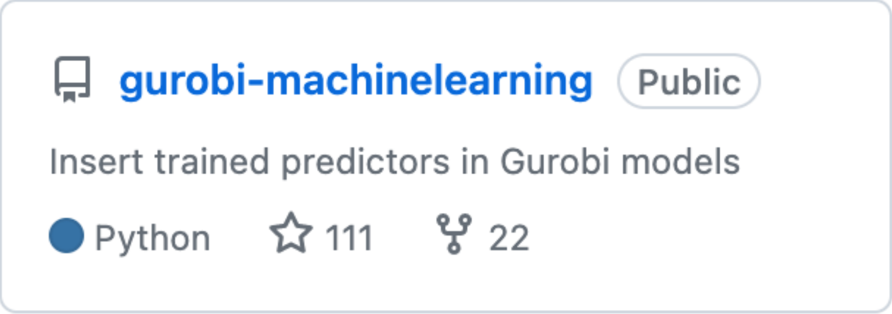


Gurobi Machine Learning:

Incorporate your Machine Learning Models into Optimization

Usage and model tuning

Zed Dean
Technical Account Manager

Alison Cozad
Gurobi Expert Team NA Manager

April 2023

Agenda

Open Source, sponsored by Gurobi

Collaborate, Experiment, and Innovate with the Gurobi Team

Gurobi Machine Learning Package

Why, What, How

Types of Model Errors/Residuals

Tuning in Machine Learning Package

Logistic regression with piece-wise approximations (PWA)

Decision Tree, Random Forest & Gradient boosting.

Neural Network

Final thoughts

GUROBI
OPTIMIZATION

Open Source Projects

Sponsored by Gurobi

We aim to foster a collaborative community around Gurobi by openly developing various optimization projects and tools, making them more accessible and user-friendly.

Our users can experiment with our innovative tools, providing **direct feedback to our developers** responsible for creating these packages.

github.com/Gurobi

gurobi-machinelearning Public

Insert trained predictors in Gurobi models

Python 111 22

gurobipy-pandas Public

Convenience wrapper for building optimization models from pandas data

Python 50 13

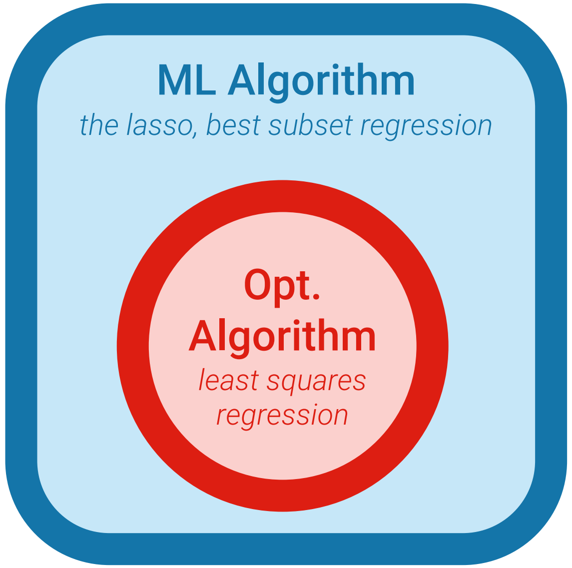
grblogtools Public

Extract and visualize information from Gurobi log files

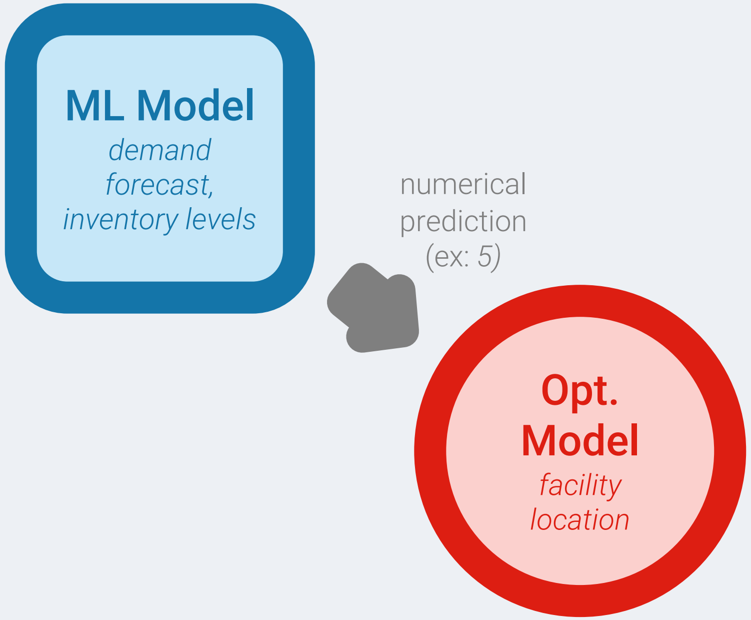
Python 61 13

Combining Machine Learning and Optimization

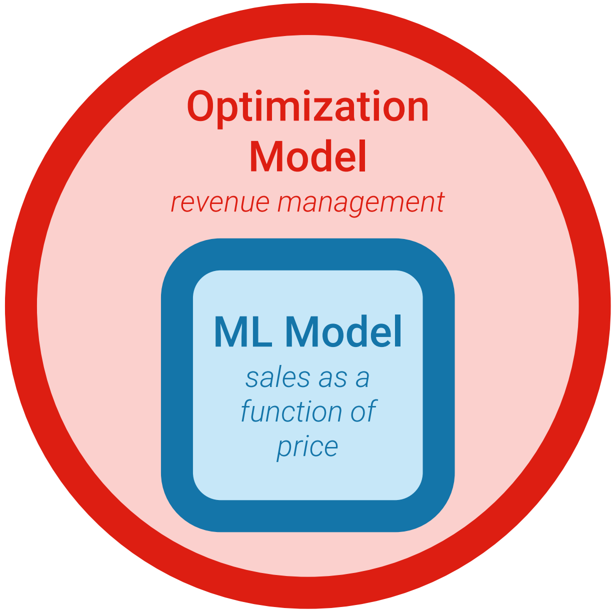
01
Training a
ML model



02
Use the ML predictions
to define the Opt. Model



03
Embed a ML model
inside the Opt. model



GUROBI OPTIMIZATION

Agenda

Open Source, sponsored by Gurobi

Collaborate, Experiment, and Innovate with the Gurobi Team

Gurobi Machine Learning Package

Why, What, How

Types of Model Errors/Residuals

Tuning in Machine Learning Package

Logistic regression with piece-wise approximations (PWA)

Decision Tree, Random Forest & Gradient boosting.

Neural Network

Final thoughts

Gurobi Machine Learning

Our Goals

- Simplify the process of **importing a trained machine learning model** built with a popular ML package into an optimization model.
- Improve **algorithmic performance** to enable the optimization model to explore a sizable space of solutions that satisfy the variable relationships captured in the ML model.
- Make it **easier** for optimization models to mix explicit and implicit constraints.

Other similar packages:

- Janos (Bergman et. al, 2019)
- ReLU_MIP (Lueg et. al, 2021)
- OptiCL (Maragno et.al, 2021)
- OMLT (Ceccon et. al, 2022)

© 2023 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 6

Gurobi Machine Learning

Regression Models Understood by Gurobi (and which has controllable errors)

scikit learn

- Linear/Logistic regression
- Decision trees
- Neural network with ReLU activation
- Random Forests
- Gradient Boosting trees
- Transformations:
  - Simple scaling of features
  - Polynomial features of degree 2
- Pipelines to combine them

Keras

- Dense layers
- ReLU layers
- Object Oriented, functional or sequential

PyTorch

- Dense layers
- ReLU layers
- Only torch.nn.Sequential models

© 2023 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 7

Gurobi Machine Learning

Generic Predictor Constraint

```
The package has four main functions:
```

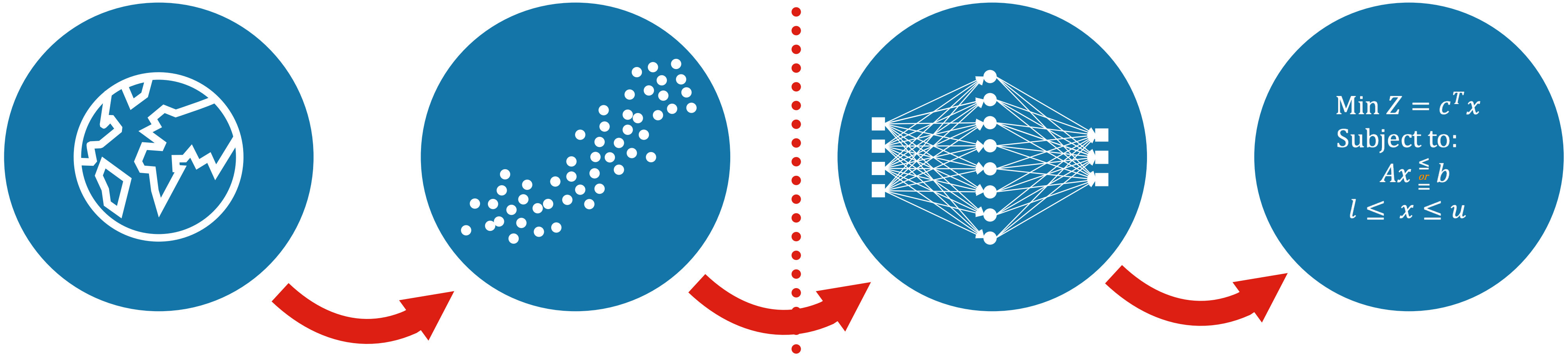
- **add_predictor_constr**(*gp_model, predictor, input_vars, output_vars=None, **kwargs*)
- **print_stats**(*abbrev=False, file=None*)
- **remove**()
- **get_error**()

© 2023 Gurobi Optimization, LLC. Confidential, All Rights Reserved | 8

Types of errors between the real system and the optimal solution

GUROBI OPTIMIZATION

Real system | **Data** | **Machine Learning Model** | **Optimization Model**



Measurement error

Model error

In some cases, errors can be introduced when translating the ML model to the optimization model

Source of Modeling Errors

In this session we will try to explain how to minimize the errors by using trained Machine Learning Predictors in Gurobi.

The source of the errors:

- Models for logistic regression use a **piecewise linear approximation** and can have approximation error (controlled by parameters).
- Models for decision tree, Random Forest and Gradient Boosting can also introduce small Modelling errors at **threshold values of node splitting** (can be controlled).
- Models for neural networks doesn’t introduce error.

Link to the example

Agenda

Open Source, sponsored by Gurobi

Collaborate, Experiment, and Innovate with the Gurobi Team

Gurobi Machine Learning Package

Why, What, How

Types of Model Errors/Residuals

Tuning in Machine Learning Package

Logistic regression with piece-wise approximations (PWA)

Decision Tree, Random Forest & Gradient boosting.

Neural Network

Final thoughts

Motivational Example: Student Enrollment

- We show how to reproduce the model of **student enrollment** from [BHB+22] with Gurobi Machine Learning.
- This model was developed in the context of the development of Janos, a toolkit similar to Gurobi Machine Learning **to integrate ML models and Mathematical Optimization.**
- This example illustrates how to use the **logistic regression** and tune the **piecewise-linear approximation** of the logistic function.
- This example illustrates how to use **Gradient Boosting** and tune the **epsilon to minimize the error.**
- We also show how to deal with fixed features in the optimization model using pandas data frames.

The objective:

The objective is to maximize of the enrolled students

The constraints:

Budget of 50000

Scholarship per student is max 2500

Useful Data form last year:

	StudentID	SAT	GPA	merit	enroll
1	1	1507	3.72	1.64	0
2	2	1532	3.93	0.52	0
3	3	1487	3.77	1.67	0
4	4	1259	3.05	1.21	1
5	5	1354	3.39	1.65	1
...	...	...	...	...	...
19996	19996	1139	3.03	1.21	1
19997	19997	1371	3.39	1.26	0
19998	19998	1424	3.72	0.85	0
19999	19999	1170	3.01	0.73	1
20000	20000	1389	3.57	0.55	0

GUROBI OPTIMIZATION

Student Enrolment

Example

- Problem Description
- Formulation & Mathematical Model
- Implementation
- Feature Discussion

Using the Data from last year we could build a logistic function (using scikit) that predict the possibility of enrollment if a student was giving certain merit (scholarship).

$$Probability_i = logisticfunction(Merit_i, SAT_i, GPA_i),$$

	StudentID	SAT	GPA	merit	enroll
1	1	1507	3.72	1.64	0
2	2	1532	3.93	0.52	0
3	3	1487	3.77	1.67	0
4	4	1259	3.05	1.21	1
5	5	1354	3.39	1.65	1
...	...	...	...	...	...
19996	19996	1139	3.03	1.21	1
19997	19997	1371	3.39	1.26	0
19998	19998	1424	3.72	0.85	0
19999	19999	1170	3.01	0.73	1
20000	20000	1389	3.57	0.55	0

GUROBI
OPTIMIZATION

Student Enrolment

Example

- Problem Description
    - The logistic function
- Formulation & Mathematical Model
- Implementation
- Feature Discussion

$$\max \sum_{i=1} Probability_i$$

subject to:

$$Probability_i = logisticfunction(Merit_i, SAT_i, GPA_i) \quad i = 1, \ldots, n,$$

$$\sum_{i=1} Merit_i \leq 50000,$$

$$0 \leq Merit_i \leq 2500.$$

Variables:

- *Probability*
- *Merit*

Probability is a predictor variable.

Merit is an optimization decision variable that was embedded into the logistic function.

GUROBI OPTIMIZATION

Student Enrolment

Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation
- Feature Discussion

The ML regressor

$$y_i = g(x_i, SAT_i, GPA_i),$$

The full model then reads:

$$\max \sum_{i=1}^{n} y_i$$

subject to:

$$\sum_{i=1}^{n} x_i \leq 0.2 * n,$$

$$y_i = g(x_i, SAT_i, GPA_i) \qquad i = 1, \ldots, n,$$

$$0 \leq x \leq 2.5.$$

GUROBI
OPTIMIZATION

Student Enrolment

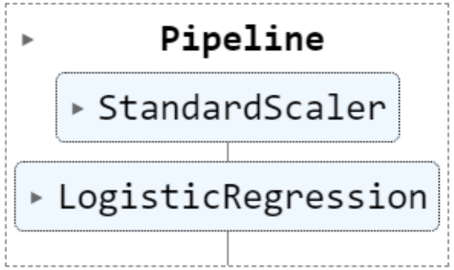
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation
- Feature Discussion

The student enrolment notebook(Logistic Regression) : Code Highlights

```
# classify our features between the ones that are fixed and the ones that will be
# part of the optimization problem
features = ["merit", "SAT", "GPA"]
target = "enroll"

# Run our regression
scaler = StandardScaler()
regression = LogisticRegression(random_state=1)
pipe = make_pipeline(scaler, regression)
pipe.fit(X=historical_data.loc[:, features], y=historical_data.loc[:, target])
```



GUROBI
OPTIMIZATION

Student Enrolment
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Logistic Regression) : Code Highlights

```
pred_constr = add_predictor_constr(
    m, pipe, studentsdata, y,
output_type="probability_1"
)

pred_constr.print_stats()
```

```
Model for pipe1:
12000 variables
8000 constraints
2000 general constraints
Input has shape (2000, 3)
Output has shape (2000, 1)

Pipeline has 2 steps:

--------------------------------------------------------------------------------
Step            Output Shape    Variables              Constraints
                                                Linear    Quadratic      General
================================================================================
std_scaler1        (2000, 3)        10000         6000            0            0

log_reg1           (2000, 1)         2000         2000            0         2000

--------------------------------------------------------------------------------
```

GUROBI OPTIMIZATION

Student Enrolment

Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Logistic Regression) :
Code Highlights

```
print(
    "Maximum error in approximating the regression {:.6}".format(
        np.max(pred_constr.get_error())
    )
)
```

Maximum error in approximating the regression
0.00715885

GUROBI OPTIMIZATION

Student Enrolment
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Logistic Regression) : Code Highlights

```
pred_constr.remove()
pwl_attributes = {
    "FuncPieces": -1,
    "FuncPieceLength": 0.01,
    "FuncPieceError": 1e-5,
    "FuncPieceRatio": -1.0,
}
pred_constr = add_predictor_constr(
    m, pipe, studentsdata, y,
output_type="probability_1",
pwl_attributes=pwl_attributes
)
m.optimize()
print(
    "Maximum error in approximating the regression
{:.6}".format(
        np.max(pred_constr.get_error())
    )
)
```

Maximum error in approximating the regression 4.47141e-06

GUROBI OPTIMIZATION

Student Enrolment

Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Decision Tree) : Code Highlights

```
# classify our features between the ones that are fixed and the ones that will be
# part of the optimization problem
features = ["merit", "SAT", "GPA"]
target = "enroll"
# Run our regression
regression = DecisionTreeRegressor(max_depth=10, max_leaf_nodes=50, random_state=1)

regression.fit(X=historical_data.loc[:, features], y=historical_data.loc[:, target])
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(max_depth=10, max_leaf_nodes=50, random_state=1)
```

GUROBI OPTIMIZATION

Student Enrolment
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Decision Tree) :
Code Highlights

```
# classify our features between the ones that are fixed and the ones that will be
# part of the optimization problem
features = ["merit", "SAT", "GPA"]
target = "enroll"
# Run our regression
regression = DecisionTreeRegressor(max_depth=10, max_leaf_nodes=50, random_state=1)

regression.fit(X=historical_data.loc[:, features], y=historical_data.loc[:, target])
```

```
▾ DecisionTreeRegressor
DecisionTreeRegressor(max_depth=10, max_leaf_nodes=50, random_state=1)
```

GUROBI
OPTIMIZATION

Student Enrolment
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Decision Tree) : Code Highlights

```
# Add Trained constraint
pred_constr = add_predictor_constr(m, regression,
studentsdata, y)

print(
    "Error in approximating the regression
{:.6}".format(
        np.max(np.abs(pred_constr.get_error()))
    )
)
```

```
Error in approximating the regression 1.0
```

GUROBI OPTIMIZATION

Student Enrolment
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

The student enrolment notebook(Decision Tree) : Code Highlights

```
# Remove pred_constr
pred_constr.remove()

# Add new constraint setting epsilon to 1e-5
pred_constr = add_predictor_constr(m, regression,
studentsdata, y, epsilon=1e-5)

m.optimize()
print(
    "Error in approximating the regression
{:.6}".format(
        np.max(np.abs(pred_constr.get_error()))
    )
)
```

```
Error in approximating the regression 5.54244e-
16
```

GUROBI OPTIMIZATION

Student Enrolment

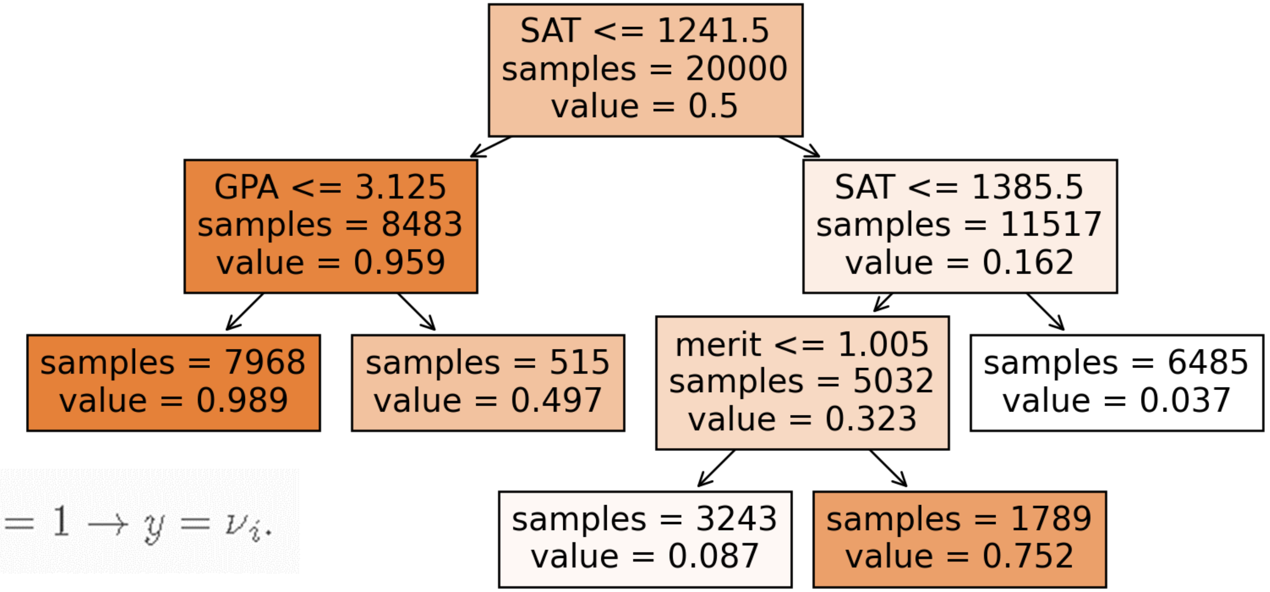
Example

- Problem Description
  - The logistic function
- Formulation & Mathematical Model
- Implementation: notebook examples
  - gurobi-machinelearning/student_admission.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
  - gurobi-machinelearning/Decision Tree.ipynb at main · Gurobi/gurobi-machinelearning (github.com)
- Feature Discussion

Tunning Decision Tree:
Decision Tree Regression and node splitting.

- We add one binary decision variable for each node of the tree (and each input vector).
- We differentiate between splitting nodes and leafs of the tree
- By definition, if a node is on the decision path, then we proceed .
- We use an indicator constraint to model that if it moves is on the decision path, the output value of the output is fixed to with an indicator constraint.

This all cause a minimal error.



$$\delta_i = 1 \rightarrow y = \nu_i.$$

$$\delta_j = 1 \rightarrow x_{s_i} \leq \theta_i,$$
$$\delta_k = 1 \rightarrow x_{s_i} \geq \theta_i + \epsilon.$$

Decision Tree Regression

- **It only corresponds to a small perturbation in the values of the input variables**
- The default value for � is 0.
- Adding small epslison value could easily address the error

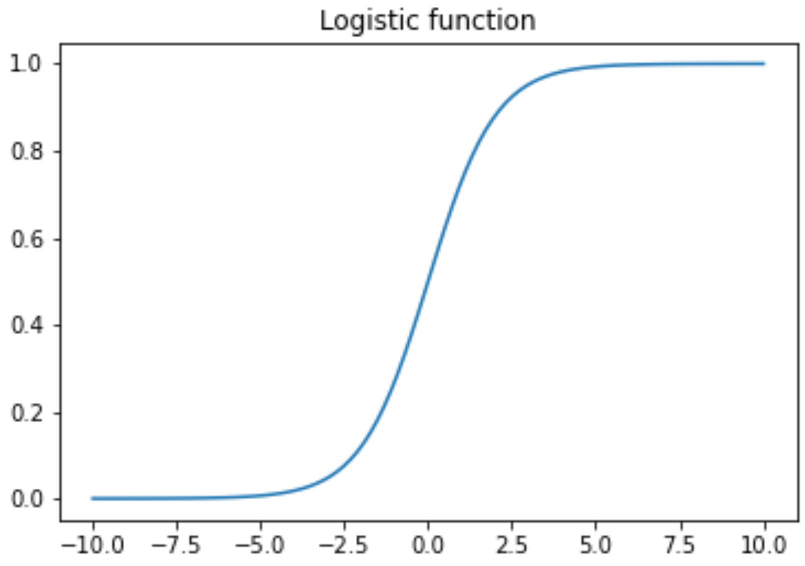
- `pred_constr = add_predictor_constr(m, regression, studentsdata, y, epsilon=1e-5)`

[Mixed Integer Formulations — Gurobi Machine Learning documentation (gurobi-machinelearning.readthedocs.io)]

Tunning Logistic Regression :
Logistic Regression and Piecewise-Linear Approximation PWA

- Function constraints in Gurobi
  - Allow to state $y = f(x)$
    - f is a predefined function
    - y and x are one-dimensional variables
  - Gurobi automatically performs a piecewise-linear approximation of f in the domain of x.
- Added logistic function to our set of predefined f.

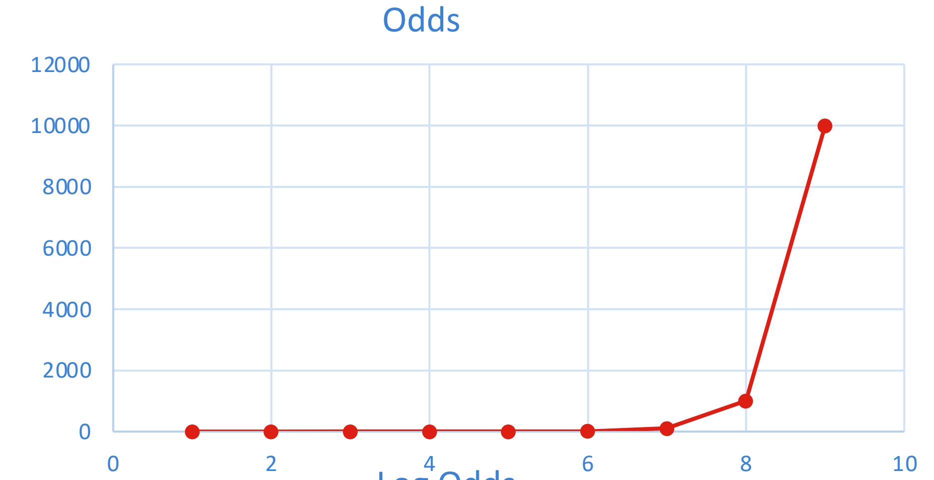
$$p(x) = \frac{1}{1 + e^{-(x)}}$$

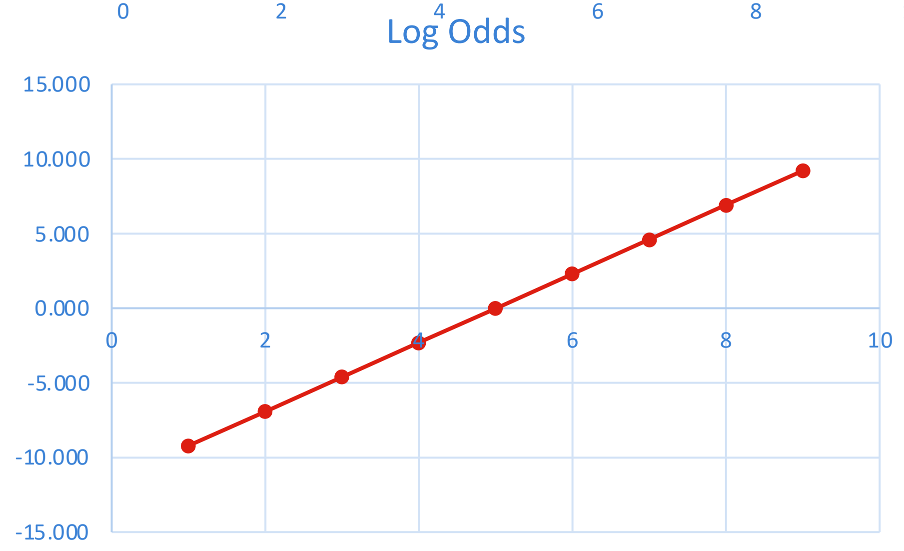


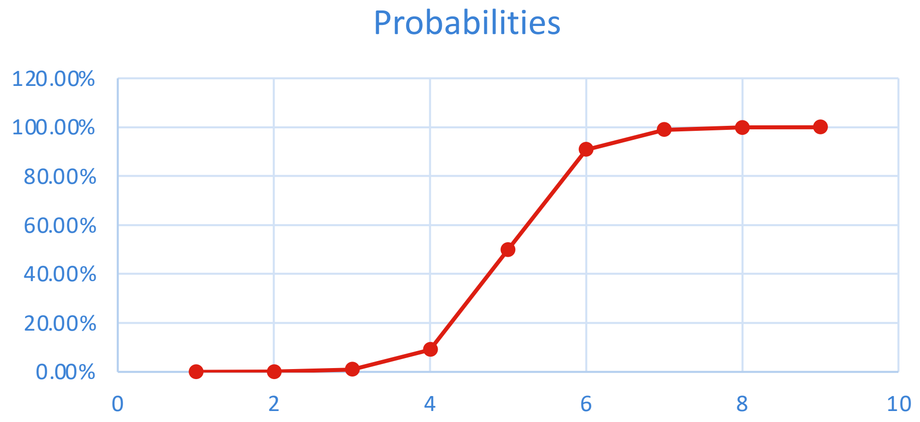
- Why it is important
- Probability = Odds / (1 + Odds)
- # y = 1 / (1 + exp(-x))
- gc = model.addGenConstrLogistic(x, y)

Why it is important ? : Making probability a linear Equation

Log Odds	Delta Log Odds	Odds on Success	Prob of Success	Delta Probability
-9.210		0.0001	0.01%	#N/A
-6.908	2.303	0.001	0.10%	0.09%
-4.605	2.303	0.01	0.99%	0.89%
-2.303	2.303	0.1	9.09%	8.10%
0.000	2.303	1	50.00%	40.91%
2.303	2.303	10	90.91%	40.91%
4.605	2.303	100	99.01%	8.10%
6.908	2.303	1000	99.90%	0.89%
9.210	2.303	10000	99.99%	0.09%





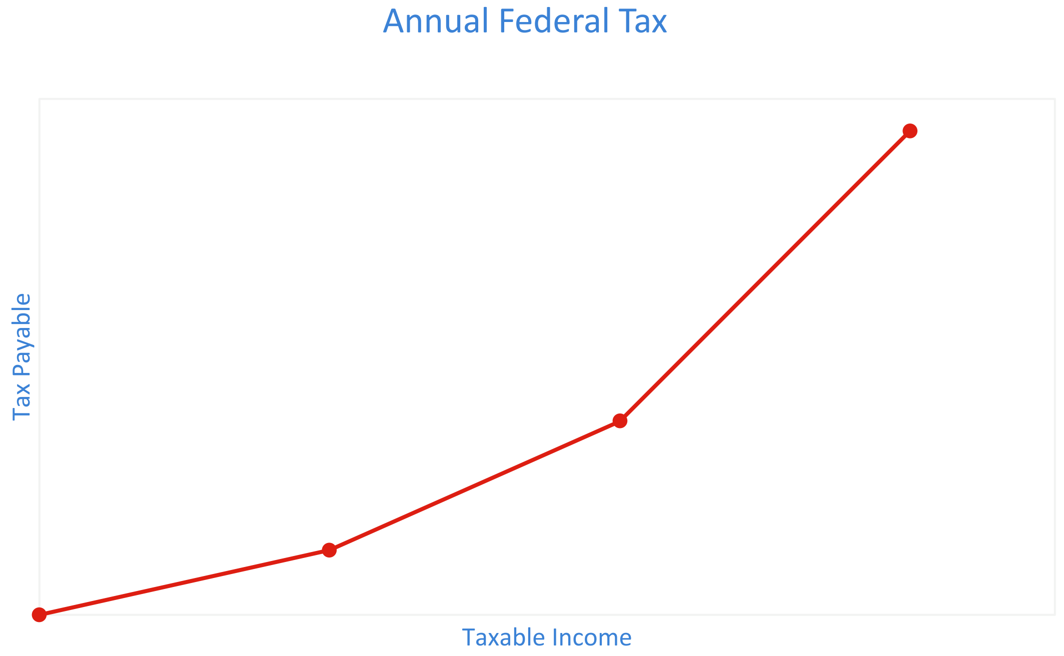


How Gurobi PWA simplifies model development process

Piecewise Linear Constraint – Without Gurobi

They consistently use piecewise linear constraints in our model.

- One such example is to calculate the annual tax payable given a taxable income.



Source:[Optimizing Your Financial Future: A Goals-Based Approach to Financial Planning - Gurobi Optimization]

How Gurobi PWA simplifies model development process

GUROBI OPTIMIZATION

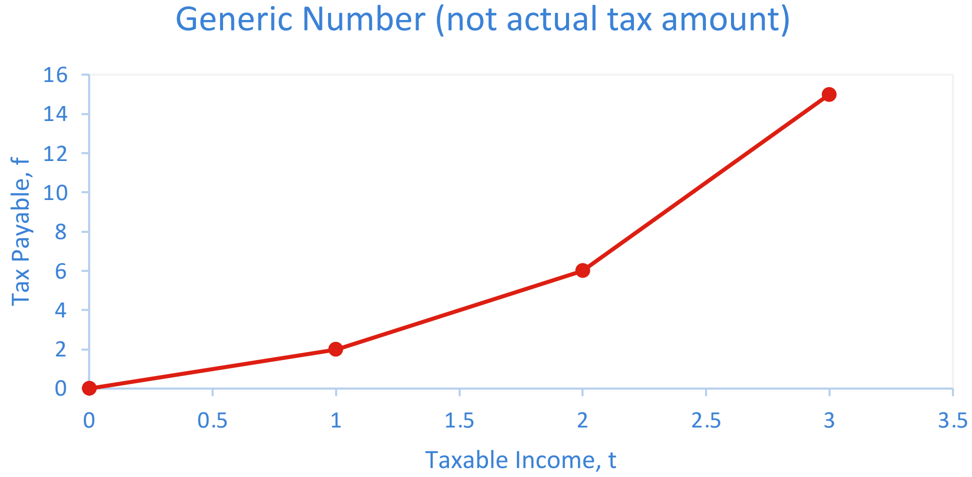
Piecewise Linear Constraint – Without Gurobi

This piecewise linear function can be modelled with the following constraints:

- $\boldsymbol{b_1 + b_2 + b_3 = 1}$
- $\boldsymbol{0 \le s_i \le b_i \quad for\ i = 1, 2, 3}$
- $t = t_1 b_1 + (t_2 - t_1)s_1 + t_2 b_2 + (t_3 - t_2)s_2 + t_3 b_3 + (t_4 - t_3)s_3$
- $t = 0 \cdot b_1 + (1 - 0)s_1 + 1b_2 + (2 - 1)s_2 + 2b_3 + (3 - 2)s_3$
- $\boldsymbol{t = s_1 + b_2 + s_2 + 2b_3 + s_3}$
- $f = f_1 b_1 + (f_2 - f_1)s_1 + f_2 b_2 + (f_3 - f_2)s_2 + f_3 b_3 + (f_4 - f_3)s_3$
- $f = 0 \cdot b_1 + (2 - 0)s_1 + 2b_2 + (6 - 2)s_2 + 6b_3 + (15 - 6)s_3$
- $\boldsymbol{f = 2s_1 + 2b_2 + 4s_2 + 6b_3 + 9s_3}$

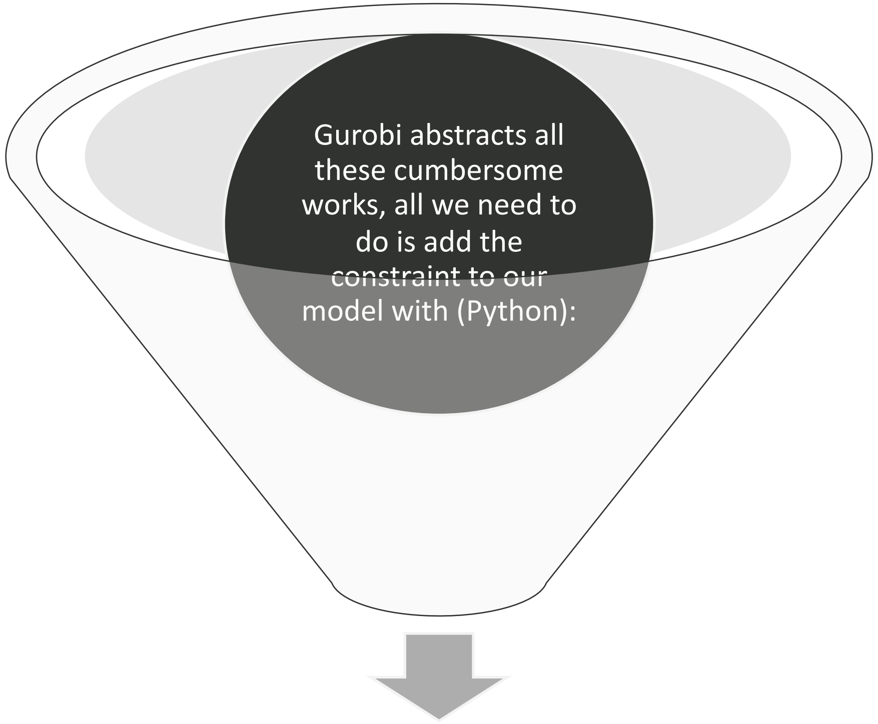
- Let t_i be the taxable income, and f_i be the tax payable.
- Let $b_1, b_2, \ldots, b_{n-1}$ be binary variables such that $b_i \in \{0, 1\}$
- Let $s_1, s_2, \ldots, s_{n-1}$ be segment variables such that $s_i \in R$

$t_i = [0, 1, 2, 3]$ and $f_i = [0, 2, 6, 15]$



How Gurobi PWA simplifies model development process

Piecewise Linear Constraint – With Gurobi




```python
model.addGenConstrPWL(xvar=t, yvar=f, xpts=[0, 1, 2, 3], ypts=[0, 2, 6, 15])
```

Parameters: Is it easy to handle?

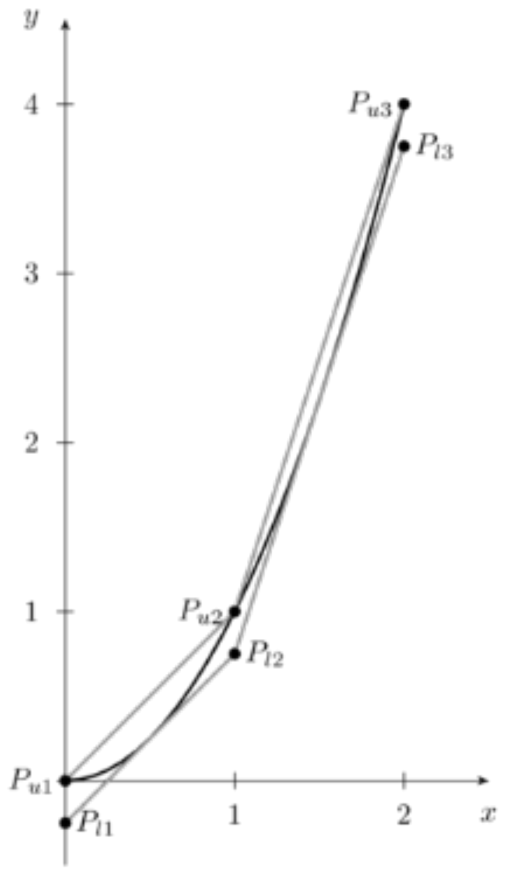
The details of the approximation are controlled using the following four attributes (or using the parameters with the same names):

FuncPieces, number of pieces.

FuncPiecesLength, desired width of each piece.

FuncPieceError, equal to the maximum absolute approximation you are willing to tolerate, **Gurobi will do pieces accordingly.**

For details, check the General Constraint discussion



Tunning the student enrolment model

	Error Before Tuning	Error After Tunning
Logistic Regression	0.00715885	4.47141e-06
Decision Tree	1.0	5.54244e-16

Tunning Neural network

- Models for neural networks don’t introduce errors.
- However, **get_error()** will still report errors if the model suffer numerical issues ,Guidelines for Numerical Issues - Gurobi Optimization

FAQ- Discussions

GUROBI OPTIMIZATION

- Which ML models would be more useful to integrate with optimization

Final thoughts

On the Gurobi ML package

- Benchmark all ML regressors before making a final decision which one to use.
- Gurobi Machine Learning Package may generate minimal controllable errors that could be handled easily
- Logistic regression error could be handled by adjusting PWA (piecewise-linear approximation) parameters
- Decision Tree, Random Forest and Gradient Boosting errors can be handled by changing an epsilon default value from 0 to very small value

github.com/Gurobi

GUROBI OPTIMIZATION

Link to the example

Thank You

For more information: gurobi.com

Zed Dean
Technical Account Manager

dean@gurobi.com

Alison Cozad
Gurobi Expert Team NA Manager

cozad@gurobi.com