Use of metamodels in health economics to bypass complexity and democratize modelling

Alik Vodyanov[1], Thomas Padgett[1] and Phil McEwan[1]

ISPOR 2025, Montréal, Quebec, Canada

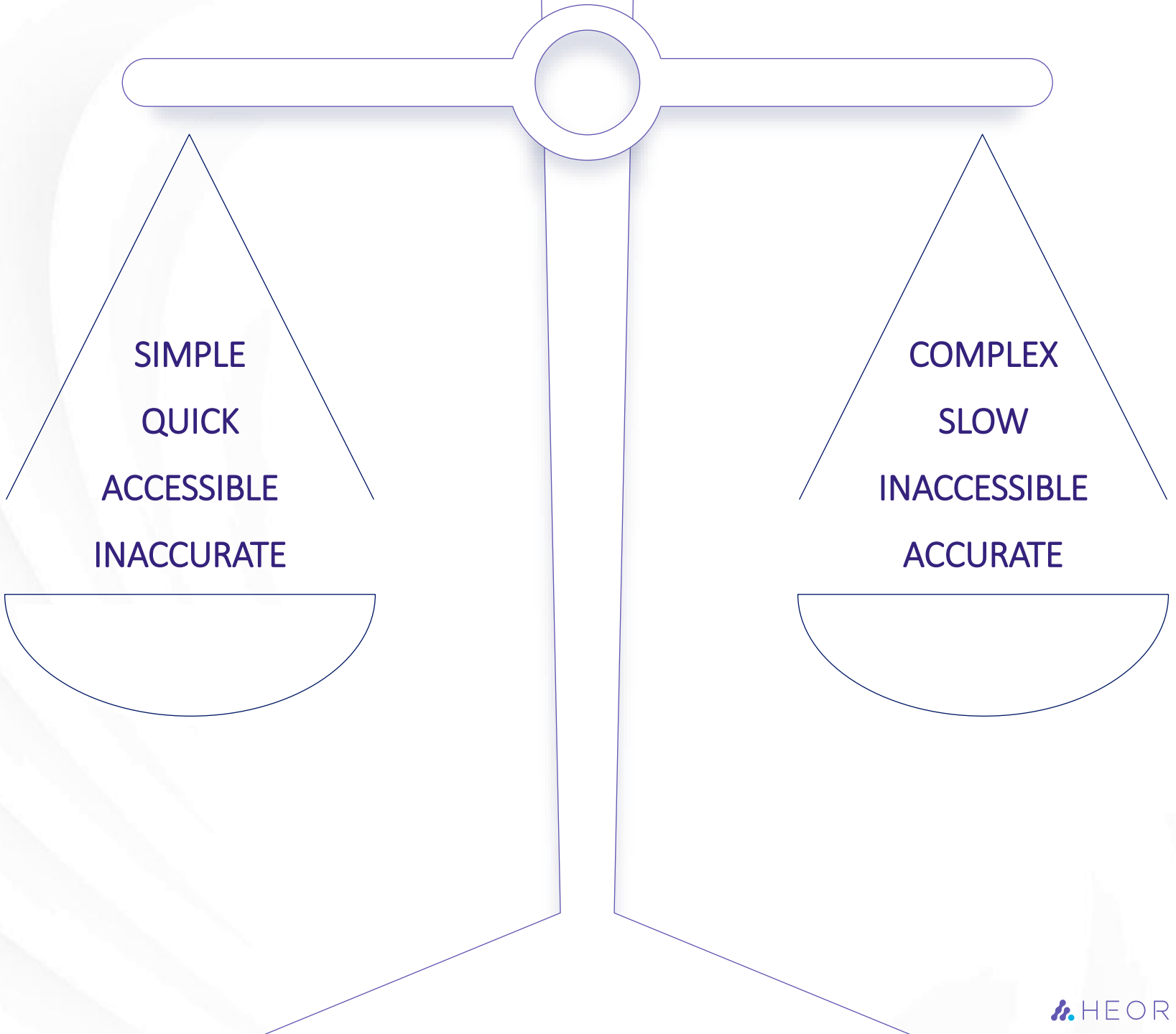
15th May 2025

[1]Health Economics and Outcomes Research Ltd., Cardiff, Wales, UK

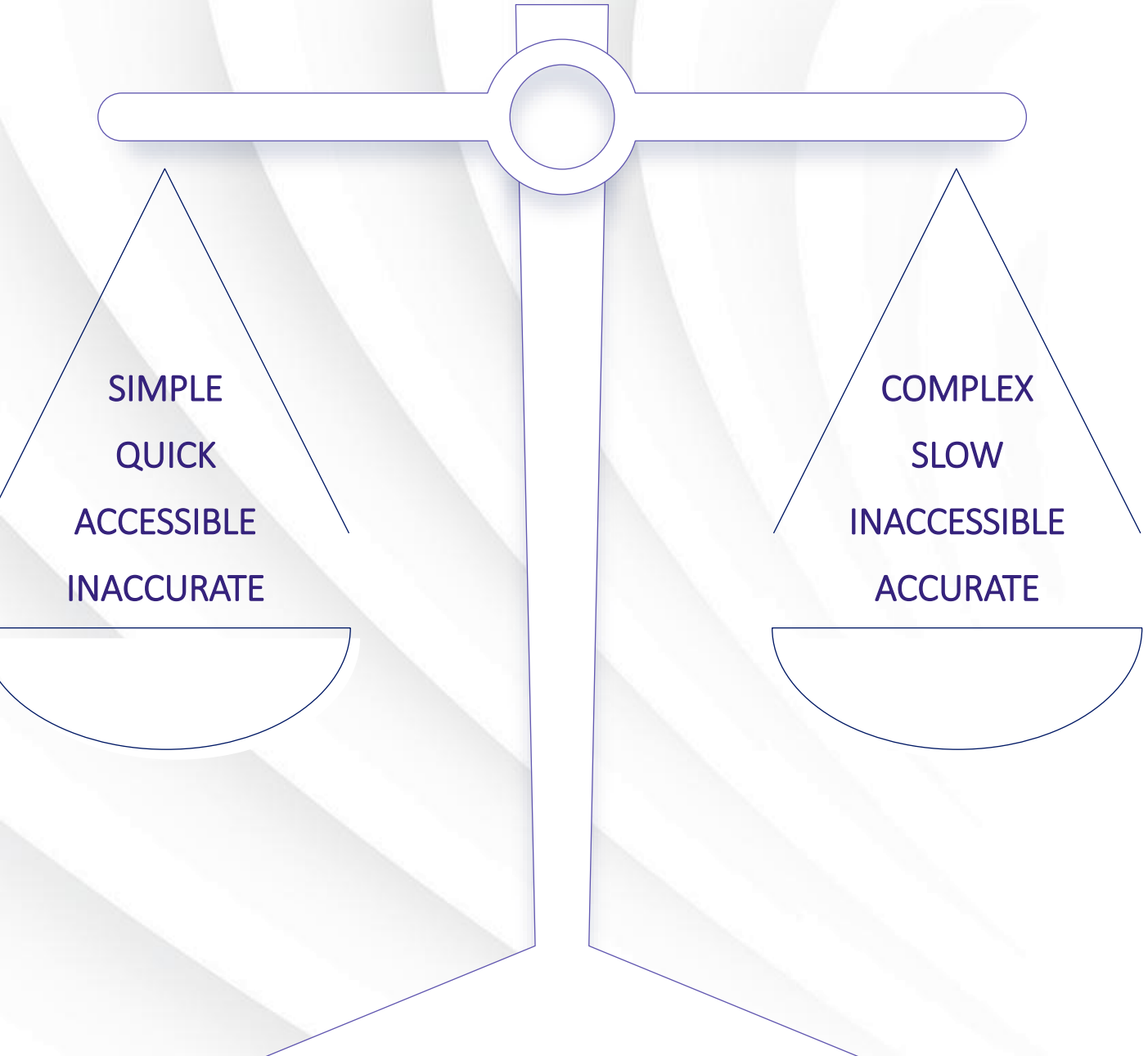
HEOR

Introduction

Modelling as a balancing act



HEOR



All models **simplify reality** with the intention of **generating insights**

Modelling is a **trade-off** between simplicity and accuracy

A model must be **sufficiently complex** enough to be fit-for-purpose

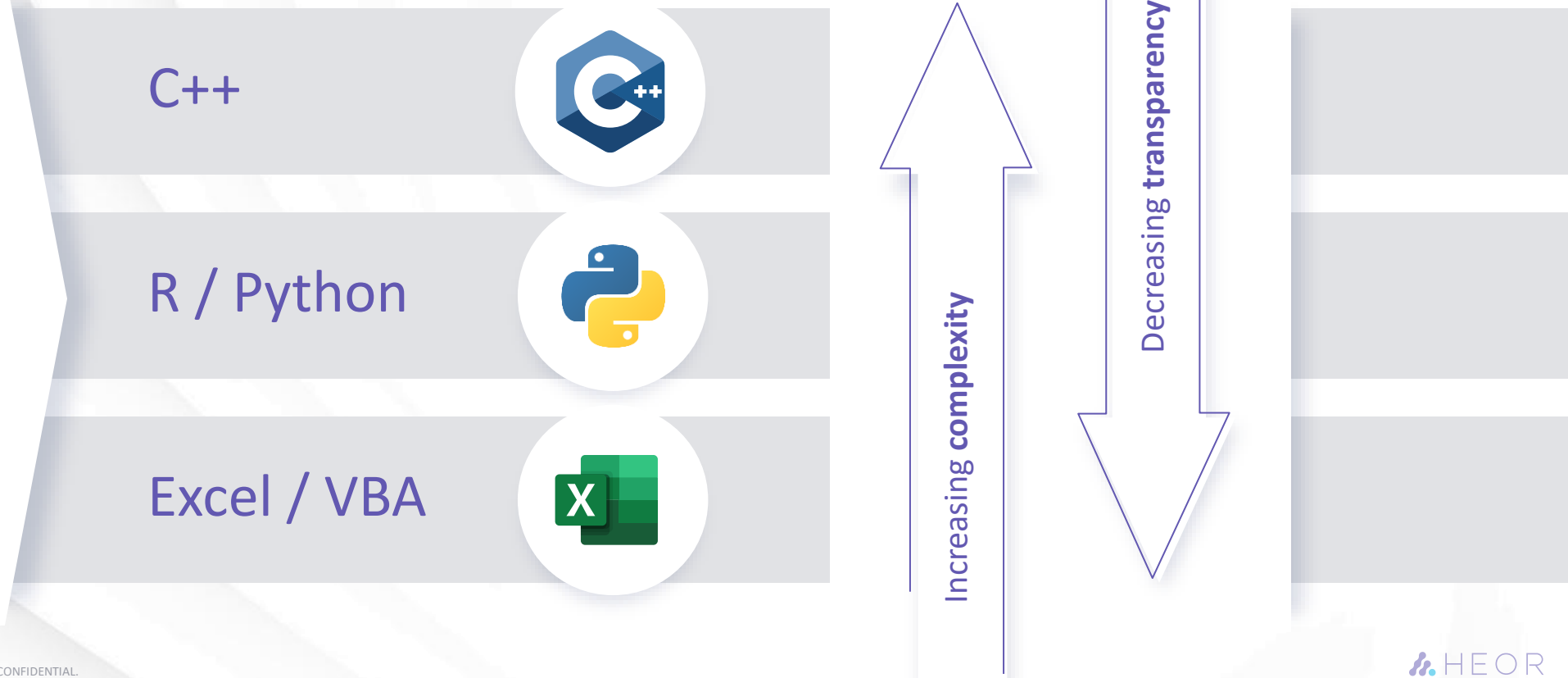
Complexity increases model runtime and makes the model inaccessible to more casual users

Impact of coding environment

Any model can give insights

The software solution/coding environment chosen for a model is informed by the problem statement and by user needs; each environment has **specific benefits**, but each **bounds what is possible**

Some models lend themselves to…

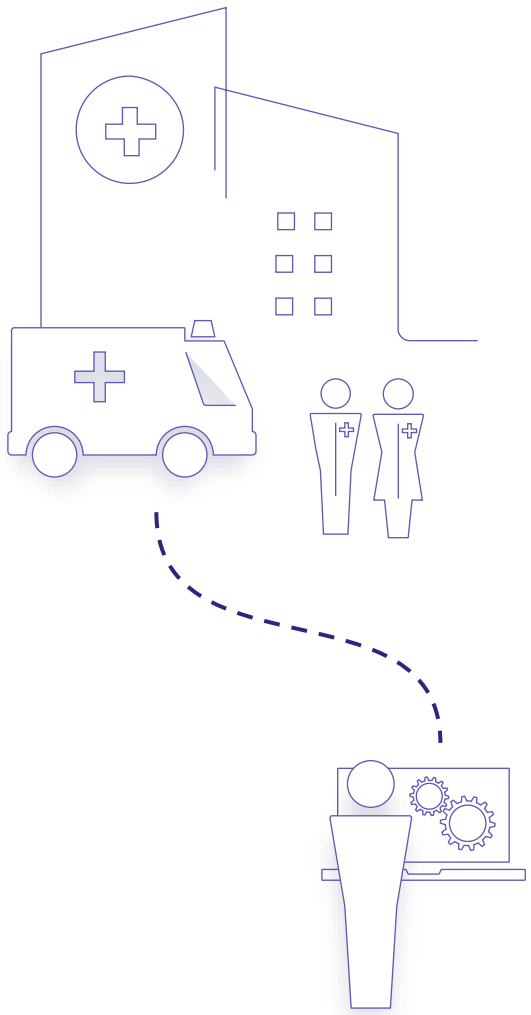


…But each environment brings **disadvantages** and **limits future** use of the model

C++		More challenging IT implementation (OS-dependent)
R / Python		Requires either an internet connection or local software installation
Excel / VBA		Relatively limited compared to other environments, but widely available

EXAMPLE

Access managers on a hospital call



1. An access manager attends a meeting with clinicians in **a hospital setting**

   They have **15 minutes** to demonstrate the value of their intervention

2. They have an iPad – **C++ is out of contention!**

   The model had to be developed in a different environment (even if that was less than ideal)

3. They get to the meeting room – **no internet connection!**

   They (somehow) manage to get a connection

4. They are asked to demonstrate the impact of uncertainty, but **PSA takes 10 minutes!**

   This meeting is turning into a cautionary tale….

Note: There are multiple ways to link programmatic approaches (e.g., in C++, Python, R) to Excel to satisfy user needs, but these still generally require internet connection or local installations

HEOR

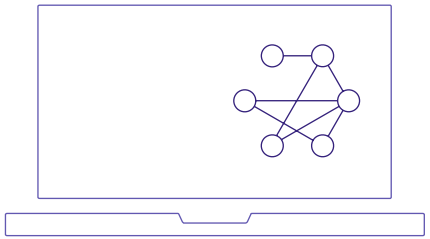
Metamodeling (models of models)

remains **relatively unexplored**
in the context of health economics
but offers the **potential** to **solve** these issues

CASE STUDY

Chronic Kidney Disease Dynamic Prevalence Model

A DPM as a foundation for a neural network and metamodel



A **dynamic prevalence model (DPM)** of chronic kidney disease (CKD) based on a system of **ordinary differential equations (ODEs)** was developed in R

The model predicts the prevalence of CKD and ESKD in Wales, with a focus on required transplantation and dialysis

We trained a neural network-based metamodel on the DPM

Latin hypercube sampling (LHS) was used to sample the design space, resulting in a **training set** of 20,000 model runs and **a test set** of 10,000 model runs based on the variance of three model parameters

We defined the design space to be broader than the user space to mitigate against increased errors at the edges of the design space

We fit the metamodel to the DPM based on mean absolute percentage errors (MAPE) between model and metamodel results (cross-validation approach)

The resulting metamodel reduced the underlying ODEs to a series of **linear algebraic expressions**

The metamodel was tested/validated using the test set

Weights and powers were extracted and implemented in the Excel environment

Results, run-times and hardware requirements for the two models (DPM and metamodel) **were compared**

Abbreviations: CKD, chronic kidney disease; DPM, dynamic prevalence model; ESKD, end stage kidney disease; LHS, Latin Hypercube Sampling; MAPE, mean absolute percentage errors; ODE, ordinary differential equations

Turning a bug into a feature: **overfitted by design**

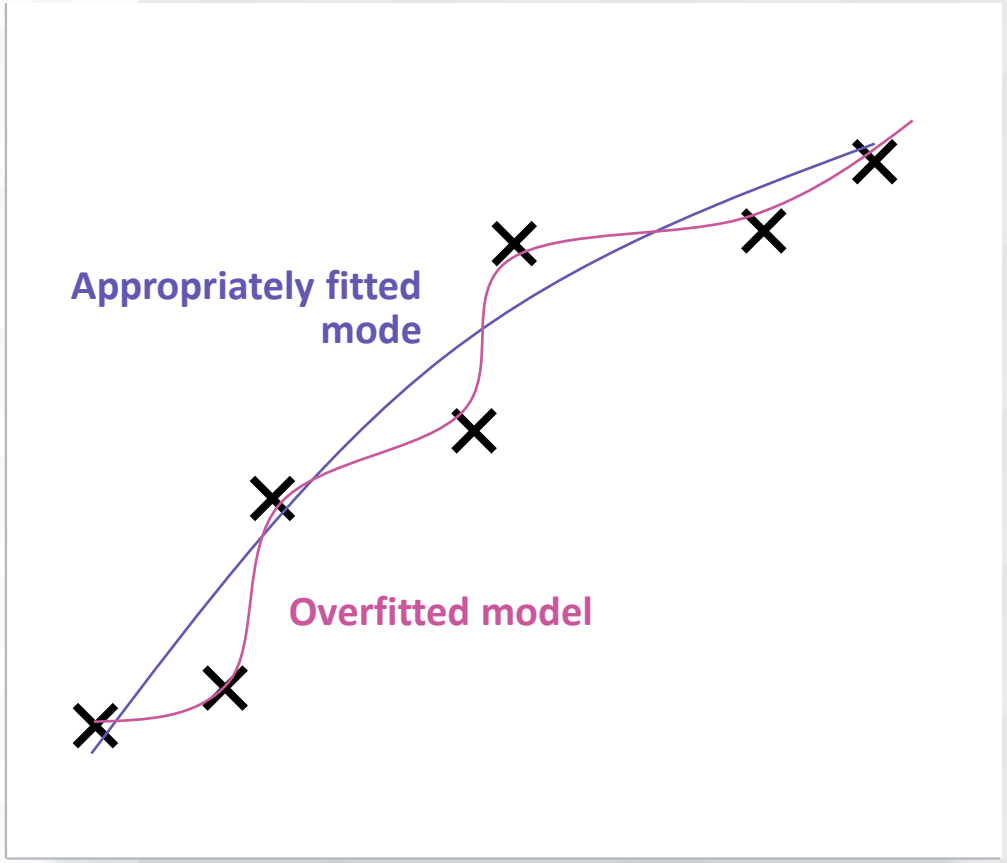
A common challenge with predictive model development is **overfitting**

An overfitted model **replicates training data too closely** and **performs poorly** when applied to other data.

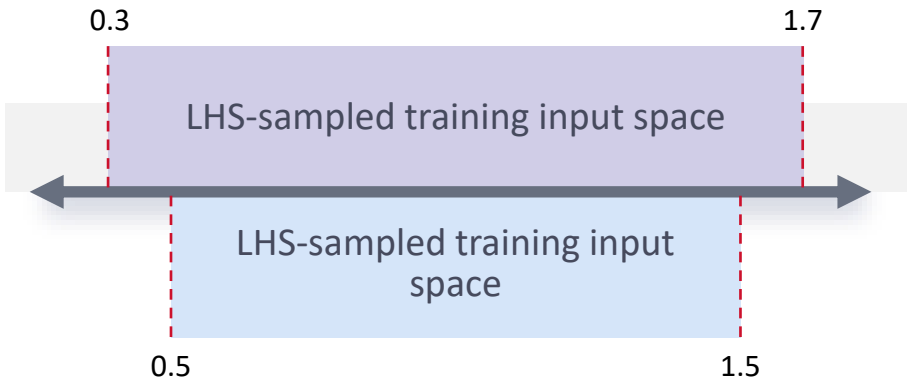
Overfitted models **lack flexibility** and can't handle uncertainty: **limited usefulness!**

Here, we sample across the whole design space and do not apply the model outside of these bounds, meaning that **overfitting is ideal**

Our **metamodel** was deliberately overfitted on a broad design space: **its only objective is to replicate the results of the base model exactly**



Why the difference between **testing** and **training?**



Metamodels are trained on a training set and tested on both training and testing sets

Performance is better on the testing set than on the training set;

why?

Expanding LHS-sampled space during training **improves performance** of the final model

Due to the overfitting approach during the training process, the metamodel is sensitive to a correctly defined input space

- Most apparent when parameter values are close to the edge values, due to few examples to train on

Metamodel performance can be further improved by:

- expanding the training parameter space
- by optimising the metamodel's neural net structure
- expanding the neural net structure
- Increasing the number of training scenarios under LHS sampling

Abbreviations: LHS, Latin hypercube sampling; MAPE, mean absolute percentage error

HEOR

RESULTS

How well does the metamodel reproduce the DPM results?

Metamodel predicts **116 categories across 36 years (4,176 datapoints)**

Testing error (MAPE) is lower than training error*

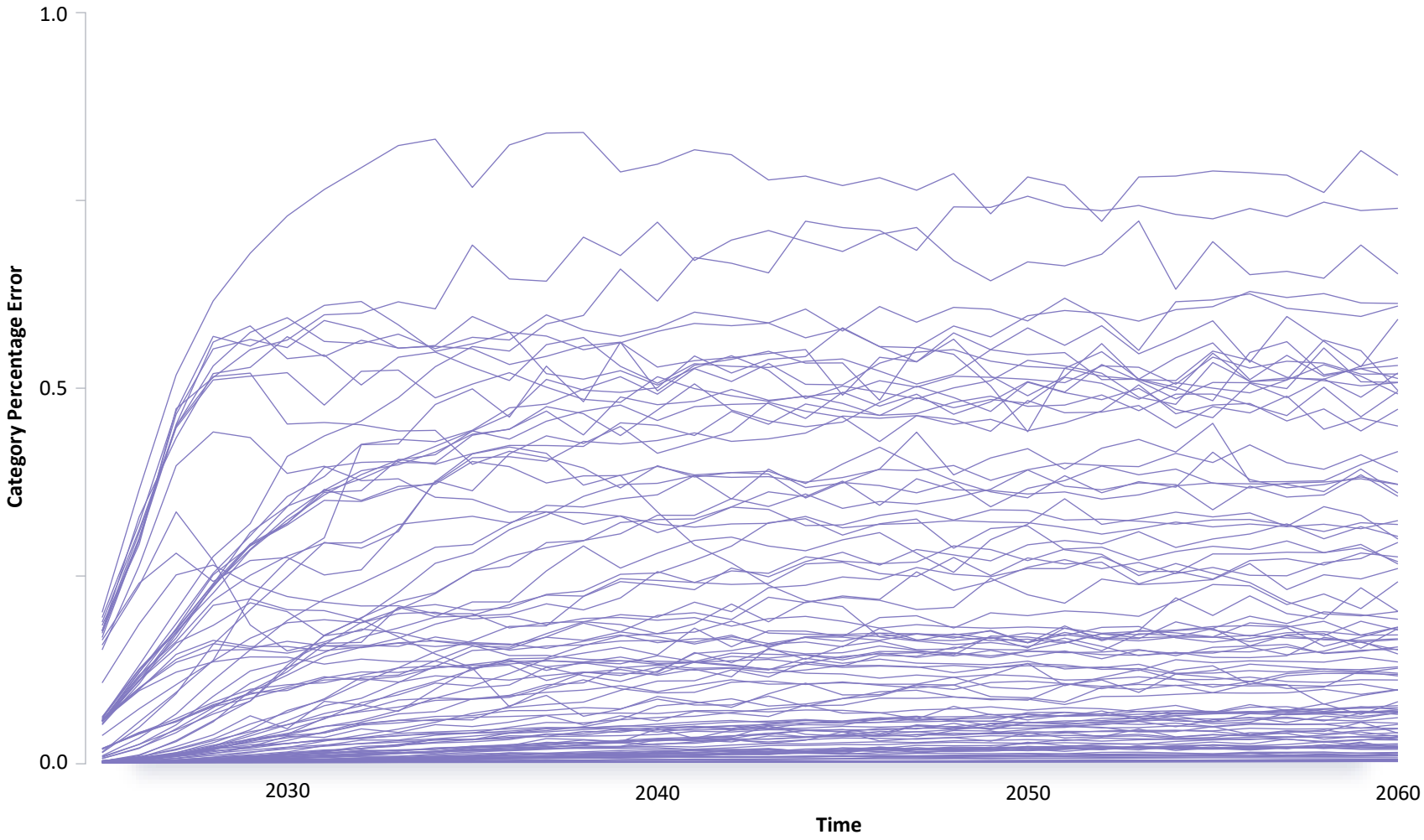
Overall MAPE is < 0.3%

Error propagates through time

Most output category errors remain < 0.25% over all years

Only four categories > 0.5% over all years

MAPE of all category predictions over time
116 total output categories



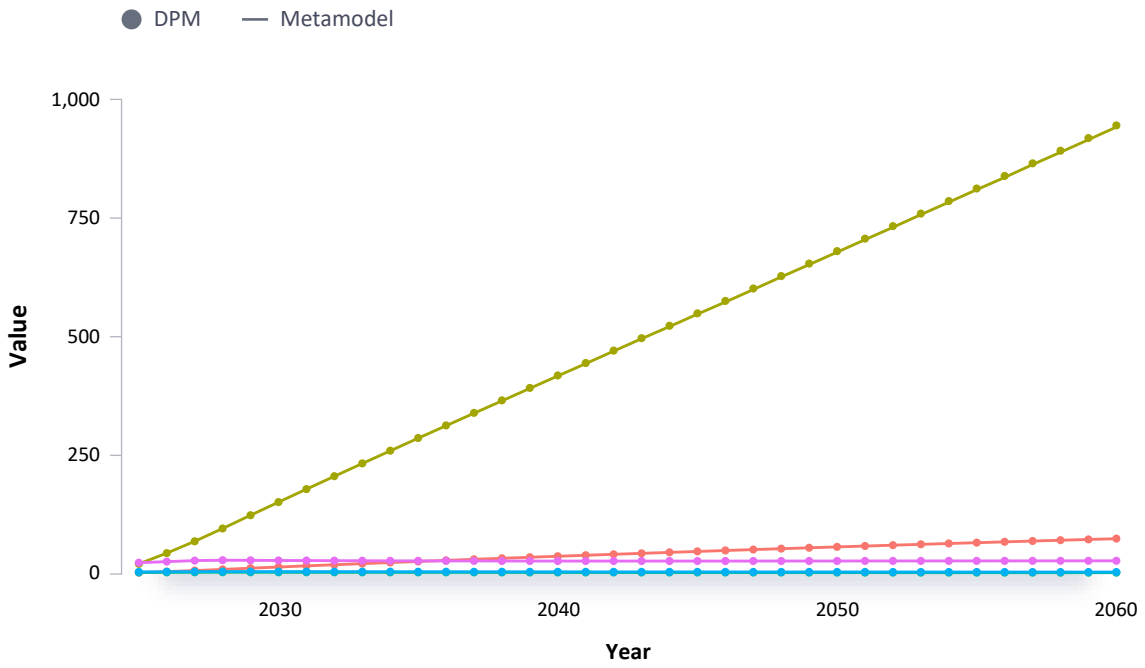
*due to the reduction of parameter space within training scenario set
Abbreviations: DPM, dynamic prevalence model; MAPE, mean absolute percentage error

HEOR

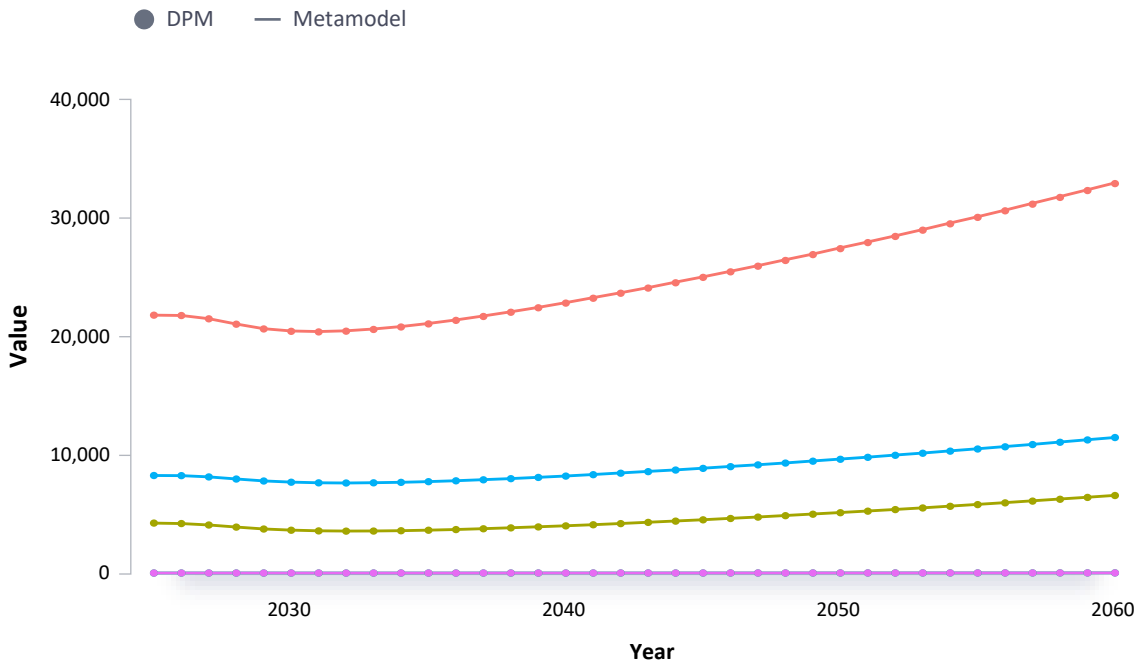
RESULTS

How well does the metamodel reproduce the DPM results?

Metamodel vs DPM absolute values
Top 5 worst performing categories in **worst** scenario



Metamodel vs DPM absolute values
Top 5 worst performing categories in **best** scenario



Indicative outcomes are shown for **worst** and **best** scenarios

Even in the worst performing categories of the worst scenario, **error remains small**

The percentage error translates to a reasonably small drift in absolute values

Abbreviations: DPM, dynamic prevalence model; MAPE, mean absolute percentage error

HEOR

RESULTS

DPM vs. Metamodel – Runtime

Model prediction runtime

The average runtime for the **DPM** performing a typical simulation was

1.83 seconds

The average runtime for the **metamodel** performing an equivalent simulation was

0.06 seconds

A difference of **~30 minutes** over a typical 1000-run analysis

The metamodel generation required roughly 5 hours of total training runtime

- A one-off time commitment required for model setup
- Most time (~4.5h) is taken up by scenario generation for model training
- This step is dependent on the underlying model engine and how quickly results can be generated, which can be improved through parallelism and additional hardware use
- Metamodel training took ~30 minutes and is a standard approach regardless of the original model structure
- Everything we have presented was done on a laptop with i7-11850H with 8 cores and 32GB of RAM

Abbreviations: DPM, dynamic prevalence model; MAPE, mean absolute percentage error

HEOR

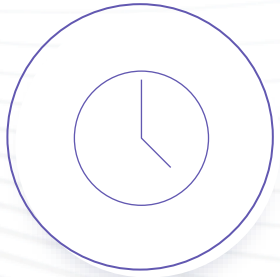
Conclusions

CONCLUSION

Metamodels for expanding access to knowledge

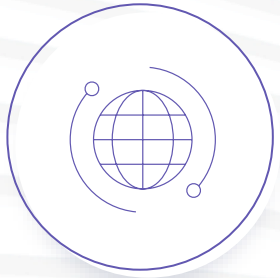
METAMODELS

are…

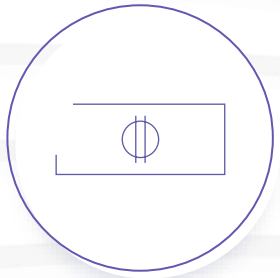


Fast

Flexible



Accessible anywhere



Cheap to run

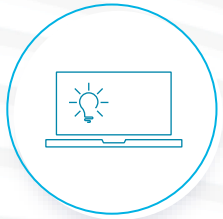
CONCLUSION

Metamodels for expanding access to knowledge

METAMODELS
allow…

Real-time demonstrations during meetings

Fast, accurate, off-line modelling



An enhanced user experience

Easy roll-out to multiple users
(low training demand)

Broadened access for varied stakeholders

HEOR

Any questions?

THANK YOU

Rhymney House, Unit A Copse Walk, Cardiff Gate Business Park, Cardiff CF23 8RB
+44 (0)2920 399146 | enquiries@heor.co.uk | www.heor.co.uk


HEOR
founded on a passion for science

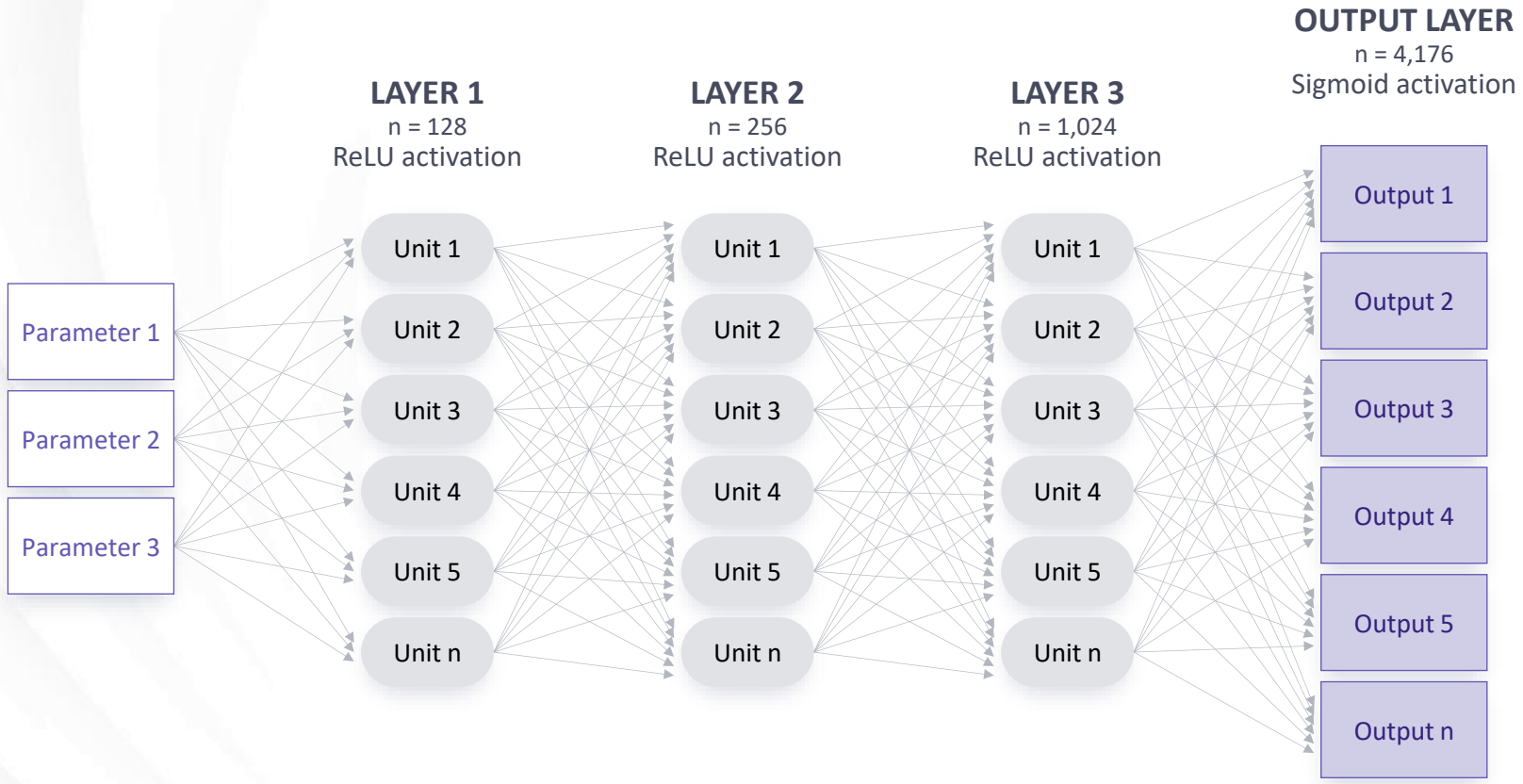
Metamodel structure

The metamodel is a **deep neural net** that was trained via keras

Mean squared error used as a loss function

Adam optimiser with learning rate of **0.001 over 400 epochs**

Since we are interested in **overfitting**, the metamodel performance can be **further improved** with an increase in size and number of hidden layers



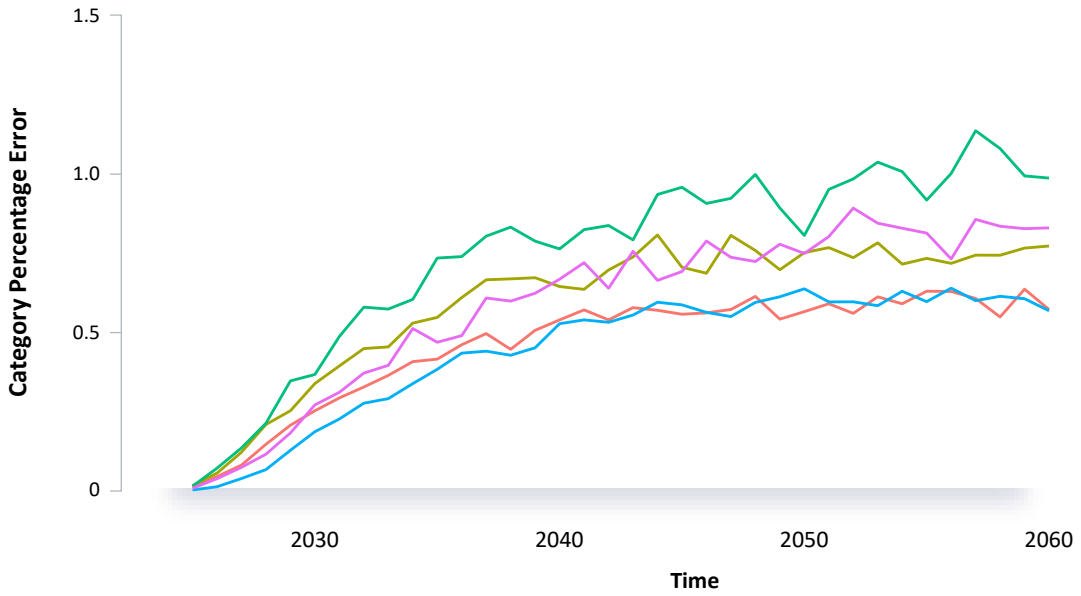
Abbreviations: ReLU, rectified linear unit

HEOR

DPM vs. Metamodel – best categories (MAPE)

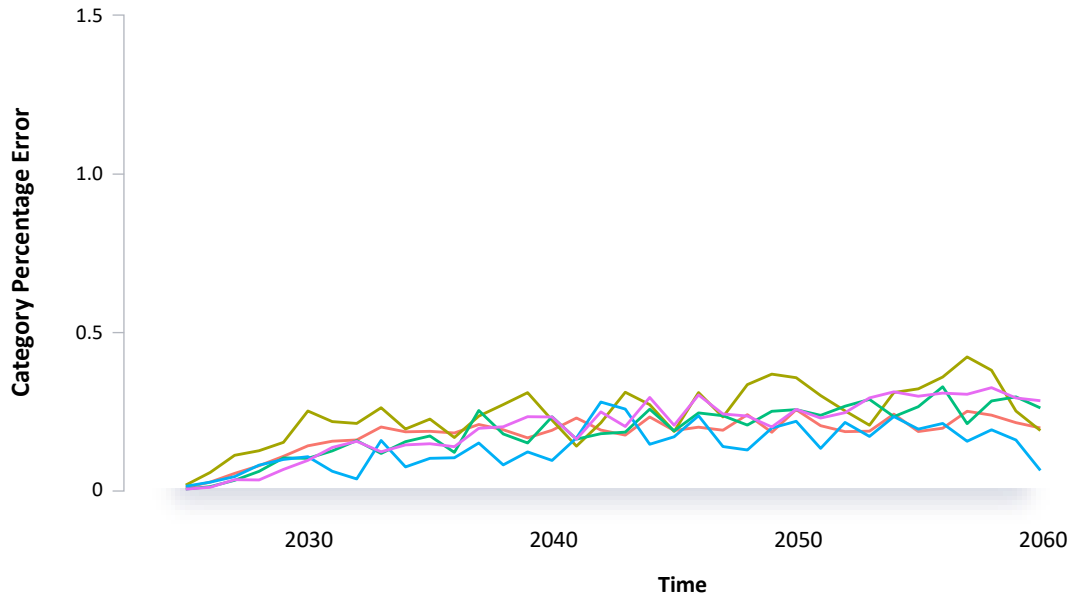
Top 5 best performing category predictions over time in the worst scenario

Mean absolute error of the scenario is 0.32%



Top 5 best performing category predictions over time in the best scenario

Mean absolute error of the scenario is 0.08%

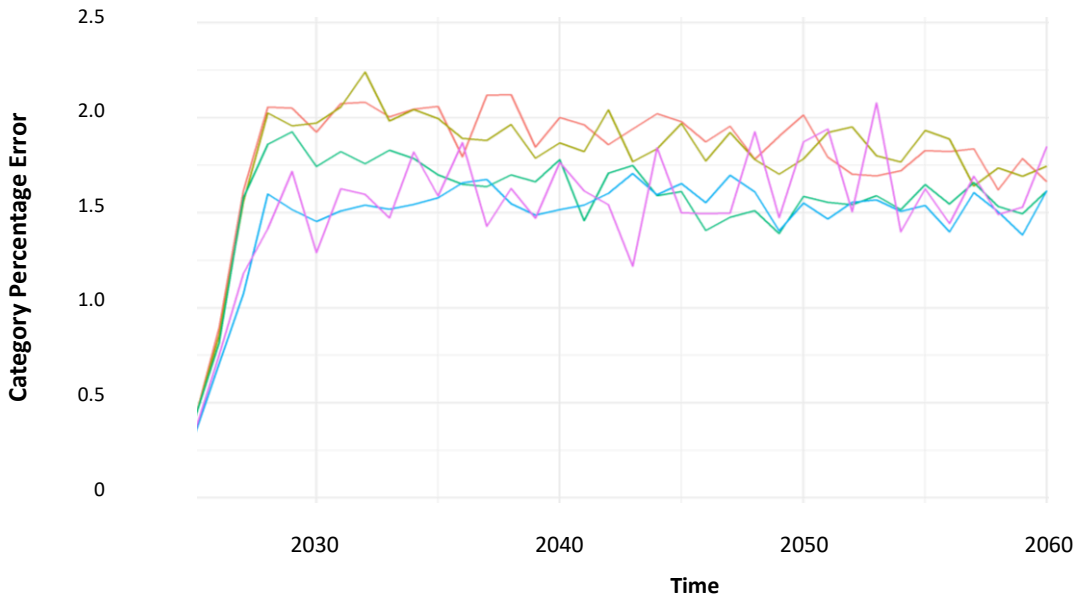


Abbreviations: DPM, dynamic prevalence model; MAPE, mean absolute percentage error

DPM vs. Metamodel – best categories (MAPE)

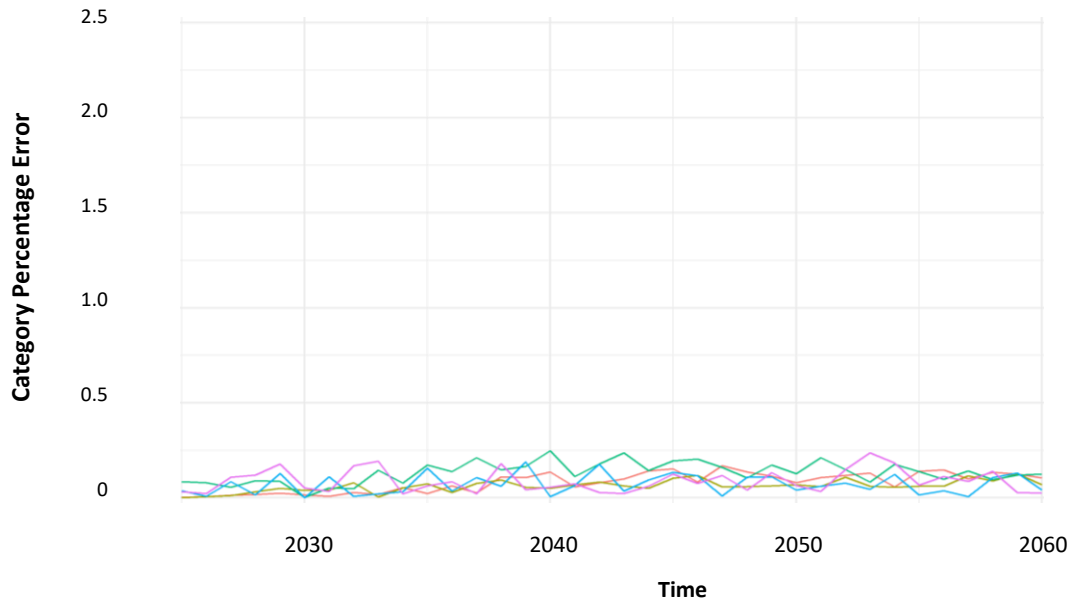
Top 5 best performing category predictions over time in the worst scenario

Mean absolute error of the scenario is 0.43%



Top 5 best performing category predictions over time in the best scenario

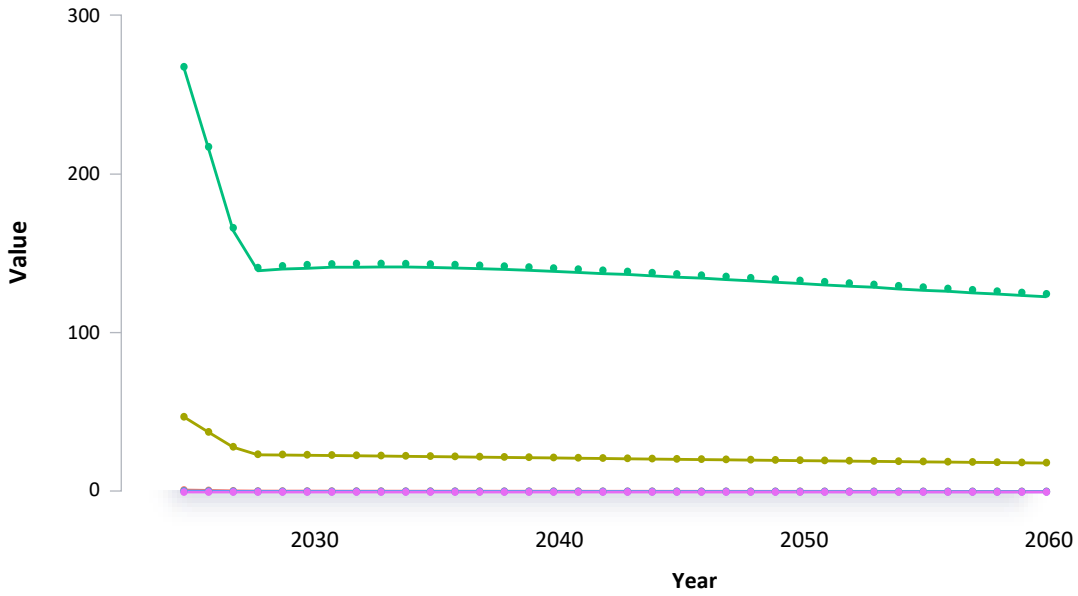
Mean absolute error of the scenario is 0.04%



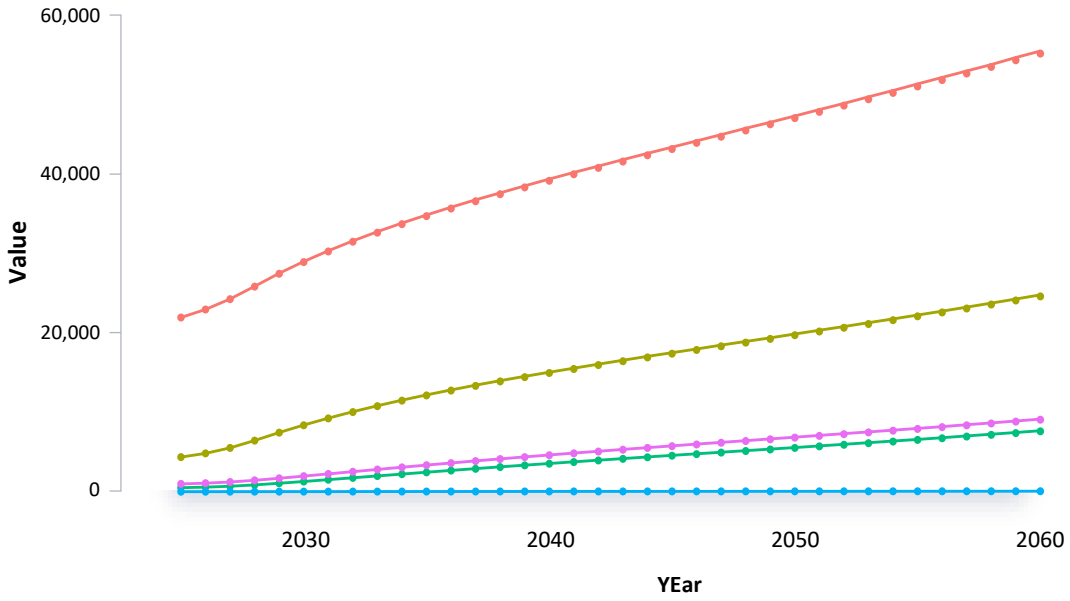
Abbreviations: DPM, dynamic prevalence model; MAPE, mean absolute percentage error

DPM vs. Metamodel – best categories (MAPE)

Metamodel vs DPM absolute values
Top 5 **worst** performing categories in worst scenario



Metamodel vs DPM absolute values
Top 5 **best** performing categories in worst scenario



Abbreviations: DPM, dynamic prevalence model; MAPE, mean absolute percentage error

Fully represented parameter space :
Latin Hypercube Sampling (LHS)

To aid with overfitting, the metamodel needs to be **trained on a set of scenarios** that would contain an exhaustive representative input set

Sampling inputs for training scenarios must be carefully considered as we need to capture as much of parameter space as possible

LHS **ensures efficient coverage** - samples are evenly distributed across the entire range of each variable, providing better coverage of the input space compared to simple random sampling

LHS can be applied to **multidimensional distributions**, making it suitable for complex models with multiple variables

LHS combines the benefits of random sampling with a structured approach, reducing the likelihood of clustering and ensuring a more representative sample

LHS is space-filling and intended to be used for **box-like domains** – it can fill the entire domain space with enough samples