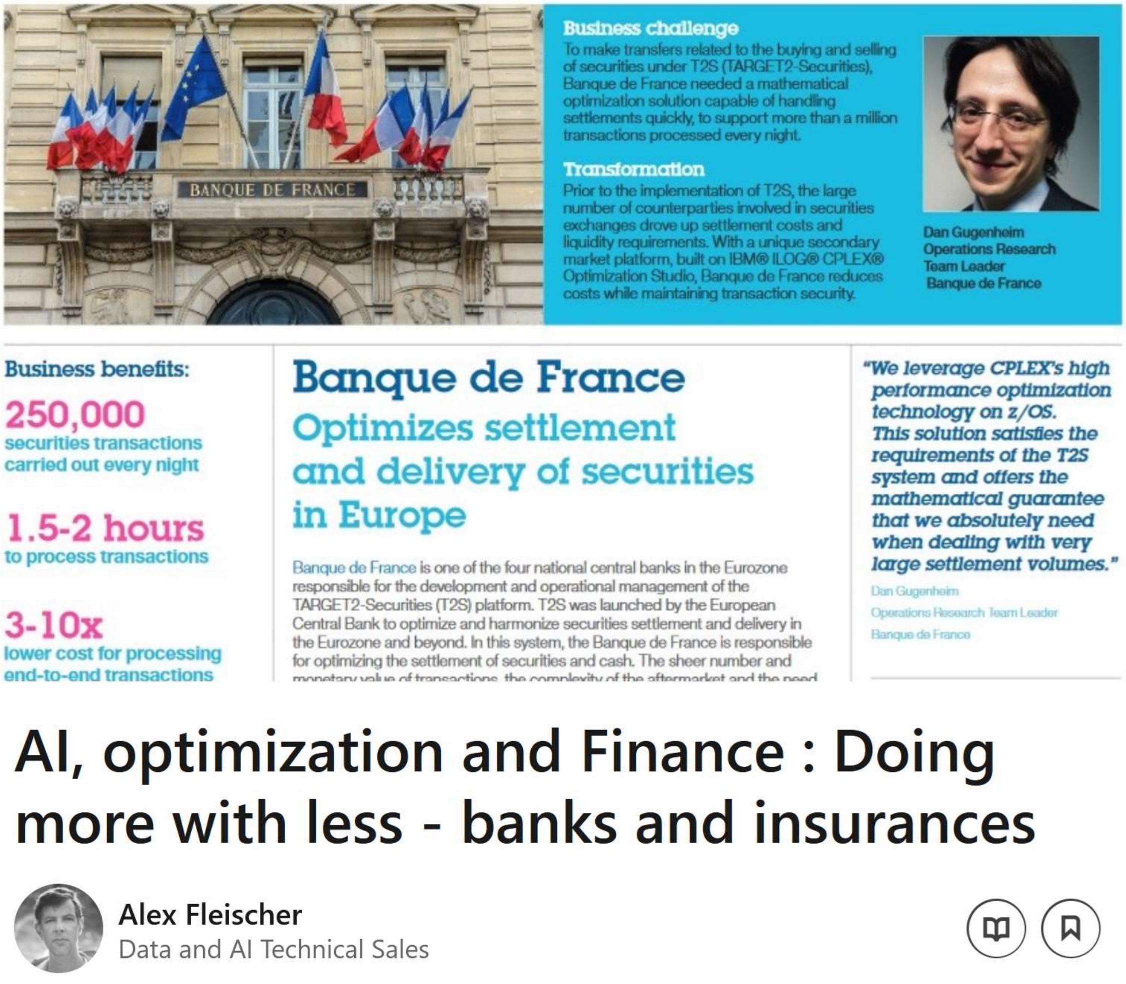
Challenge July 2024
Smart Investment

A solution with watsonx Llama3 405B that was released yesterday and CPLEX by Alex Fleischer afleischer@fr.ibm.com

2 comments:

1) With these values, a human can find the optimal solution quickly. LMM is the best stock so let's have as much as we can in LMM, 5000. 1000 for the 2 worst stocks XYZ and TTT and the 3000 remaining in the second best share, ABC.

2) 40+ banks rely on IBM CPLEX to optimize portfolios as I wrote in



Business challenge

To make transfers related to the buying and selling of securities under T2S (TARGET2-Securities), Banque de France needed a mathematical optimization solution capable of handling settlements quickly, to support more than a million transactions processed every night.

Transformation

Prior to the implementation of T2S, the large number of counterparties involved in securities exchanges drove up settlement costs and liquidity requirements. With a unique secondary market platform, built on IBM® ILOG® CPLEX® Optimization Studio, Banque de France reduces costs while maintaining transaction security.

Dan Gugenheim
Operations Research
Team Leader
Banque de France

Business benefits:

250,000
securities transactions carried out every night

1.5-2 hours
to process transactions

3-10x
lower cost for processing end-to-end transactions

Banque de France

Optimizes settlement and delivery of securities in Europe

Banque de France is one of the four national central banks in the Eurozone responsible for the development and operational management of the TARGET2-Securities (T2S) platform. T2S was launched by the European Central Bank to optimize and harmonize securities settlement and delivery in the Eurozone and beyond. In this system, the Banque de France is responsible for optimizing the settlement of securities and cash. The sheer number and monetary value of transactions, the complexity of the aftermarket and the need

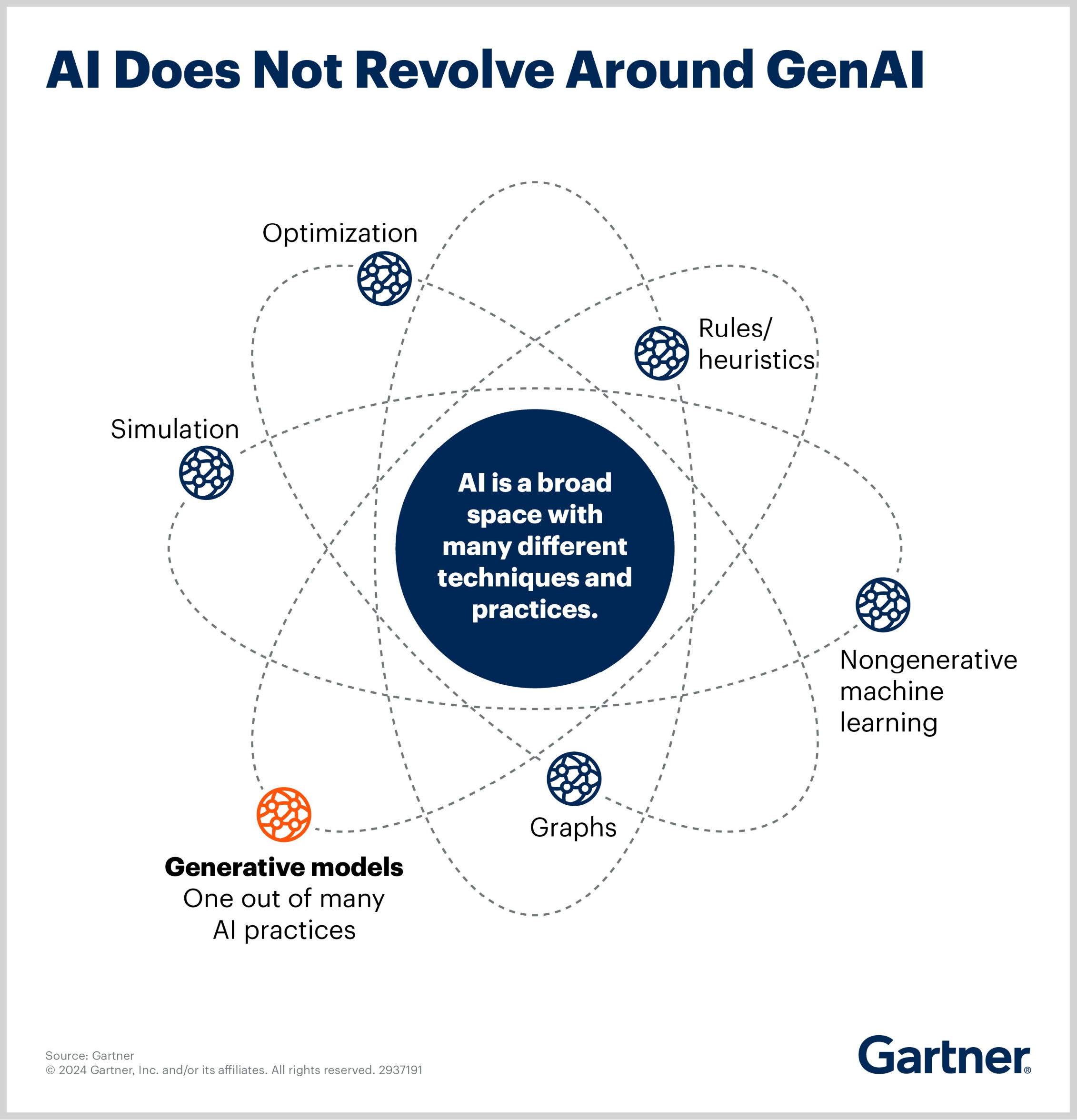
"We leverage CPLEX's high performance optimization technology on z/OS. This solution satisfies the requirements of the T2S system and offers the mathematical guarantee that we absolutely need when dealing with very large settlement volumes."

Dan Gugenheim
Operations Research Team Leader
Banque de France

AI, optimization and Finance : Doing more with less - banks and insurances

Alex Fleischer
Data and AI Technical Sales

AI is wider than generative AI as Gartner wrote. In IBM AI platform watsonx we have both generative AI and decision optimization.



AI Does Not Revolve Around GenAI

Optimization
Rules/ heuristics
Simulation
AI is a broad space with many different techniques and practices.
Nongenerative machine learning
Graphs
Generative models
One out of many AI practices
Gartner

Source: Gartner

© 2024 Gartner, Inc. and/or its affiliates. All rights reserved. 2937191

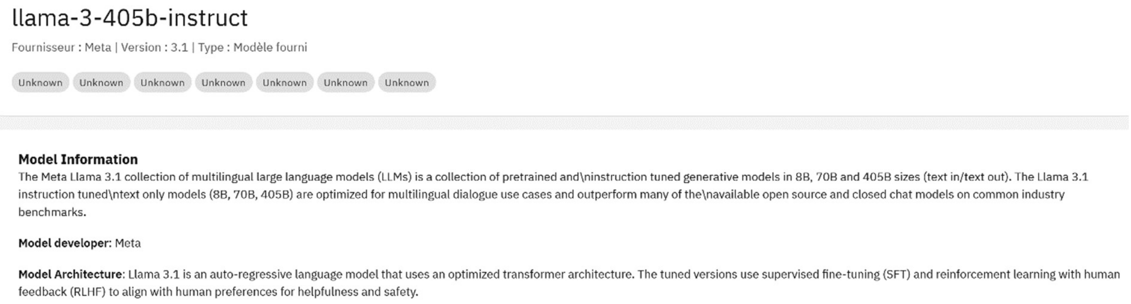
Anyway, let me share how to solve this puzzle in IBM watsonx with Llama3 and CPLEX.

Let me use this prompt:

A client of an investment firm has $10000 available for investment. He has instructed that his money be invested in particular stocks, so that no more than $5000 is invested in any

one stock but at least $1000 be invested in each stock. He has further instructed the firm to use its current data and invest in the manner that maximizes his overall gain during a one-year period. The stocks, the current price per share and the firm's predicted stock price a year from now are summarized below. ABC $25 $35 XYZ $50 $60 TTT $100 $125 LMN $25 $40 Write a docplex CPLEX python decision model that can be used to make a smart investment while satisfying the client requirements for different combinations of stocks.

And then choose the LLM (Large Language Model)



llama-3-405b-instruct

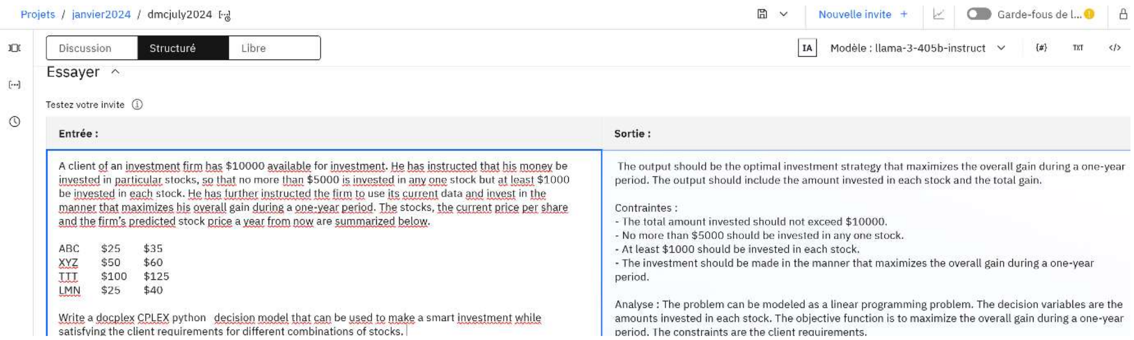
Fournisseur : Meta | Version : 3.1 | Type : Modèle fourni

Model Information
The Meta Llama 3.1 collection of multilingual large language models (LLMs) is a collection of pretrained and\ninstruction tuned generative models in 8B, 70B and 405B sizes (text in/text out). The Llama 3.1 instruction tuned\ntext only models (8B, 70B, 405B) are optimized for multilingual dialogue use cases and outperform many of the\navailable open source and closed chat models on common industry benchmarks.

Model developer: Meta

Model Architecture: Llama 3.1 is an auto-regressive language model that uses an optimized transformer architecture. The tuned versions use supervised fine-tuning (SFT) and reinforcement learning with human feedback (RLHF) to align with human preferences for helpfulness and safety.

It looks like



Entrée :

A client of an investment firm has $10000 available for investment. He has instructed that his money be invested in particular stocks, so that no more than $5000 is invested in any one stock but at least $1000 be invested in each stock. He has further instructed the firm to use its current data and invest in the manner that maximizes his overall gain during a one-year period. The stocks, the current price per share and the firm's predicted stock price a year from now are summarized below.

ABC	$25	$35
XYZ	$50	$60
TTT	$100	$125
LMN	$25	$40

Write a docplex CPLEX python decision model that can be used to make a smart investment while satisfying the client requirements for different combinations of stocks.

Sortie :

The output should be the optimal investment strategy that maximizes the overall gain during a one-year period. The output should include the amount invested in each stock and the total gain.

Contraintes :
- The total amount invested should not exceed $10000.
- No more than $5000 should be invested in any one stock.
- At least $1000 should be invested in each stock.
- The investment should be made in the manner that maximizes the overall gain during a one-year period.

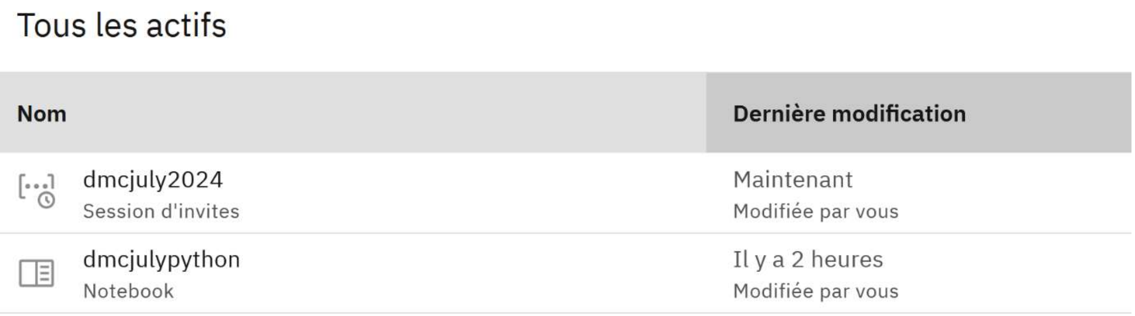
Analyse : The problem can be modeled as a linear programming problem. The decision variables are the amounts invested in each stock. The objective function is to maximize the overall gain during a one-year period. The constraints are the client requirements.

And the output is

```
from docplex.mp.model import Model

# Create a model
model = Model("Investment")

# Define the decision variables
X_ABC = model.continuous_var(name="X_ABC", lb=1000, ub=5000)
X_XYZ = model.continuous_var(name="X_XYZ", lb=1000, ub=5000)
```

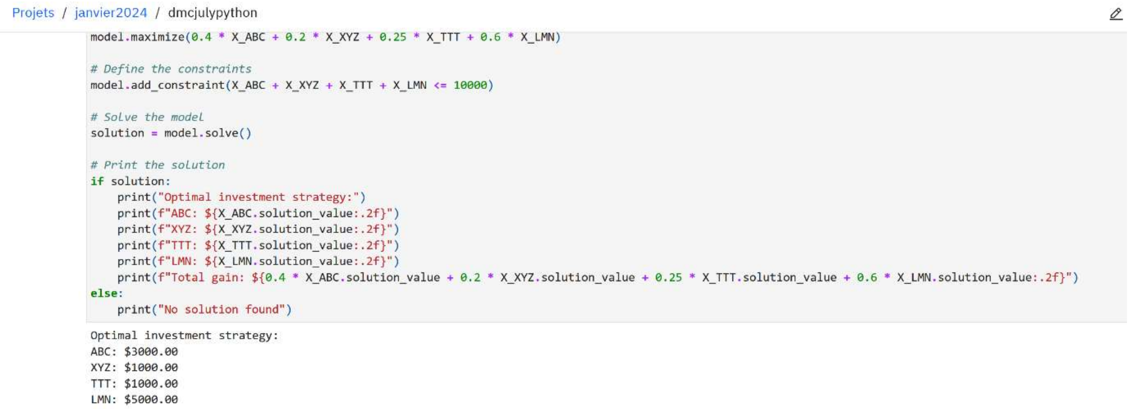
```
X_TTT = model.continuous_var(name="X_TTT", lb=1000, ub=5000)
X_LMN = model.continuous_var(name="X_LMN", lb=1000, ub=5000)

# Define the objective function
model.maximize(0.4 * X_ABC + 0.2 * X_XYZ + 0.25 * X_TTT + 0.6 * X_LMN)

# Define the constraints
model.add_constraint(X_ABC + X_XYZ + X_TTT + X_LMN <= 10000)

# Solve the model
solution = model.solve()

# Print the solution
print("Optimal investment strategy:")
print(f"ABC: {solution[X_ABC]}")
print(f"XYZ: {solution[X_XYZ]}")
print(f"TTT: {solution[X_TTT]}")
print(f"LMN: {solution[X_LMN]}")
print(f"Total gain: {0.4 * solution[X_ABC] + 0.2 * solution[X_XYZ] + 0.25 * solution[X_TTT] + 0.6 * solution[X_LMN]}")
```

which I can directly run within watsonx.AI through a python notebook



Tous les actifs

Nom	Dernière modification
dmcjuly2024 Session d'invites	Maintenant Modifiée par vous
dmcjulypython Notebook	Il y a 2 heures Modifiée par vous



```python
model.maximize(0.4 * X_ABC + 0.2 * X_XYZ + 0.25 * X_TTT + 0.6 * X_LMN)

# Define the constraints
model.add_constraint(X_ABC + X_XYZ + X_TTT + X_LMN <= 10000)

# Solve the model
solution = model.solve()

# Print the solution
if solution:
    print("Optimal investment strategy:")
    print(f"ABC: ${X_ABC.solution_value:.2f}")
    print(f"XYZ: ${X_XYZ.solution_value:.2f}")
    print(f"TTT: ${X_TTT.solution_value:.2f}")
    print(f"LMN: ${X_LMN.solution_value:.2f}")
    print(f"Total gain: ${0.4 * X_ABC.solution_value + 0.2 * X_XYZ.solution_value + 0.25 * X_TTT.solution_value + 0.6 * X_LMN.solution_value:.2f}")
else:
    print("No solution found")
```

```
Optimal investment strategy:
ABC: $3000.00
XYZ: $1000.00
TTT: $1000.00
LMN: $5000.00
```

And I see

```
Optimal investment strategy:
ABC: $3000.00
XYZ: $1000.00
TTT: $1000.00
LMN: $5000.00
Total gain: $4650.00
```

Mixing generative AI and optimization … that’s hybrid AI!