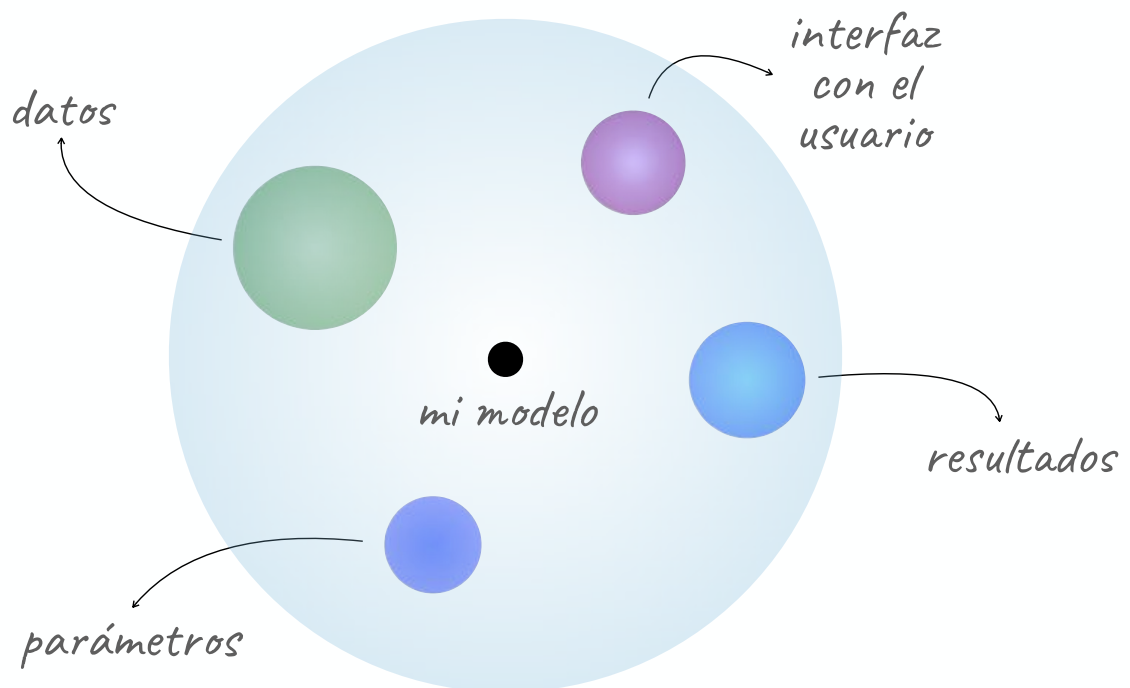
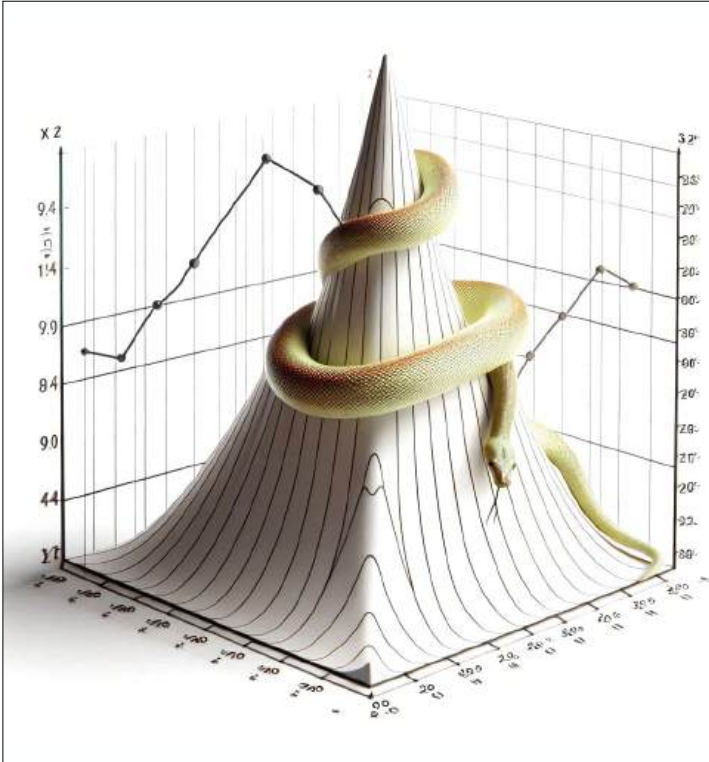


INVESTIGACIÓN OPERATIVA SUPERIOR

al fin ...¡optimización!

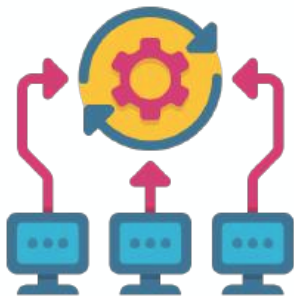
Presencial





*optimización
con python*

integración



*posibilidad de integración con
todas las partes de mi modelo
ampliado*

*gratuito**



*gratuito frente a los altos
costos en dólares del
software comercial*

el modelo más simple ...

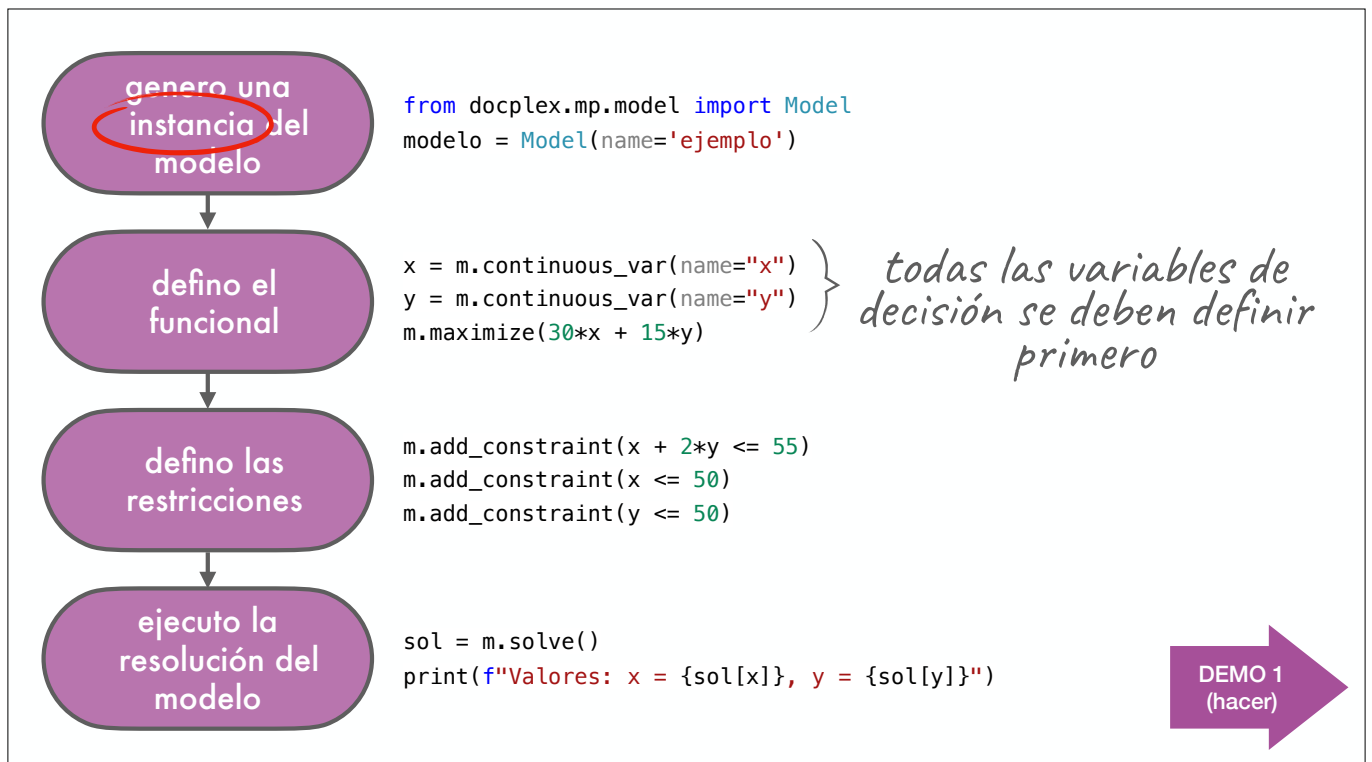
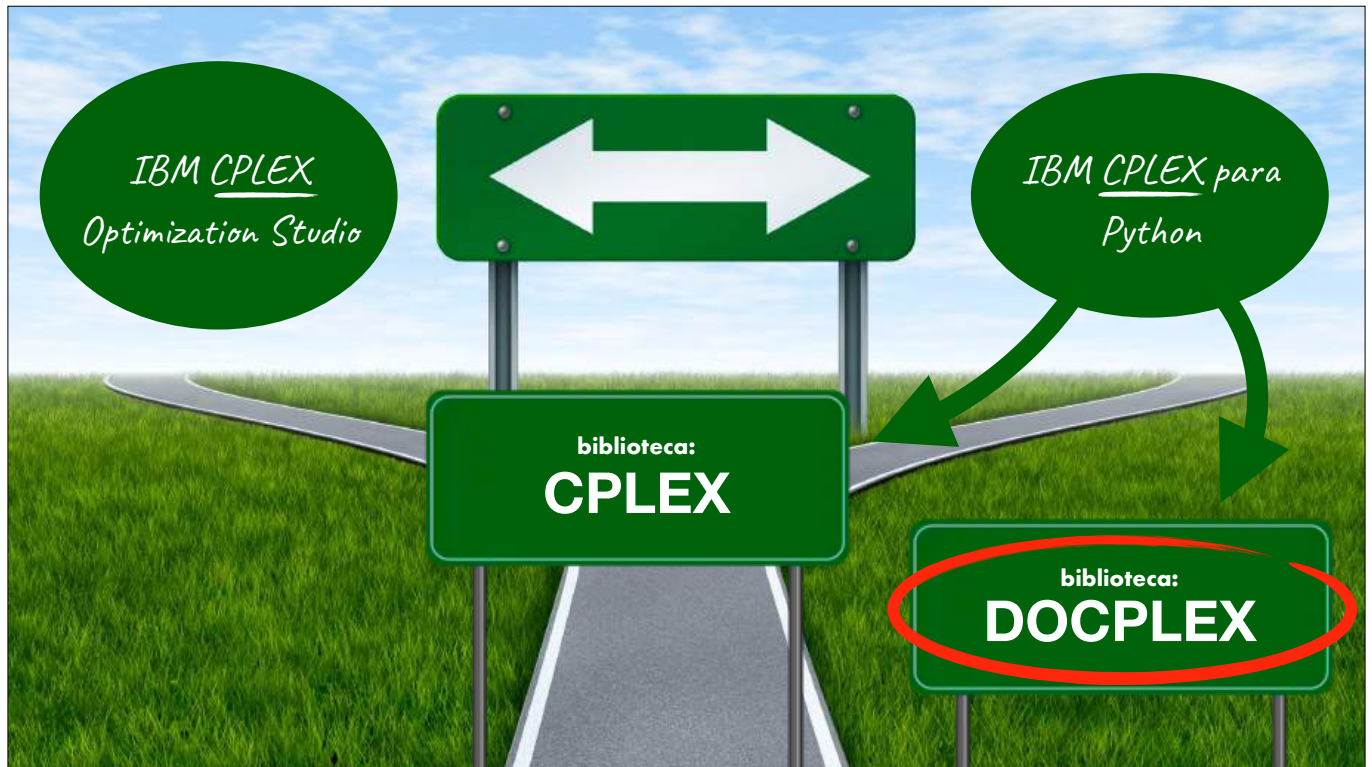
*maximizar $30^*x + 15^*y$*

*$x + 2^*y \leq 55$*

$x \leq 50$

$y \leq 50$





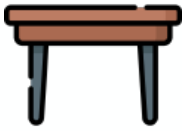
¿preguntas?



optimización de necesidades de stock



una empresa vende tres productos ...



Mesa (M)

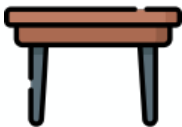


Silla (O)



Biblioteca (B)

cada semana se pregunta ¿qué cantidades de cada producto me conviene comprar/fabricar?



- margen
- costo
- demanda estimada
- stock actual
- stock de seguridad
- peso unitario
- volumen unitario
- stock máximo



sugerencias de compra/fabricación para reposición

ecuaciones

- (1) debemos maximizar la ganancia (funcional)
- (2) el espacio ocupado por estanterías no debe superar 84000 m²
- (3) ningún producto puede estar por debajo del stock de seguridad definido
- (4) ningún producto puede superar el máximo stock definido
- (5) el peso de mesas y sillas sumados no pueden superar los 1900 kg
- (6) no pueden superar el importe máximo de 1.200.000 ARS/semana
- (7) el stock final no debe superar $1,1 \cdot$ demanda estimada para cada producto

veamos una por una las
ecuaciones ...

① maximizar la ganancia

$$\text{maximizar } \underbrace{m_m * Q_m}_{\text{mesa}} + \underbrace{m_s * Q_s}_{\text{silla}} + \underbrace{m_b * Q_b}_{\text{biblioteca}}$$

variable de decisión (entera) \longrightarrow m_i : margen del producto i
 Q_i : cantidad a comprar de producto i

② limitación de superficie



$$S_b * Stk_b + S_b * Q_b \leq 84.000$$

S_B : superficie de una biblioteca (m^2)

Stk_B : stock inicial de bibliotecas (m^2)

Q_B : cantidad a comprar de bibliotecas

③ stock de seguridad

 $\longrightarrow Stk_m + Q_m \geq Seg_m$

 $\longrightarrow Stk_s + Q_s \geq Seg_s$


 $\longrightarrow Stk_b + Q_b \geq Seg_b$

Stk_i : stock actual producto i
Seg_i : stock de seguridad producto i

④ stock máximo

 $\longrightarrow Stk_m + Q_m \leq Smax_m$

 $\longrightarrow Stk_s + Q_s \leq Smax_s$

 $\longrightarrow Stk_b + Q_b \leq Smax_b$

Stk_i : stock actual producto i
Smax_i : stock máximo del producto i

5) peso máximo

$$Peso_m * Q_m + Peso_s * Q_s \leq P_{max}$$



Peso_i : peso unitario del producto i
P_{max} : peso máximo de carga

6) inversión máxima

$$C_m * Q_m + C_s * Q_s + C_b * Q_b \leq 1.200.000$$



C_i : costo del producto i

7 demanda estimada



$$Stk_m + Q_m \leq 1,1 * D_m$$



$$Stk_s + Q_s \leq 1,1 * D_s$$



$$Stk_b + Q_b \leq 1,1 * D_b$$

Stk_i : stock actual producto i

D_i : demanda del producto i

DEMO 2



el código para
nosotros son las
ecuaciones

[HardCoded]

no se debería
"hardcodear" ningún
dato que en un futuro
pueda variar

¿y si varía el
stock?

¿y si varían los
parámetros?

¿y si varía el
margen?

¿y si varía el
presupuesto?



¿o si varía
cualquier otro
valor?

¿y si agrego
un producto?

¿y si agrego
20 productos?

tratar con
gran cantidad
de datos

obtener
información
de distintas
fuentes

modelización avanzada y modelos dinámicos

resolver la
variabilidad
de los datos

debemos comenzar planteando las
ecuaciones en forma genérica ...

$$\text{maximizar } \sum_{i=1}^n M_i \times Q_i \quad (1)$$

$$(2) \quad Sup_b \times Stk_b + Sup_b \times Q_i \leq Sup_{max}$$

$$(3) \quad Q_i + Stk_i \geq Seg_i \quad \forall i \in 1...n$$

$$(4) \quad Q_i + Stk_i \leq Stk_{max_i} \quad \forall i \in 1...n$$

$$(5) \quad Peso_m \times Q_m + Peso_s \times Q_s \leq Peso_{max}$$

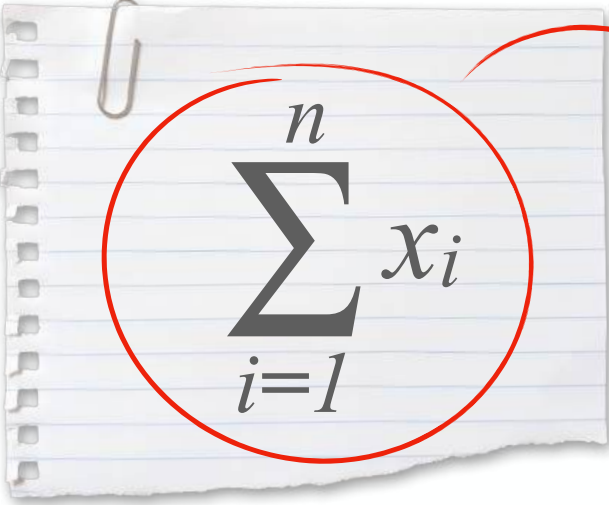
$$(6) \quad \sum_{i=1}^n C_i \times Q_i \leq Costo_{max}$$

$$(7) \quad Q_i + Stk_i \leq 1,1 \times D_i \quad \forall i \in 1...n$$

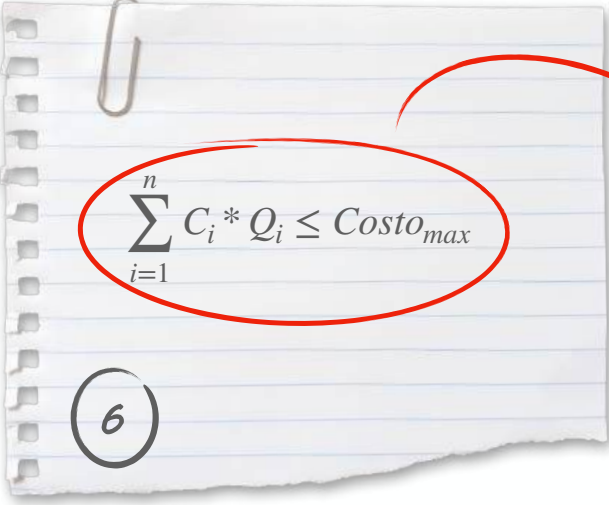
y busco dos cosas ...

las sumatorias van a implicar, el 99% de las veces, el uso de listas




$$\sum_{i=1}^n x_i$$

`modelo.sum(x[i] for i in range(n))`


$$\sum_{i=1}^n C_i * Q_i \leq Costo_{max}$$

`modelo.add_constraint(
 modelo.sum(
 costo[i]*Q[i]
 for i in range(len(producto))
) <= inv_max)`

6

DEMO 3



el "para todo" va a implicar, el 99%, de las veces el uso de bucles de repeticiones

$$Q_i \geq (Seg_i - Stk_i) \quad \forall i \in 1...n$$

3

```
for i in range(len(producto)):
    modelo.add_constraint(
        Q[i] >= seg[i] - stk[i]
    )
```

DEMO 4



*!ya tenemos nuestro
modelo dinámico j*

*otras bibliotecas de
optimización*

```

from docplex.mp.model import Model

modelo = Model(name='ejemplo')

x = modelo.continuous_var(name="x")
y = modelo.continuous_var(name="y")
modelo.maximize(30*x + 15*y)

modelo.add_constraint(x + 2*y <= 55)
modelo.add_constraint(x <= 50)
modelo.add_constraint(y <= 50)

sol = modelo.solve()

print("x", "=", sol[x])
print("y", "=", sol[y])
print("Objetivo: ", sol.get_objective_value())

```

docplex

Decision Optimization CPLEX



```
!pip install docplex
```



APACHE LICENSE, VERSION 2.0

YA LO HICIMOS
EN LA DEMO 1

```

import cplex

modelo = cplex.Cplex()

modelo.variables.add(obj=[30, 15], lb=[0, 0], ub=[50, 50], names=["x", "y"])
modelo.objective.set_sense(modelo.objective.sense.maximize)
modelo.linear_constraints.add(lin_expr=[["x", "y"], [1, 2]], senses=["L"], rhs=[55])
modelo.solve()

print("x", " = ", modelo.solution.get_values(["x"]))
print("y", " = ", modelo.solution.get_values(["y"]))
print("Objetivo: ", modelo.solution.get_objective_value())

```

IBM CPLEX



```
!pip install cplex
```



COMERCIAL LICENSE, IBM

EJEMPLO 1

```

from pycipopt import Model

modelo = Model()

x = modelo.addVar("x")
y = modelo.addVar("y")
modelo.setObjective(30*x + 15*y, sense="maximize")

modelo.addCons(x + 2*y <= 55)
modelo.addCons(x <= 50)
modelo.addCons(y <= 50)

modelo.optimize()

print("x", " = ", modelo.getBestSol()[x])
print("y", " = ", modelo.getBestSol()[y])
print("Objetivo: ", modelo.getObjVal())

```

pySCIPOpt



!pip install pycipopt



APACHE LICENSE, VERSION 2.0

EJEMPLO 2

```

import glpk

modelo = glpk.LPX()
modelo.obj.maximize = True

modelo.rows.add(3)
modelo.rows[0].bounds = None, 55.0
modelo.rows[1].bounds = None, 50.0
modelo.rows[2].bounds = None, 50.0

modelo.cols.add(2)
modelo.cols[0].bounds = 0.0, None
modelo.cols[1].bounds = 0.0, None

modelo.obj[:] = [30.0, 15.0]

modelo.matrix = [
    1.0, 2.0,
    1.0, 0.0,
    0.0, 1.0
]

modelo.simplex()

print("x", "=", modelo.cols[0].primal)
print("y", "=", modelo.cols[1].primal)
print("Objetivo: ", modelo.obj.value)

```

GLPK



EJEMPLO 3

```

from ortools.linear_solver import pywraplp

modelo = pywraplp.Solver.CreateSolver("GLOP")

x = modelo.NumVar(0, modelo.infinity(), "x")
y = modelo.NumVar(0, modelo.infinity(), "y")

objective = modelo.Objective()
objective.SetCoefficient(x, 30)
objective.SetCoefficient(y, 15)
objective.SetMaximization()

ct1 = modelo.Constraint(0, 55, "ct")
ct1.SetCoefficient(x, 1)
ct1.SetCoefficient(y, 2)

ct2 = modelo.Constraint(0, 50, "ct")
ct2.SetCoefficient(x, 1)

ct3 = modelo.Constraint(0, 50, "ct")
ct3.SetCoefficient(y, 1)

modelo.Solve()

print("x", "=", x.solution_value())
print("y", "=", y.solution_value())
print("Objetivo: ", objective.Value())

```



Google OR-Tools

!pip install ortools



APACHE LICENSE, VERSION 2.0

EJEMPLO 4

¿preguntas?

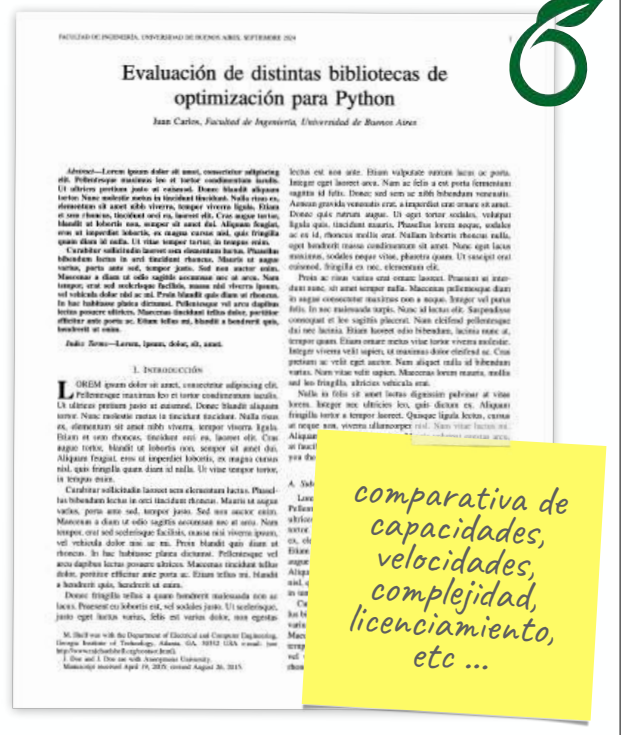




trabajo de investigación

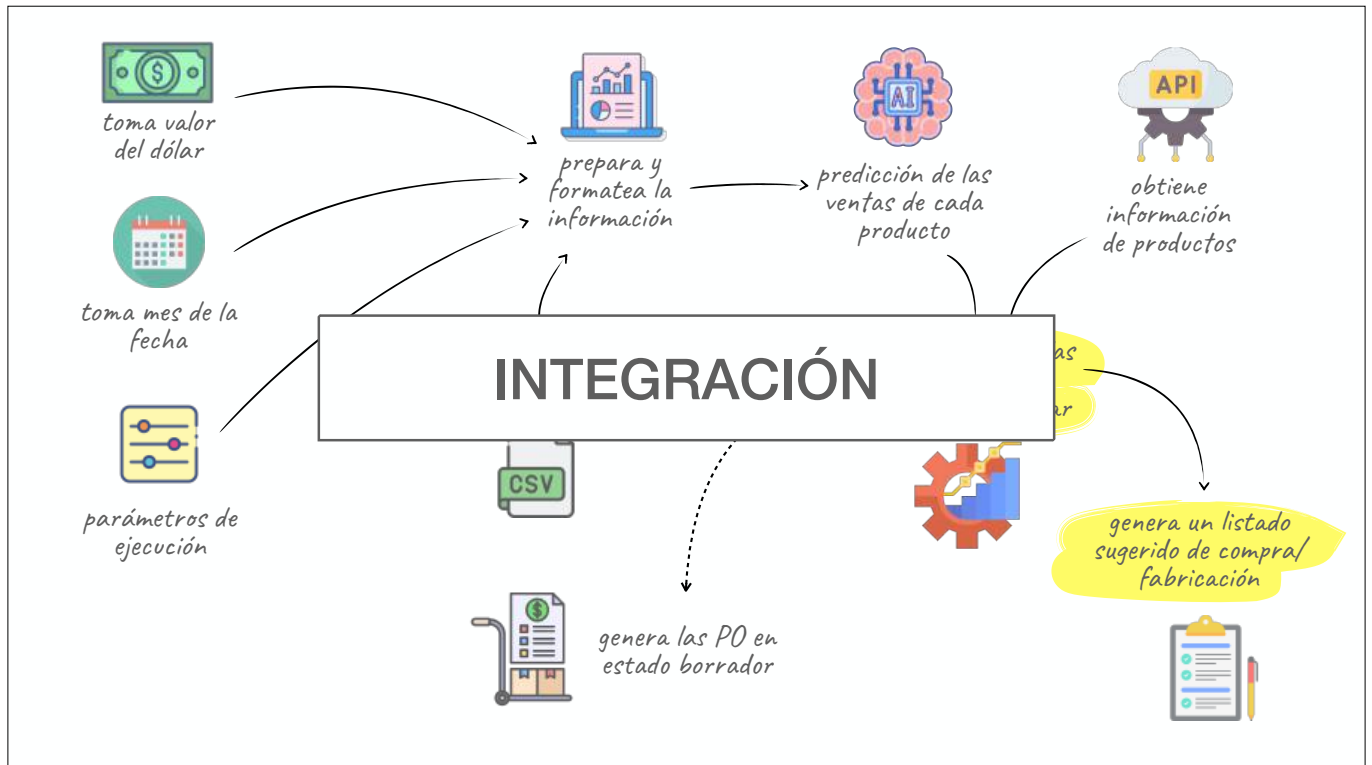
ex-coloquio

- Formato paper (overleaf [latex]).
- Aproximadamente 5-6 páginas.
- Si lo desean, les ayudamos a publicarlo.
- Los temas no se repiten ni se solapan.



ejemplo integrador



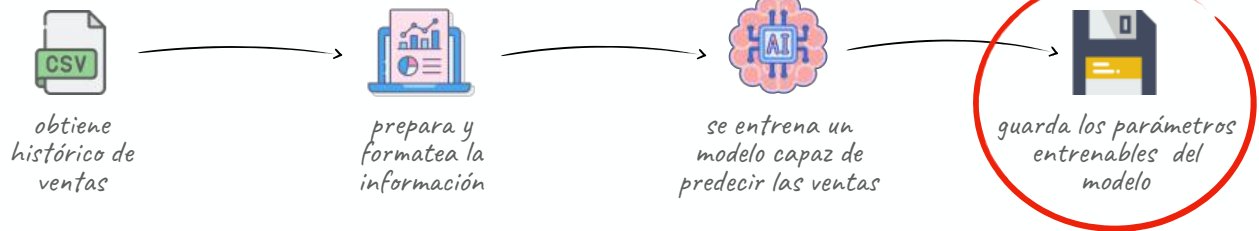


una aclaración

dado que hay un algoritmo de inteligencia artificial que debe ser entrenado, el esquema anterior tiene dos partes ...

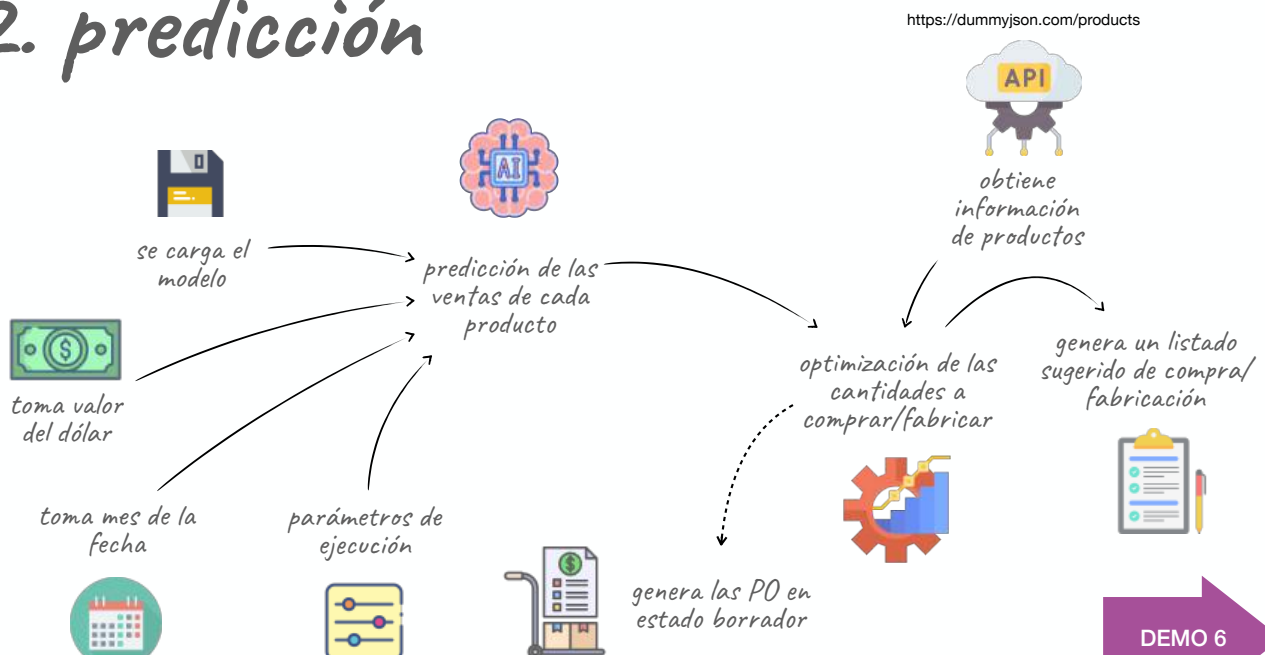
1. entrenamiento

```
torch.save(modelo.state_dict(), 'modelo_ia.pth')
```



DEMO 5

2. predicción



DEMO 6

algunas practicas optativas ...



- (1) hacer el ejercicio integrador cambiando el motor de optimización y exportar los resultados a un listado csv.
- (2) investigar posibilidades, ventajas y desventajas de compilar un archivo python (pista: ".pyc").



introducción
a python



modelo
ampliado



optimización
con python



formato
xml



intercambio
de datos



inteligencia
artificial



preprocesamiento
de datos



archivos
separados por
comas



interfaz de
programación de
aplicaciones



formato
json



próxima clase:

clase virtual

*nuevo módulo: planificación
introducción, programación de
restricciones y scheduling*

bibliografía y otros ...

[Python] Google ORTOOLS:

<https://developers.google.com/optimization>
<https://medium.com/google-or-tools>

[Python] SCIP:

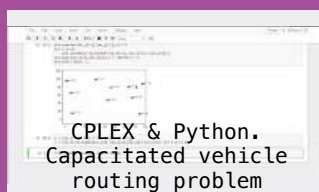
<https://www.scipopt.org/>
<https://scipopt.github.io/PySCIPOpt>

[Python] GLPK:

<https://pyglpk.readthedocs.io>
<https://www.gnu.org/software/glpk/>

[Python] IBM CPLEX

https://ibmdecisionoptimization.github.io/tutorials/html/Linear_Programming.html



CPLEX & TSP (Inglés)



ORTools (Inglés)

INVESTIGACIÓN OPERATIVA SUPERIOR

¡muchas gracias!