Empirical Model Learning

An Industrial Point of View

Alessio Bonfietti
Thomas Bridi

11-03-2019

MindIT

Good News

Statement 1: declarative Combinatorial Optimization methods can be applied successfully to real world problems

Declarative Combinatorial Optimization

- Mixed Integer Linear Programming
- Constraint Programming
- SAT
- ...

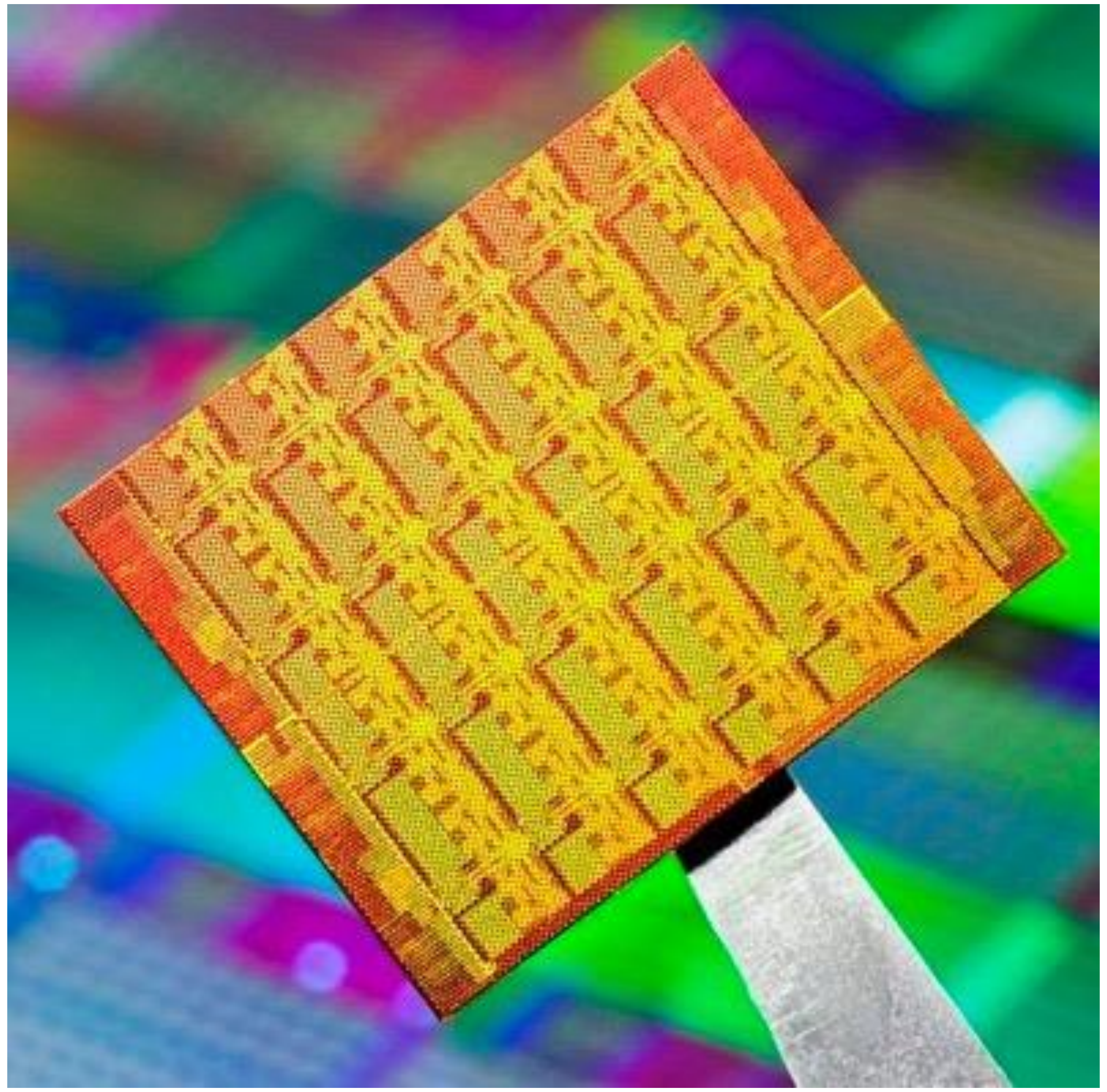
Bad News

Statement 2: declarative Combinatorial Optimization methods are not always successful on real world problems

Sometimes they are not even worth a try

If someone asks, I didn’t said this, ok? ;-)

Know your enemy



An experimental CPU by Intel:

Intel SCC
Single-chip Cloud Computer

- The "father" of Xeon Phi
- **48 cores**
- connected via a **Network on Chip**
- designed to process **job batches**

*This work has been partially funded by the **FP7 ERC Multithermal Project** and by the **Google Focused Grant Program** on Mathematical Optimization and Combinatorial Optimization in Europe*

Know your enemy

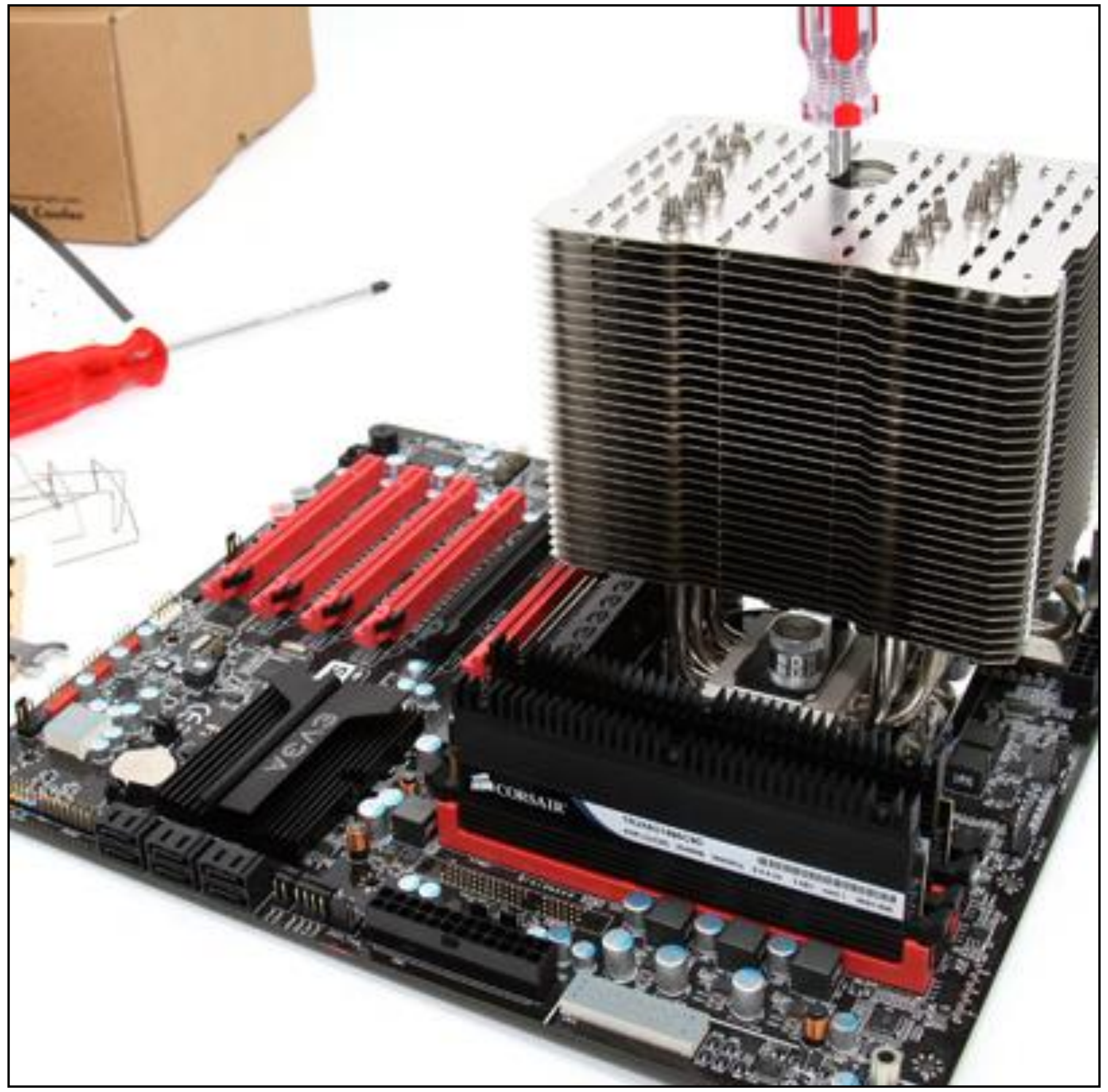
An experimental CPU by Intel:

Intel SCC
Single-chip Cloud Computer

- The “father” of Xeon Phi
- **48 cores**
- connected via a **Network on Chip**
- designed to process **job batches**

This thing is a burner!

Know your enemy

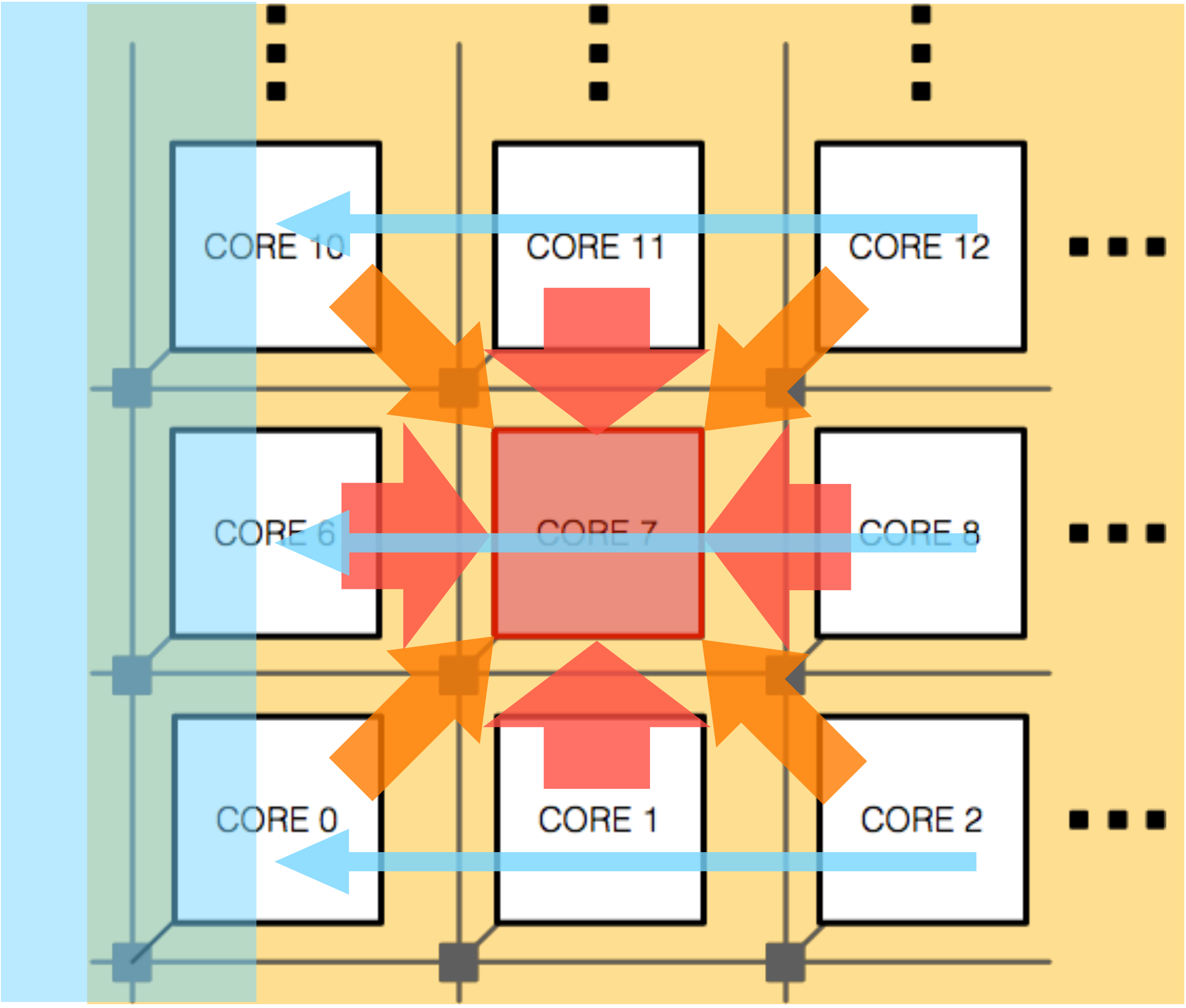


The current solution

☺

Know your enemy

The effect of dispatching decisions is hard to predict



Because it is affected by:

- the room temperature
- the core workload
- the neighbor workload
- the heat sink position
- ...

Know your enemy

You have...

- A lot of interacting agents?
- Autonomous behavior / feedback?
- Memory effects?
- Complex, dynamic rules, regulating your system?
- ....

Then you'll have:

Modeling Difficulties

(and you are in good company)

Know your enemy

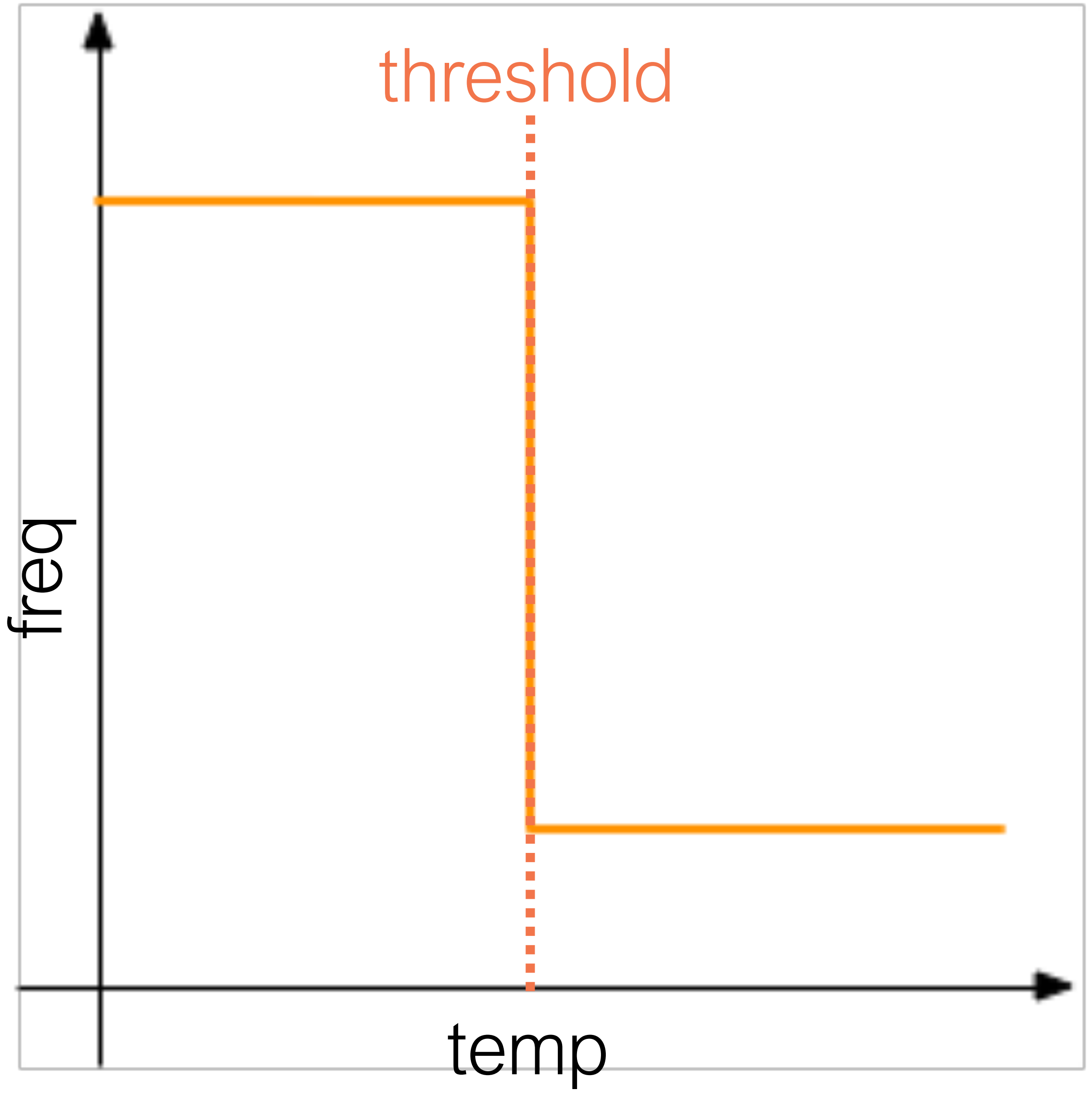
You have...

- A lot of interacting agents?
- Autonomous behavior / feedback?
- Memory effects?
- Complex, dynamic rules, regulating your system?
- ....

Let’s be precise

- It’s hard to build a declarative model
- But you can often build a numerical model (e.g. a simulator)
- Which can be slow...

Know your enemy



Intel SCC

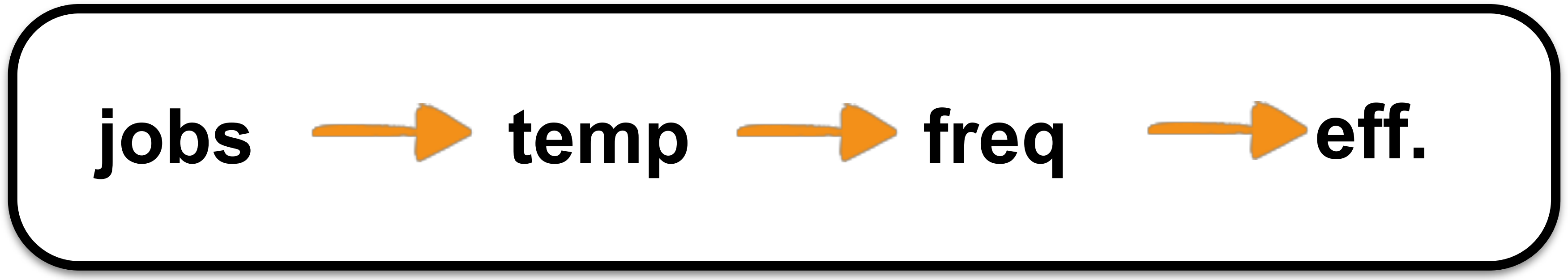
- The “father” of Xeon Phi
- 48 cores
- Accepts job batches

This thing is a burner!

Sol: **thermal controller**

temp → **freq** → **eff.**

Know your enemy



An interesting problem:

- Map a batch of jobs
- Maximize #cores with high efficiency

How to model jobs → eff?

- No declarative model, but a simulator is available
- Use the simulator? → **no way, too slow**

We learn the relation
`mapping` → … → `efficiency`
via an Artificial Neural Network

Empirical Model Learning

EML is a method to deal with complex, hard to model, systems in declarative combinatorial optimization

EMPIRICAL MODEL LEARNING

Usage Directions:

- Obtain an approximate system via your favorite Machine Learning technique
- Embed it into a Combinatorial Optimization Model
- Solve and enjoy

42000 06200

Revenue Maximization

Another example

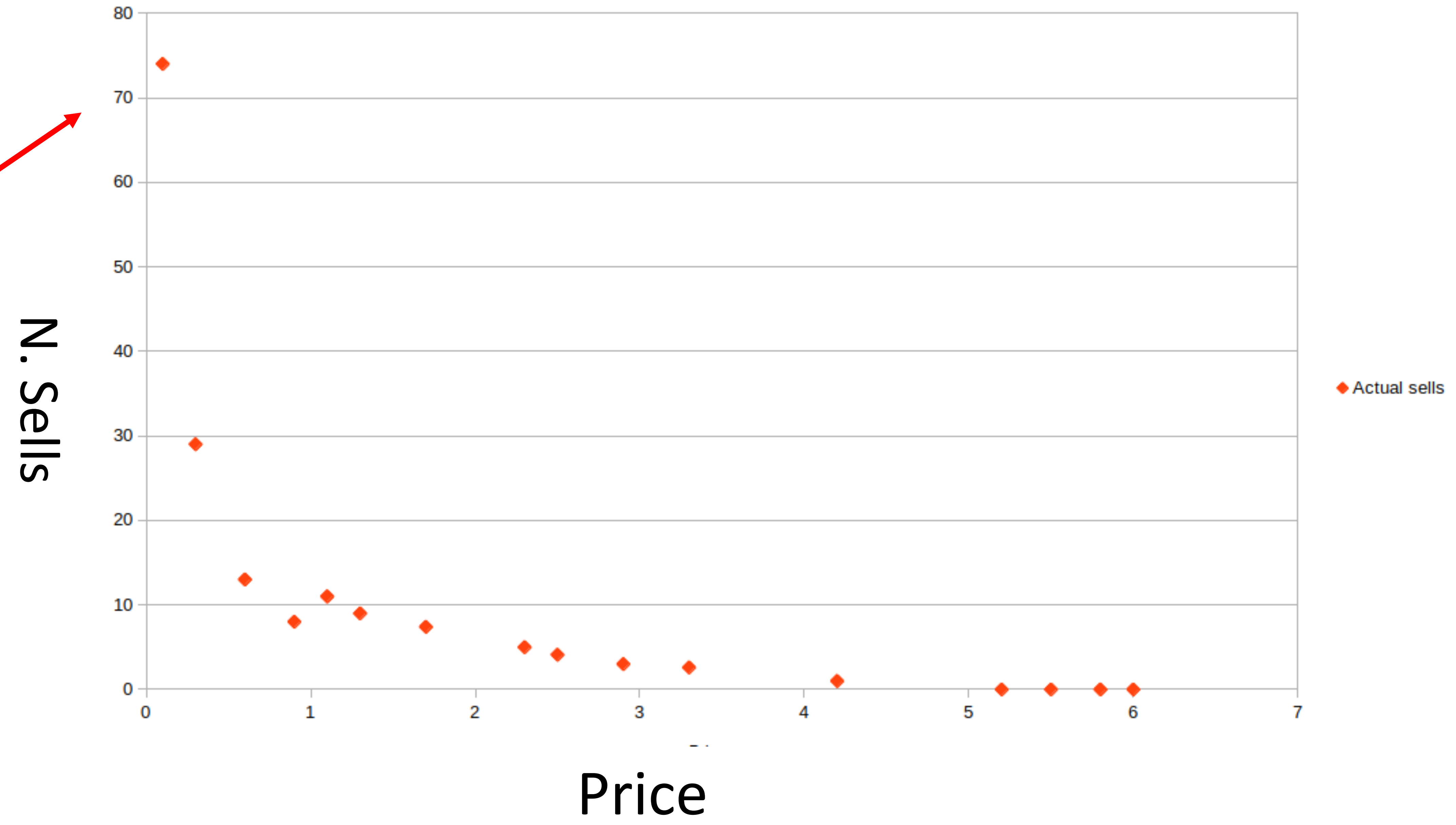
- Given a product to sel, set the price in order to maximize the revenue

Suppose you are selling oranges and you want to set the price to maximize the revenue.

Revenue Maximization

Another example

- Try different prices and record the number of sells



low prices corresponds to a high number of sells

Revenue Maximization

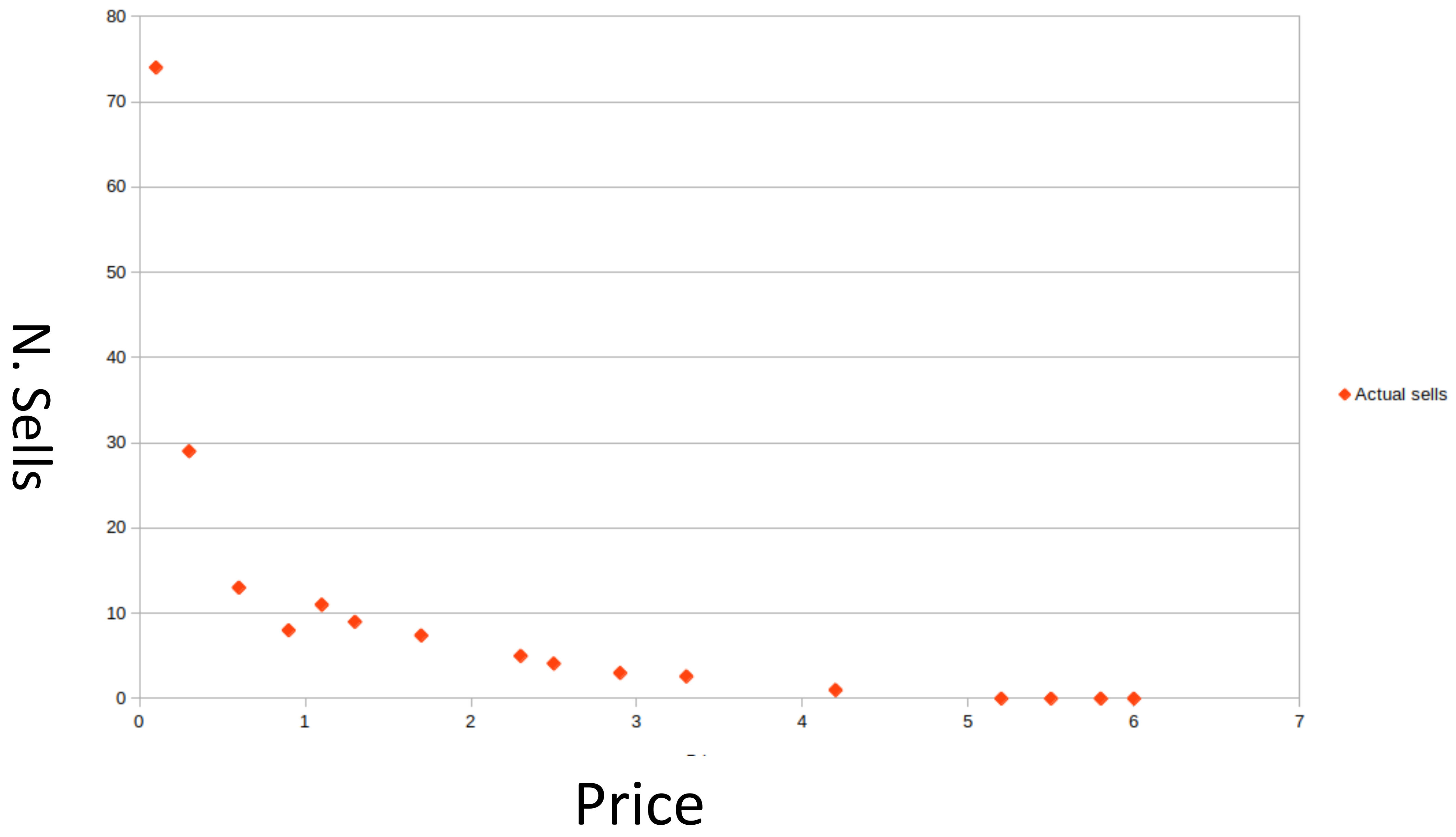
Another example

- And then? How can I continue to model this problem?

It is difficult to continue to model this problem to find an optimal solution.

I could just compute the Revenue and choose the price between those tested that gives the maximum revenue.

But we can do better!



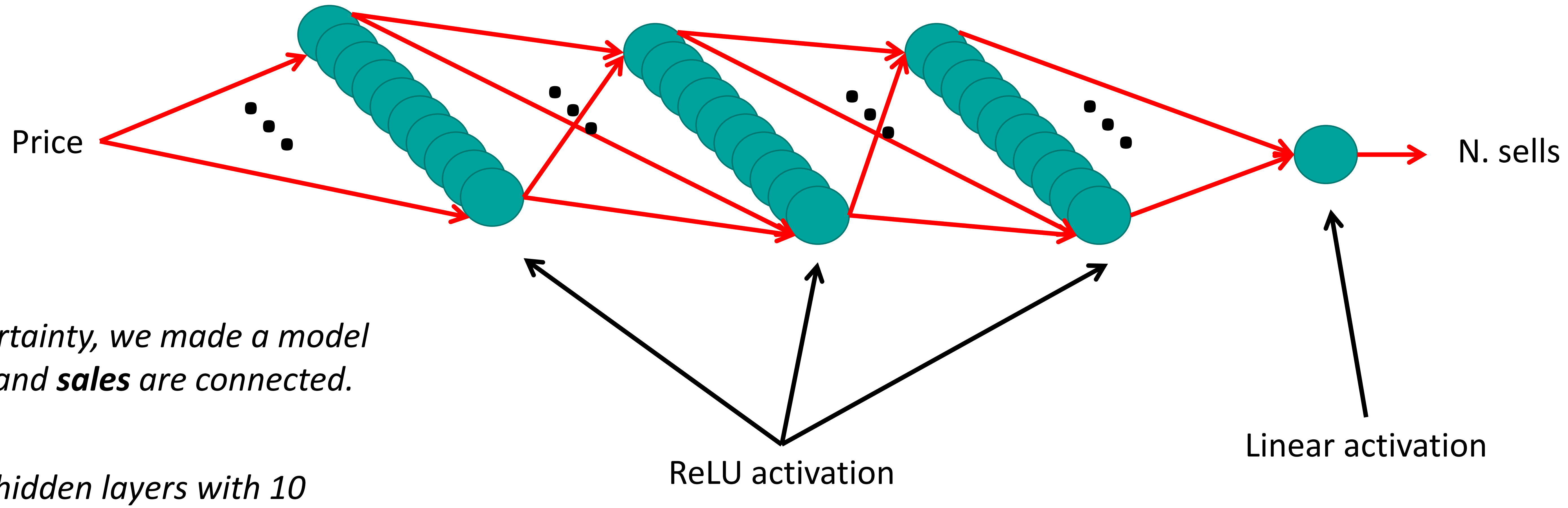
Empirical Model Learning (EML)

- Model uncertainty through machine learning
- Train and validate the model
- Encode and embed the ML model into the optimization model
- Solve

Revenue Maximization

Model uncertainty through machine learning

- Let’s learn how the price affects the number of sells through a machine learning model



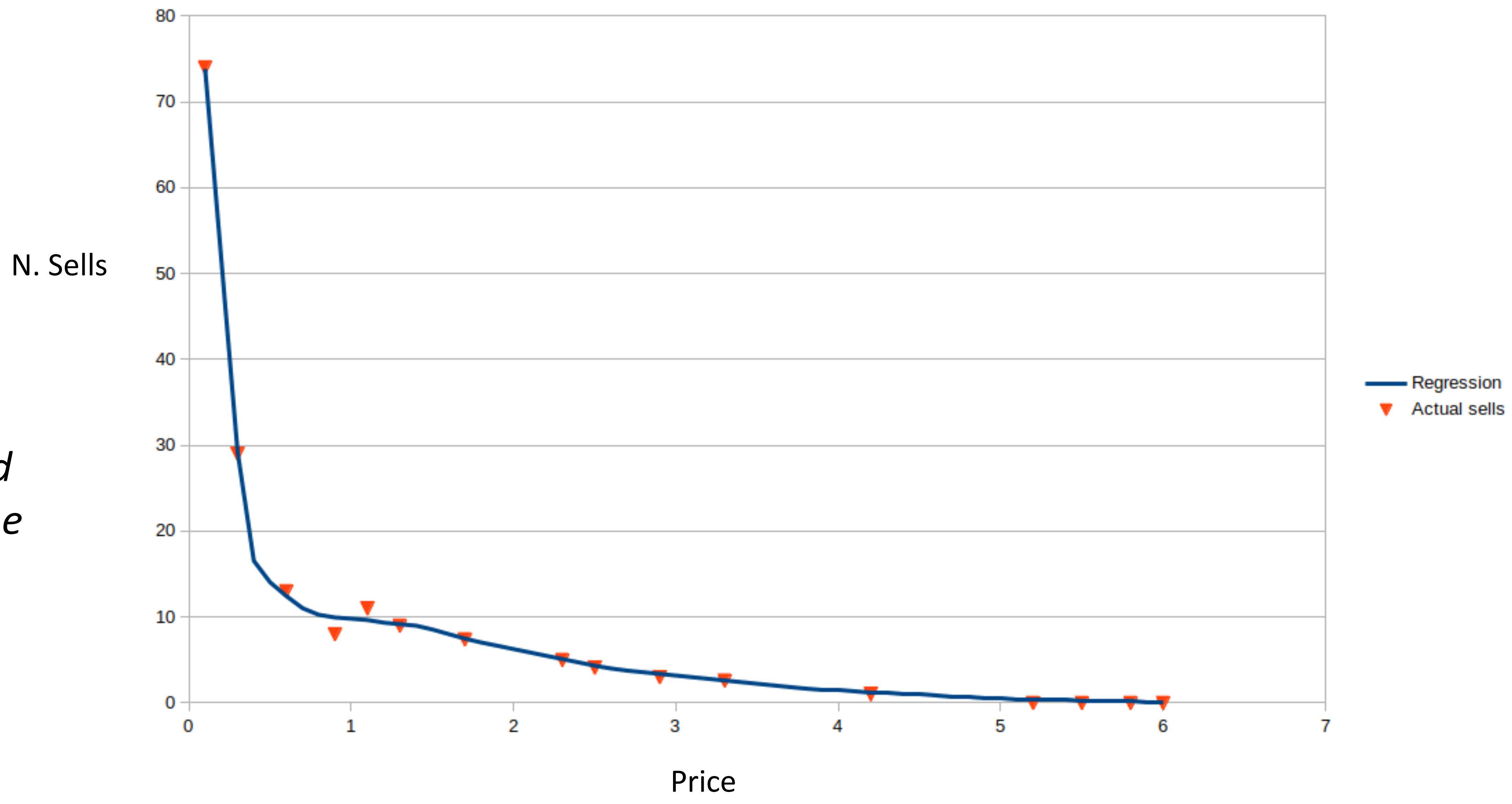
*Starting with the uncertainty, we made a model that learns how **Price** and **sales** are connected.*

The network counts 3 hidden layers with 10 neurons each with the Rectifier Linear Unit (ReLU) as activation function.

Revenue Maximization

Validate the model

- Let’s learn how the price affects the number of sells through a machine learning model



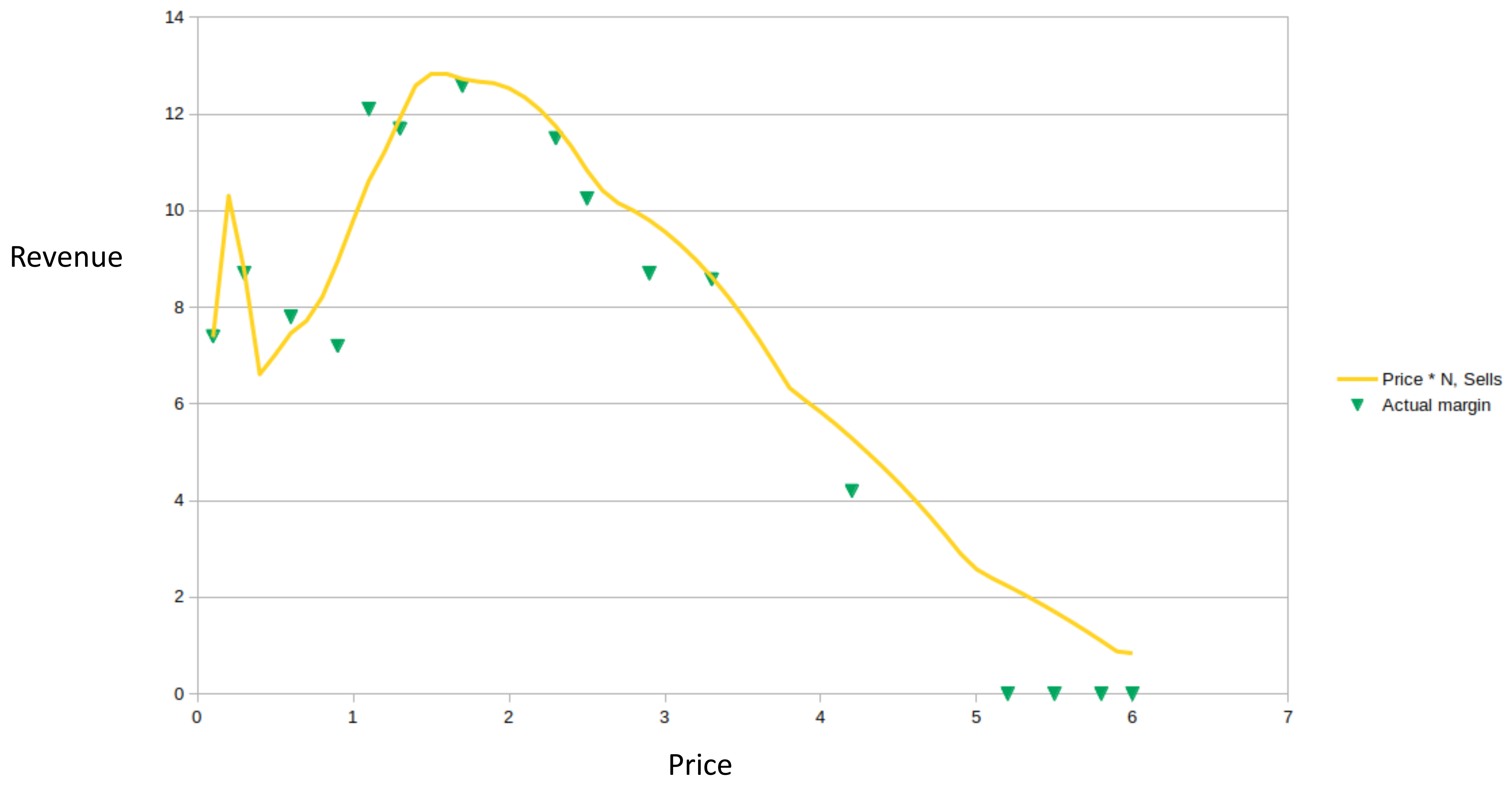
It seems quite a good result and being a regression it gives us the possibility to forecast the sales whatever the input price

Revenue Maximization

Obtain the revenue

- Revenue = Predicted N. of sells * Price

We can therefore compute the Revenue as the number of sells times the price

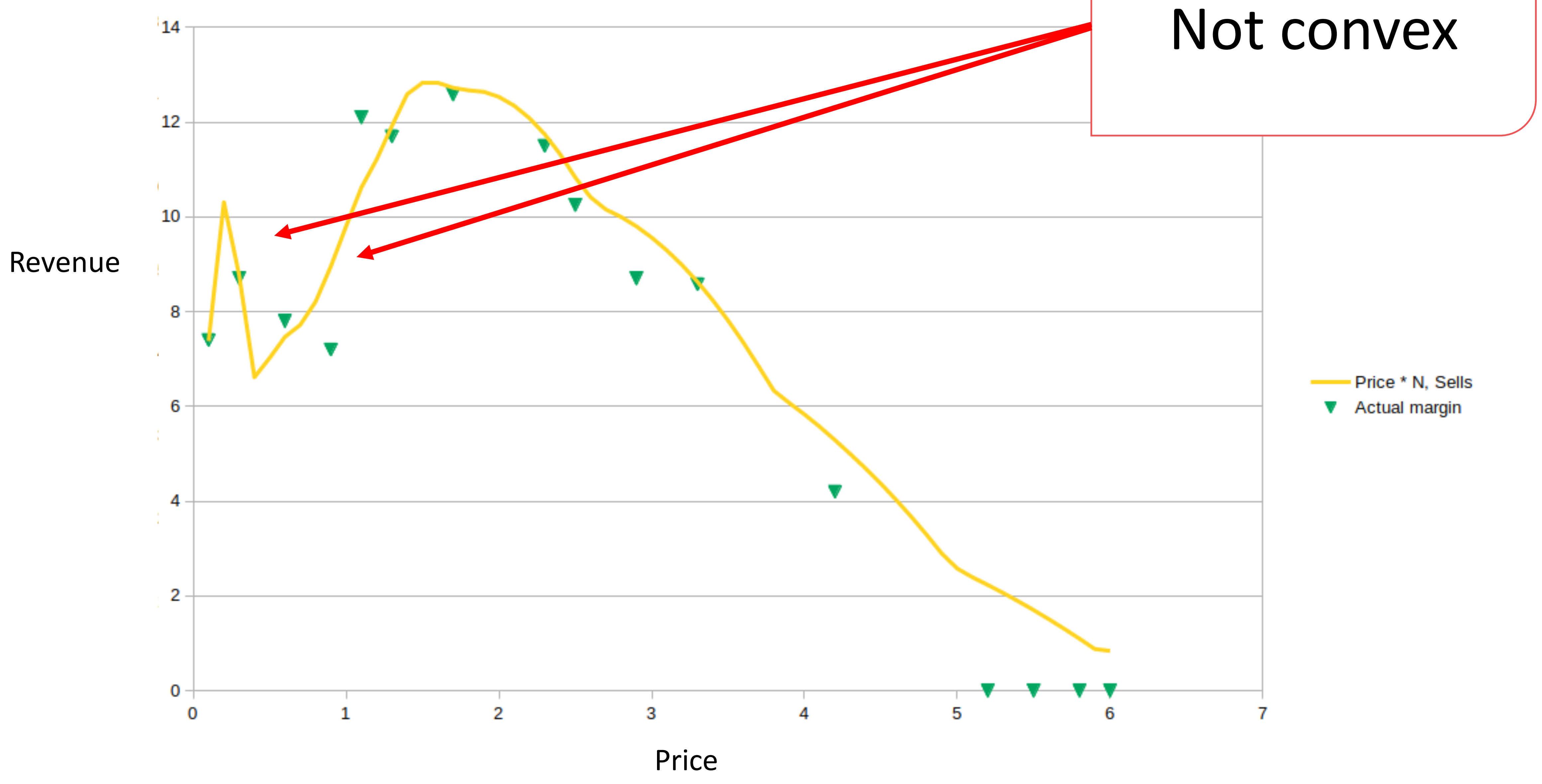


the actual revenue in green and the result of our neural network multiplied the price in yellow

Revenue Maximization

Obtain the revenue

- Revenue = Predicted N. of sells * Price



Not convex

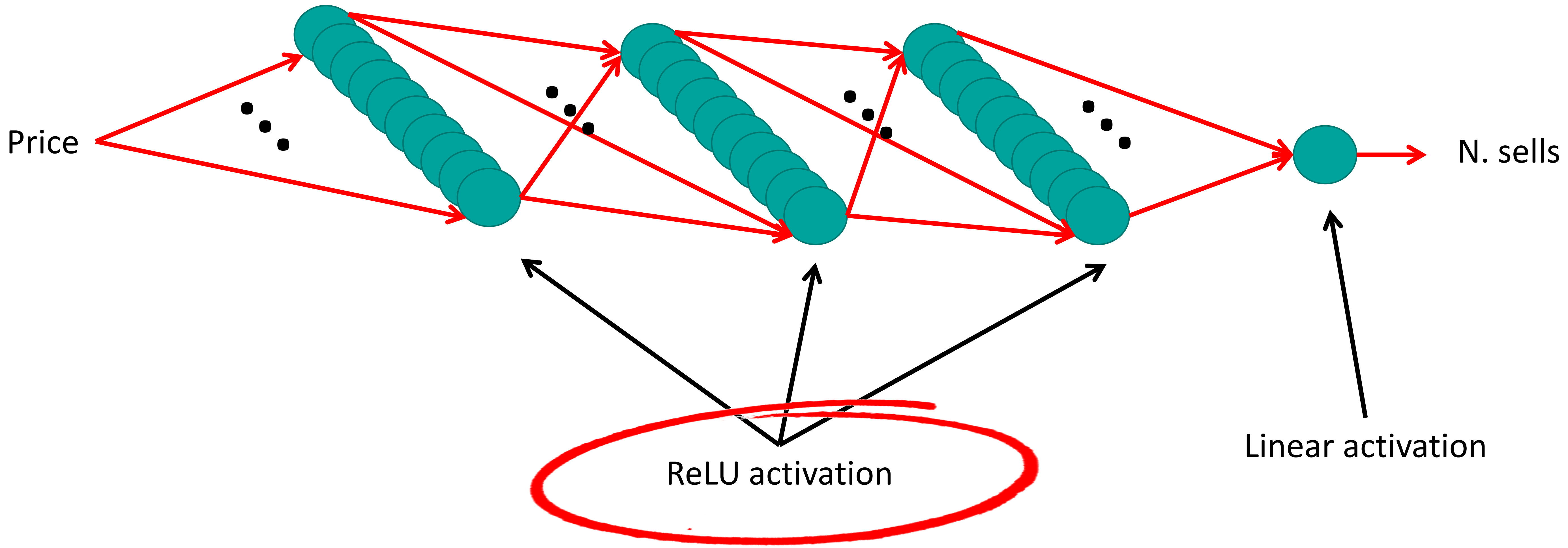
Empirical Model Learning

- ✓ Model uncertainty through machine learning
- ✓ Validate the model
- Encode and embed the ML model into an optimization model
- Solve

Let’s choose Mixed Integer Programming

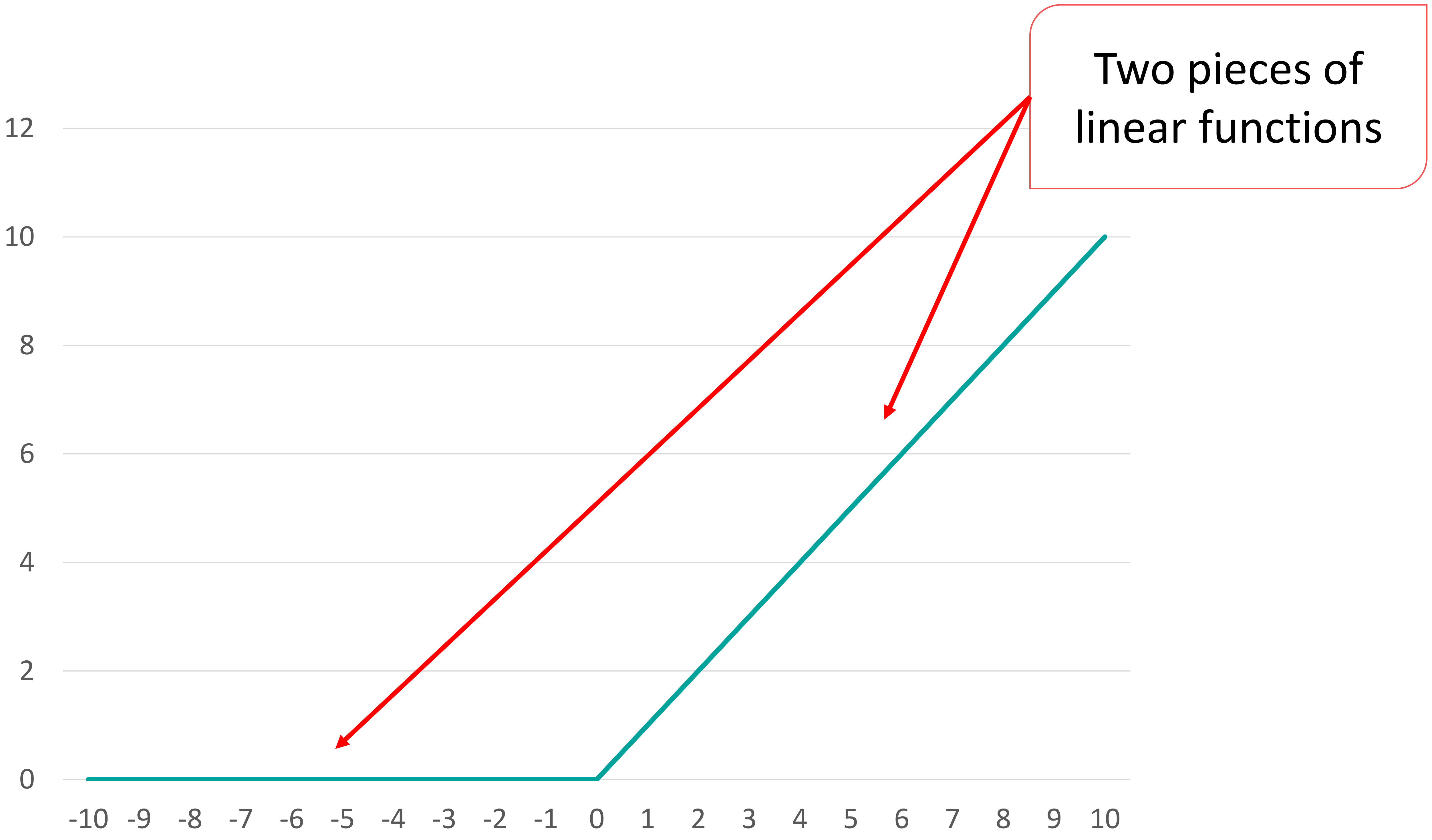
Revenue Maximization

Machine Learning



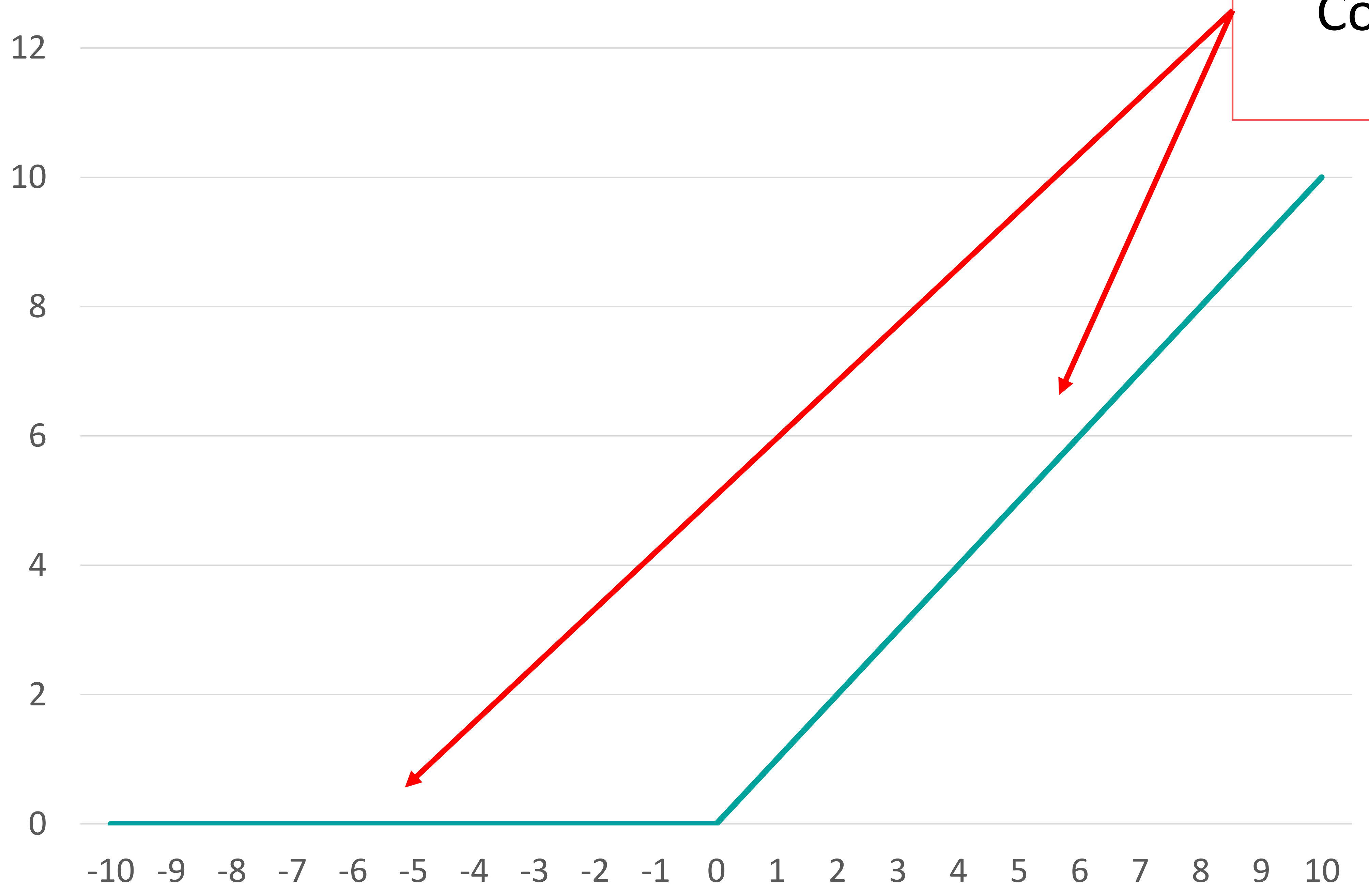
ReLU

This function is zero for negative inputs and linear in the input for positive inputs



two pieces of linear functions

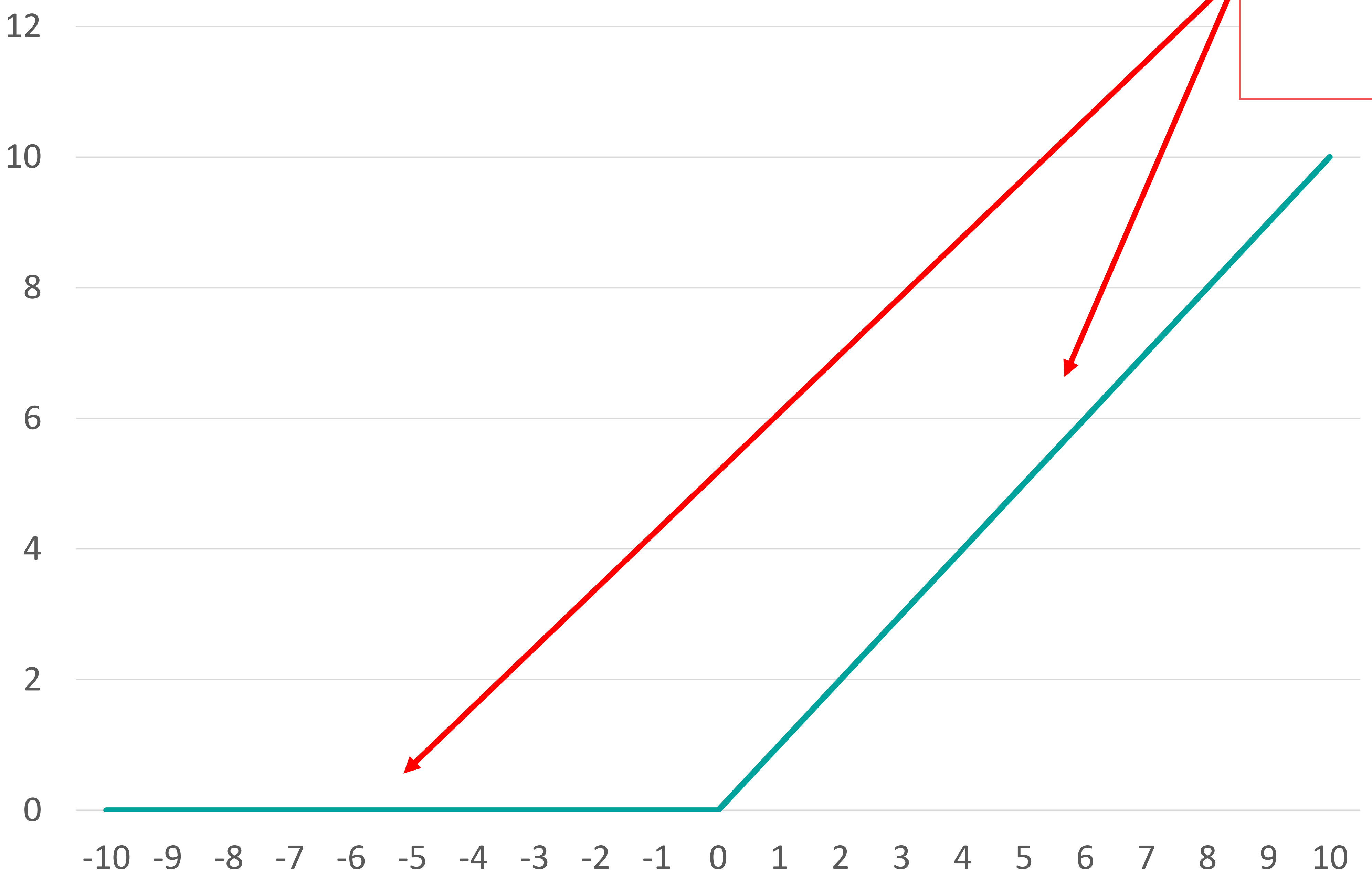
ReLU



<u>Special Order Set</u>

Constraint of
type 1

ReLU



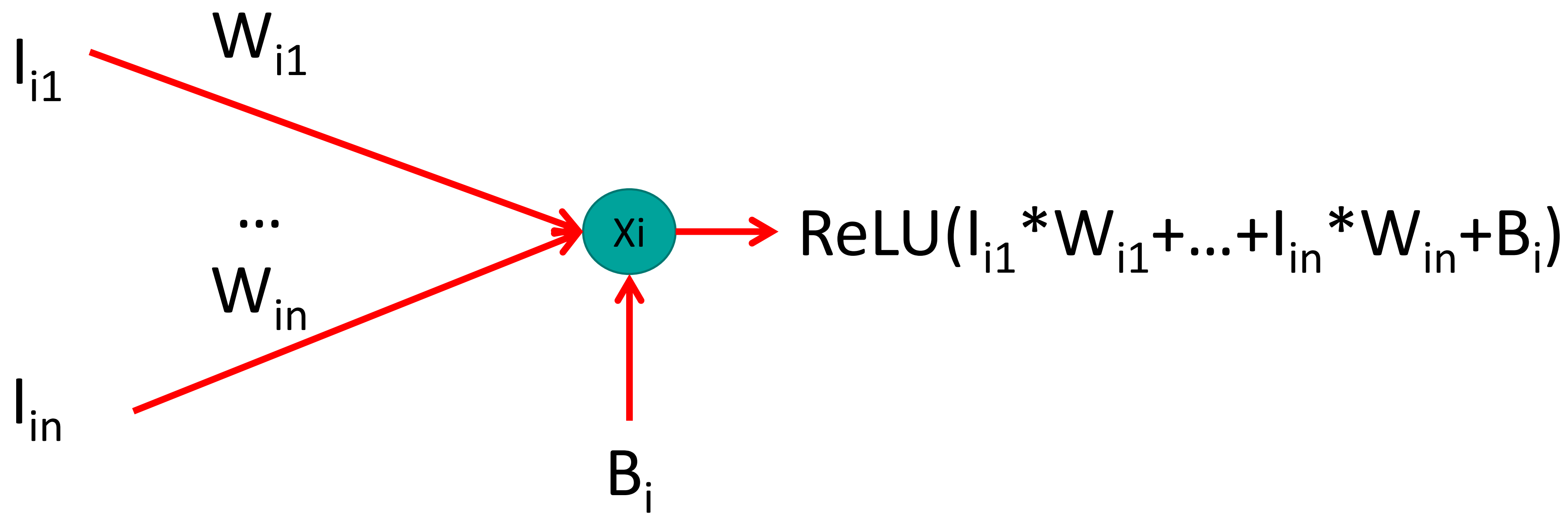
Being the domain [-inf, inf]
Just one Integer variable per neuron!!!!

This means that our optimization model needs only one integer variable for each neuron of the neural network.

Revenue Maximization

Encode and embed the ML model into an optimization model

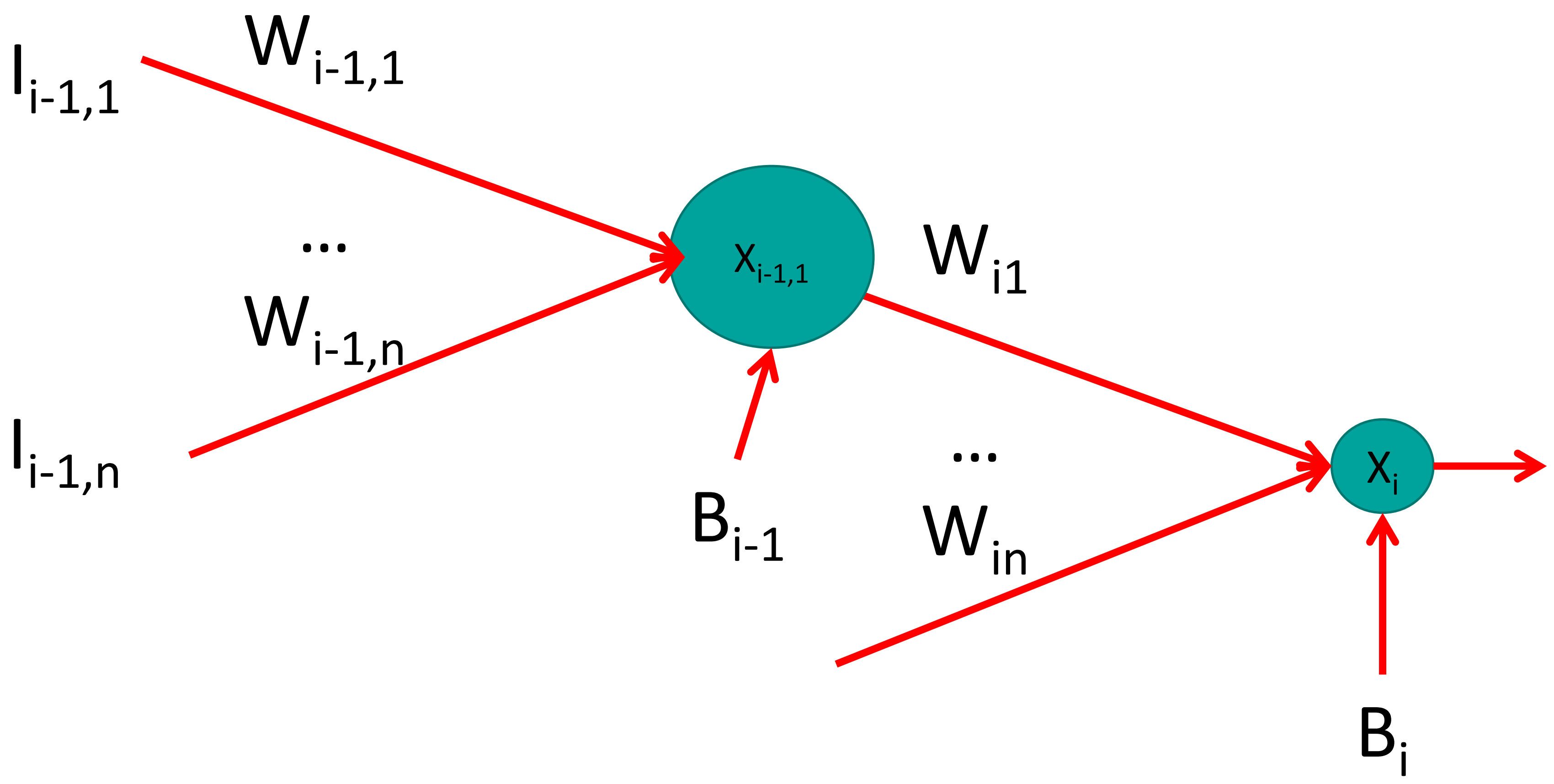
- The input of a neuron is a sum of multiplications (I_{ij} * W_{ij})
- The bias of a neuron is a sum ($+B_i$)
- The ReLU activation is a special case of a SOS1 constraint (ReLU(X_i))



Revenue Maximization

Encode and embed the ML model into an optimization model

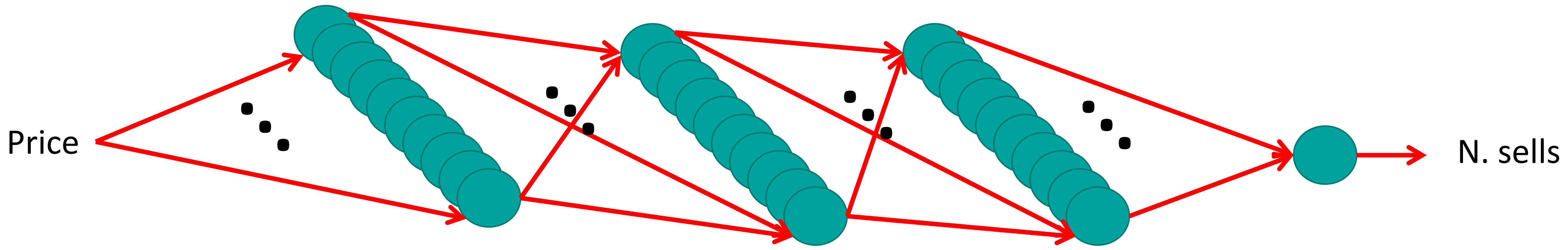
- Repeat recursively for each neuron from the last to the first



$\mathrm{ReLU}(\mathrm{ReLU}(X_{i-1,1})*W_{i1}+\ldots+\mathrm{ReLU}(X_{i-1,k})*\ *W_{in}+B_i)$

Revenue Maximization

Repeat for each neuron and compose your NeuralNetwork(Price) Constraint



The NeuralNetwork cst constraints the sales to be equal to the value learned by our machine learning model given a price.

Revenue Maximization

Finally, the model:

Minimize: -Price * Sells

s.t.

Sells = NeuralNetwork(Price)

Price > 0, Sells >0, …

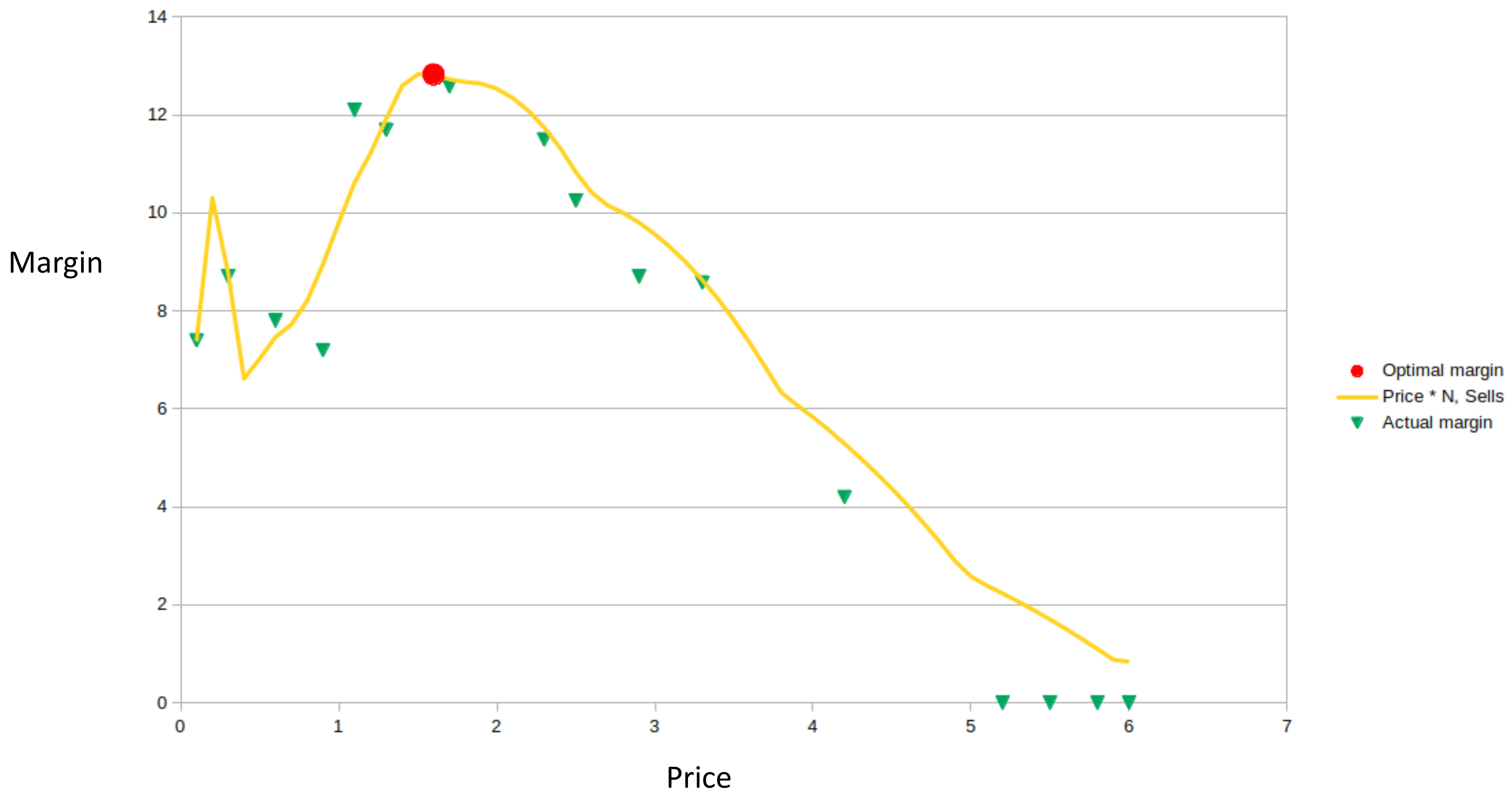
Mixed Integer Quadratic Program

Empirical Model Learning

- ✓ Model uncertainty through machine learning
- ✓ Validate the model
- ✓ Encode and embed the ML model into an optimization model
- Solve

Revenue Maximization

Solve



EML

Final Remarks

- Initially CP implementation: Neural Networks, Decision Trees, Random Forests

- High modeling constraints (e.g. you should not use MILP with Sigmoid as neuron activation)
- Needs some effort to obtain good performance
- Needs off-line training
- Needs a good dataset

It's not about ideas.
It's about making ideas happen

MindIT Contacts

www.minditsolutions.it

info@minditsolutions.it

Location
Viale Fanin 48
Bologna 40127
IT

Mobile
+393703458445

Iscr.Reg.Impr.BO, P.I. e C.F: 03614631202

MindIT

Make IT Simple