada.

Figura 60

```python
import sympy as sp

x = sp.symbols('x')
C = 7000 + 2*x
C_deriv = sp.diff(C, x)
```

Evaluamos el costo marginal en x=500:

Figura 61

```python
costo_marginal_500 = C_deriv.subs(x, 500)
print(f"Costo marginal a 500 tanques: {costo_marginal_500}")
```

Calculamos el costo exacto de producir el tanque 501:

Figura 62

```python
C_500 = C.subs(x, 500)
C_501 = C.subs(x, 501)
costo_exacto_501 = C_501 - C_500
print(f"Costo exacto del tanque 501: {costo_exacto_501}")
```

13.2 Herramientas de cálculo con Python para modelaje de funciones aplicado a negocios: oferta y demanda y producción total

En esta sección, vamos a aplicar la teoría aprendida sobre costos, ingresos y lucros marginales utilizando Python. Esto nos permitirá visualizar de manera práctica cómo se comportan estas funciones en un entorno de negocios. ¡Vamos a preparar nuestro entorno y comenzar con los cálculos!

Preparación del entorno

Primero, necesitamos preparar nuestro entorno de trabajo en Python. Definiremos nuestras funciones de costo y precio para luego calcular el ingreso y el lucro.

Definición de funciones

Supongamos que la función de costo total C(x)C(x) está dada por:

$C(x)=50+20x$

Y la función de precio está definida por:

$P(x)=100-2x$

Ahora, definimos estas funciones en Python:

Figura 63

```python
import sympy as sp

# Definición de variables
x = sp.symbols('x')

# Funciones de costo y precio
C = 50 + 20*x
P = 100 - 2*x

# Función de ingreso total
R = P * x

# Función de lucro
L = R - C
```

Derivadas y funciones marginales

Ahora, vamos a calcular las derivadas de estas funciones para obtener el costo marginal, ingreso marginal y lucro marginal.

Costo marginal:

$C'(x)=d/dx(50+20x)$

Ingreso marginal:

$R'(x)=d/dx(P(x)•x)$

Lucro marginal:

$L'(x)=d/dx(R(x)−C(x))$

En Python, esto se realiza de la siguiente manera:

Figura 64

```python
# Derivadas
C_marginal = sp.diff(C, x)
R_marginal = sp.diff(R, x)
L_marginal = sp.diff(L, x)
```

Visualización

Para comprender mejor cómo se comportan estas funciones, vamos a visualizar los resultados en gráficos. Supongamos que estamos produciendo entre 0 y 40 unidades. Calcularemos y graficaremos el costo, el ingreso y el lucro total, así como sus derivadas marginales.

Figura 65

```python
import numpy as np
import matplotlib.pyplot as plt

# Definición de rango de producción
x_vals = np.linspace(0, 40, 400)
C_vals = [C.subs(x, val) for val in x_vals]
R_vals = [R.subs(x, val) for val in x_vals]
L_vals = [L.subs(x, val) for val in x_vals]
C_marginal_vals = [C_marginal.subs(x, val) for val in x_vals]
R_marginal_vals = [R_marginal.subs(x, val) for val in x_vals]
L_marginal_vals = [L_marginal.subs(x, val) for val in x_vals]

# Graficar los resultados
plt.figure(figsize=(10, 6))
plt.plot(x_vals, C_vals, label='Costo Total')
plt.plot(x_vals, R_vals, label='Ingreso Total')
plt.plot(x_vals, L_vals, label='Lucro Total')
plt.plot(x_vals, R_marginal_vals, label='Ingreso Marginal')
plt.plot(x_vals, C_marginal_vals, label='Costo Marginal')
plt.plot(x_vals, L_marginal_vals, label='Lucro Marginal')
plt.xlabel('Cantidad Producción')
plt.ylabel('Valores')
plt.legend()
plt.title('Análisis de Costo, Ingreso y Lucro Marginal')
plt.grid(True)
plt.show()
```

Interpretación de resultados

Al visualizar los resultados, podemos observar lo siguiente:

1. **Costo total**: Incrementa linealmente con la cantidad producida debido al término 20x.
2. **Ingreso total**: Inicialmente incrementa, pero luego decrece debido a la función de demanda P(x).
3. **Lucro total**: Es la diferencia entre el ingreso y el costo. Se observa el punto máximo de lucro.
4. **Ingreso marginal**: Muestra cómo cambia el ingreso total con cada unidad adicional ven-

dida.

5. **Costo marginal**: Muestra cómo cambia el costo total con cada unidad adicional producida.
6. **Lucro marginal**: Indica la rentabilidad de producir una unidad adicional.

Referencias

Gómez, M., & Rodríguez, J. (2019). *GeoGebra y su aplicación en la enseñanza de las matemáticas*. Editorial Educativa.

Martínez, S., & Torres, A. (2020). *Simulación y modelado en ingeniería: herramientas y aplicaciones*. Editorial Técnica.

Pérez, L., & López, J. (2021). *Tecnologías educativas: integración de software en la enseñanza de las ciencias*. Editorial Académica.

Glosario de los términos citados

Máximo relativo: Es el punto en una función donde el valor de la función es mayor que en los puntos inmediatamente adyacentes. También conocido como máximo local, es un punto $x=c$ tal que $f(c)\geq f(x)$ para todos los x cercanos a c.

Mínimo relativo: Es el punto en una función donde el valor de la función es menor que en los puntos inmediatamente adyacentes. También conocido como mínimo local, es un punto $x=c$ tal que $f(c)\leq f(x)$ para todos los x cercanos a c.

PUCE Virtual

La excelencia no se improvisa

síguenos

ICYS - 12 - 241