Explainable AI - Lecture 1

Preliminaries, terminology, interpretability and global surrogate models

norwegian open ai lab
NTNU

This course

Week 1: Interpretability, surrogate models, local model-agnostic explanations

Day 1 (Monday 12. Jan): Preliminaries, terminology, interpretability, global surrogate models
Day 2 (Tuesday 13. Jan): Counterfactuals
Day 3 (Wednesday 14. Jan): Local surrogate models and LIME
Day 4 (Thursday 15. Jan): The Shapley decomposition
Day 5 (Friday 16. Jan): SHAP

The first assignment is due on Monday the 19th.
The coding assignments are crucial for you to get familiar with the methods.

This course

Week 2: Global model-agnostic explanations and model specific explanations

Day 6 (Monday 19. Jan): Coding time and assignment demonstration

Day 7 (Tuesday 20. Jan): Partial Dependence Plots and Accumulated Local Effects

Day 8 (Wednesday 21. Jan): Global feature importance, LOFO and SAGE. Feature importance metrics

Day 9 (Thursday 22. Jan): Concept based explanations

Day 10 (Friday 23. January): Transformer explanation and interpretation

The second assignment is due on Tuesday the 27th of January

The coding assignments are crucial for you to get familiar with the methods.

This course

Week 3: Model specific explanations and exam

Day 11 (Monday 26. Jan): CNN interpretation and explanation methods

Day 12 (Tuesday 27. Jan): Coding time and assignment demonstration

Day 13 (Wednesday 28. Jan): Exam

The second assignment is due on Tuesday the 27th of January

The coding assignments are crucial for you to get familiar with the methods.

Evaluation

Assignments count 40% (20% each)

Final exam counts 60%

This course

Each lecture has

1) Additional resources, usually the paper describing the method(s) discussed in the lecture. You should have a look at these and can use them for increased depth, but *everything you need to know for the exam is contained in the lectures.*

2) Homework, constituting part of the ongoing assignment. *I strongly recommend that you work a bit on the assignment every day.*

Preliminaries

Machine learning

Machine learning is an optimisation problem where data is used by an optimisation algorithm to adjust parameters in a way that minimizes some loss function, resulting in a trained model.

*In this course, we will use **linear regression, scikit-learn decision trees, XGBoost and neural networks in pytorch**. If you are not familiar with these, please set aside extra time for practicing this.*

Notation

Our machine learning models are usually denoted f, shorthand for with f_θ, with θ denoting the fitted parameters.

Notation

Our machine learning models are usually denoted f, shorthand for with f_θ, with θ denoting the fitted parameters.

The true underlying data distribution is in general unknown, and we draw samples from it to generate datasets for training / validation / testing. We assume that this process be random and representative. Since it is common to use capital letters to denote random variables, datasets can be labelled X_{train}, X_{test}, ...

Notation

Our machine learning models are usually denoted f, shorthand for with f_θ, with θ denoting the fitted parameters.

The true underlying data distribution is in general unknown, and we draw samples from it to generate datasets for training / validation / testing. We assume that this process be random and representative. Since it is common to use capital letters to denote random variables, datasets can be labelled X_{train}, X_{test}, ...

Still, the data - loaded to and stored on our computers - is a fixed resource, consisting of actual data points. Our machine learning models f predict on specific data points, denoted by x. The data has a column per feature, x_i denoting the feature with index i, and data points are sorted into rows, $x^{(i)}$ denoting row index i. The literature is sometimes inconsistent regarding use of upper- or lowercase symbols. (sorry)

In supervised learning, the target is denoted y.

XAI motivation

AI: trying to make machines intelligent (over a summer school) since 1956

Photo: Margaret Minsky

A Proposal for the

DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE

June 17 - Aug. 16

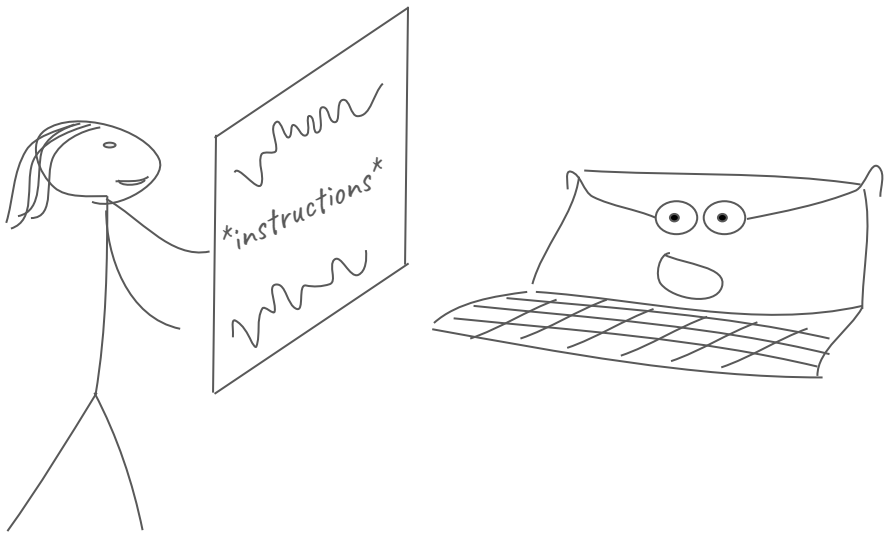
We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

AI: symbolic

Human knowledge represented inside computers.

Expert systems, rules, planning, reasoning, search, …

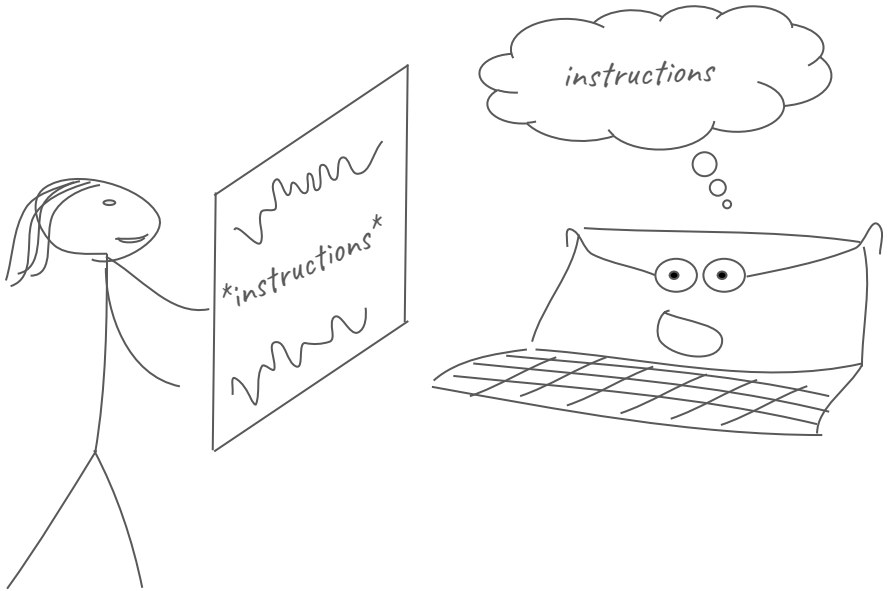
Dominant until the 1980s.



AI: symbolic

Human knowledge represented inside computers.

⇒ logic of processing and relations between symbols represented in a way that is meaningful to humans.

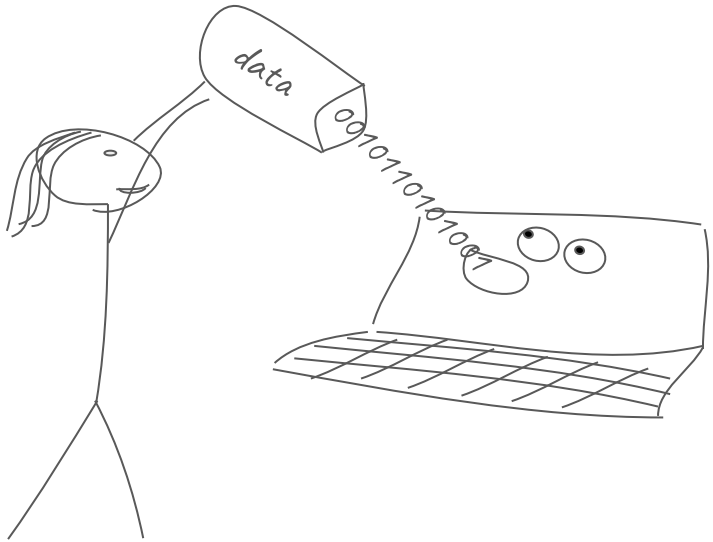


AI: sub-symbolic

How to handle the uncertainties of the real world?

How to adapt to different contexts?

How to gain perceptual abilities?

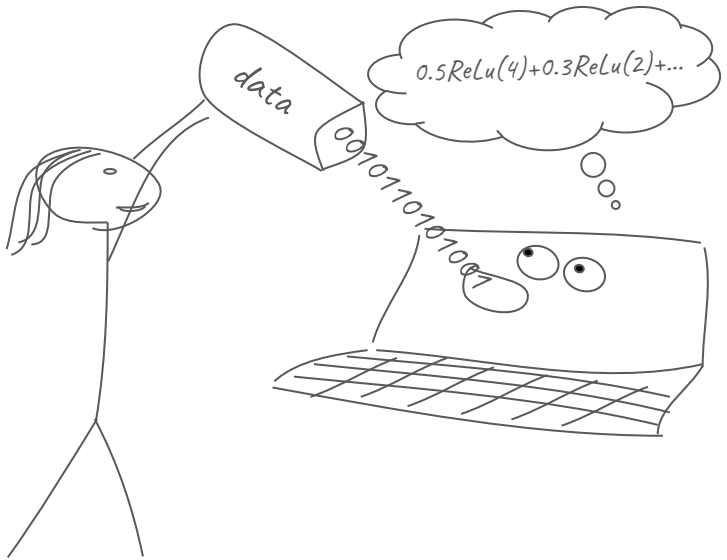


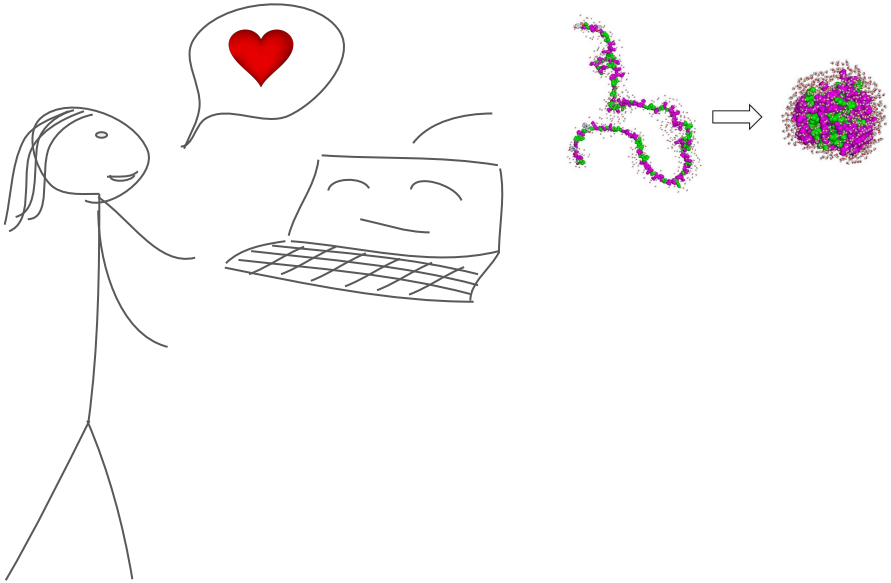
AI: sub-symbolic

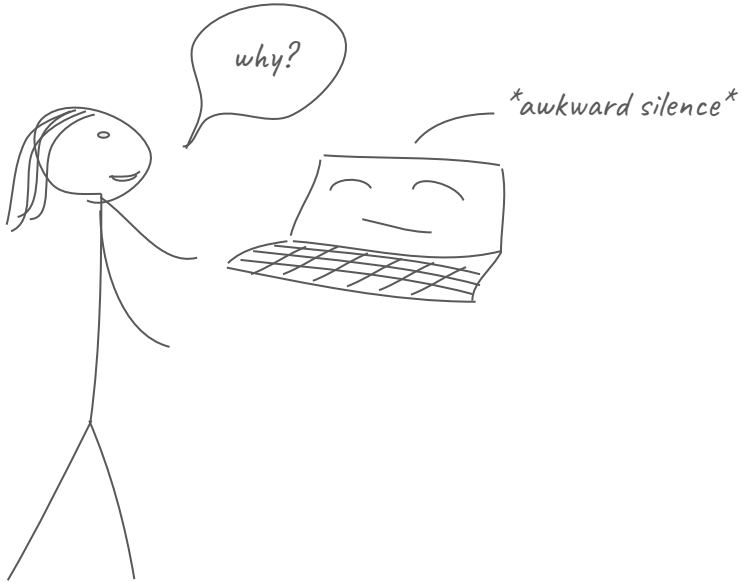
Use statistical methods to “learn”, i.e. detect correlations from data.

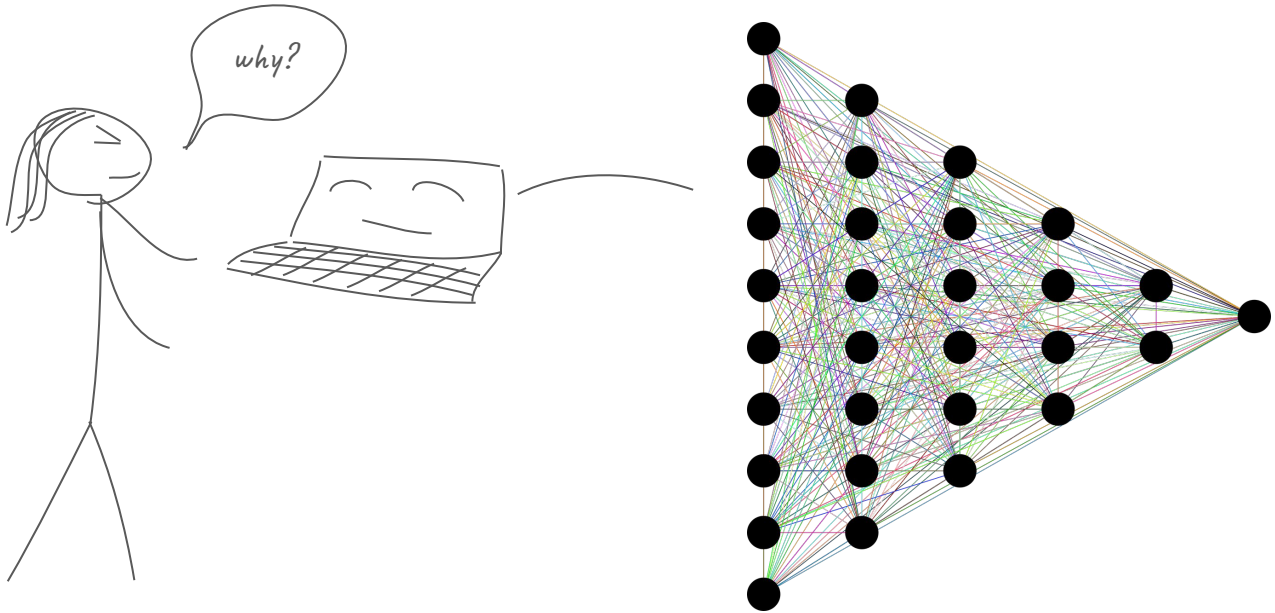
⇒ the extracted correlations are not represented in a way that is meaningful to humans.

The knowledge is *underneath* the symbolic: sub-symbolic.





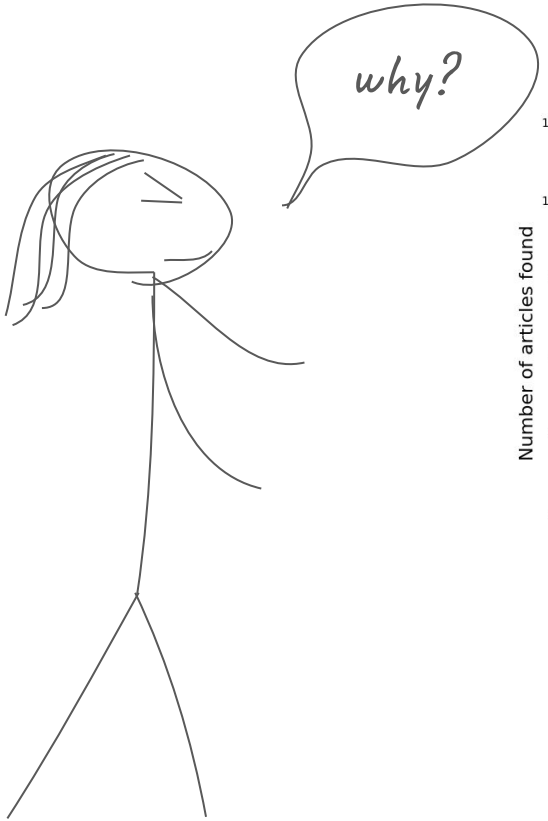

why?
awkward silence

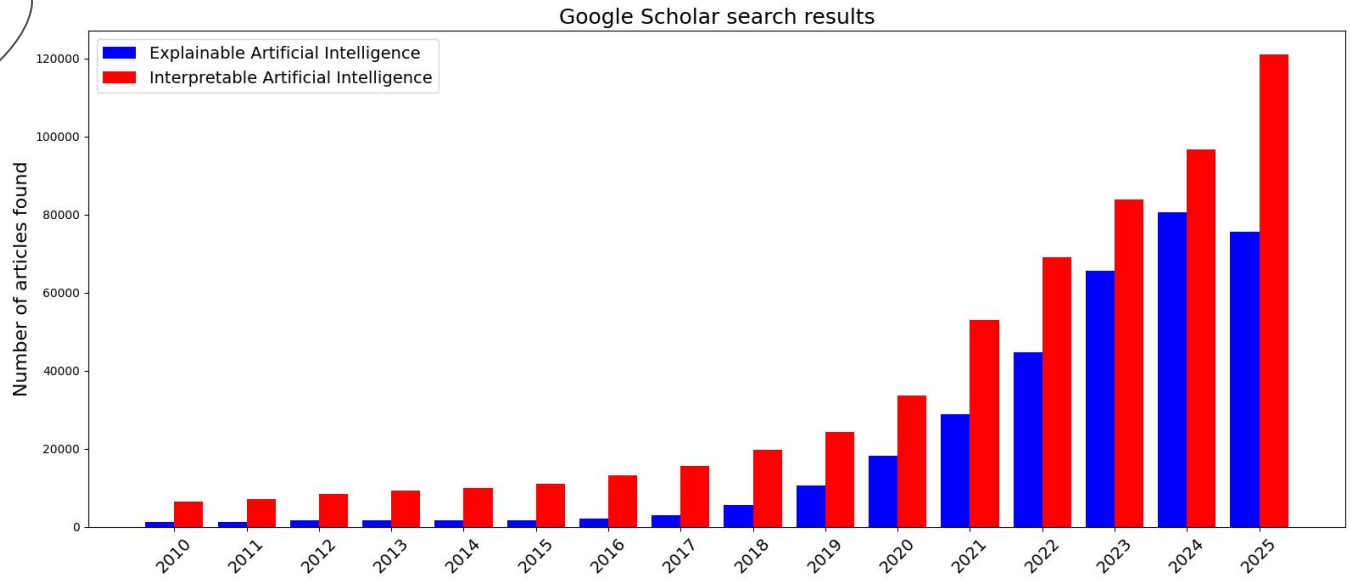


Modern machine learning model types, like gradient-boosted trees and various neural network architectures, are *traceable* but not *interpretable.*

⇒ we don't understand the model no matter how hard and long we stare at it.

We need help / tools / methods to **interpret** or **explain** the models.





norwegian open ai lab

NTNU

USA: DARPA XAI Program (4 years)

2017: “create a suite of new or modified machine learning **techniques that produce explainable models** that, when combined with effective explanation techniques, enable end users to understand, appropriately trust, and effectively manage the emerging generation of AI systems.”

2021: “**There currently is no universal solution to XAI**. Different user types require different types of explanations.”

“The results have produced a more nuanced understanding of XAI uses and users, the psychology of XAI, the challenges of measuring explanation effectiveness, as well as producing a new portfolio of XAI ML and HCI techniques.”

“XAI will continue as an active research area for some time.”

DARPA's explainable AI (XAI) program: A retrospective, https://doi.org/10.1002/ail2.61

Legal perspectives - Europe

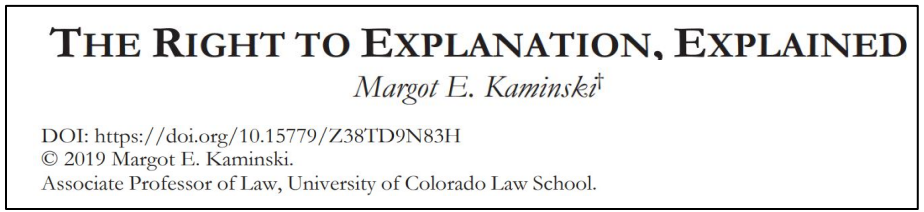
The EU General Data Protection Regulation, GDPR (2016), has in many respects established global data and privacy protection regulation standards.

Articles 22 and 35(3)(a) give data owners a right to:

"**meaningful information** about the logic of processing"

"the significance and envisaged consequences"

"an **explanation** of the decision reached after [algorithmic] assessment"


THE RIGHT TO EXPLANATION, EXPLAINED

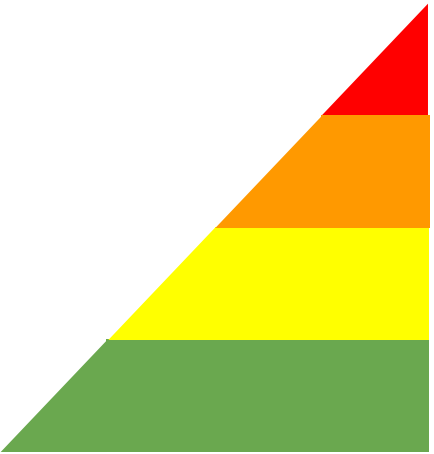
Margot E. Kaminski†

DOI: https://doi.org/10.15779/Z38TD9N83H

Associate Professor of Law, University of Colorado Law School.

Legal perspectives - Europe

The EU is currently implementing the world's first *AI specific regulation*, the AI Act, structured according to the AI system's *systemic risk.*



Unacceptable Risk	AI systems that pose significant harm or violate fundamental rights.	Social scoring, Manipulative AI that influences behavior without consent
High Risk	Regulated systems that could impact health, safety, or rights significantly.	Biometric identification systems, AI in critical infrastructure (transport, healthcare)
Limited Risk	Subject to transparency obligations; users must be informed they are interacting with AI.	Chatbots, Deepfakes
Minimal Risk	Unregulated systems that pose little to no risk.	Video games, spam filters

https://artificialintelligenceact.eu

Legal perspectives - Europe

The AI Act: Articles 11 and 13, and Annex IV

"High-risk AI systems shall be designed and developed in such a way to ensure that their operation is **sufficiently transparent** to **enable users to interpret** the system's output and use it appropriately. An appropriate type and degree of transparency shall be ensured"

"assessment of the human oversight measures needed, including an assessment of the technical measures needed to **facilitate the interpretation of the outputs** of AI systems **by the users**"

Legal requirements

Various regulations around the world give users the ***"right to an explanation", "ensured transparency" and "facilitated interpretation"***

Europe: *GDPR (General Data Protection Regulation), Artificial Intelligence Act, ...*

US: *California, Colorado, Virginia, Connecticut, New York, ... State privacy laws*

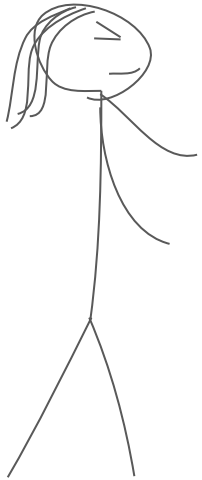
South America: *LGPD (Lei Geral de Proteção de Dados)*

Africa: *South Africa, Kenya, Uganda, Tanzania, ... Data Protection Acts*

... all grant certain rights to subjects whose data are processed.

... all mention rights to insight into inferences, transparency, explanations, but **none of these clarify what exactly is required** from an explanation.

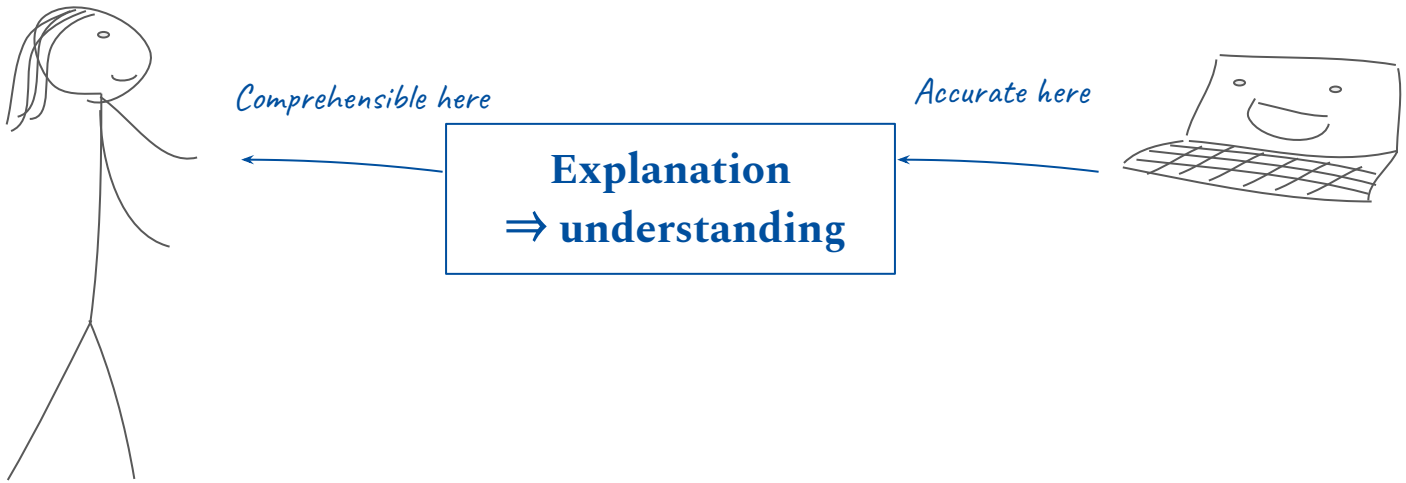
Explainable AI (XAI)



What do we need here?

Explainable AI (XAI)

Methods that explain the decision processes *or* outputs of AI systems in a way that increases human understanding.



Terminology (one variation...)

Interpretability: The property of being directly understandable to humans (especially without further tools).

Explainability: The notion of explanation as an interface between humans and the model that is, at the same time, both an accurate proxy of the model and comprehensible to humans.

Explicability: The degree to which a model can be interpreted or has been explained is often referred to as explicatility in law and ethics. AI researchers rarely use this term.

Transparency: Can mean several things:

Sometimes used as a collective term for interpretability and explainability,
sometimes used to describe models that are in and of themselves understandable, and
sometimes used referring to full access to the model, i.e. the model not being closed source/proprietary.

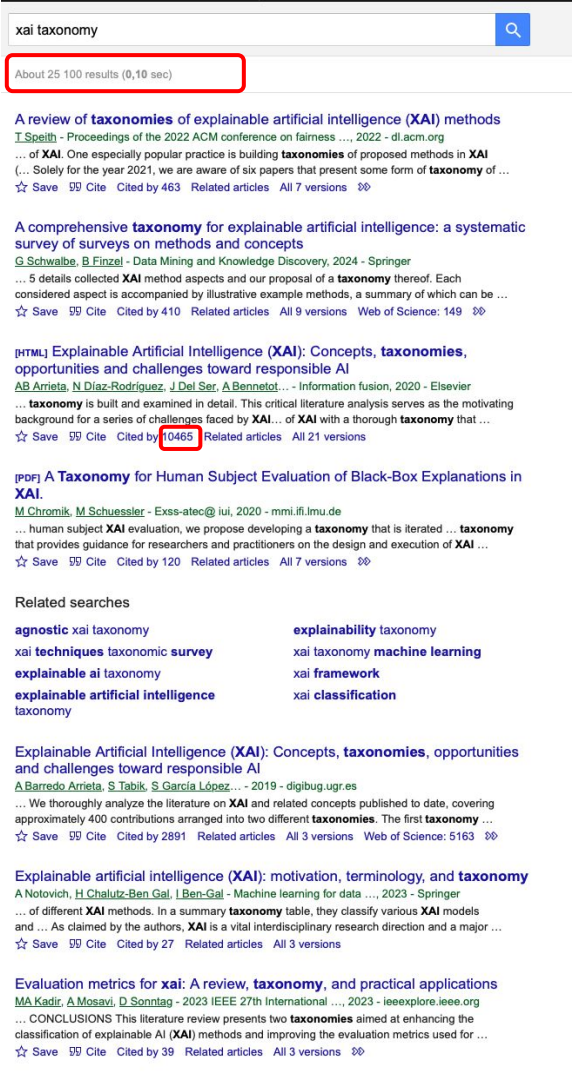
Terminology

Mechanistic interpretability (coined in 2024 [1]) aims to understand the workings of neural networks by analyzing their computations as mechanisms. It can be regarded a subfield of XAI, while some regard it as a separate subfield of neural network research.

The approach is akin to how all computer programs can be reverse-engineered to understand their functions.

We'll talk more about the "difference" between XAI and mechint in a few slides :)

[1] https://aclanthology.org/2024.blackboxnlp-1.30.pdf



The literature is so huge and messy that people have started writing reviews of all the taxonomies :)

Taxonomy

There are many ways to organise XAI, but no fully agree-upon categorisation. However, the following terms are well-established:

When the explanation is generated

Ante-hoc: The explanation functionality is integrated into the model training ← *not covered in this course*

Post-hoc: The explanation generated for a fitted model, i.e. after training is finished.

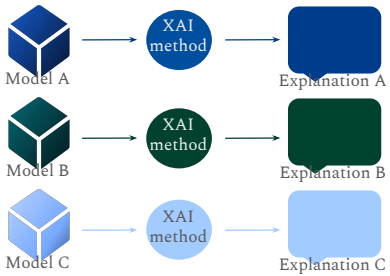
Taxonomy

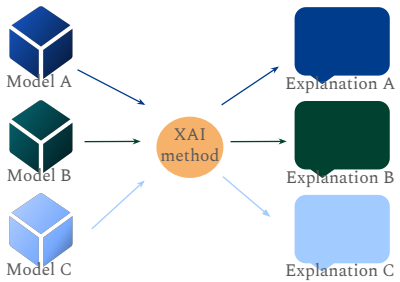
There are many ways to organise XAI, but no fully agree-upon categorisation. However, the following terms are well-established:

What kind of model the explanation method expects

Model-specific: The explanation method exploits or depends on the model's architecture structure.

Model-agnostic: The explanation method does not exploit or depend on the model itself, and can be used for any model (that receives data and produces a prediction).





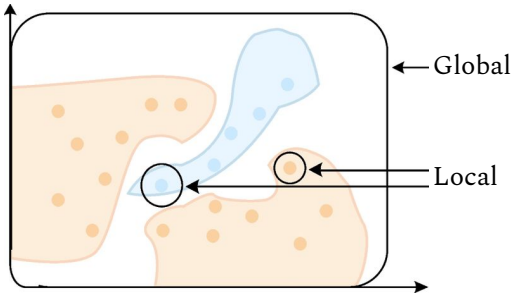
Taxonomy

There are many ways to organise XAI, but no fully agree-upon categorisation. However, the following terms are well-established:

What the method generates an explanation about

Local: The explanation is for a single data point / prediction.

Global: The explanation is for the whole model.
Note that such an explanation can depend on the specific data points used to generate the explanation, but the explanation itself is still not a statement about the model's behaviour for specific data points.



Taxonomy

Ante-hoc *(not what we're doing)*

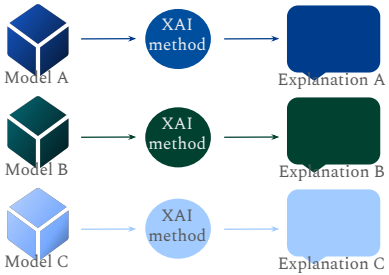
Post-hoc *(what we're doing)*

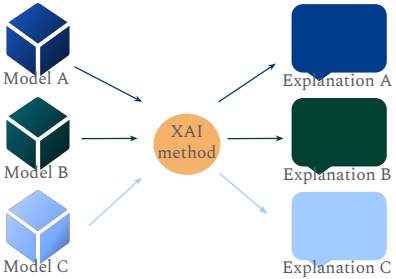
Model-specific

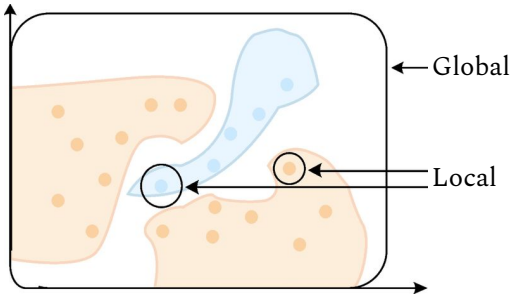
Model-agnostic

Local

Global





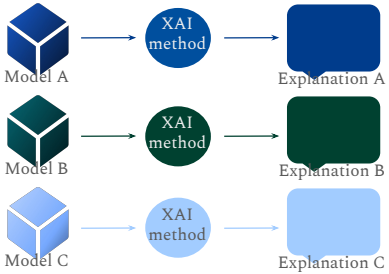


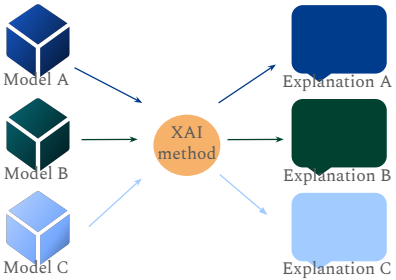
Taxonomy

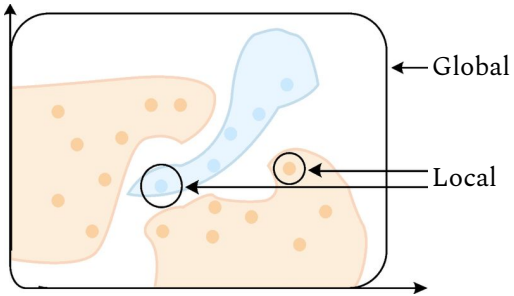
Ante-hoc *(not what we're doing)*
The explanation functionality is integrated into the model training
Post-hoc *(what we're doing)*
The explanation generated for a fitted model.

Model-specific
The explanation method exploits or depends on the model architecture.
Model-agnostic
The explanation method does not exploit or depend on the model.

Local
The explanation is for a single data point / prediction.
Global
The explanation is for the whole model.







Placement of post-hoc methods

At the end of this course, you should be able to place the methods we study in the following table

Post-hoc methods	Model-agnostic	Model-specific
Local		
Global		

This course

We will touch upon interpretability, but primarily focus on **explanation methods**.

There is no XAI textbook (yet), but the Interpretable ML book (available online) can be a good resource.

The field is still young, and the "old classics" are all from around 2017.

There is a *vast* number of XAI methods, and we will study a selection that is intended to cover the most widely used methods and also be representative of the broadness of XAI approaches.

For each lecture, the original papers will be provided as recommended reading. However, this course is condensed, and you should **prioritise the coding assignments**.

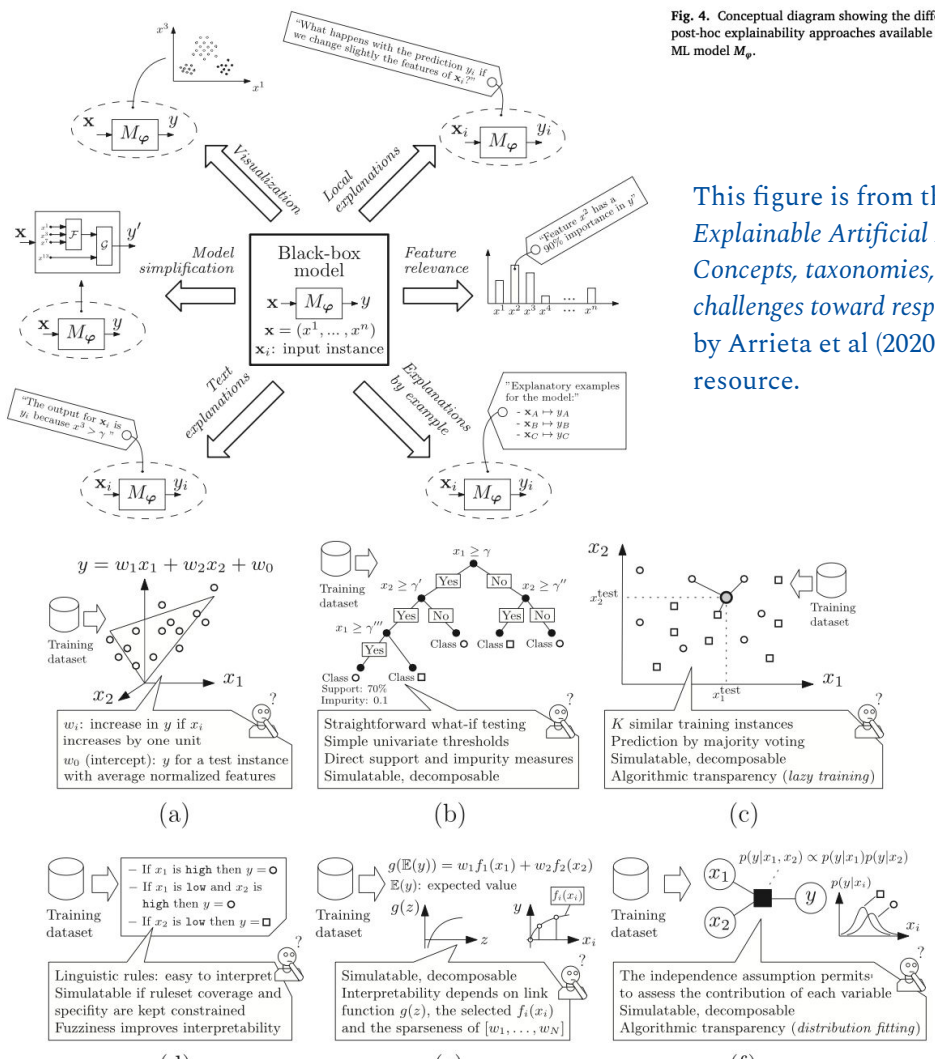


Fig. 4. Conceptual diagram showing the different post-hoc explainability approaches available for a ML model M_φ.

This figure is from the highly cited *Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI* by Arrieta et al (2020) - today's additional resource.

One way to mentally organise XAI - intuitively

A mystical black box falls down from the sky

Three approaches to understanding the box:

Find a phenomenon that behaves like the box ← **Surrogate box explanation**

Poke the vox and see what happens ← **Box agnostic explanation**

Open the box and look inside ← **Box specific explanation**

One way to mentally organise XAI

We have a machine learning model f, approximating some function.

This takes inputs x, and is defined by its parameters θ:

$$f(x,\theta)$$

One way to mentally organise XAI

We have a machine learning model f, approximating some function.

This takes inputs x, and is defined by its parameters θ:

$$f(x,\theta)$$

There are three things here we can study:

Can f be replaced by an interpretable function?	**← Surrogate model explanation**
Sensitivity analysis of the model wrt x	**← Model agnostic explanation**
Structure of f and significance of θ	**← Model specific explanation**

... and fit mecint into the picture

XAI

Encompasses surrogate model, model agnostic *and* model specific methods.

The epistemic goal is a human-understandable account of the model.

"Why does the model behave as it does?"
"Why did the model produce this output?"

Mechint

Requires access to the model.

The epistemic goal is a causal, mechanistic understanding of the model.

"How does the model work?"
"How does the model compute its outputs?"

If this is confusing, don't worry. This course is about XAI methods.
For illustration, we will do some mechint in the part on model-specific methods (end of week 2 + week 3).

One way to mentally organise XAI

We have a machine learning model f, approximating some function.

This takes inputs x, and is defined by its parameters θ:

$$f(x,\theta)$$

There are three things here we can study:

Can f be replaced by an interpretable function?	**← Surrogate model explanation**
Sensitivity analysis of the model wrt x	**← Model agnostic explanation**
Structure of f and significance of θ	**← Model specific explanation**

This course - reminder

Week 1: Interpretability, surrogate models, local model-agnostic explanations

Terminology, taxonomy, interpretability and global surrogate models

Counterfactuals

Local surrogate models and LIME

The Shapley decomposition

SHAP

Week 2: Global model-agnostic explanations and model specific explanations

Partial Dependence Plots and Accumulated Local Effects

Global feature importance, LOFO and SAGE. Feature importance metrics

Concept based explanations

Transformer explanation and interpretation

Week 3: Model specific explanations and exam

CNN interpretation and explanation methods

Interpretable models

Can you think of any interpretable model types?

Can you think of any interpretable model types?

Linear regression

Logistic regression

Decision trees

Rule lists

1-layer neural networks?

Naive Bayes?

k-Nearest Neighbors?

Sparse polynomial models?

... depends on the recipient, perhaps.

What makes a model interpretable?

~“a model is interpretable if a user can correctly and efficiently predict its results”

Kim et al (2016)

Linear models

Linear models

In linear regression, the target value is modeled as a linear combination of the d features,

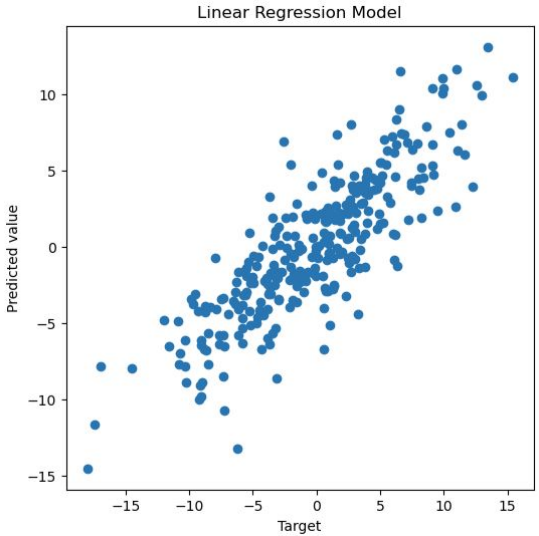
$$y = w_1 x_1 + w_2 x_2 + ... + w_d x_d + b$$

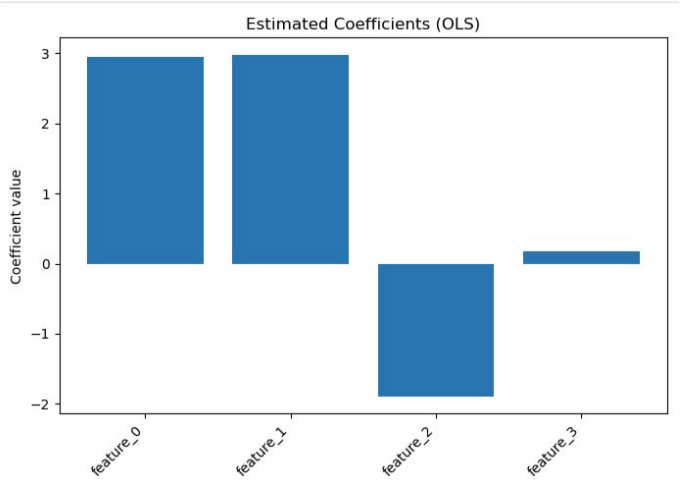
In binary logistic regression, the sigmoid function is applied to the above expression to yield the class probability.

Linear regression example

	feature_0	feature_1	feature_2	feature_3	target
0	-1.348002	0.578496	0.647689	-0.138264	-6.973793
1	-0.303994	0.666743	1.579213	-0.234137	-4.009799
2	0.658207	0.069100	-0.463418	0.542560	0.391087
3	0.168199	-0.543045	-1.724918	-1.913280	0.298653
4	1.677755	-0.108684	-0.908024	0.314247	5.943660

Given a dataset with 4 features and a continuous target, we fit an sklearn LinearRegression (OLS) model



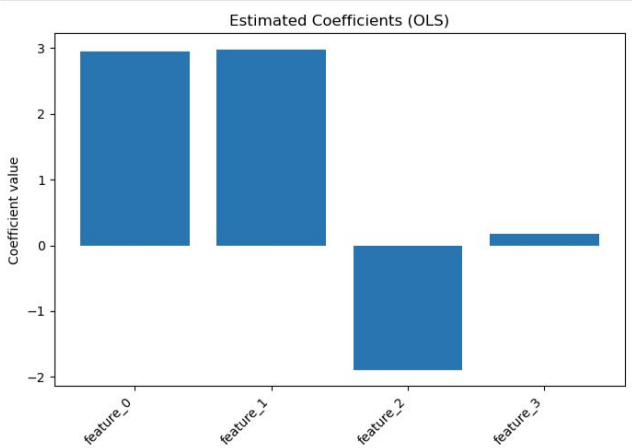


The model is a decent fit, and we have the coefficients [2.94, 2.98, -1.90, 0.17]

Interpreting coefficients

$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$

What can you say about the features based on this?



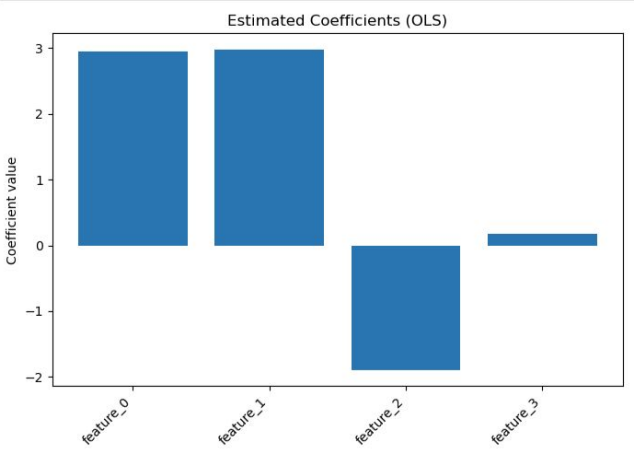
Interpreting coefficients

$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$

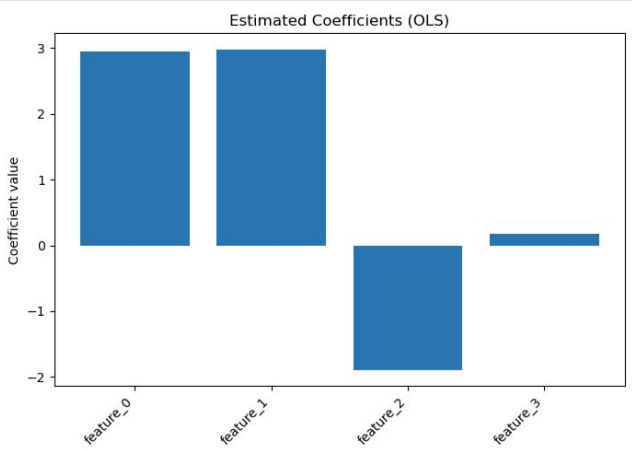
Changing features x_0 and x_1 affect the model prediction the most.

Feature x_2 has a negative relative effect on the model prediction

Changing feature x_3 has the least impact on the model prediction.



Interpreting coefficients



$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$

Changing features x_0 and x_1 affect the model prediction the most.

Feature x_2 has a negative relative effect on the model prediction

Changing feature x_3 has the least impact on the model prediction.

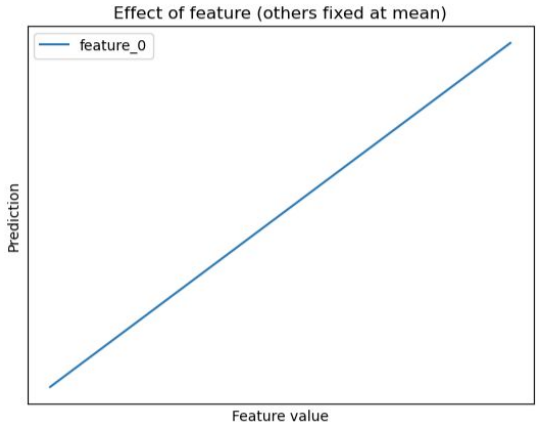
Say look at one feature at a time, varying it from its minimum to its maximum value, and fit a linear model to the target value y.

Next, we plot the range of this one feature against this model's prediction.

Finally, we repeat the process for all features. What will the resulting plots look like?

Interpreting coefficients

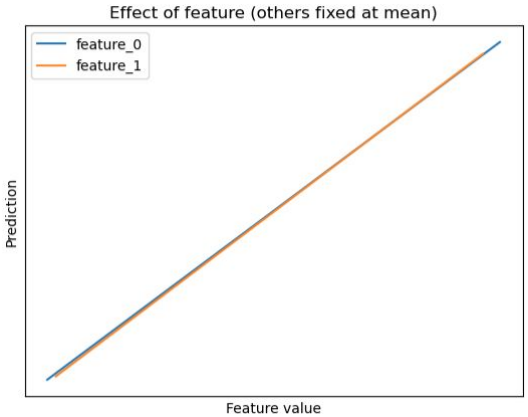
$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$



Assuming the same range for feature x_1, what should the next line look like?

Interpreting coefficients

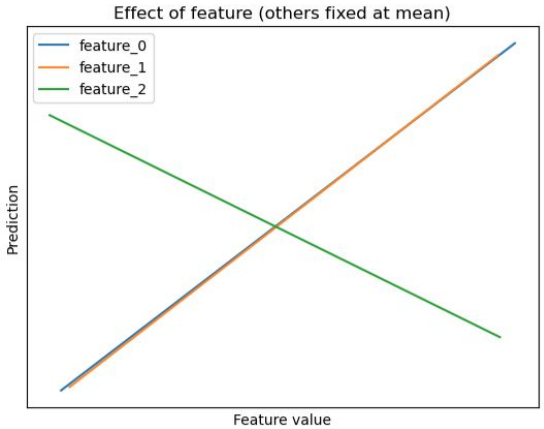
$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$



Assuming the same range for feature x_2, what should the next line look like?

Interpreting coefficients

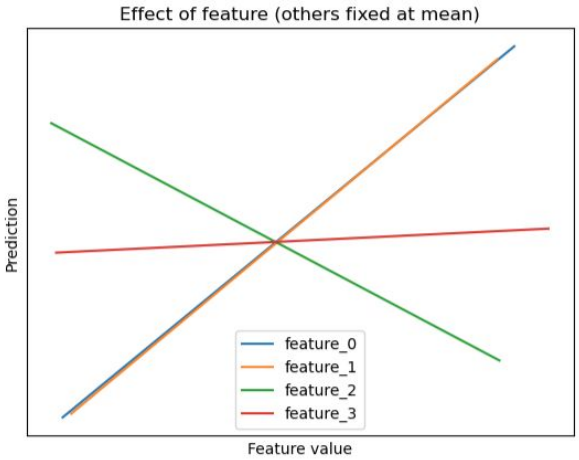
$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$



Assuming the same range for feature x_3, what should the next line look like?

Interpreting coefficients

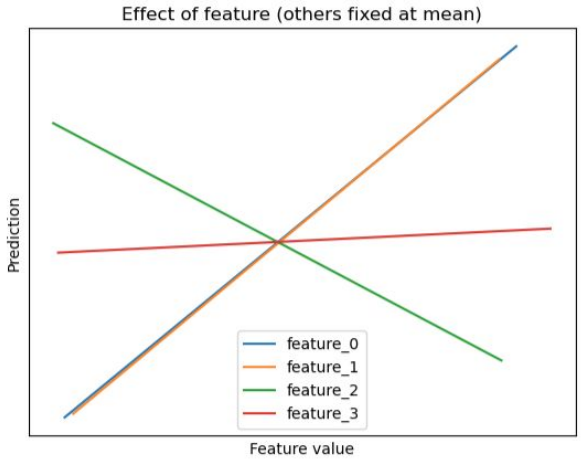
$$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$$



Does this make sense? What are we looking at?

Interpreting coefficients

$$\hat{y} = 2.94\, x_0 + 2.98\, x_1 - 1.90\, x_2 + 0.17\, x_3$$



Each line shows the feature coefficient in a linear model of y using only that feature. ***This reflects the feature's marginal contribution to y.***

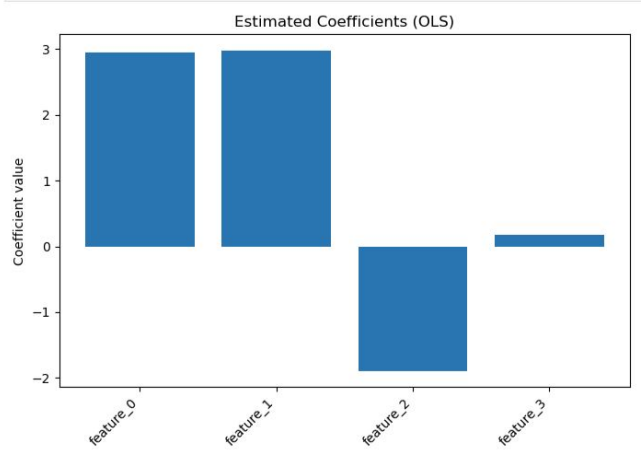
There is no conditioning on other features.
The lines do not say anything about interactions in the full model.

Interpreting coefficients - common misunderstanding

$\hat{y} = 2.94\ x_0 + 2.98\ x_1 - 1.90\ x_2 + 0.17\ x_3$

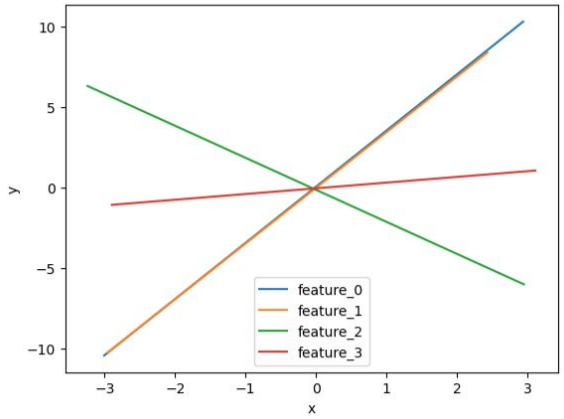
Note: The coefficients in the full linear regression model are not the same as the single regression coefficients.

Still, they are very similar. *Why could this be?*




```python
# Single variable regression
for feat in feature_names:
    _ols = LinearRegression()
    _X_train = X_train[feat].to_numpy().reshape(-1, 1)
    _X_test = X_test[feat].to_numpy().reshape(-1, 1)
    _ols.fit(_X_train, y_train.to_numpy())
    y_pred = _ols.predict(_X_test)

    plt.plot(np.sort(_X_test.flatten()), _ols.predict(np.sort(_X_test, axis=0)), label=feat)
    print(_ols.coef_)

plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.show()
```

```
[3.487754]
[3.4566194]
[-1.98948201]
[0.35456995]
```

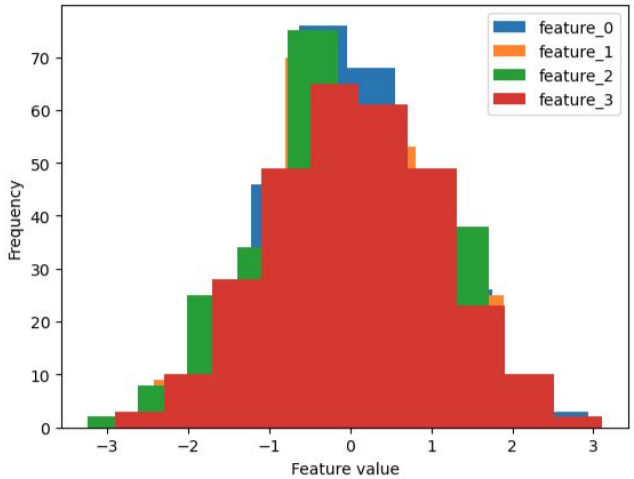


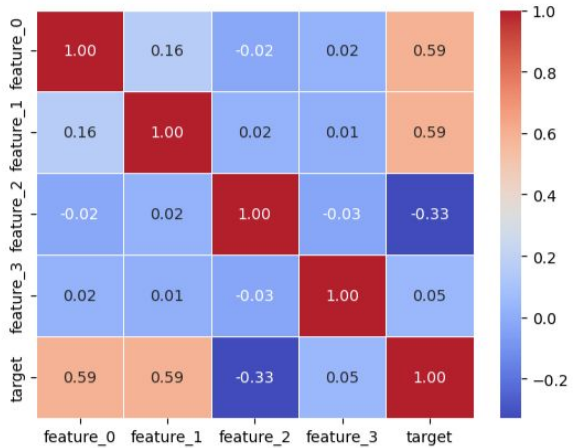
norwegian open ai lab NTNU

Interpreting coefficients

I made the data. The true functional relationship is $y = 3\,x_0 + 3\,x_1 - 2\,x_2 + 0.2\,x_3$

In this data, the features have the same scale, and are only weakly correlated.

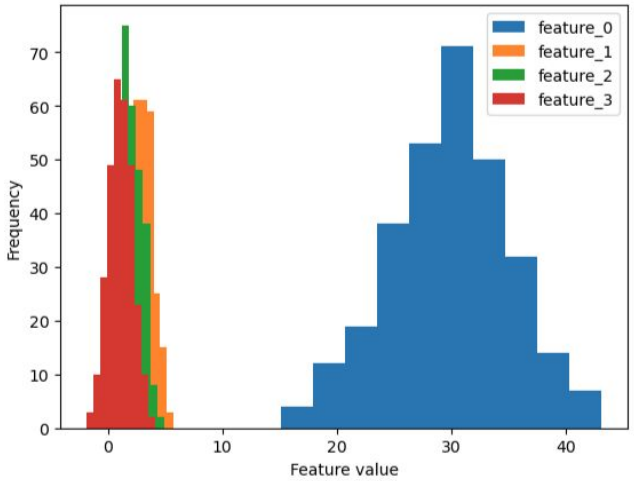


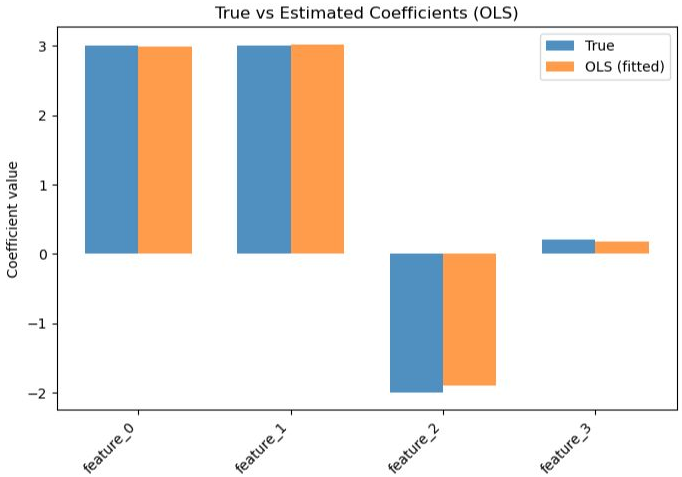


Interpreting coefficients - different scales

We keep the functional relationship, $y = 3\,x_0 + 3\,x_1 - 2\,x_2 + 0.2\,x_3$, but change the distribution of x_0.

We fit a new model, and get ~the same coefficients.

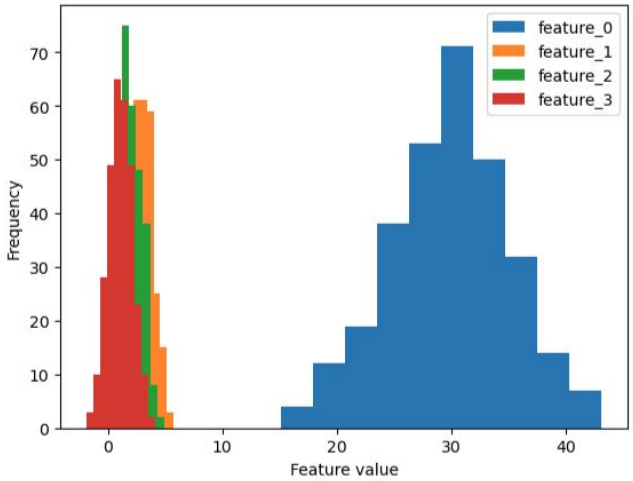


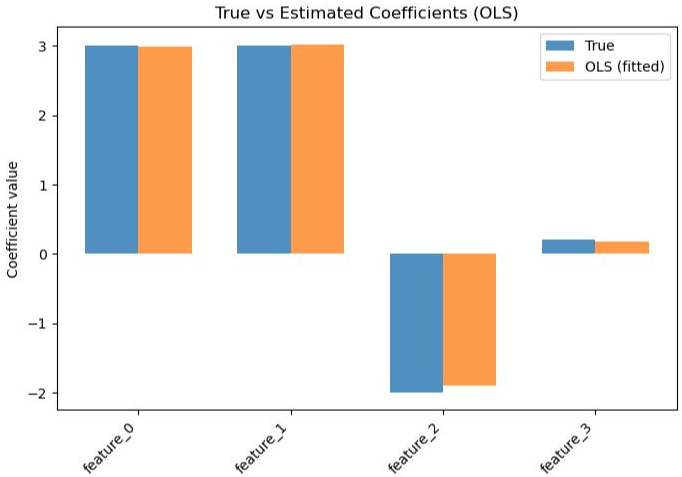


Interpreting coefficients - different scales

If the features have different scales, the relative coefficient sizes might not be very informative.

1) *Why?*

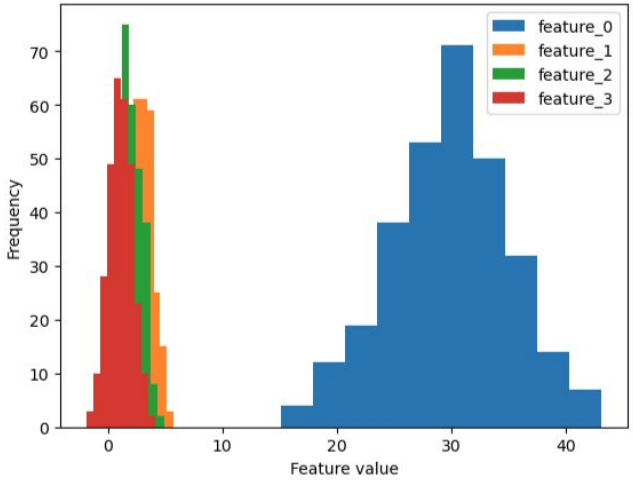


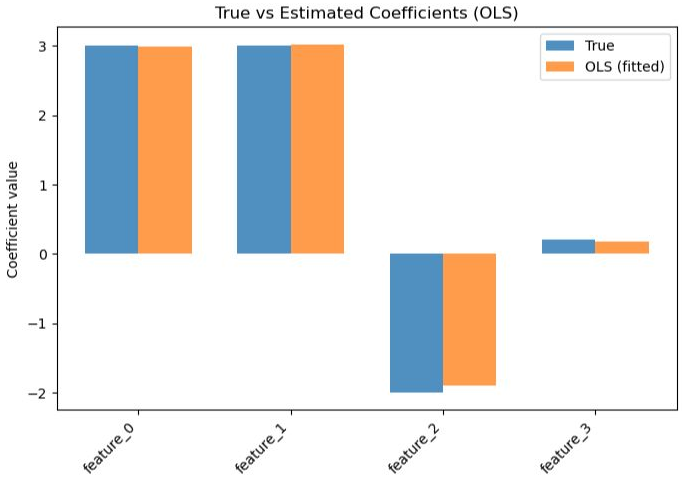


Interpreting coefficients - different scales

If the features have different scales, the relative coefficient sizes might not be very informative.

1) *Because changing x_0 in its distribution will have a greater effect on the model than the other features.*
2) *What can we do instead, to get feature importance estimates?*

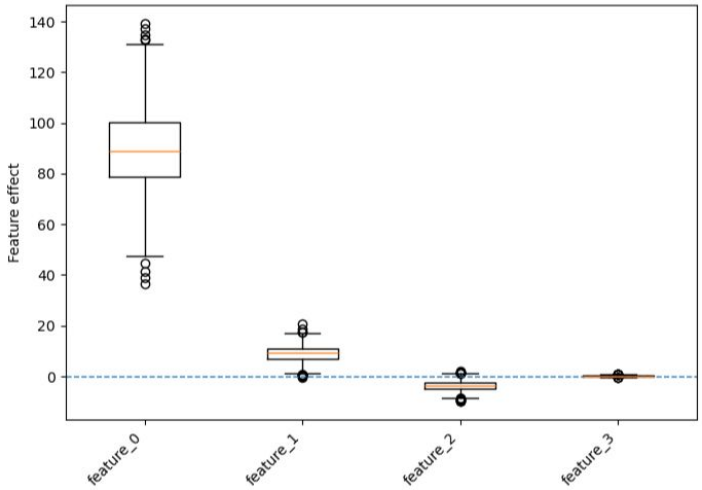


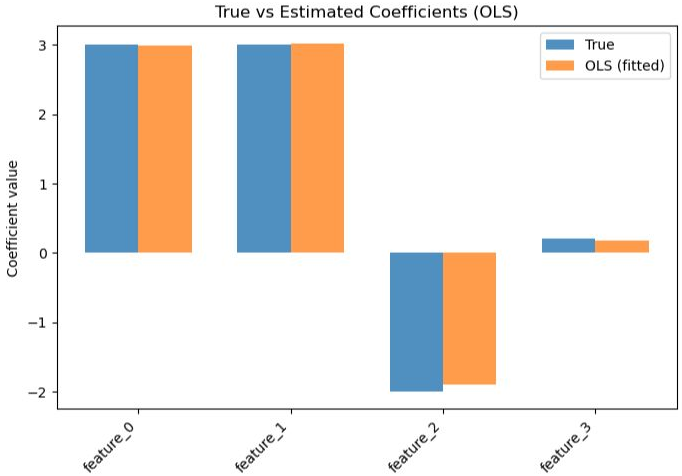


Interpreting coefficients - different scales

The $\text{effect}_j = x^{(i)}_j w_j$ is more informative, especially if the features have different scales

Here, we see that the *effect* of x_0 on the model is much greater than that of the other features.





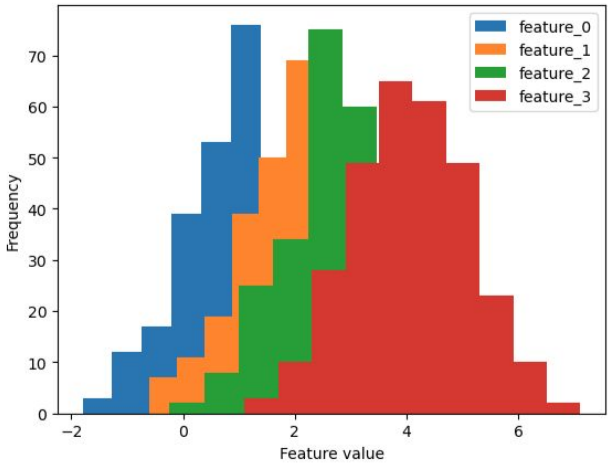

```
for i, feat in enumerate(feature_names):
    _effects = X_train[feat].to_numpy()*ols_coefs[i]
    effects[feat] = _effects
```

Interpreting coefficients - correlated features

Coefficients in multiple linear models represent the relationship between the given feature x_i and the target y, assuming that all the other features remain constant.

In the case of correlated features, this assumption often fails in practice.

Let's go back to features with the same scale, but where two are highly correlated

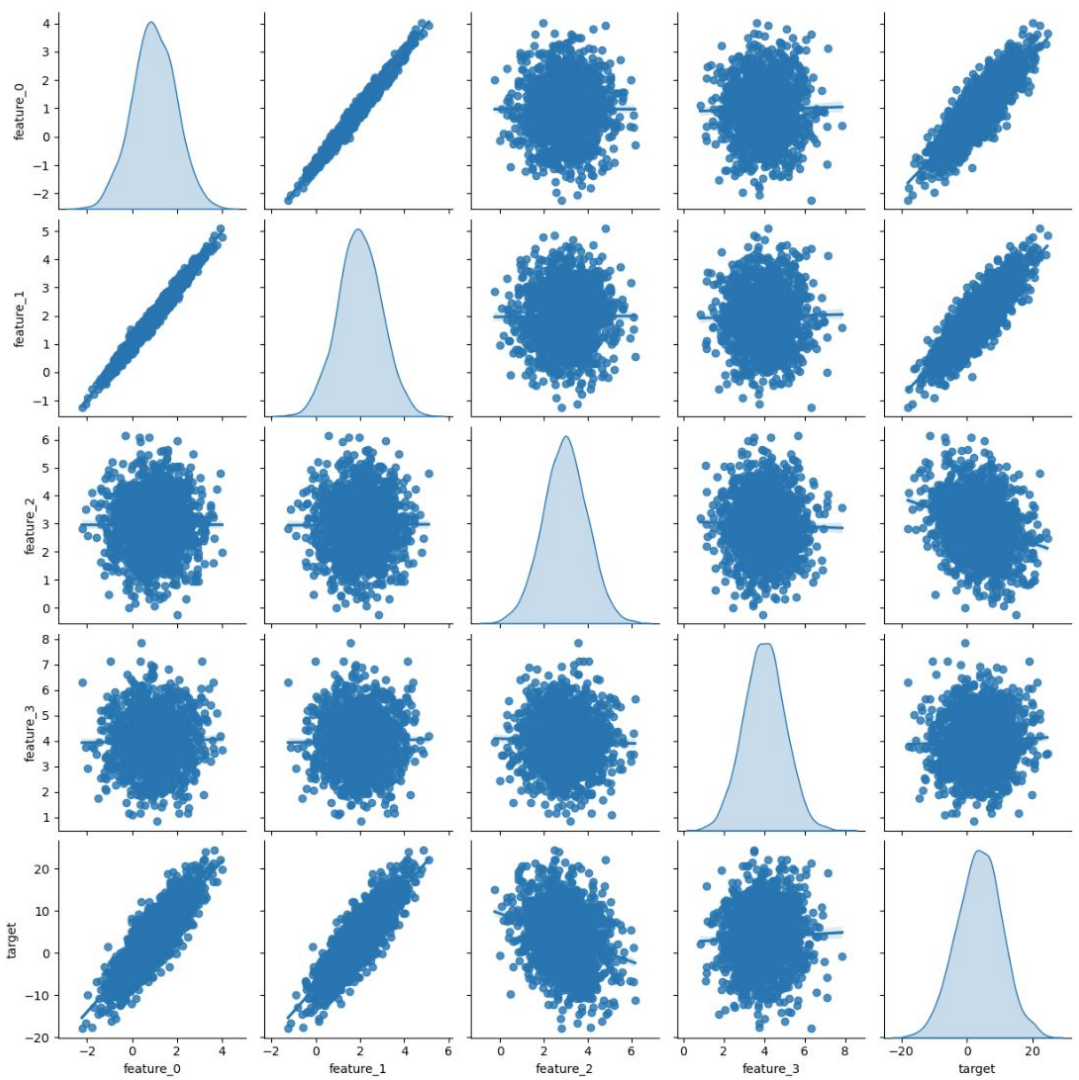


Correlated features

Still, $y = 3\,x_0 + 3\,x_1 - 2\,x_2 + 0.2\,x_3$.

Features have the same scale, but
two are highly correlated (which two? :))

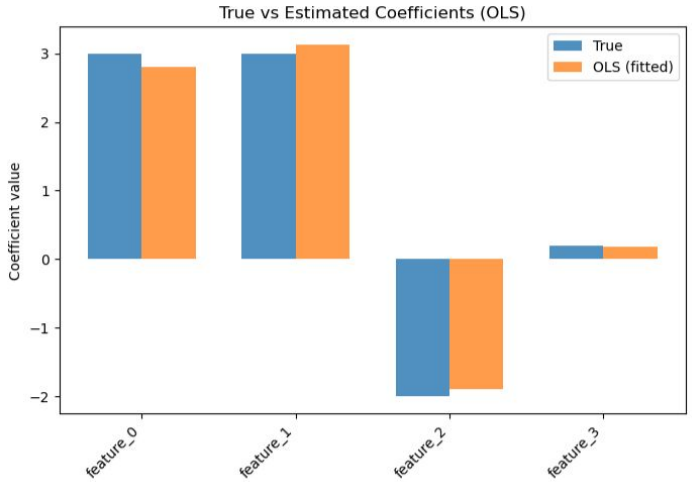
We fit a model and look at the coefficients.



Interpreting coefficients - correlated features

This looks nice, but we know that correlated features can confuse models since they represent each other's variability. What if we were just lucky this time around?

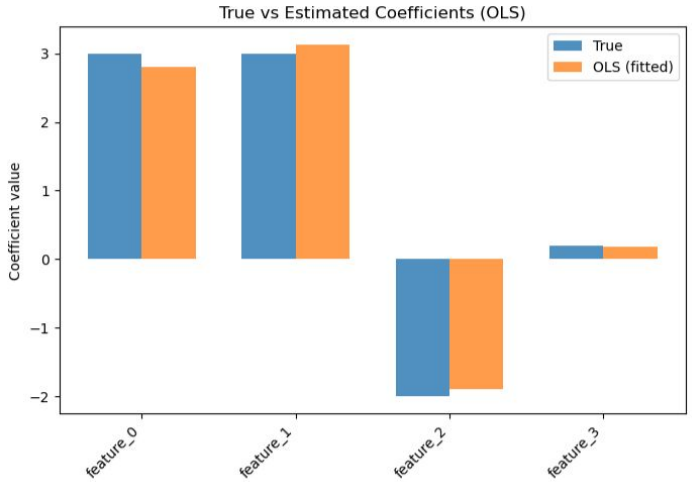
What can we do to find out how lucky we got, and how much we should expect the coefficient values to vary?



Interpreting coefficients - correlated features

This looks nice, but we know that correlated features can confuse models since they represent each other's variability. What if we were just lucky this time around?

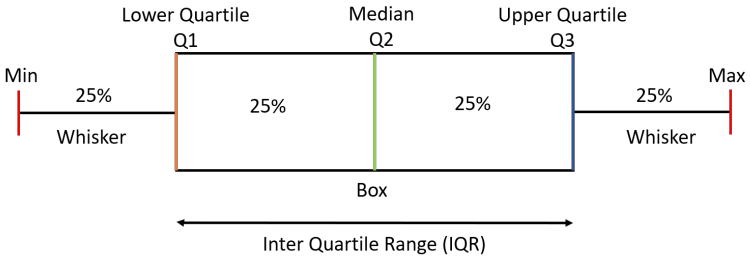
To get a sense of the variation, we can perform bootstrapping.



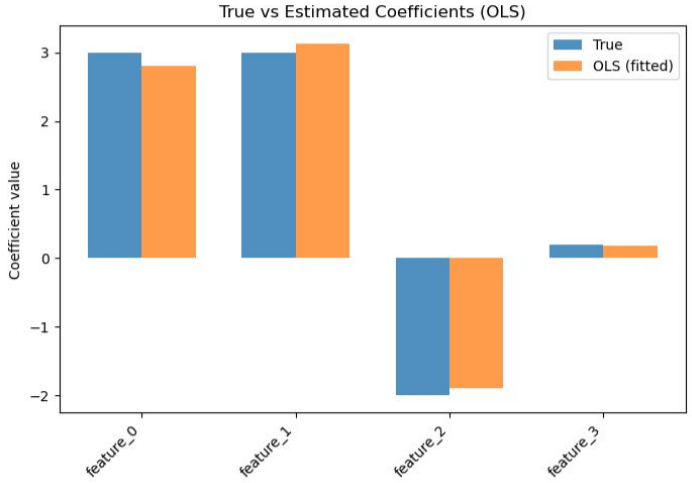
Correlated features

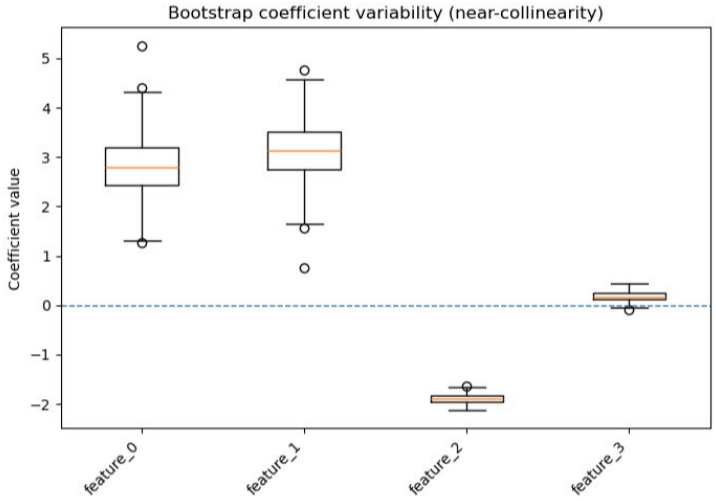
```
for each bootstrap iteration:
      draw a sample from the data
      fit a linear regression model
      store the coefficients for each feature

make an ax.boxplot for each feature
```

a boxplot in python by default has 50% of the data inside the box (Q1-Q3) and indicates the most extreme points.



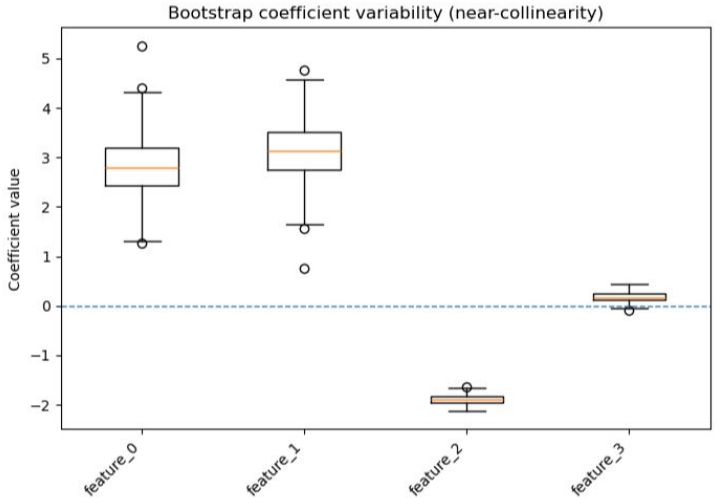
What does the box size tell us?





Interpreting coefficients - correlated features

```
for each bootstrap iteration:
      draw a sample from the data
      fit a linear regression model
      store the coefficients for each feature

make an ax.boxplot for each feature
a boxplot in python by default has 50% of the data inside
the box (Q1-Q3) and indicates the most extreme points.
```



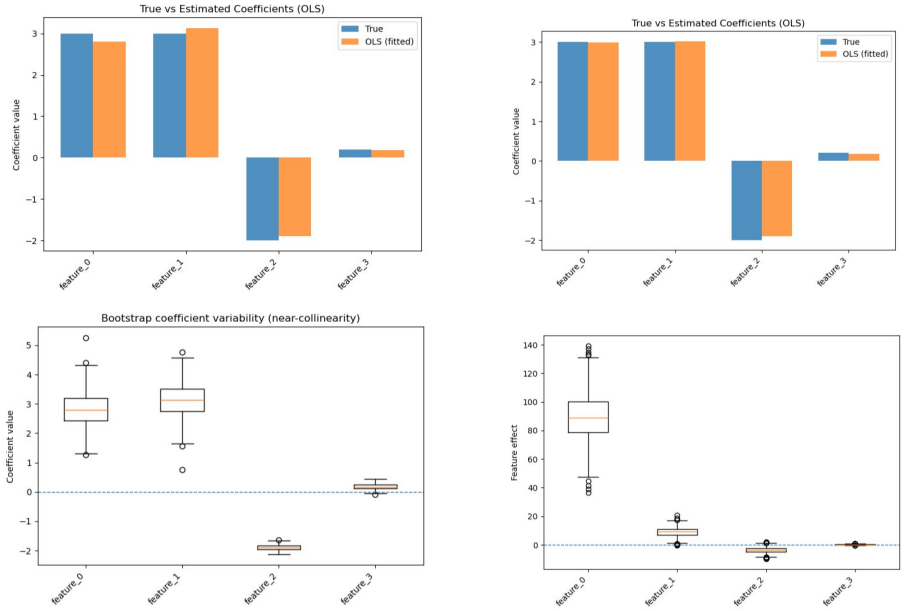
The correlation in the data makes the estimation of these coefficients uncertain.

There is no fix for this problem. Correlated features destabilise linear (and logistic, for that matter) regression models.

Interpreting coefficients - so far

We can look at the magnitude of a coefficient w_j, or the total $\text{effect}_j = x^{(i)}_j \, w_j$ of a coefficient and feature, to say how important model features are.

In XAI terminology, is this a **global** or a **local** explanation (or, interpretation)?



Decision trees

Decision trees

How to plant:

Give the entire training dataset to the root note.

Use some measure of information or entropy to find the best criterion on which to split the dataset, creating a new decision node.

Divide the dataset into two subsets based on the possible values of the selected criterion.

Repeat the above steps for all the subsets until all subsets' data belong to the same class, a max depth is reached or no further splitting is possible.

Interpreting decision trees

Starting from the root node, the decision nodes are selected based on some selection criterion (maximize information gain / minimize entropy / …).

Each node of the decision tree is an IF-sentence:
"if this feature is greater than some number…"

The nodes of the decision tree are connected by the AND-operation.
Starting at the root node, if a feature satisfies criterion 1 AND criterion 2 AND … until the leaf node.

It is fairly easy for humans to follow this process.

Or? Is it always?

Example: Bike demand

`Bake_Sharing_Demand` is a dataset often used for the regression task of predicting how many bikes will be used, based on features such as weather, day of the week, etc.

We train a decision tree regressor to predict the 'count' variable, indicating bike demand.

From here on and out:
I'll often use label encoding instead of one-hot encoding, to keep the data easier to discuss and interpret for us. For modelling purposes, one-hot encoding is usually more stable and thus preferable. The implications of our discussions stay the same.

```
bike = fetch_openml("Bike_Sharing_Demand", version=2, as_frame=True)
df = bike.frame
features = df.columns
df
```

	season	year	month	hour	holiday	weekday	workingday	weather	temp	feel_temp	humidity	windspeed	count
0	spring	0	1	0	False	6	False	clear	9.84	14.395	0.81	0.0000	16
1	spring	0	1	1	False	6	False	clear	9.02	13.635	0.80	0.0000	40
2	spring	0	1	2	False	6	False	clear	9.02	13.635	0.80	0.0000	32
3	spring	0	1	3	False	6	False	clear	9.84	14.395	0.75	0.0000	13
4	spring	0	1	4	False	6	False	clear	9.84	14.395	0.75	0.0000	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
17374	spring	1	12	19	False	1	True	misty	10.66	12.880	0.60	11.0014	119
17375	spring	1	12	20	False	1	True	misty	10.66	12.880	0.60	11.0014	89
17376	spring	1	12	21	False	1	True	clear	10.66	12.880	0.60	11.0014	90
17377	spring	1	12	22	False	1	True	clear	10.66	13.635	0.56	8.9981	61
17378	spring	1	12	23	False	1	True	clear	10.66	13.635	0.65	8.9981	49

17379 rows × 13 columns

```
df_enc, enc = encode_string_features(df)
df_enc
```

	season	year	month	hour	holiday	weekday	workingday	weather	temp	feel_temp	humidity	windspeed	count
0	1	0	1	0	0	6	0	0	9.84	14.395	0.81	0.0000	16
1	1	0	1	1	0	6	0	0	9.02	13.635	0.80	0.0000	40
2	1	0	1	2	0	6	0	0	9.02	13.635	0.80	0.0000	32
3	1	0	1	3	0	6	0	0	9.84	14.395	0.75	0.0000	13
4	1	0	1	4	0	6	0	0	9.84	14.395	0.75	0.0000	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...
17374	1	1	12	19	0	1	1	2	10.66	12.880	0.60	11.0014	119
17375	1	1	12	20	0	1	1	2	10.66	12.880	0.60	11.0014	89
17376	1	1	12	21	0	1	1	0	10.66	12.880	0.60	11.0014	90
17377	1	1	12	22	0	1	1	0	10.66	13.635	0.56	8.9981	61
17378	1	1	12	23	0	1	1	0	10.66	13.635	0.65	8.9981	49

17379 rows × 13 columns

```
for col, le in enc.items():
    mapping = {cls: idx for idx, cls in enumerate(le.classes_)}
    print(col, mapping)
```

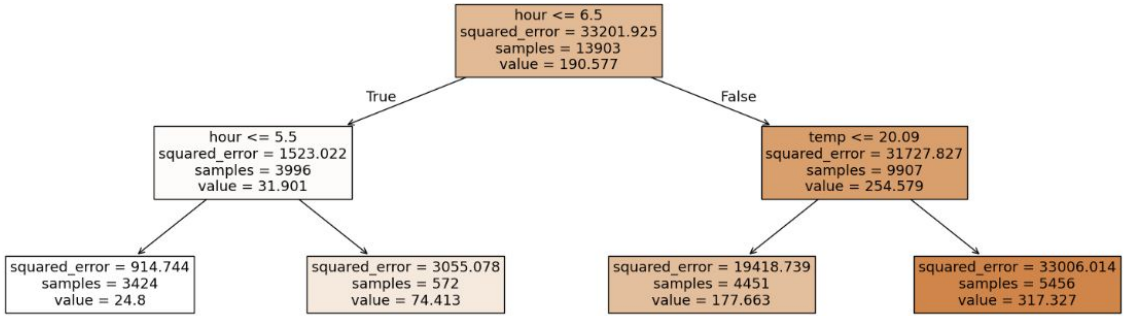
```
season {'fall': 0, 'spring': 1, 'summer': 2, 'winter': 3}
holiday {'False': 0, 'True': 1}
workingday {'False': 0, 'True': 1}
weather {'clear': 0, 'heavy_rain': 1, 'misty': 2, 'rain': 3}
```

Example: Bike demand

```
model = DecisionTreeRegressor(max_depth=2)
model.fit(X_train, y_train);
print("Nodes in decision tree:", model.tree_.node_count)
plt.figure(figsize=(20,6))
plot_tree(model, filled=True, feature_names=features, fontsize=13)
plt.savefig("tree_depth_2.pdf", format='pdf')
```

note max_depth=2

```
Nodes in decision tree: 7
```



The mapping:

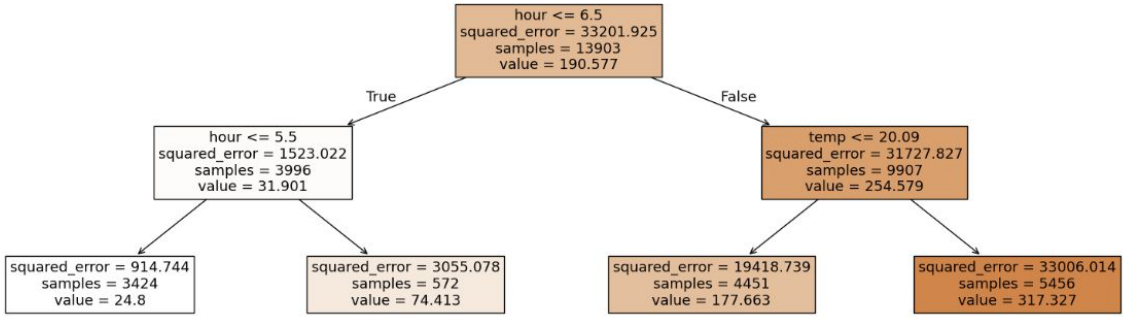
```
for col, le in enc.items():
    mapping = {cls: idx for idx, cls in enumerate(le.classes_)}
    print(col, mapping)
```

```
season {'fall': 0, 'spring': 1, 'summer': 2, 'winter': 3}
holiday {'False': 0, 'True': 1}
workingday {'False': 0, 'True': 1}
weather {'clear': 0, 'heavy_rain': 1, 'misty': 2, 'rain': 3}
```

Example: Bike demand

```
model = DecisionTreeRegressor(max_depth=2)
model.fit(X_train, y_train);
print("Nodes in decision tree:", model.tree_.node_count)
plt.figure(figsize=(20,6))
plot_tree(model, filled=True, feature_names=features, fontsize=13)
plt.savefig("tree_depth_2.pdf", format='pdf')
```

```
Nodes in decision tree: 7
```



How can we explain the prediction for this data point?

```
X_test.sample(random_state=2)
```

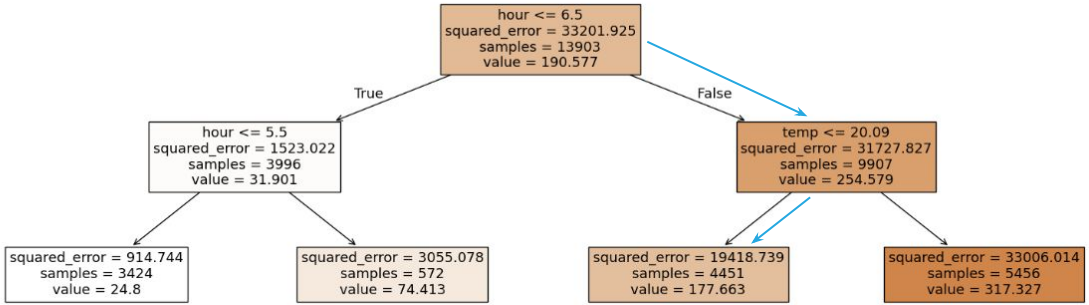
	season	year	month	hour	holiday	weekday	workingday	weather	temp	feel_temp	humidity	windspeed
8201	3	0	12	9	0	2	1	0	10.66	12.88	0.6	11.0014

The mapping:

```
for col, le in enc.items():
    mapping = {cls: idx for idx, cls in enumerate(le.classes_)}
    print(col, mapping)
```

```
season {'fall': 0, 'spring': 1, 'summer': 2, 'winter': 3}
holiday {'False': 0, 'True': 1}
workingday {'False': 0, 'True': 1}
weather {'clear': 0, 'heavy_rain': 1, 'misty': 2, 'rain': 3}
```

Interpreting a decision tree prediction




```
X_test.sample(random_state=2)
```

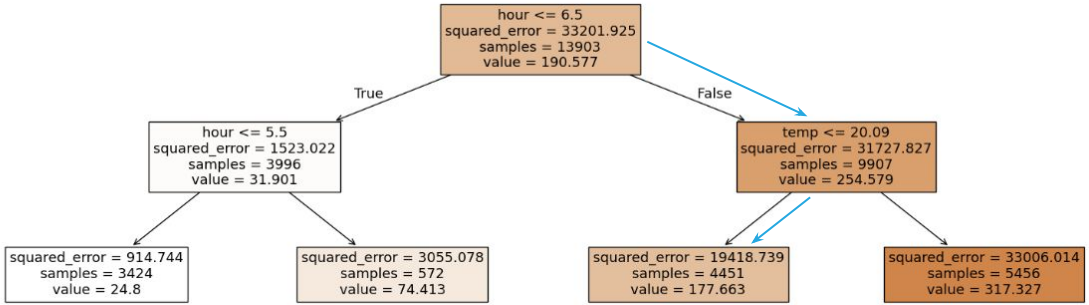
	season	year	month	hour	holiday	weekday	workingday	weather	temp	feel_temp	humidity	windspeed
8201	3	0	12	9	0	2	1	0	10.66	12.88	0.6	11.0014

```
model.predict(X_test.loc[[8201]]).item()
```

```
177.6634464165356
```

How does the decision tree model arrive at the prediction of 177.663?

Interpreting a decision tree prediction




```
X_test.sample(random_state=2)
```

	season	year	month	hour	holiday	weekday	workingday	weather	temp	feel_temp	humidity	windspeed
8201	3	0	12	9	0	2	1	0	10.66	12.88	0.6	11.0014

```
model.predict(X_test.loc[[8201]]).item()
```

```
177.6634464165356
```

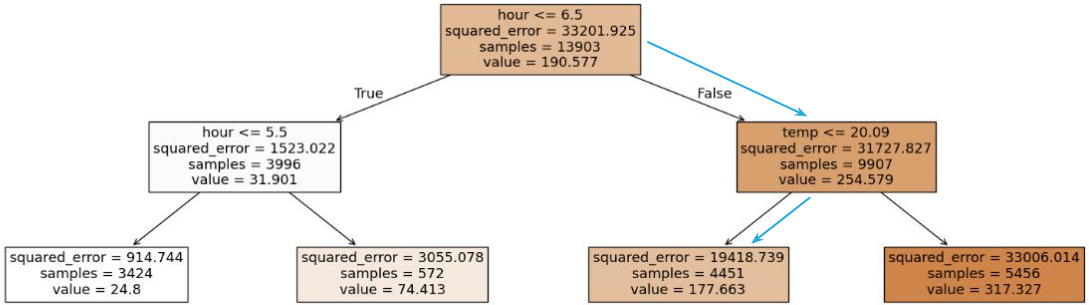
The prediction follows from:

hour <= 6.5 False
temp <= 20.09 True

⇒ value (prediction) = 177.663

Is this an explanation?

Interpreting a decision tree prediction




```
X_test.sample(random_state=2)
```

	season	year	month	hour	holiday	weekday	workingday	weather	temp	feel_temp	humidity	windspeed
8201	3	0	12	9	0	2	1	0	10.66	12.88	0.6	11.0014

```
model.predict(X_test.loc[[8201]]).item()
```

```
177.6634464165356
```

The prediction follows from:

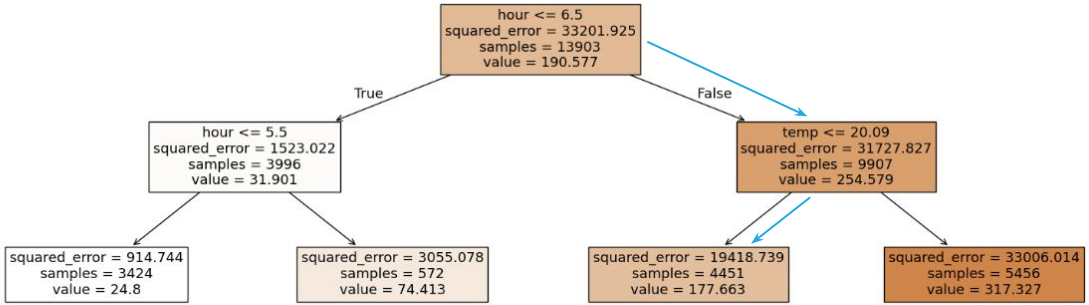
hour <= 6.5 False
temp <= 20.09 True

⇒ value (prediction) = 177.663

Is this an explanation?

In XAI terminology, is it local or global?

Interpreting a decision tree prediction



We can explain/interpret single predictions by following the decision path to the predicted value.

In XAI terminology, this is a *local* explanation.

What would a global explanation look like? Any suggestions?

Global explanations of decision trees

```
df = pd.read_csv('../data/diabetes_prediction_dataset.csv')

mapping = {"Female": 0, "Male": 1}
df["gender"] = df["gender"].map(mapping)
print("gender:", mapping)
le = LabelEncoder()
df["smoking_history"] = le.fit_transform(df["smoking_history"])
print("smoking history:", [list(enumerate(le.classes_))])

X = df.drop(columns=["diabetes"])
features = X.columns
y = df["diabetes"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
X_train
```

```
gender: {'Female': 0, 'Male': 1}
smoking history: [[(0, 'No Info'), (1, 'current'), (2, 'ever'), (3, 'former'), (4, 'never'), (5, 'not current')]]
```

	gender	age	hypertension	heart_disease	smoking_history	bmi	HbA1c_level	blood_glucose_level
75220	1.0	73.0	0	0	3	24.77	3.5	80
48955	1.0	80.0	0	0	4	24.60	5.7	145
44966	0.0	38.0	0	0	2	24.33	4.0	158
13568	0.0	26.0	0	0	5	18.87	5.0	100
92727	0.0	61.0	1	0	1	22.11	4.5	85
...	...	...	...	...	...	...	...	...
6265	1.0	49.0	0	0	4	32.98	5.7	80
54886	0.0	15.0	0	0	4	28.10	5.0	159
76820	1.0	42.0	0	0	4	26.14	5.8	85
860	0.0	37.0	0	0	4	24.96	6.2	158
15795	0.0	23.0	0	0	4	27.99	5.0	159

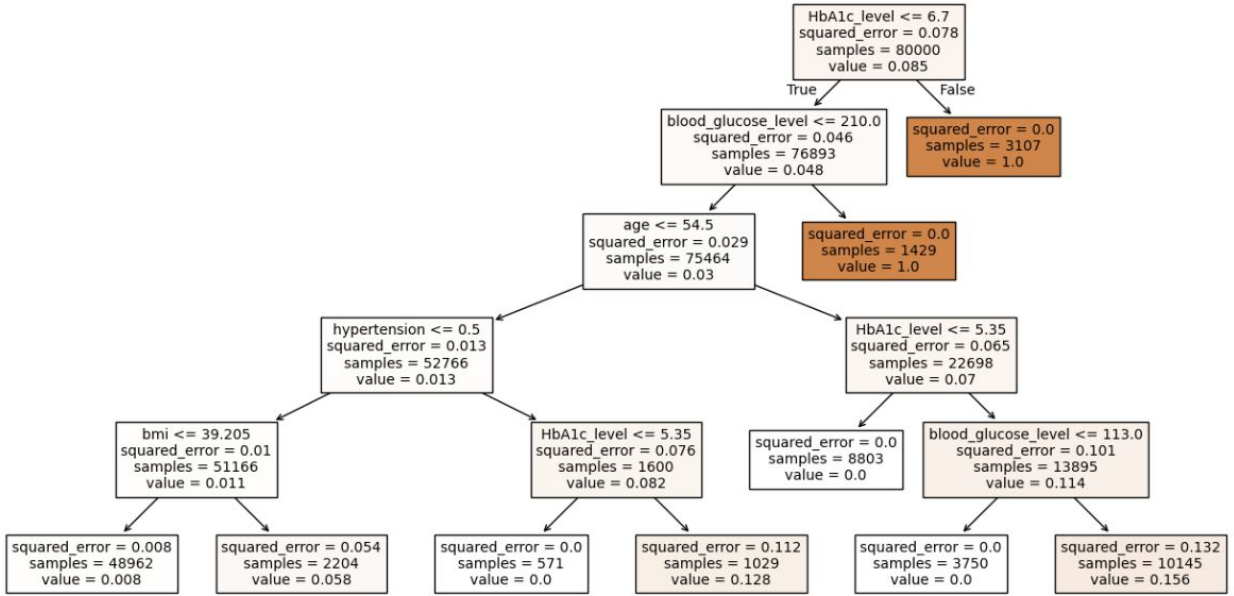
Let's use a decision tree classifier for this example.

This is the well-known `diabetes` dataset, where the aim is to predict whether somebody has diabetes based on gender, age, medical history and various blood test results.

Feature importance for decision trees

```
model = DecisionTreeRegressor(max_depth=5)
model.fit(X_train, y_train)
print("Model accuracy:", accuracy(y_test, model.predict(X_test).round()))
model.tree_.node_count
plt.figure(figsize=(16,8))
plot_tree(model, filled=True, feature_names=features, fontsize=10)
plt.show()
```

```
Model accuracy: 0.97215
```



This decision tree classifier achieves an accuracy of 97,2%

What is the most important feature?

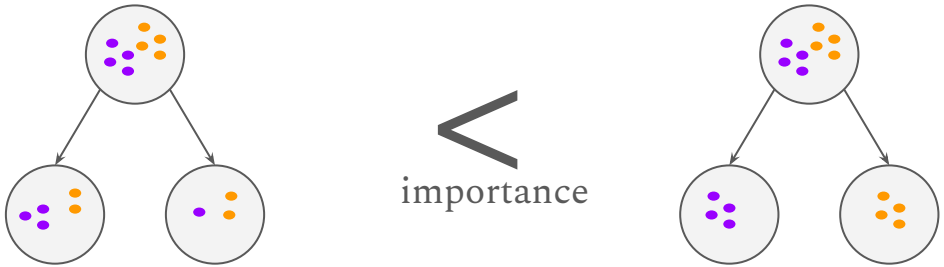
Why?

Feature importance for decision trees

The importances of the features in a tree are calculated as the normalized total reduction of [loss criterion] by feature.

Features that are split on earlier are not *necessarily* more important.

Intuitively: a feature's importance depends on how much splitting on it reduces the impurity of the data - in total, every time it is used in the tree.



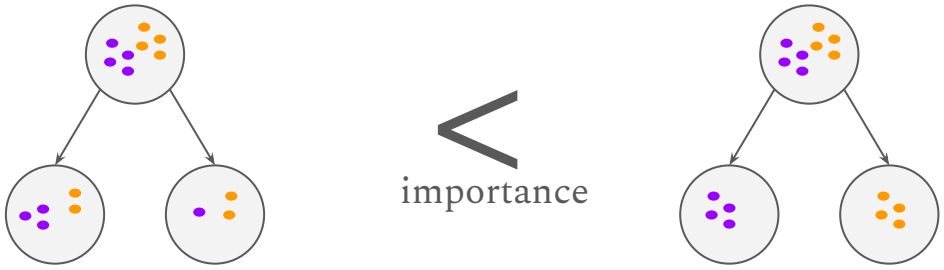
Feature importance for decision trees

Gini impurity: A metric quantifying the impurity of the data (entering a node in a decision tree).

We can calculate the Gini importance by summing the reductions in Gini impurity achieved by a feature across all splits where it is used.

$$\text{Importance}(f) = \sum_{\text{splits on } f} (\text{Gini}_{\text{parent}} - \text{Gini}_{\text{children}})$$

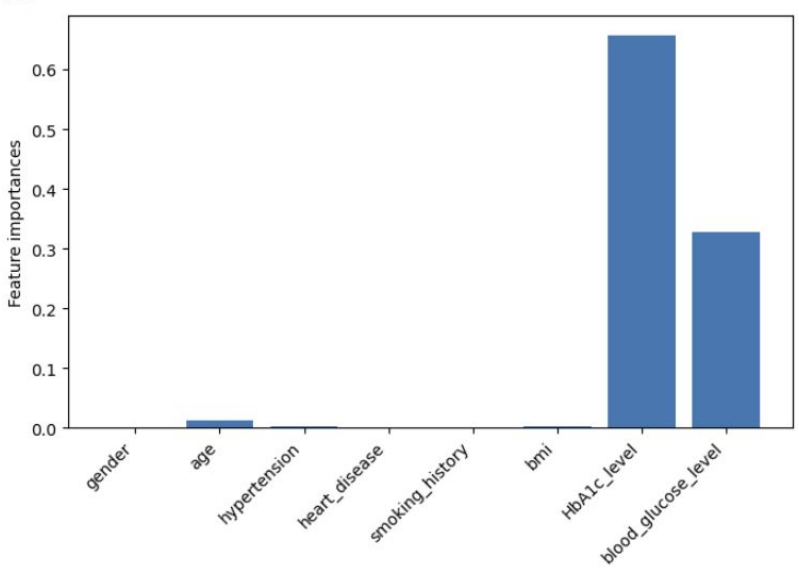
Simply put: The Gini impurity measures how mixed the classes are. A feature that sorts the data better earns more importance.



Feature importance for decision trees

```
importances = model.feature_importances_
print(sum(importances))

x_pos = np.arange(len(features))
fig, ax = plt.subplots(figsize=(7,5))
ax.bar(x_pos, importances)
ax.set_xticks(x_pos)
ax.set_xticklabels(features, rotation=45, ha="right")
ax.set_ylabel("Feature importances")
plt.tight_layout()
plt.show()
```

```
1.0
```



The sum can be implemented using a `for`-loop - or we can use sklearn's built-in `feature_importances_` to get a list of normalized importances.

Based on this result: how small do you think we can make the tree without decreasing the accuracy significantly?

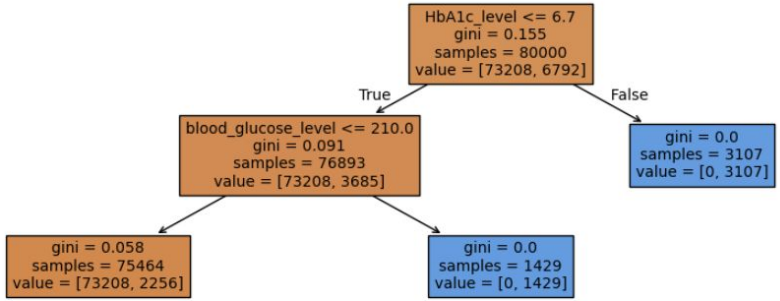
Feature importance for decision trees

Using only two features (depth=2), this decision tree classifier achieves an accuracy of 97,2%!

This makes sense, as HbA1C and blood glucose levels are used in actual tests for diabetes.*

```python
def accuracy(true_values, predictions):
    return np.mean(true_values == predictions)
```

```python
model = DecisionTreeClassifier(max_depth=2)
model.fit(X_train, y_train)
model.tree_.node_count
plt.figure(figsize=(12,4))
plot_tree(model, filled=True, feature_names=features, fontsize=10)
plt.show()
print("Model accuracy:", accuracy(y_test, model.predict(X_test).round()))
```




```
Model accuracy: 0.97215
```

*the split value for HbA1C that the decision tree chooses (6.7) is close to the conventionally accepted threshold of HbA1C levels used to detect diabetes (6.5).

Feature importance for decision trees

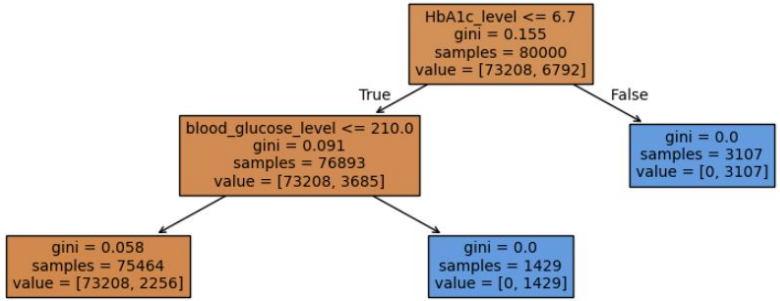
Using only two features (depth=2), this decision tree classifier achieves an accuracy of 97,2%!

The most important feature is HbA1C.

How can we use the Gini impurity to generate this explanation?

```python
def accuracy(true_values, predictions):
    return np.mean(true_values == predictions)
```

```python
model = DecisionTreeClassifier(max_depth=2)
model.fit(X_train, y_train)
model.tree_.node_count
plt.figure(figsize=(12,4))
plot_tree(model, filled=True, feature_names=features, fontsize=10)
plt.show()
print("Model accuracy:", accuracy(y_test, model.predict(X_test).round()))
```




```
Model accuracy: 0.97215
```

Feature importance for decision trees

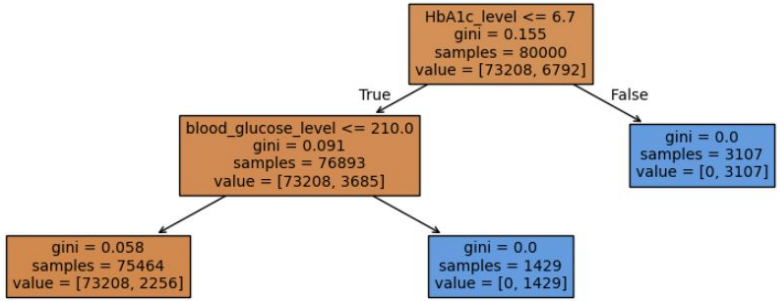
Using only two features (depth=2), this decision tree classifier achieves an accuracy of 97,2%!

The most important feature is HbA1C, because it reduces the Gini impurity by 0.064 (=0.155–0.091).

The second most important feature, blood_glucose_level, reduces the Gini impurity by 0.033 (=0.091–0.058).

```python
def accuracy(true_values, predictions):
    return np.mean(true_values == predictions)
```

```python
model = DecisionTreeClassifier(max_depth=2)
model.fit(X_train, y_train)
model.tree_.node_count
plt.figure(figsize=(12,4))
plot_tree(model, filled=True, feature_names=features, fontsize=10)
plt.show()
print("Model accuracy:", accuracy(y_test, model.predict(X_test).round()))
```




```
Model accuracy: 0.97215
```

How large trees can we interpret?

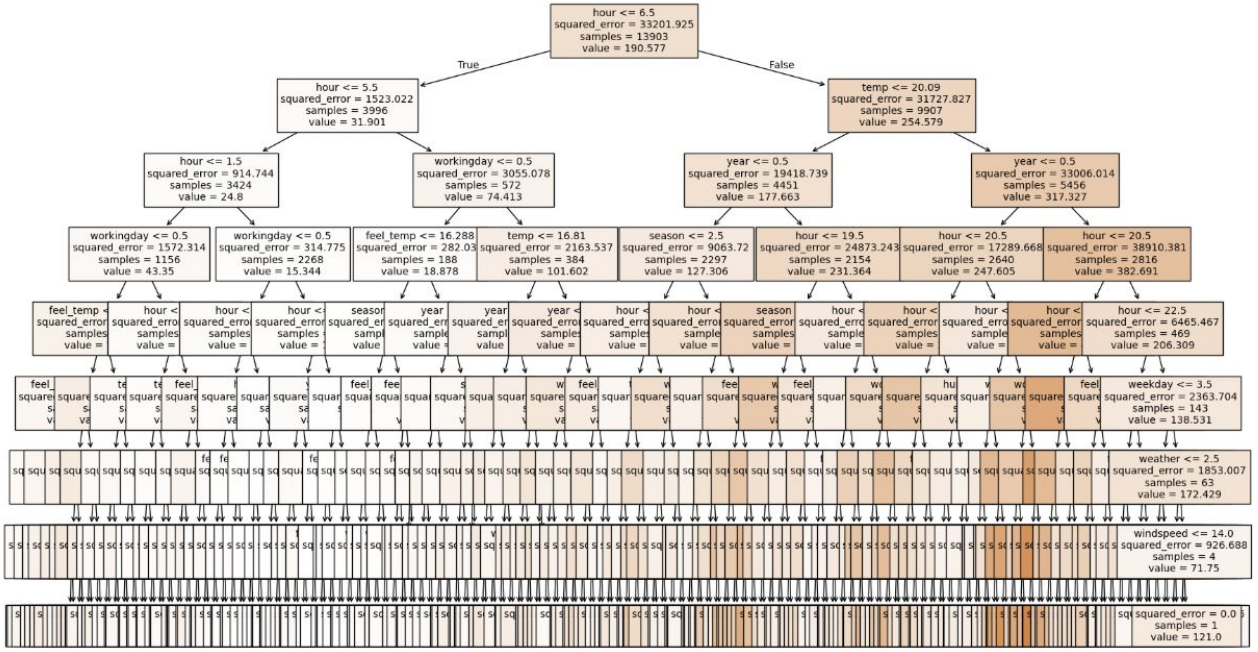
The local explanation/interpretation we gave earlier for a single data point involved manually traversing the tree.

We used a tree with depth=2. The following is a tree with depth=8.

How large trees can we interpret?

```
model = DecisionTreeRegressor(max_depth=8)
model.fit(X_train, y_train);
print("Nodes in decision tree:", model.tree_.node_count)
plt.figure(figsize=(20,12))
plot_tree(model, filled=True, feature_names=features, fontsize=10)
plt.show()
```

```
Nodes in decision tree: 505
```



max_depth=*n* does *not* mean *n* splits: It means the longest path from root to leaf is *n* nodes:

depth 0 → root
depth 1
depth 2
…
depth n → max decision depth

Has the interpretability decreased as we've gone from depth 2 to depth 8?

What do you think?

How large trees can we interpret?

If we let sklearn decide how deep the tree should be for achieving the most accurate predictions, we get

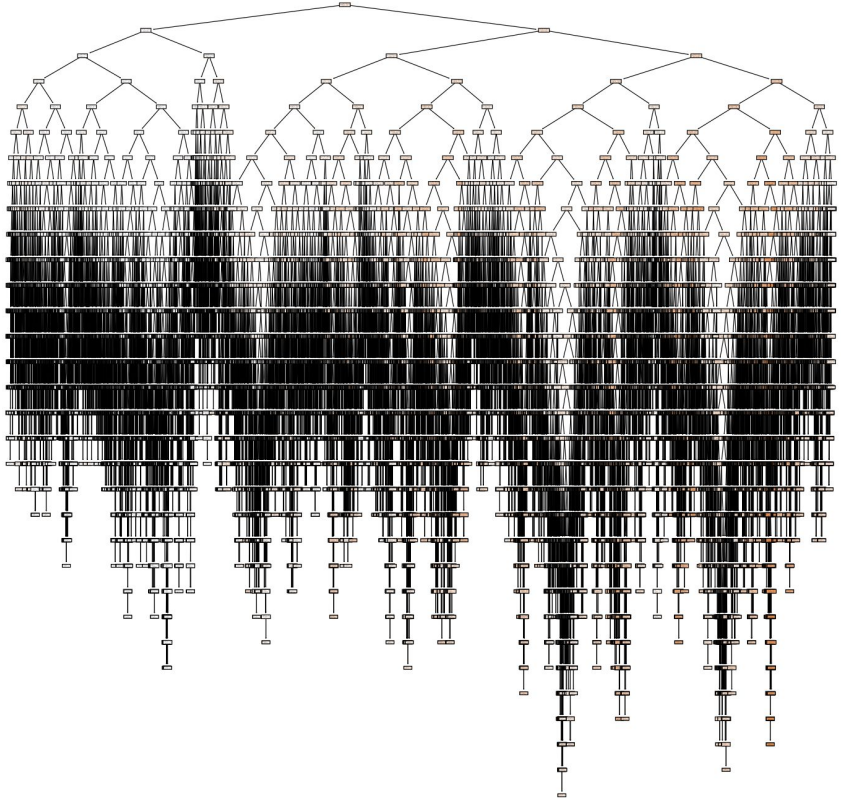
```
model = DecisionTreeRegressor()
model.fit(X_train, y_train);
```

```
model.tree_.node_count
```

```
26377
```

26,377 nodes (± some variability due to random data split) for the bike demand data.

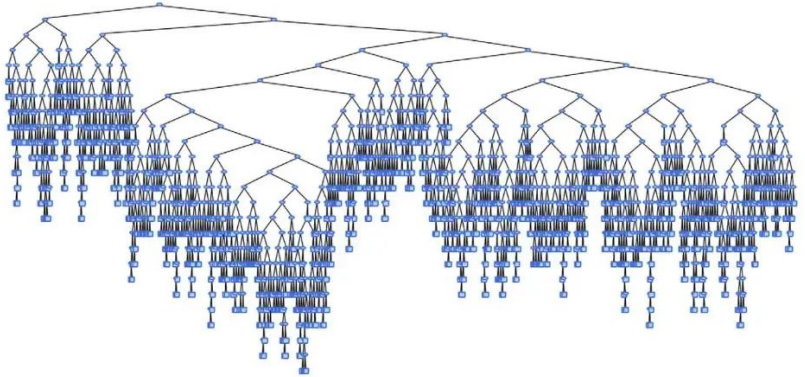
How long do you think it would take to tell this story? :)



Decomposability

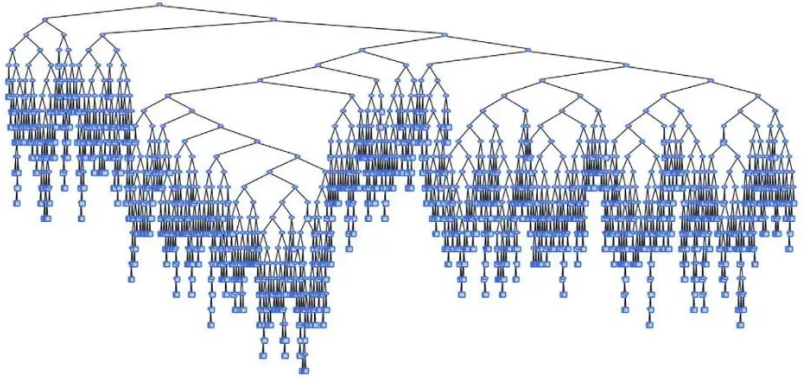
Are trees *always* interpretable?

*When are decision trees **interpretable**?*



[1] The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery, DOI:10.1145/3236386.3241340

Are trees *always* interpretable?



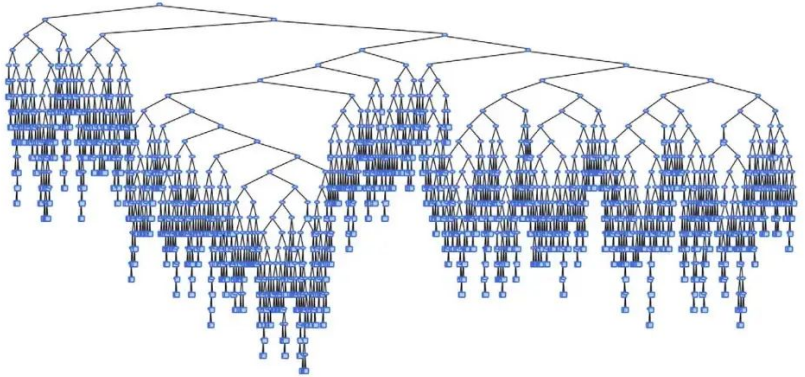
Short answer: Decision trees are interpretable,
provided they're not too large.

Lipton (2018) [1] suggests the following levels of interpretability (although they use the term "transparency"):

- **Simulatable** ← so simple that a human can simulate the outcome given an input
- **Decomposable** ← humans can keep track of the data throughout (it isn't transformed)
- **Algorithmic** ← it is possible to evaluate the method mathematically

[1] The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery, DOI:10.1145/3236386.3241340

Are trees *always* interpretable?



Short answer: Decision trees are interpretable, provided they're not too large.

Lipton [1] suggests the following levels of interpretability (although they use the term "transparency"):

- **Simulatable** ← so simple that a human can simulate the outcome given an input
- **Decomposable** ← humans can keep track of the data throughout (it isn't transformed)
- **Algorithmic** ← it is possible to evaluate the method mathematically

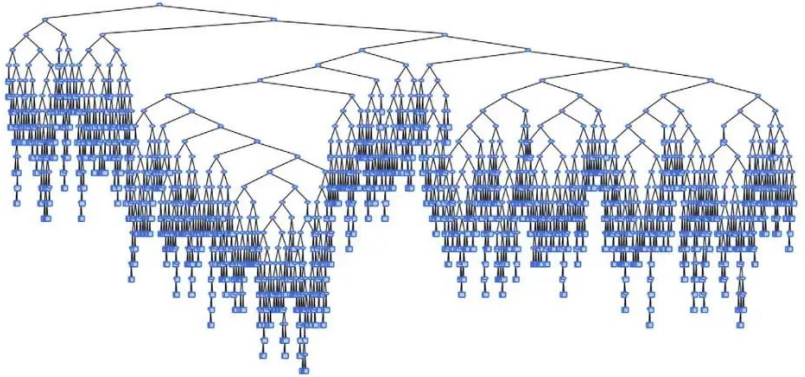
Which of these holds for small decision trees?

Which of these holds for very large decision trees?

What about neural networks?

[1] The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery, DOI:10.1145/3236386.3241340

Are trees *always* interpretable?



Short answer: Decision trees are interpretable, provided they're not too large.

Lipton [1] suggests the following levels of interpretability (although they use the term "transparency"):

- **Simulatable** ← so simple that a human can simulate the outcome given an input
- **Decomposable** ← humans can keep track of the data throughout (it isn't transformed)
- **Algorithmic** ← it is possible to evaluate the method mathematically

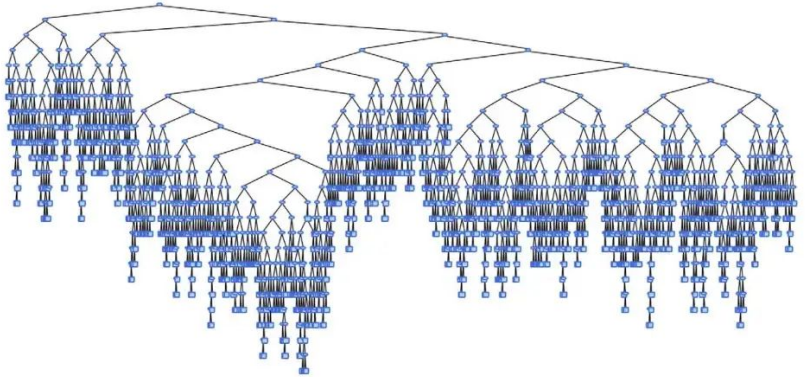
Small decision trees small trees are simulatable (directly and fully interpretable)

Which of these holds for very large decision trees?

What about neural networks?

[1] The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery, DOI:10.1145/3236386.3241340

Are trees *always* interpretable?



Short answer: Decision trees are interpretable, provided they're not too large.

Lipton [1] suggests the following levels of interpretability (although they use the term "transparency"):

- **Simulatable** ← so simple that a human can simulate the outcome given an input
- **Decomposable** ← humans can keep track of the data throughout (it isn't transformed)
- **Algorithmic** ← it is possible to evaluate the method mathematically

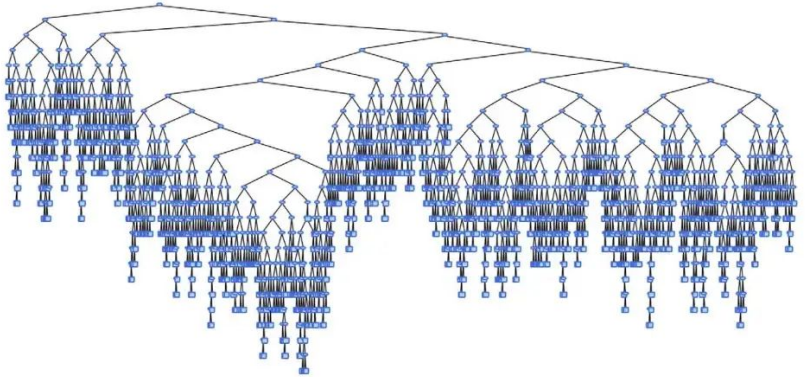
Small decision trees small trees are simulatable (directly and fully interpretable)

Large decision trees are decomposable (only piecewise interpretable)

What about neural networks?

[1] The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery, DOI:10.1145/3236386.3241340

Are trees *always* interpretable?



Short answer: Decision trees are interpretable, provided they're not too large.

Lipton [1] suggests the following levels of interpretability (although they use the term "transparency"):

- **Simulatable** ← so simple that a human can simulate the outcome given an input
- **Decomposable** ← humans can keep track of the data throughout (it isn't transformed)
- **Algorithmic** ← it is possible to evaluate the method mathematically

Small decision trees small trees are simulatable (directly and fully interpretable)

Large decision trees are decomposable (only piecewise interpretable)

Neural networks are only algorithmic interpretable (they transform the data, so even a single operation isn't necessarily interpretable to humans)

[1] The Mythos of Model Interpretability: In machine learning, the concept of interpretability is both important and slippery, DOI:10.1145/3236386.3241340

Interpretability of decision trees

A decision tree is locally interpretable (a human can follow the prediction) if

1. the tree is sufficiently small, and
2. the split criteria are interpretable.

For (global) feature importance, a calculation is necessary: it is not necessarily the case that a feature is more important because it is splitted on in an early node!

Global surrogate models

How to use model interpretations in an XAI setting

Given a machine learning model f we cannot interpret (a neural network, boosted forest, huge decision tree, ...), we can train an interpretable model g to mimic the behaviour of f, and use the interpretation of g as an explanation of f. We say that g is a surrogate model for explaining the full model f.

How to use model interpretations in an XAI setting

Given a machine learning model f we cannot interpret (a neural network, boosted forest, huge decision tree, ...), we can train an interpretable model g to mimic the behaviour of f, and use the interpretation of g as an explanation of f. We say that g is a surrogate model for explaining the full model f.

Step 1: Train a full model f on data x with target y, to predict $f(x) = \hat{y}$

Step 2: Train an interpretable model g on the same data x. What to use as targets?

How to use model interpretations in an XAI setting

Given a machine learning model f we cannot interpret (a neural network, boosted forest, huge decision tree, ...), we can train an interpretable model g to mimic the behaviour of f, and use the interpretation of g as an explanation of f. We say that g is a surrogate model for explaining the full model f.

Step 1: Train a full model f on data x with target y, to predict $f(x) = \hat{y}$

Step 2: Train an interpretable model g on the same data x, with the predictions $\hat{y}$ as targets. The aim is for g to approximate f.

How to use model interpretations in an XAI setting

Given a machine learning model f we cannot interpret (a neural network, boosted forest, huge decision tree, …), we can train an interpretable model g to mimic the behaviour of f, and use the interpretation of g as a replacement for the (missing) explanation of f.

Step 1: Train a full model f on data x with target y, to predict $f(x) = \hat{y}$

Step 2: Train an interpretable model g on the same data x, with the predictions $\hat{y}$ as targets. The aim is for g to approximate f.

Step 3: Explain the behaviour of the full model f by interpreting the surrogate model g.

Explanations from interpreting surrogate models

Where do we place this in the taxonomy?

Post-hoc?

Model-agnostic or model-specific?

Local or global?

Post-hoc methods	Model-agnostic	Model-specific
Local		
Global		

Explanations from interpreting surrogate models

Where do we place this in the taxonomy?

Post-hoc: *We got a model and want to explain its behaviour.*

Model-agnostic or model-specific?

Local or global?

Post-hoc methods	Model-agnostic	Model-specific
Local		
Global		

Explanations from interpreting surrogate models

Where do we place this in the taxonomy?

Post-hoc: *We got a model and want to explain its behaviour.*

Model-agnostic: *We use only the model's prediction to train a surrogate model.*

Local or global?

Post-hoc methods	Model-agnostic	Model-specific
Local		
Global		

Explanations from interpreting surrogate models

Where do we place this in the taxonomy?

Post-hoc: *We got a model and want to explain its behaviour.*

Model-agnostic: *We use only the model's prediction to train a surrogate model.*

Global and local, depending on use: *We train the interpretable surrogate model on the whole dataset with all predictions from the full model as targets. Still, we can interpret the surrogate model locally, for single data points.*

Post-hoc methods	Model-agnostic	Model-specific
Local		
Global		

Explanations from interpreting surrogate models

Where do we place this in the taxonomy?

Post-hoc: *We got a model and want to explain its behaviour*

Model-agnostic: *We use only the model's prediction to train a surrogate model*

Global and local, depending on use: *We train the interpretable surrogate model on the whole dataset with all predictions from the full model as targets. Still, we can interpret the surrogate model locally, for single data points.*

Post-hoc methods	Model-agnostic	Model-specific
Local	*Global surrogate models*	
Global	*Global surrogate models*	

Global surrogate models

Important: We use the full model *f* for prediction and the surrogate model *g* for explanation - otherwise, we just replace the full model and use an interpretable model instead.

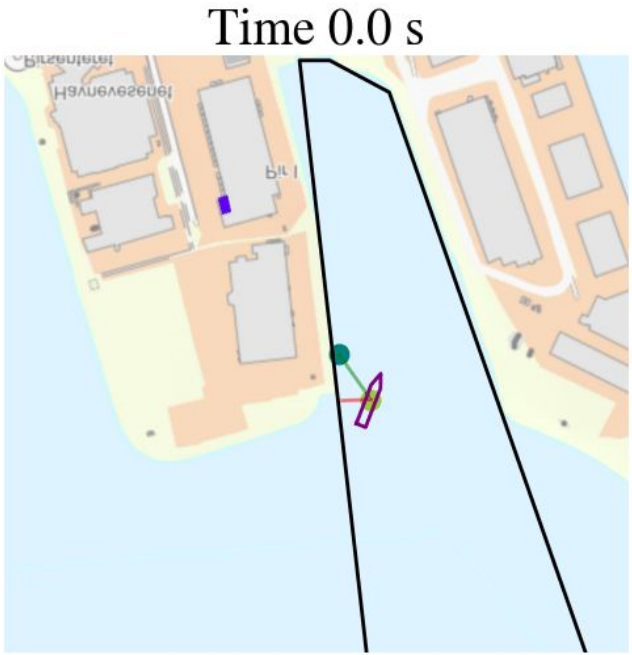
If an interpretable model can be made to suit your needs, always use that!

Real-world example

Example from a real-time application

We have a robotics application, in which a neural network is used to steer an autonomous surface vehicle (a ship).

The captain is responsible for approving the predictions from the neural network model, and needs explanations to know what the predictions are based on.

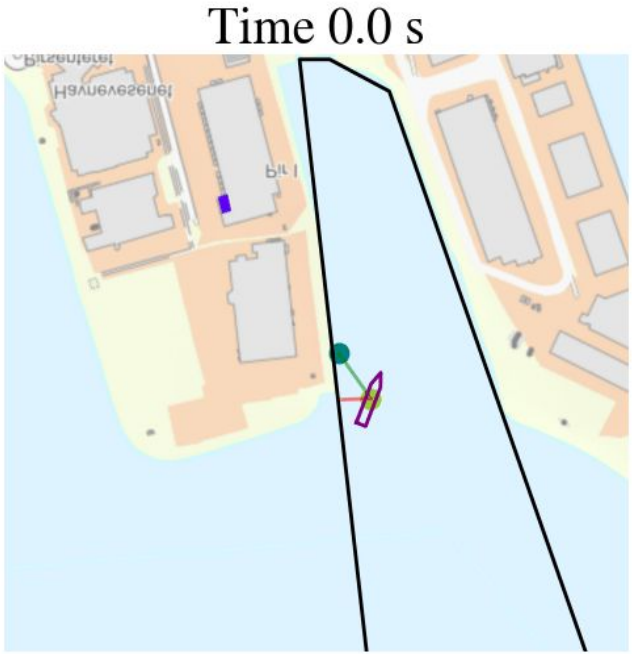


Example from a real-time application

We have a robotics application, in which a neural network is used to steer an autonomous surface vehicle (a ship).

The captain is responsible for approving the predictions from the neural network model, and needs explanations to know what the predictions are based on.

Most XAI methods take a while to generate explanations, but if explanations are needed real-time, they should take ~no longer to generate than the model takes to predict.

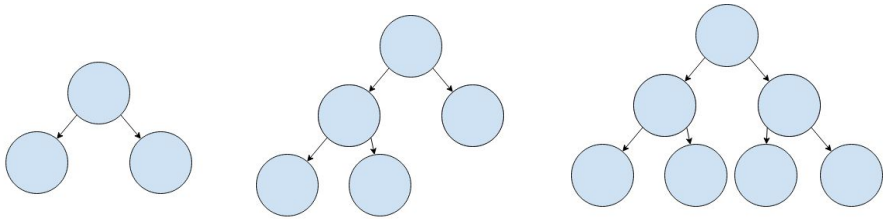


Example from a real-time application

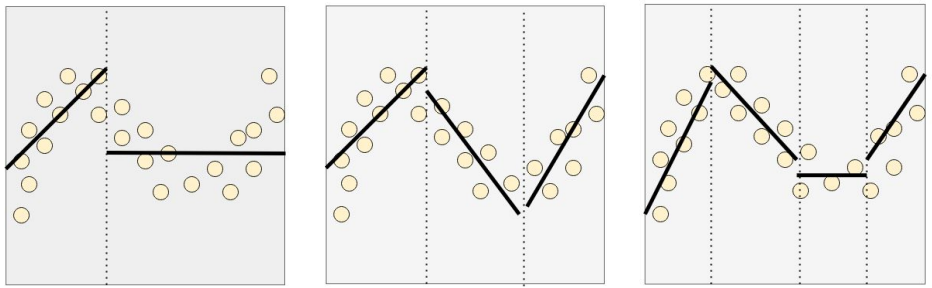
If explanations are needed real-time, they should take ~no longer to generate than the model takes to predict.

A global surrogate model can be trained in advance, and be ready when the full model is used.

Surrogate model: model tree (a linear function in each decision node)



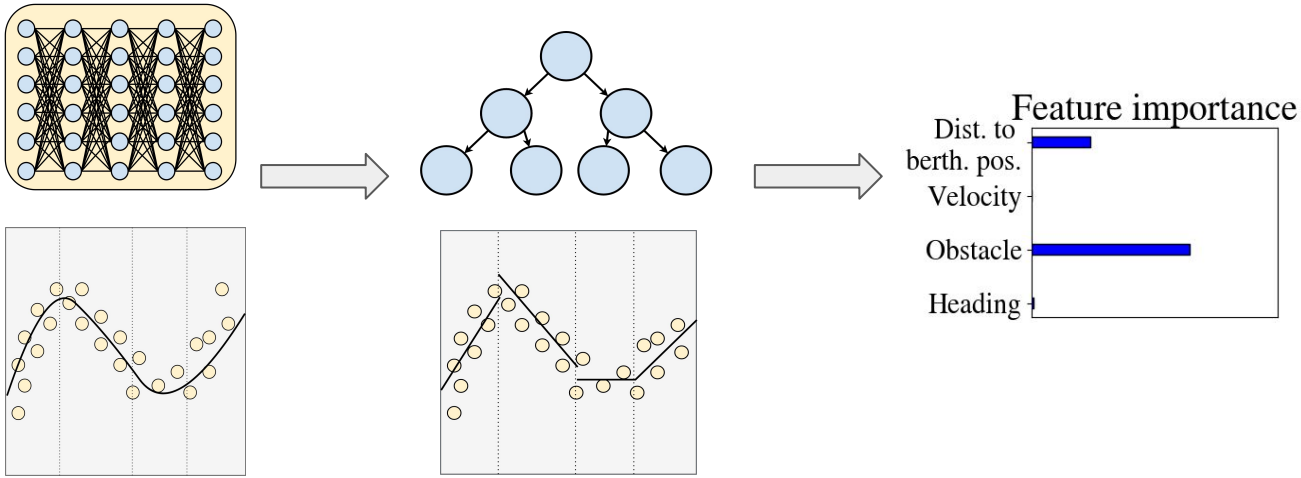
Piecewise linear function approximation of the full model's predictions

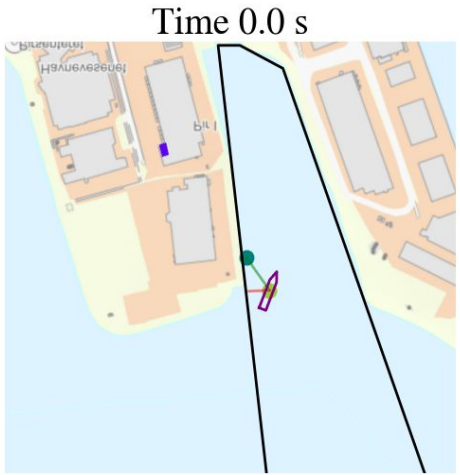


Example from a real-time application

Linear Model Tree = decision tree with a linear model in each root node.

The full model is used to steer the vehicle, and the surrogate model is used to generate explanations for each prediction.





Explaining a Deep Reinforcement Learning Docking Agent Using Linear Model Trees with User Adapted Visualization, DOI:10.1016/j.neucom.2022.10.014

Global surrogate models - benefits

Global surrogate models - benefits

Flexibility: Any interpretable model can be used as surrogate model. Both the full model and the surrogate model can be exchanged without having to change the rest of the pipeline.
Say one team is familiar with rule lists / decision trees, while another prefers linear models. You can train two surrogate models (linear model and decision tree) for the full model and offer both surrogate models for explanation. If you can later improve the surrogate model, the form of the explanations doesn't change.

Intuition: Surrogate models are chosen for being in themselves interpretable, so the explanation is immediately intuitive to the recipient.

Efficiency: The surrogate model can be trained in advance, and - as it is less complex - takes shorter to run than the full model, making it possible to generate explanations efficiently.
As we will see, some XAI methods can take very long to run.

Global surrogate models - challenges

Global surrogate models - challenges

Fidelity: It is not given how high accuracy the surrogate model should have in order to be considered faithful to the full model. Also, during use, you can never guarantee that the surrogate model will behave like the full model; they are different models. The surrogate model can be faithful to the full model for one set of data, but highly divergent for another.

Assumptions: Every model comes with its underlying assumptions (linear models assume only linear relationships in the data, decision trees make piecewise constant / linear predictions). These assumptions might be stronger than those of the full model.

Interpretability: It can be argued (as some do) that no models are actually interpretable, and that any interpretation is therefore an illusion / misleading.

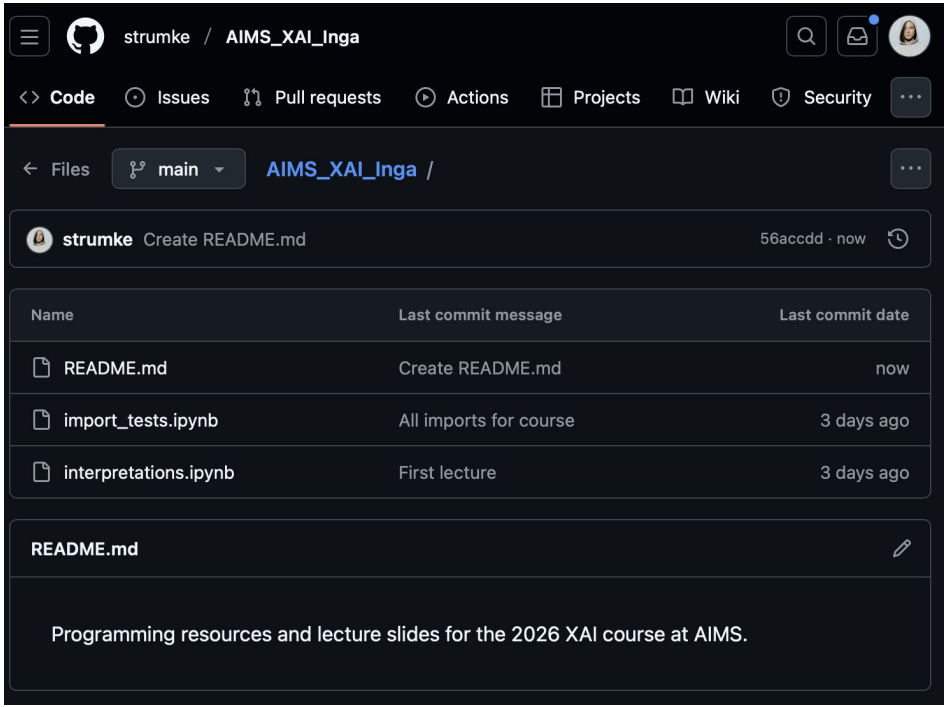
The most troubling aspect: one model is used to explain another, although the two are different. No matter how faithful the surrogate model is to the full model, the interpretation is of the surrogate model, and not the full model.

Homework

1. Write down briefly what an explanation is, what makes an explanation local or global, model-agnostic and model-specific.
2. Choose a tabular dataset for regression or classification from sklearn.datasets, find one on Kaggle Datasets, or use one from the notebook provided.
3. Train a neural network or XGBoost model on it (this is f), for regression or classification.
4. Train a linear/logistic regression and a decision tree classifier/regressor surrogate model on the data and predictions from f (this is g).
5. Create a global explanation from the surrogate model.
6. Pick a single datapoint and explain the prediction of f locally, using the surrogate model g.

Programming resources

github.com/strumke/AIMS_XAI_Inga



Notebooks to help you with the assignments, as well as the slides used in the lectures will be published here daily.

Additional resource

ELSEVIER

Information Fusion

Volume 58, June 2020, Pages 82-115

Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI

Alejandro Barredo Arrieta [a], Natalia Díaz-Rodríguez [b], Javier Del Ser [a c d], Adrien Bennetot [b e f], Siham Tabik [g], Alberto Barbado [h], Salvador Garcia [g], Sergio Gil-Lopez [a], Daniel Molina [g], Richard Benjamins [h], Raja Chatila [f], Francisco Herrera [g]

https://www.sciencedirect.com/science/article/abs/pii/S1566253519308103

Get comfortable interpreting the simple models, we'll start on XAI methods tomorrow :)

See you!

“But linear/logistic regression aren’t useful for anything”

There is a trend in machine learning of spending little time on feature engineering, and large computational resources to fit highly non-linear models with billions of parameters.

Interpretable modelling requires more resources spent on understanding the data and constructing meaningful features. Today, *data scientist* often refers to a person who knows many neural network architectures. But the term *data science* originally refers to the *science* of analysing and extracting information from large sets of data.

Model interpretability - what we've learned

For interpretable models to actually be interpretable, the input features must

- be sufficiently meaningful and expressive to allow for simple, sparse modelling - this is a feature engineering task that may require domain knowledge and multiple iterations.
- have an overseeable correlation structure,

All models are wrong - but some are useful

No model equals reality.

It is possible to create multiple conflicting yet convincing explanations for the same model

Generating explanations is like modelling:

No explanation equals the model.

⇒ "All explanations are wrong - but some are useful" ?

The need to explain

Bell (1985): **"[expert systems] may also fail because the problem are dealing with is not or cannot be understood"**

J. Opl Res. Soc. Vol. 36, No. 7, pp. 613–619, 1985
Printed in Great Britain.

0160-5682/85 $3.00 + 0.00

Why Expert Systems Fail

MICHAEL Z. BELL
Artificial Intelligence Group, Cambridge Consultants Ltd

Although many papers have been published on the subject of expert systems, relatively few of them have been concerned with systems which failed.

In this paper some of the major reasons why expert systems fail are described, along with some techniques for alleviating these problems.

Expert system developments fail for the same reasons as conventional software systems. However, they may also fail because the problem they are dealing with is not or cannot be understood, because they require common sense or knowledge of the limitations of their knowledge, because they cannot be tested, or because they cannot or will not be trusted.

www.jstor.org/stable/pdf/2582480.pdf

Lecture can be extended with the following preliminaries if needed

Random variables, distributions,

p(.) marginal p(.|.) conditional, p(.;.) og p(.,.) joint

Calculate expected value, additivity of expected value

Marginal distribution, conditional distribution,

Markov chain, markov decision processes,

Matrix multiplication .dot, @, etc

Lecture can be extended with the following preliminaries if needed

Data loading, torch loader. Encoding and scaling, train test split,

Imagenet

Classification vs regression

Metrics: confusion matrix & accuracy, MSE

Train a decision tree

Train a linear regression model

Train an XGBoost model

Build and train a neural network in pytorch. Loss, forward(),

Dropout

The interpretability - accuracy “tradeoff”

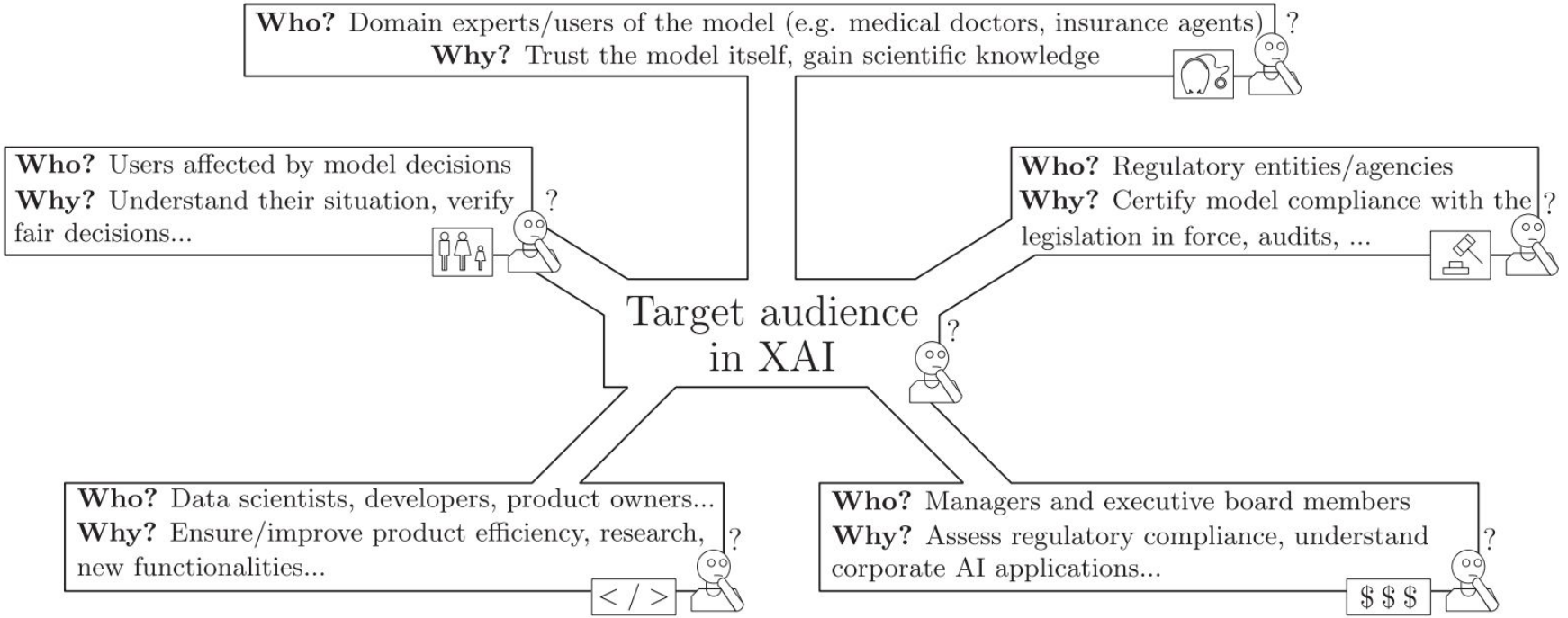
There are tradeoffs in statistics and machine learning, like the bias-variance tradeoff, the precision-recall tradeoff, etc. These represents quantities that (are on opposite ends of a see-saw)

A DARPA announcement from 2016 introduced a tradeoff between interpretability and accuracy, illustrated by the following graph.

This has been adopted to some extent by the community, but I want to give you a vaccine: this “tradeoff” has no mathematical foundation, like other famous tradeoffs in the field. The figure was created using one dataset and a narrow selection of models, and does not represent real-world modelling approaches, that involve data collection, selection, curation and analysis.

Yes, large neural networks with billions of parameters are non-interpretable and highly accurate. *However*, this does not constitute a proof that models *must* be non-interpretable to be accurate. In practice we observe that the trade-off between interpretability and accuracy generally occurs. This is as much an observation about the trend in machine learning (use large resources to fit billions of parameters) as it is an observation about the potential of interpretable models. (more on interpretable models last time)

End-user perspective



… and explanations would be *really* nice

Models have internalised something we humans would like to learn. Explanations providing understanding can lead to **knowledge discovery**.

Some systems function so well that (almost) no human intervention is necessary
⇒ machine autonomy. Understanding is a prerequisite for **accountability** and vital for protecting human **autonomy**.

When decisions affect human lives (medicine, law, national defence, …) understanding is required for **human control** and **oversight**.

Classic example: “computer says no”

Person applies for a loan at a bank

“Computer says no”

Person has the “right to an explanation”
⇒ person must put in a position where they can change the outcome

Understanding why things happen to us and what we can do to affect the outcome preserves our autonomy.

We can't test our way to robustness

Testing provides a measure of performance on a selection of data, but does not

1. tell us how the model will perform in new situations
2. tell us on what the model predictions are based.

Various (attempts at) real world applications have demonstrated that NN models are non-robust (e.g. vulnerable to adversarial attacks)

Yes, fixable. But: you'd *never* think of testing this specific scenario!

Robust Physical-World Attacks on Deep Learning Visual Classification, Kevin Eykholt et al, 2018