Machine Learning Interpretability:

the key to ML adoption in the enterprise

Alberto Danese

About me

Day time

- Senior Data Scientist @ Cerved since 2016
- Innovation & Data Sources Team

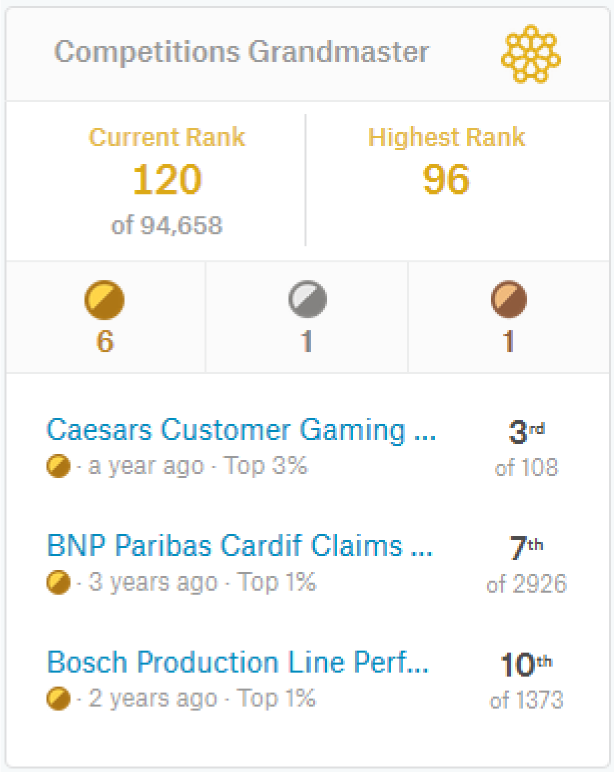
Background

- Manager @ EY
- Senior Consultant @ Between
- Computer Engineer @ Politecnico di Milano (2007)

Night time

- Active kaggler since 2016 (*albedan*)
- Just 8 competitions completed so far

- Kaggle Grandmaster after my 6th competition
- 6 gold medals, 4 solo gold, 1 solo prize in masters only competition



kaggle days

LVMH Christian Dior COUTURE LOUIS VUITTON SEPHORA LOGICAI kaggle

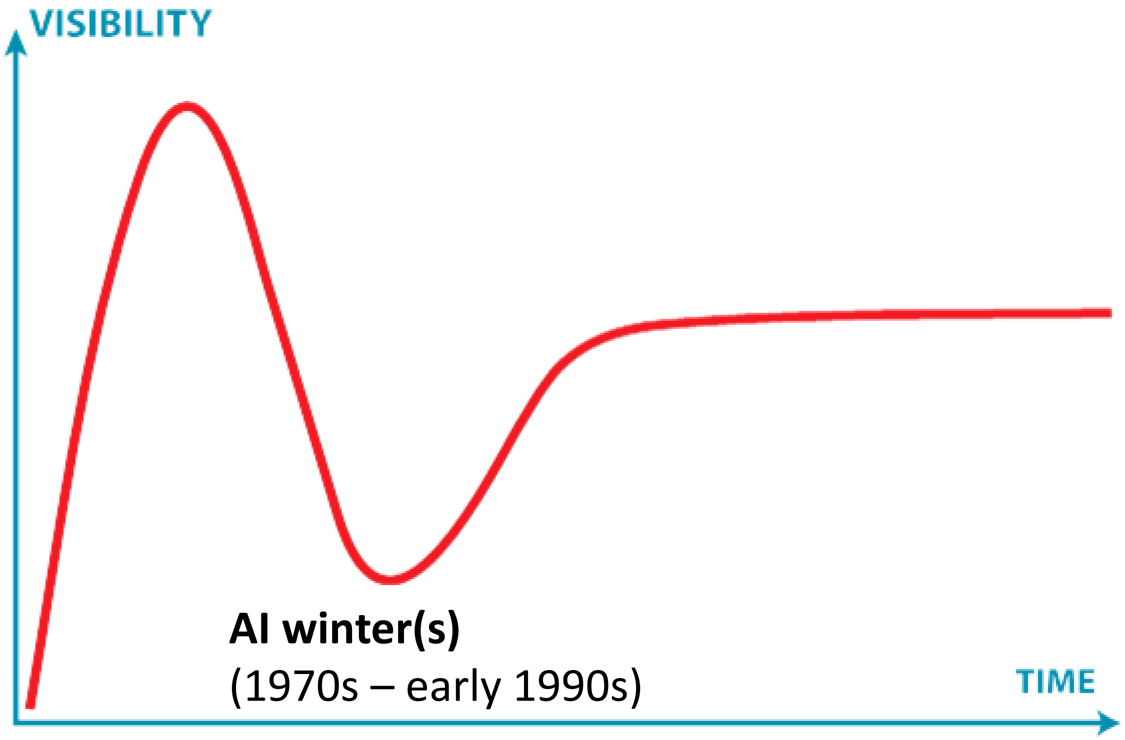
Talk overview

- *Why ML Interpretability is key*
- *How (and when) to use ML Interpretability*
- *Drill-down on methods, techniques and approaches*

Why ML Interpretability is key

AI & ML history in a nutshell

Early days
(1940s – 1970s):
- Turing test
- Dartmouth workshop
- First neural network and perceptrons



AI winter(s)
(1970s – early 1990s)

AI success and new developments
(from the early 1990s)
- Achievements cover by media
    - Deep Blue vs Kasparov (1996-97)
    - AlphaGo vs Lee Sedol (2016)
- Tech (HW) advancements
    - GPUs (2006)
    - TPUs (2016)
- Free and open source languages and frameworks
- Competitions and culture
    - Netflix prize (2006)
    - Kaggle (2010)
    - HBR – Data scientist: the sexiest job of 21^{st} century (2012)

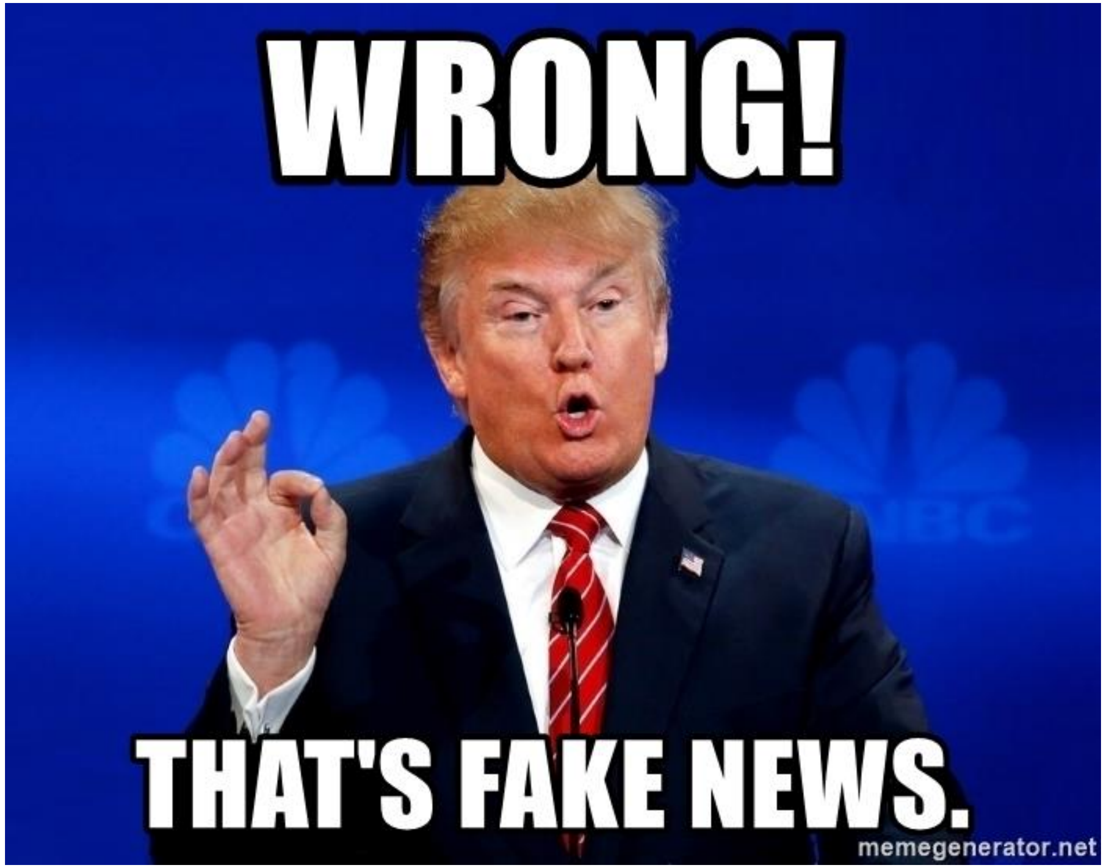
And now, in 2019

Let’s use some simple logic:

- 60 years of AI ➕
- 20 years of advancements and accomplishments ➕
- 1 cultural change 🟰

AI & ML solutions are now widely **deployed in production**, across all industries and companies

Right?

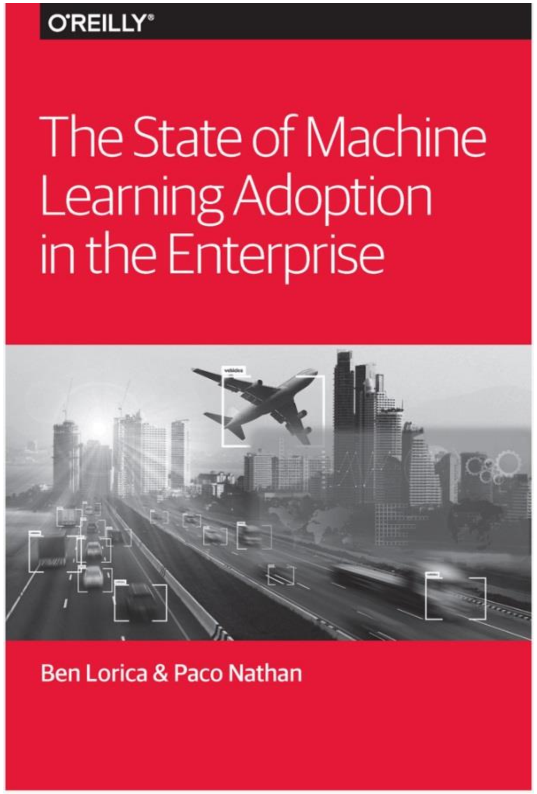


kaggle days

And now, in 2019 **(hype-free)**

- *It's Still Early Days for Machine Learning Adoption*[1]

- Nearly half (49%) of the 11,400+ data specialists who took O'Reilly's survey in June 2018 indicated they were in the exploration phase of machine learning and <u>have not deployed any machine learning models into production</u>



A two-speed world

Technology-first companies

- Large use of ML in production
- Very frequent use, at scale, often on not critical topics (movie recommendations, targeted ads)

Well-established companies (banks, insurances, healthcare)

- More POCs than actual deployments
- Crucial but not so frequent decisions (accepting a mortgage request, health diagnosis)

- Why? **Trust & communication**, i.e. **ML Interpretability**
  - Humans need explanations – especially **regulators** – and classic statistical methods (e.g. linear regressions) are easily interpretable
  - Till a few years ago, many ML models were complete black boxes

ML interpretability besides regulation

- **Husky vs. Wolf classification** as in the paper “Why should I trust you”[2]
    1. The authors trained a biased classifier (on purpose): every wolf picture had snow in the background
    2. They asked 27 ML students if they trusted the model and to highlight potential features (with and without ML interpretability)

(a) Husky classified as wolf (b) Explanation

Is *snow* a key feature?
Yes, for **12 out of 27**

Is *snow* a key feature?
Yes, for **25 out of 27**

- Other areas where interpretability matters: hacking / adversarial attacks

How (and when) to use ML Interpretability

Let’s agree on the basics

1. **Interpretability**: *the ability to explain or to present in understandable terms to a human*[3]
2. **Accuracy vs. Interpretability** is a tradeoff[4], i.e. you can get:
   - Accurate models with approximate explanations
   - Approximate models with accurate explanations
3. **Global vs. Local Interpretability**[4]:
   - Global: explain how the model works in predicting unseen data
   - Local: explain the "reason" of a specific prediction (i.e. of a single record)
4. **Model agnostic vs. model specific interpretability models**

Four levels of interpretability

0	Know your **data**	• Your **model** is only **as interpretable as your data** • Understand **processes / responsabilities** on data
1	**Before** building a model	• **Enforcing specific properties** in a model, to allow for a better understanding of its internal behaviour • Fine tuning what the model is **optimizing**
2 / 3	2 **After** the model has been built (***globally***) 3 **After** the model has been built (***locally***)	• Showing **global behaviour** of the model, i.e. the relation between the predicted target and one or more variables • **Giving more information** on a specific prediction, e.g. what features impacted **that prediction** the most

FOCUS

My quick framework

- **Name and main concepts** of the interpretability model / approach
- **When** is it applied (B before / A after building the model)
- **Where** it applies (L local / G global)
- **Other notes / restrictions** (model agnostic or specific, etc.)
- **What can you say if you use this method / approach**

- Main focus on *structured* data

Drill down on methods, techniques and approaches

1 Monotonicity constraints (1/2) B G

- Main concepts:
  - Some features are expected to show a **monotonic behaviour** (constantly non-descending or non-ascending **with respect to the target variable**)
  - Usually, due to the specific training set, a tree-based greedy model (e.g. GBT) is likely to "catch up" irregularities, even when the overall trend is non-descending or non-ascending
  - **Enforcing** a monotonic constraint, a specific tree-based model only "accepts" splits that are in line with the constraint
  - This can help limiting overfitting if the monotonic behaviour is consistent
- Monotonicity constraints are applied **before** building a model
- They act globally and are specific of tree-based models

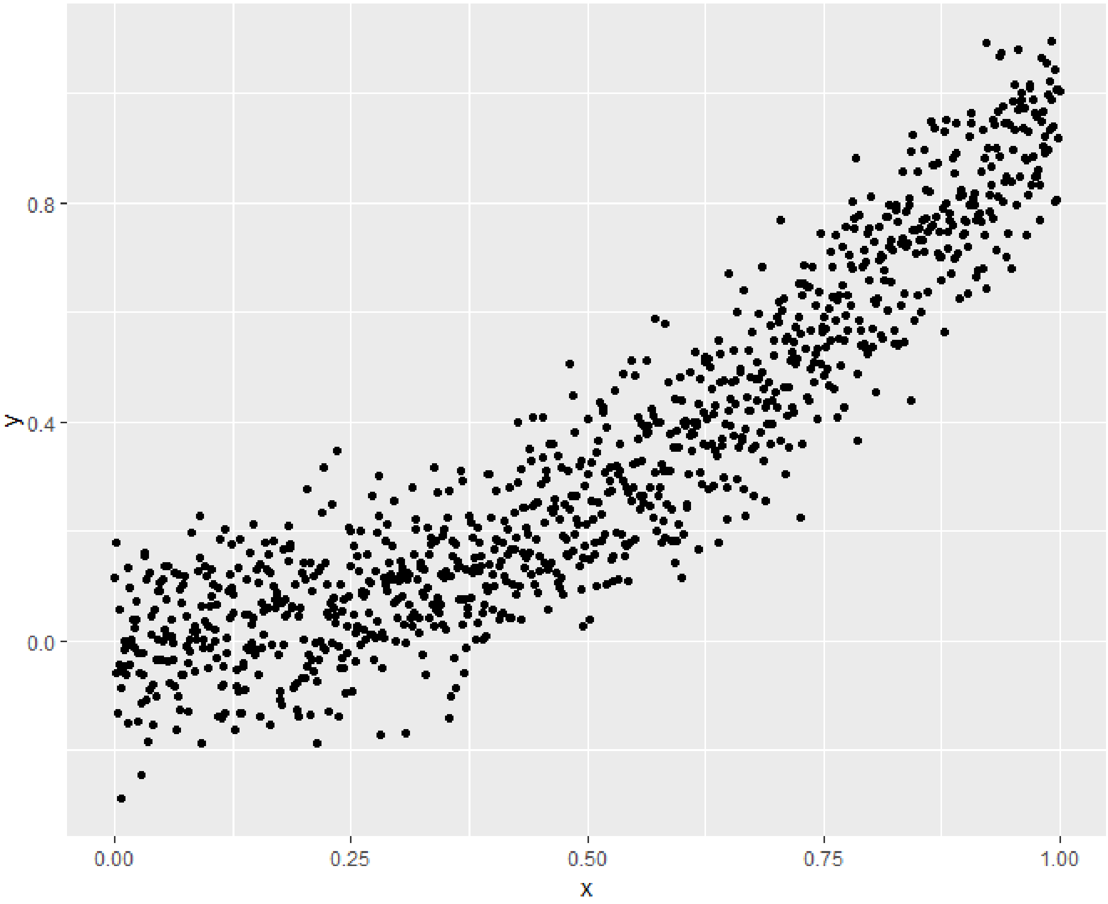
- When you enforce a monotonic constraint on **feature X** with respect to **target Y**, you can safely say: *<u>when all other features are equal, there's no way that increasing **X** will result in a decreasing prediction of **Y**</u>*

1 Monotonicity constraints (2/2) B G

- Let's take a simple parabola with added gaussian noise

```r
x <- seq(0, 1, by = 0.001)
y <- x^2 + rnorm(length(x), sd = 0.1)
```

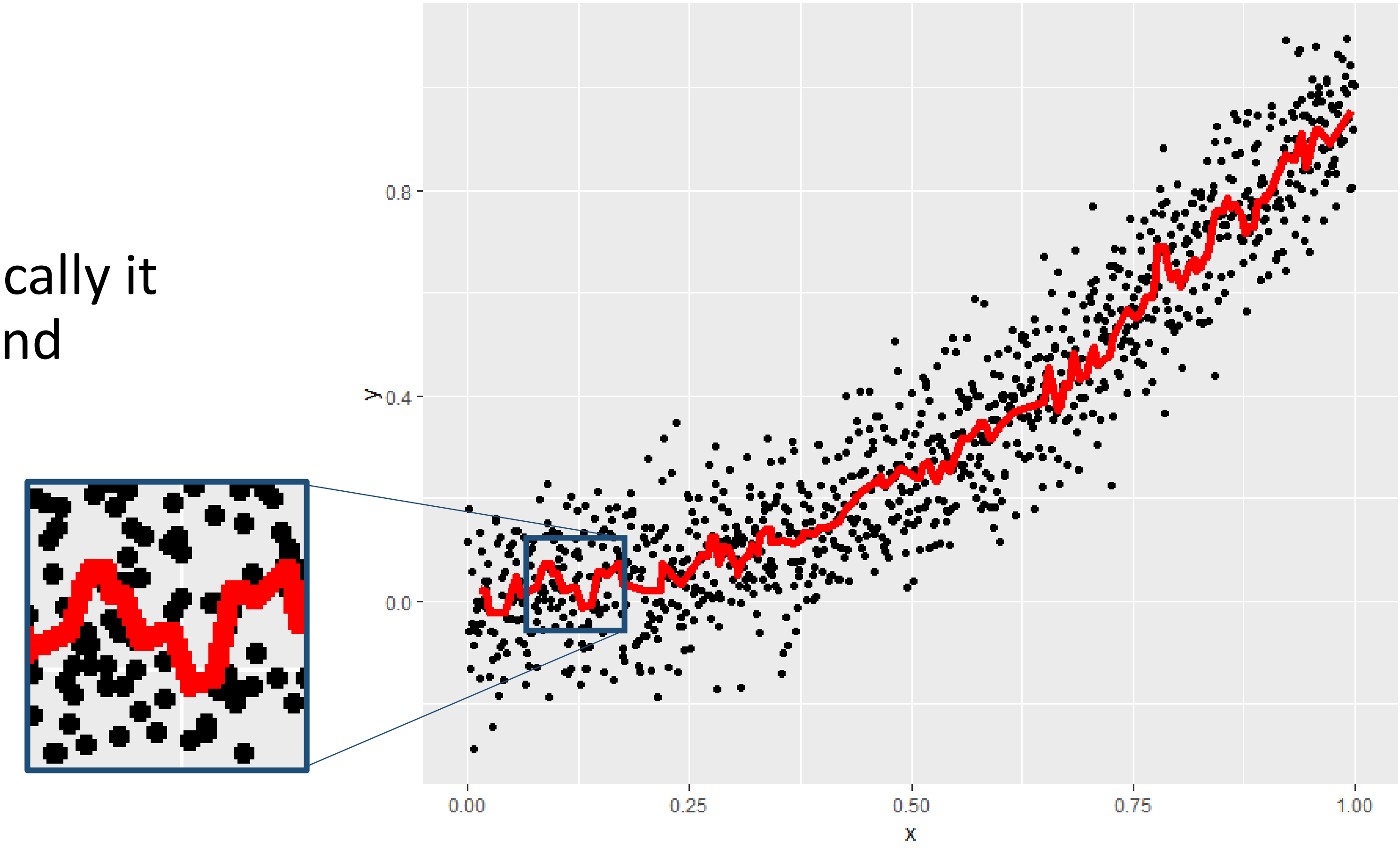
- It's a one feature, one target problem



1 Monotonicity constraints (2/2) B G

- Let's fit a simple Lightgbm

- Overall a decent fit, but locally it can show a decreasing trend



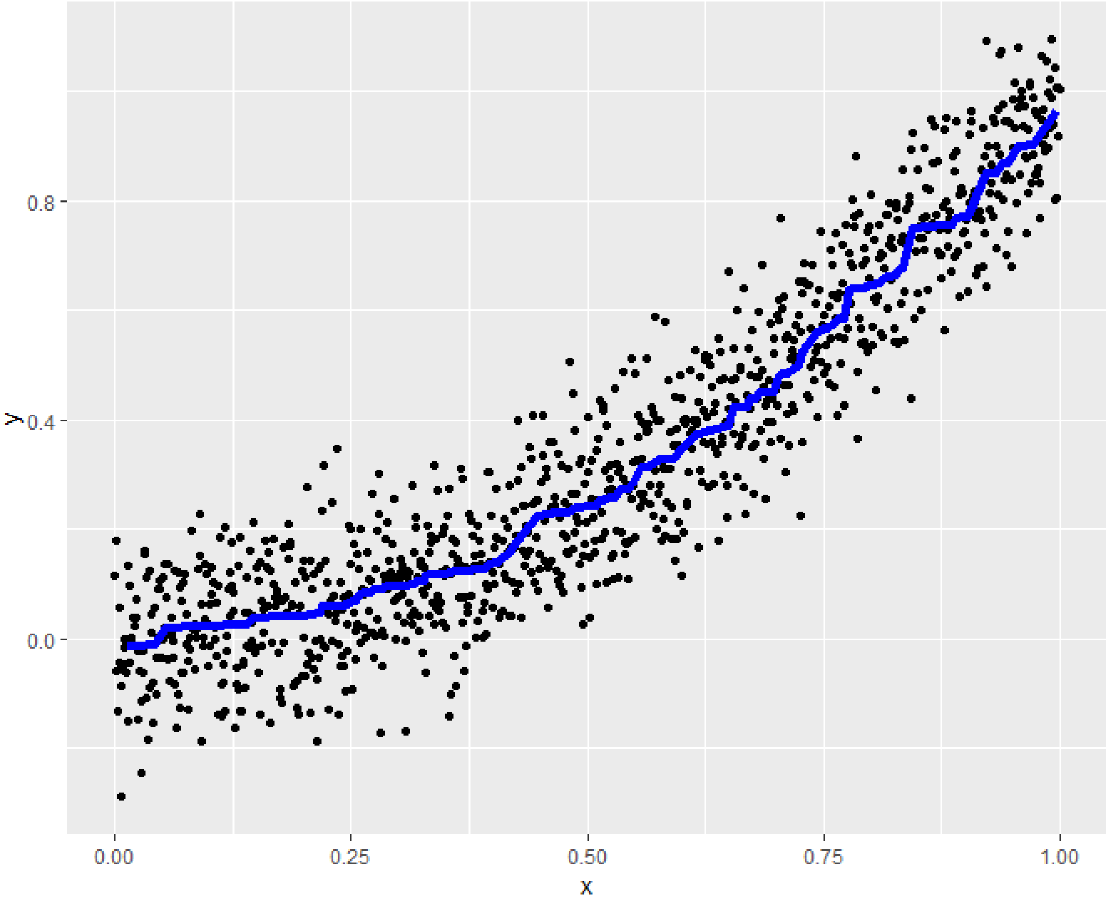
1 Monotonicity constraints (2/2) B G

- Let's add this to the lgb parameters:

```
monotone_constraints = '1'
```

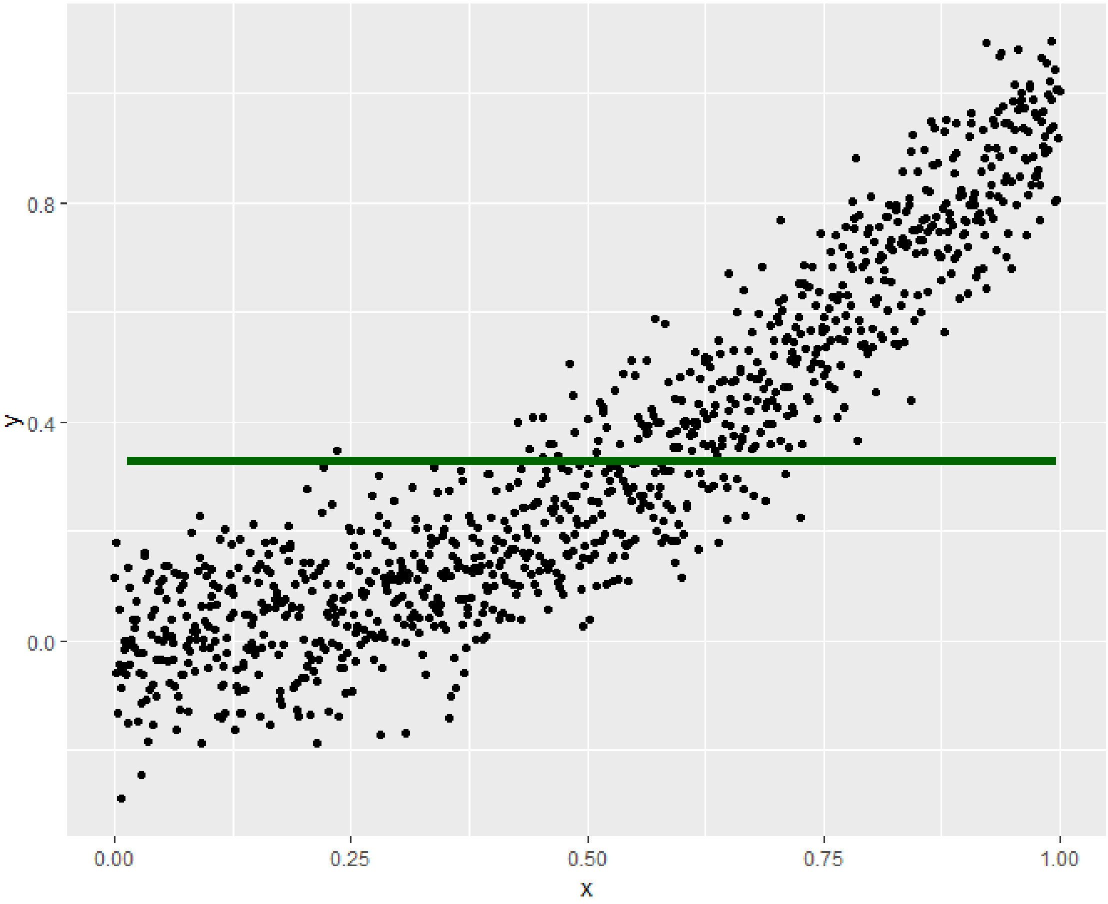
  - +1 = non-descending
  - 0 = no constraint
  - -1 = non-ascending

- Much better fit!



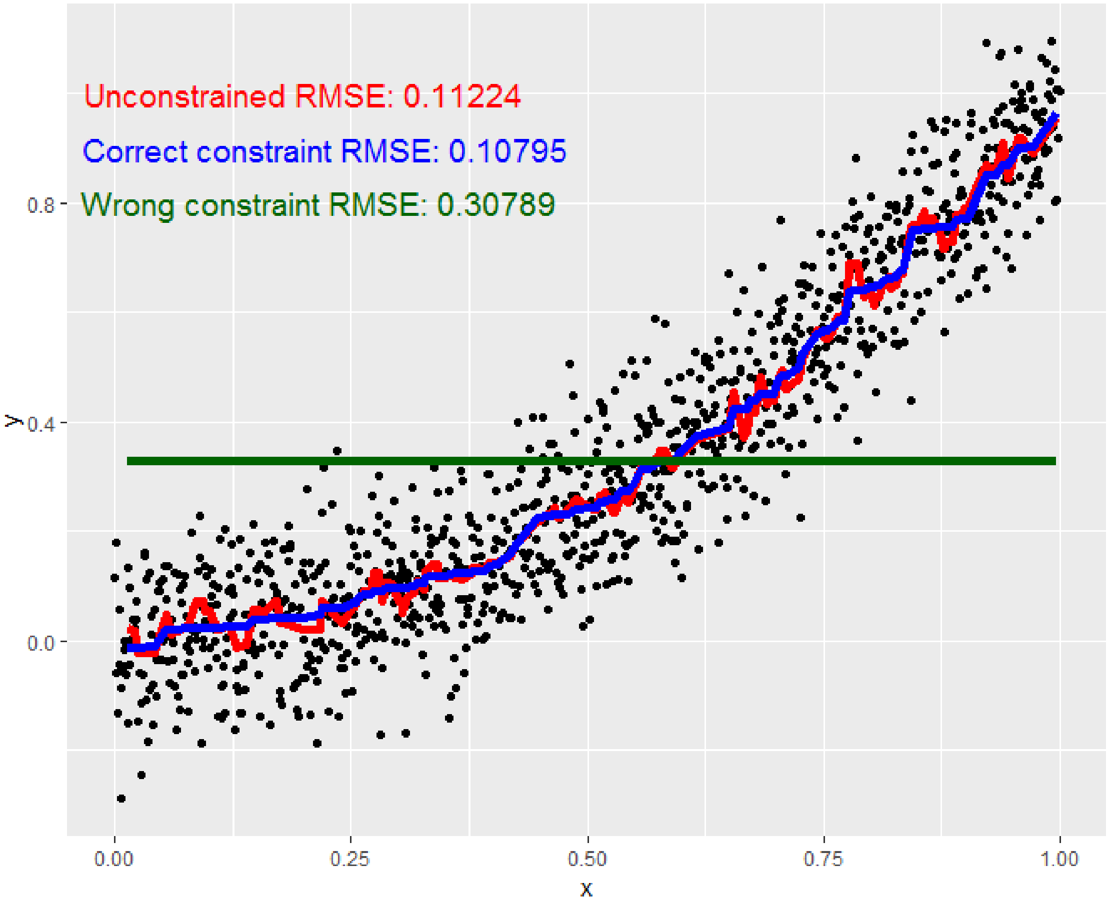
1 Monotonicity constraints (2/2) B G

- If I set -1 as monotonic constraint, the model doesn't even find a single valid split



① Monotonicity constraints (2/2) B G

- Actually the model with the correct constraint not only gives a degree of explainability, it's also the best performing model!



1 Other examples B G

- Build a model that **optimizes a custom metric**
  - Custom evaluation function
  - Custom objective function

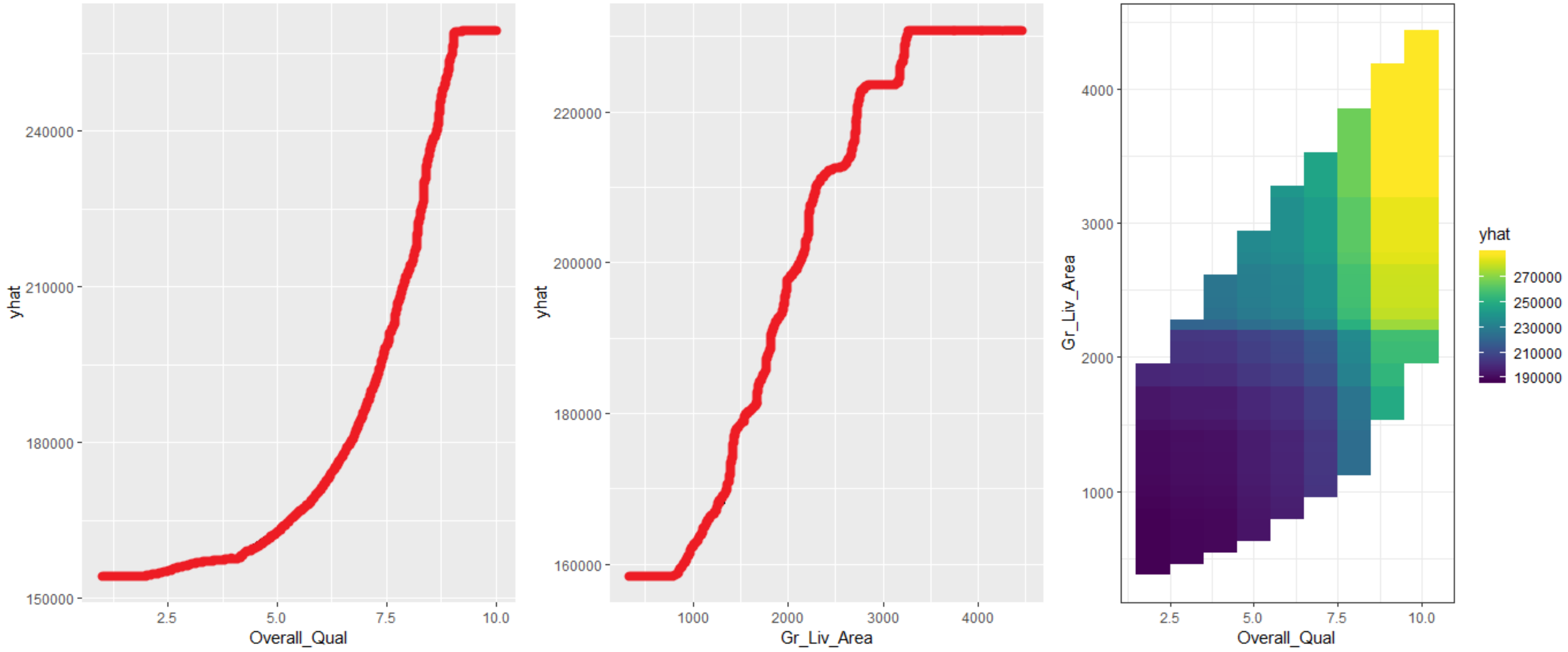
- When you apply custom objective and/or evaluation functions, you can say: *my model (in)directly optimizes this specific metric*

2 Partial dependence plots (1/2) A G

- Main concepts:
  - Once you have highlighted the most important features, it is useful to understand how these features affect the predictions
  - The *partial dependence plots* "average out" the other variables and usually represents the effect of one or two features with respect to the outcome[7]
- PDP analysis is performed **after** a model has been built and is a **global measure**, typically model-agnostic
- With PDP, you can say: *on average, the predictions have this specific behaviour with respect to this one variable (or two of them)*

2 Partial dependence plots (2/2)

A G



Partial Dependence Plots

2-features PDP

3 LIME (1/2) A L

- Main concepts:
  - LIME stands for **Local Interpretable Model-Agnostic Explanations**
  - It assumes that, **locally**, complex models can be approximated with simpler linear models
  - It's based on 4 phases, starting from the single prediction we want to explain
    - **Perturbations**: alter your dataset and get the black box predictions for new observations
    - **Weighting**: the new samples are weighted by their proximity to the instance of interest
    - **Fitting**: a weighted, interpretable model is fitted on the perturbed dataset
    - **Explanation**: of the (simple) local model
  - It works on tabular data, text and images and it's completely model-agnostic

- With LIME, you can say: *this specific prediction is affected by these features, each with its own relevance*

3 LIME (2/2) A L

```
# LIME
explainer <- lime(subset(ames_test, select = -Sale_Price)
, model = ames_xgb)
explanation <- explain(subset(ames_test[1:3,], select = -
Sale_Price), explainer, n_features = 8)
plot_features(explanation, ncol = 1)
```

Built with R package "lime"

First, build an *explainer* with data and a model (here, a classic XGBoost)
Then, create explanations specifying the number N of features you want to include

3 Other examples A L

- **Shapley** additive explanations
  - Uses **game theory** to explain predictions
  - For each prediction, you get the contribution of each variable to that specific case
  - Natively a local model, can be extended to analyze globally the behaviour of a model
  - Fast implementation for GBTs available

- With Shapley, you can say: *in this specific prediction, each feature gives this exact contribution*

ML Interpretability – Recap & Examples

Need	Example	Approach
Enforce some kind of *expected behaviour* in a ML model 1	**Real estate**: floor surface vs. house price for two identical apartments (location, quality, etc.)	**Enforce monotonicity before building a ML model** B G
Show the effects of different features on a specific target, across a large population 2	**Healthcare**: most **important variables** that are linked to a form of illness and what their **impact** is	**Feature importance + PDPs** A G
Understand a single prediction and possibly define ad hoc strategies based on individual analysis 3	**Customer churn**: for each customer in the top N% likely to churn, identify the main reason(s) and give actionable insights to define **individual marketing campaigns**[10]	**LIME + Shapley** A L

Wrap up

ML Interpretability – Conclusions

- Regulation, human nature and in general the desire of **fair, robust and transparent models** are all good reasons to dig into ML interpretability
- There are **many ways** to make a ML model interpretable with radically distinct approaches, but always consider:
  - Data first
  - Tricks and expedients to use before building a model
  - Ex-post global explanations
  - Ex-post local explanations
- This field had a **tremendous growth** in the last 2-3 years and currently allows high-performance models (like GBTs) to have a more than reasonable level of interpretability
- An **interpretable model** can be more robust, insightful, actionable… simply **better**

References

- [1] https://www.datanami.com/2018/08/07/its-still-early-days-for-machine-learning-adoption/
- [2] "Why Should I Trust You?": Explaining the Predictions of Any Classifier – Marco Tulio Ribeiro, Sameer Singh, Carlos Guestrin (2016) – https://arxiv.org/abs/1602.04938
- [3] Towards A Rigorous Science of Interpretable Machine Learning – Finale Doshi-Velez, Been Kim (2017) - https://arxiv.org/abs/1702.08608
- [4] An introduction to Machine Learning Interpretability – Patrick Hall and Navdeep Gill – O'Reilly
- [5] https://github.com/dmlc/xgboost/blob/master/R-package/demo/custom_objective.R
- [6] https://medium.com/the-artificial-impostor/feature-importance-measures-for-tree-models-part-i-47f187c1a2c3
- [7] https://bgreenwell.github.io/pdp/articles/pdp-example-xgboost.html
- [8] https://christophm.github.io/interpretable-ml-book/
- [9] https://github.com/slundberg/shap
- [10] https://medium.com/civis-analytics/demystifying-black-box-models-with-shap-value-analysis-3e20b536fc80

- *Icons made by Freepik from www.flaticon.com*

Thank you!