Uitvoeringsorganisatie
Bedrijfsvoering Rijk
Ministerie van Binnenlandse Zaken en Koninkrijksrelaties

Explainable AI with Python

Nino van Halem

About me

Artificial Intelligence BSc @ Nijmegen

Artificial Intelligence MSc @ Nijmegen

Software engineer/ Data scientist @ NFI

Data scientist @ Rijks ICT Gilde

Agenda

- Explainability
- Machine learning models
- Explainability methods
- Wrap up and conclusion

Explainability

- **What?**

  Why am I getting a discount and my neighbour isn’t?

  Why are we predicting people to buy certain products?

  How certain is the car, what is it’s priority?

- **Why?**

  GDPR, non-discrimination, transparency, fairness, trust, control

Agenda

- Explainability
- **Machine learning models**
- Explainability methods
- Wrap up and conclusion

Machine learning models

- **Directly interpretable (easy to explain)**

  Linear regression

  Logistic regression

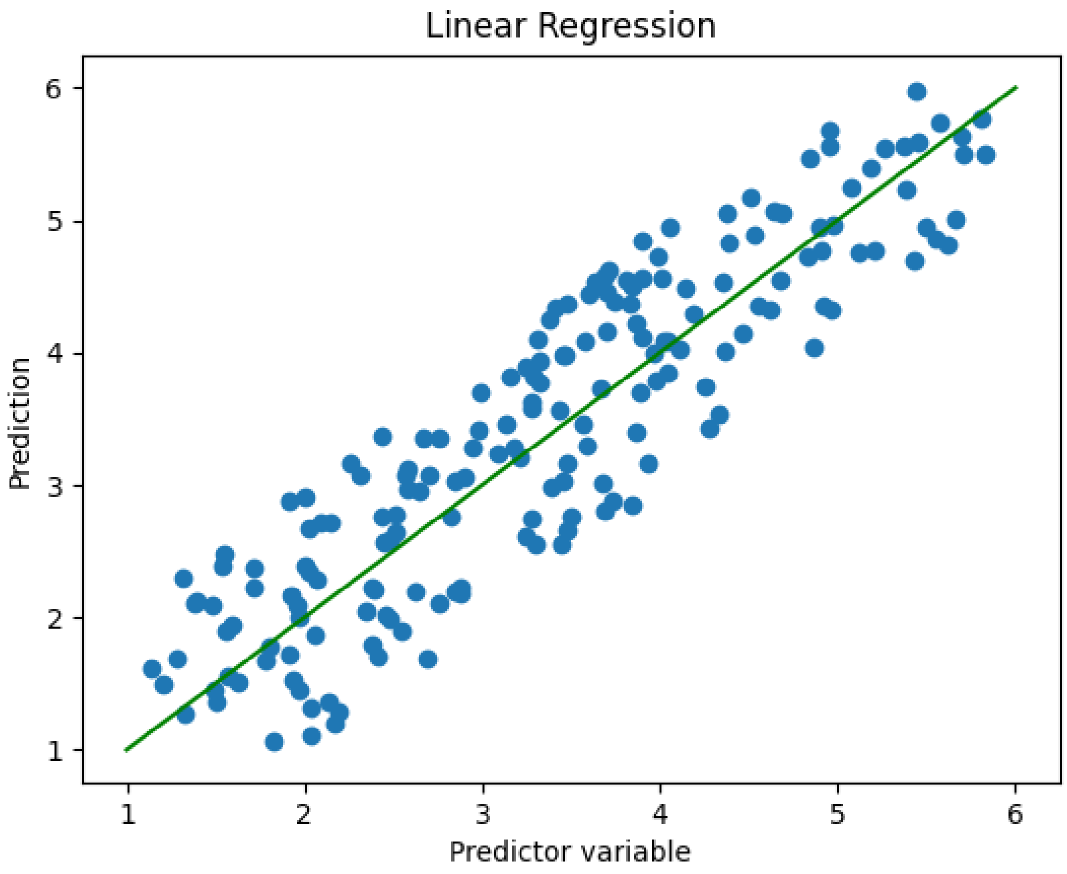
  Decision trees

- **Not directly interpretable (hard to explain)**

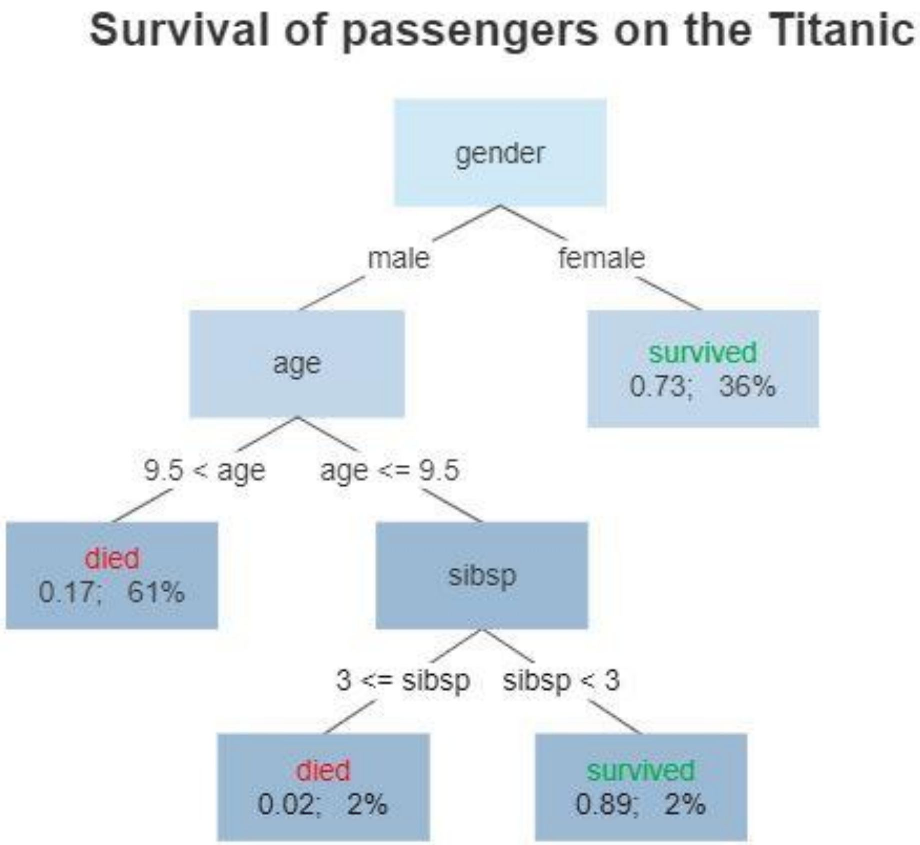
  Neural networks

  Random forests

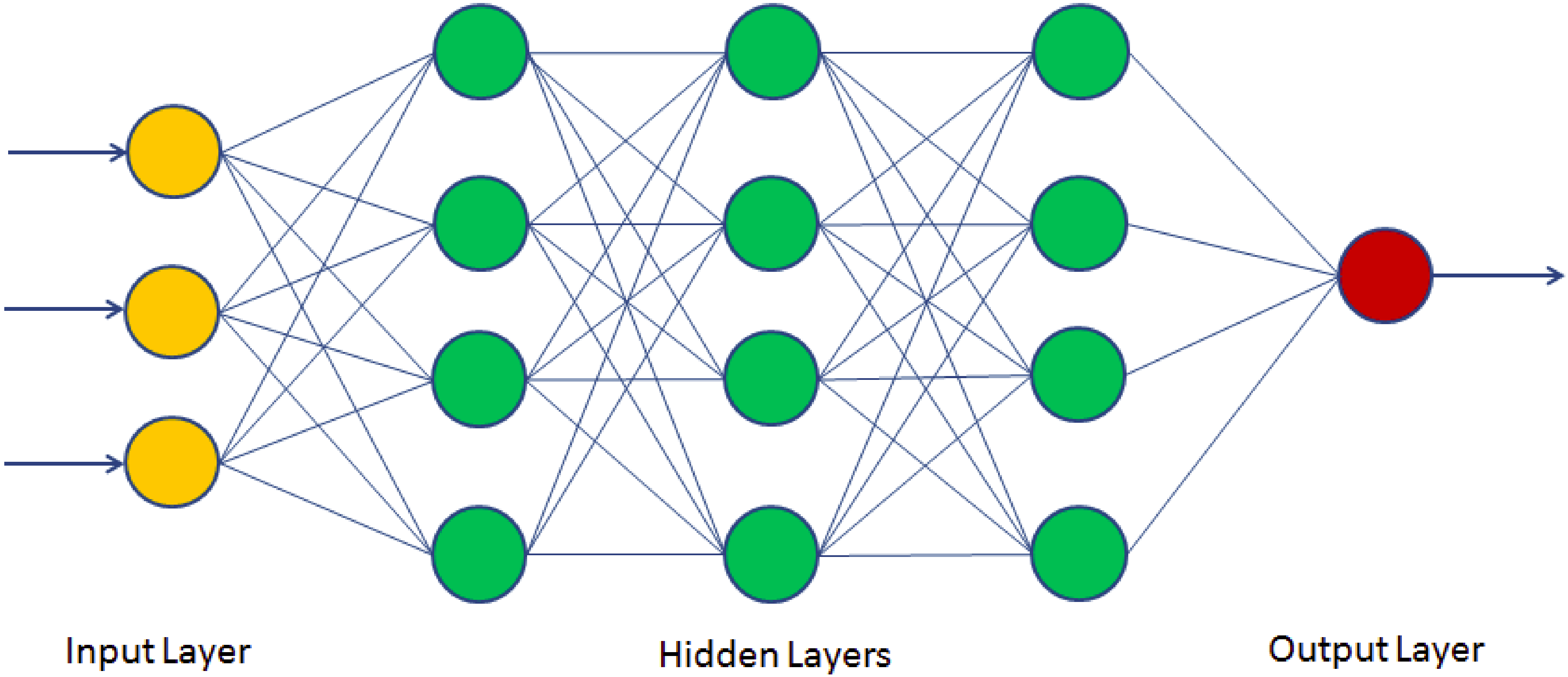
Linear regression



Decision tree



Neural network



Random forest

Agenda

- Explainability
- Machine learning models
- **Explainability methods**
- Wrap up and conclusion

Explainability

- **Model-agnostic**

  Works for any model
- **Model-specific**

  Uses the structure of the model

- **Global**

  About the behaviour of the entire model
- **Local**

  About a certain prediction

California block prices

- **Ground truth:**

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
9463	2.7	32.0	5.56338	1.06338	380.0	2.676056	39.44	-123.73	1.65

- **Prediction:**

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
9463	2.7	32.0	5.56338	1.06338	380.0	2.676056	39.44	-123.73	

Breakdown Plot

- **What?**

  Change in average prediction (on training set) after fixing each variable to test value

- **How?**

  Fix variable

  Keep track of change in average prediction

  Repeat!

California block prices

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
5247	15.0001	36.0	9.368263	1.173653	862.0	2.580838	34.09	-118.44	5.00001
19550	2.0885	35.0	4.812065	1.106729	1687.0	3.914153	37.62	-121.01	0.73700
18764	1.4911	18.0	6.215859	1.453744	494.0	2.176211	40.75	-122.31	0.75800
15779	3.8625	52.0	8.758621	2.482759	153.0	5.275862	37.78	-122.41	3.50000
8870	6.6343	52.0	7.166189	1.037249	748.0	2.143266	34.06	-118.40	5.00001
...	...	...	...	...	...	...	...	...	...
10967	4.8000	34.0	5.223881	1.044776	723.0	3.597015	33.76	-117.89	1.92700
17310	11.7794	39.0	14.666667	1.809524	59.0	2.809524	34.35	-119.50	5.00001
5199	1.4329	21.0	3.057762	1.003610	1283.0	4.631769	33.93	-118.28	0.94100
12187	3.1832	9.0	6.288462	1.083333	596.0	3.820513	33.67	-117.31	1.57400
235	2.3036	35.0	4.620513	1.176923	1009.0	2.587179	37.79	-122.20	1.26000

2.078

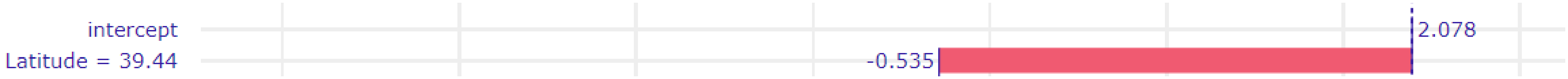
Breakdown Plot

California block prices

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
5247	15.0001	36.0	9.368263	1.173653	862.0	2.580838	39.44	-118.44	???
19550	2.0885	35.0	4.812065	1.106729	1687.0	3.914153	39.44	-121.01	???
18764	1.4911	18.0	6.215859	1.453744	494.0	2.176211	39.44	-122.31	???
15779	3.8625	52.0	8.758621	2.482759	153.0	5.275862	39.44	-122.41	???
8870	6.6343	52.0	7.166189	1.037249	748.0	2.143266	39.44	-118.40	???
...	...	...	...	...	...	...	...	...	...
10967	4.8000	34.0	5.223881	1.044776	723.0	3.597015	39.44	-117.89	???
17310	11.7794	39.0	14.666667	1.809524	59.0	2.809524	39.44	-119.50	???
5199	1.4329	21.0	3.057762	1.003610	1283.0	4.631769	39.44	-118.28	???
12187	3.1832	9.0	6.288462	1.083333	596.0	3.820513	39.44	-117.31	???
235	2.3036	35.0	4.620513	1.176923	1009.0	2.587179	39.44	-122.20	???

1.543

Breakdown Plot

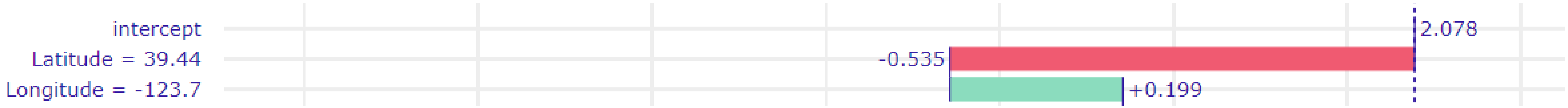


California block prices

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
5247	15.0001	36.0	9.368263	1.173653	862.0	2.580838	39.44	-123.7	???
19550	2.0885	35.0	4.812065	1.106729	1687.0	3.914153	39.44	-123.7	???
18764	1.4911	18.0	6.215859	1.453744	494.0	2.176211	39.44	-123.7	???
15779	3.8625	52.0	8.758621	2.482759	153.0	5.275862	39.44	-123.7	???
8870	6.6343	52.0	7.166189	1.037249	748.0	2.143266	39.44	-123.7	???
...	...	...	...	...	...	...	...	...	...
10967	4.8000	34.0	5.223881	1.044776	723.0	3.597015	39.44	-123.7	???
17310	11.7794	39.0	14.666667	1.809524	59.0	2.809524	39.44	-123.7	???
5199	1.4329	21.0	3.057762	1.003610	1283.0	4.631769	39.44	-123.7	???
12187	3.1832	9.0	6.288462	1.083333	596.0	3.820513	39.44	-123.7	???
235	2.3036	35.0	4.620513	1.176923	1009.0	2.587179	39.44	-123.7	???

1.742

Breakdown Plot

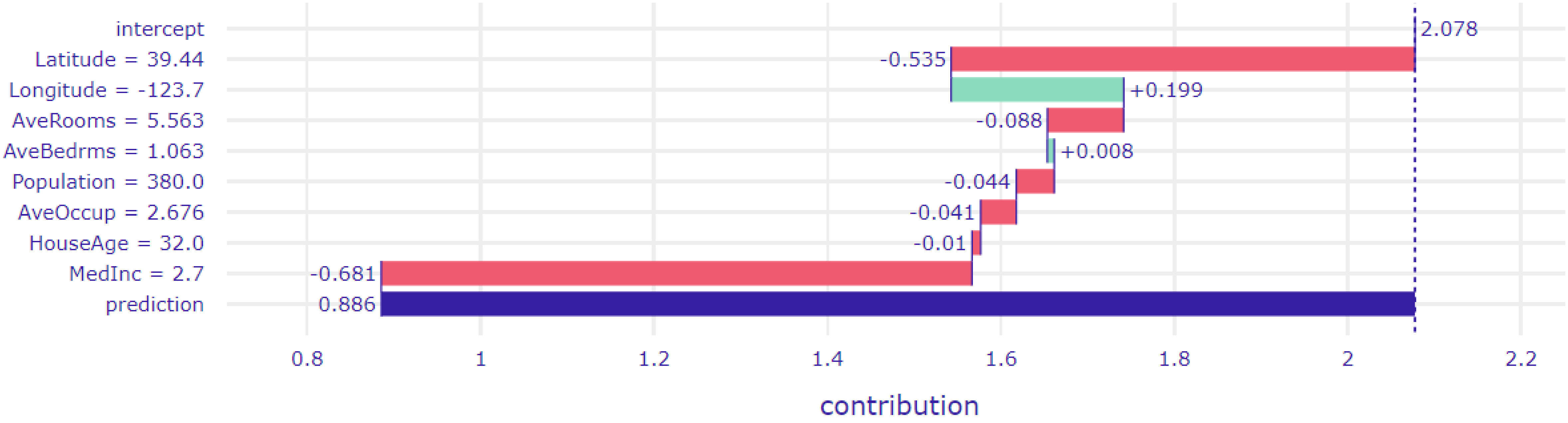


California block prices

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	MedHouseVal
5247	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
19550	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
18764	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
15779	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
8870	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
...	...	...	...	...	...	...	...	...	...
10967	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
17310	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
5199	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
12187	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???
235	2.7	32.0	5.563	1.063	3.8	2.676	39.44	-123.7	???

0.886

Breakdown Plot



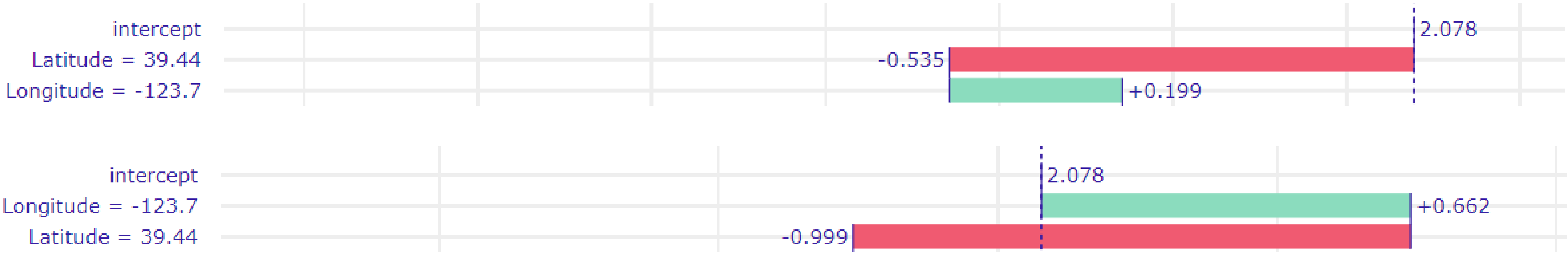
Breakdown Plot

```
import dalex as dx

exp = dx.Explainer(model, X_train, y_train)
breakdown = exp.predict_parts(X_test_sample,
                              type='break_down',
                              order=np.array(['Latitude', 'Longitude', 'AveRooms', 'AveBedrms',
                                              'Population', 'AveOccup', 'HouseAge', 'MedInc']),
                              random_state=1)
breakdown.plot()
```

Breakdown Plot

```
import dalex as dx

exp = dx.Explainer(model, X_train, y_train)
breakdown = exp.predict_parts(X_test_sample,
                              type='break_down',
                              order=np.array(['Longitude', 'Latitude', 'AveRooms', 'AveBedrms',
                                              'Population', 'AveOccup', 'HouseAge', 'MedInc']),
                              random_state=1)
breakdown.plot()
```

Breakdown Plot

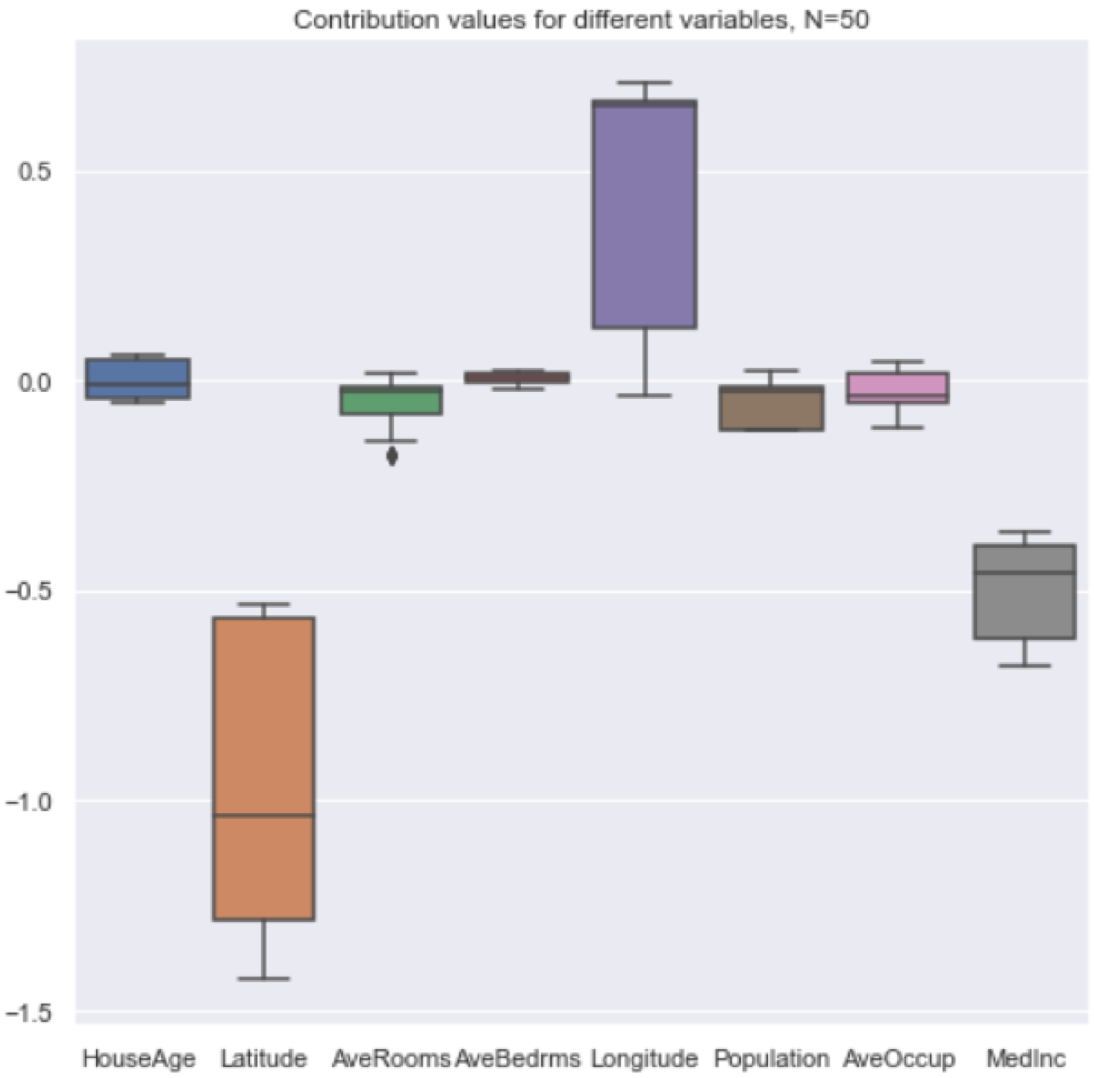


- **Order matters!**

Breakdown Plot

```
n = 50
random.seed(1)
contributions = defaultdict(lambda: [])

for ordering in tqdm(sample(list(itertools.permutations(data.feature_names)), n)):
  breakdown = exp.predict_parts(X_test_sample, type='break_down', order=list(ordering))
  for item in list(zip(breakdown.result.variable_name, breakdown.result.contribution))[1:-1]:
    contributions[item[0]].append(item[1])

sns.boxplot(data=pd.DataFrame(contributions))
_ = plt.title(f'Contribution values for different variables, N={n}')
```

Breakdown Plot



Breakdown Plot

- **Pros**

  Understandable

  Compact

  Intuitive

- **Cons**

  Misleading for interactions

  Order matters

Shapley Values

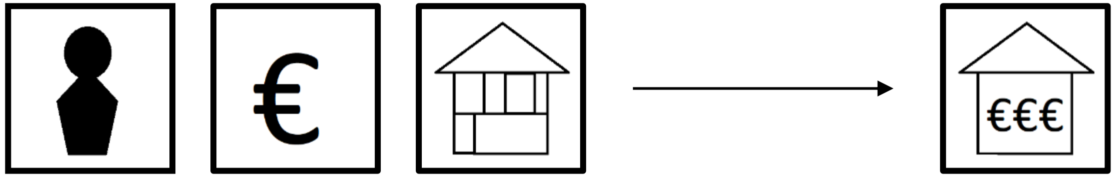
- **Calculate the ‘contribution’ of each variable/player (game theory)**

- **How?**

  Add value of the test instance for a variable to a coalition of other variables

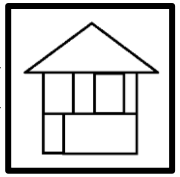
  Compare this with random value for variable of interest

Shapley Values



- **Variables:**
  Number of occupants
  Total income
  Number of rooms

Shapley Values

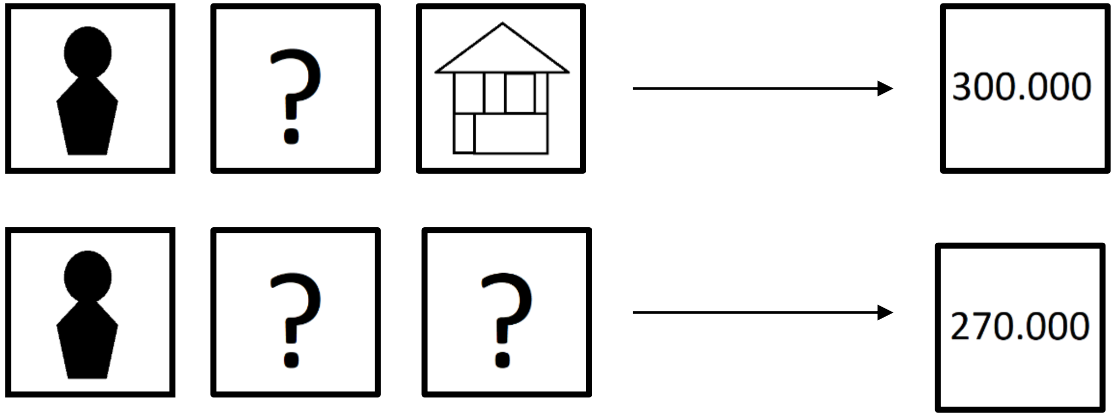
- **Shapley value(**  **)**

- **Coalitions:**

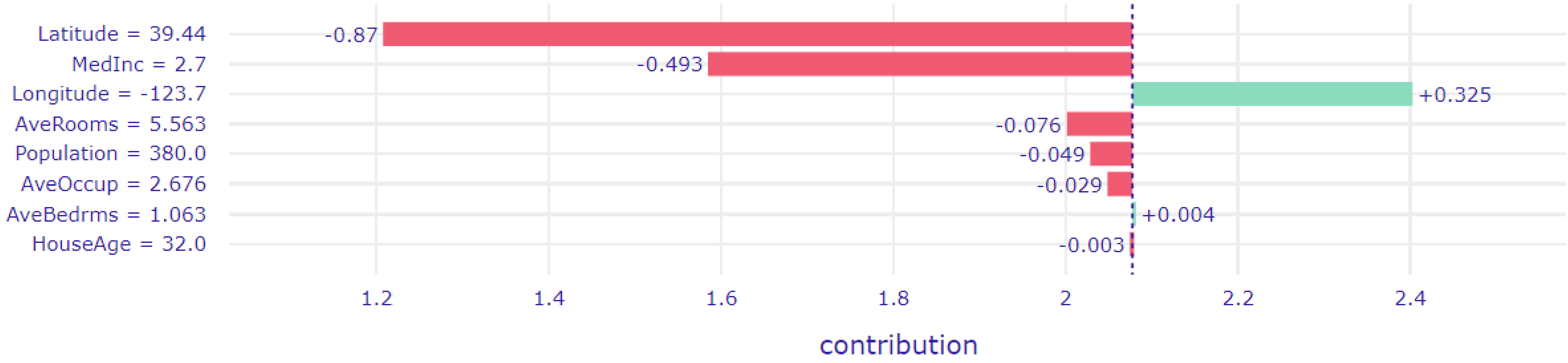
 / € /  € / Ø

Shapley Values

- **Coalition(**  **)**



Shapley Values



Shapley Values

```python
import dalex as dx

exp = dx.Explainer(model, X_train, y_train)
shapley_values = exp.predict_parts(X_test_sample, type='shap', random_state=1)
shapley_values.plot()
```

Shapley Values

- **Pros**

  Single value

  Strong foundation in Game Theory

  Works well for additive contributions

- **Cons**

  Does not show interaction effects

  Can be very time consuming

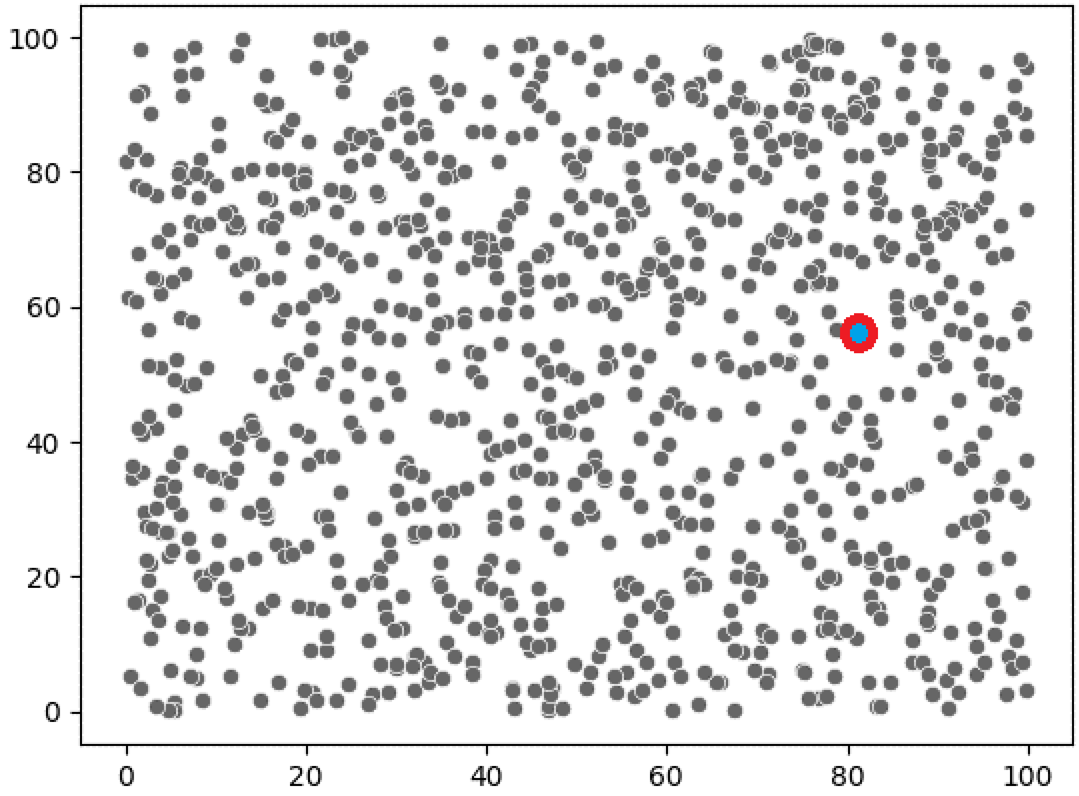
LIME

- Local Interpretable Model-agnostic Explanations
- Fit a simple model on a black box model to explain an instance
- Oversimplification

LIME steps

- Permute data

Permute data

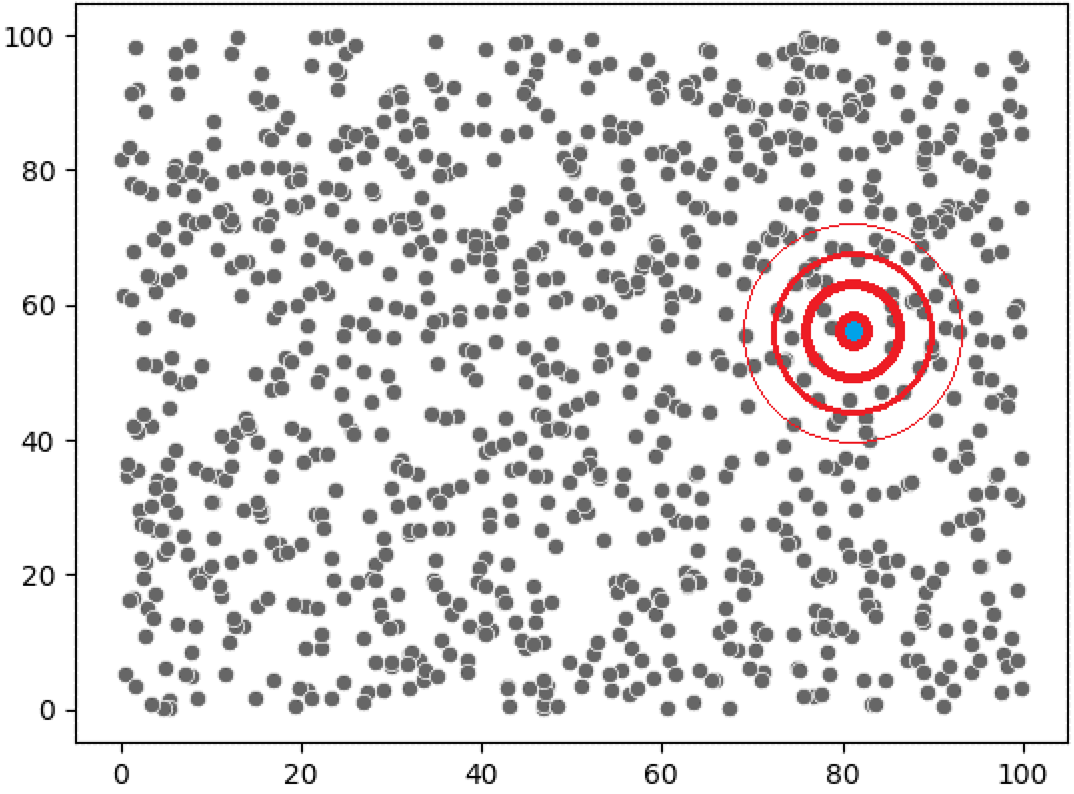


LIME steps

- Permute data

- Calculate distance measure

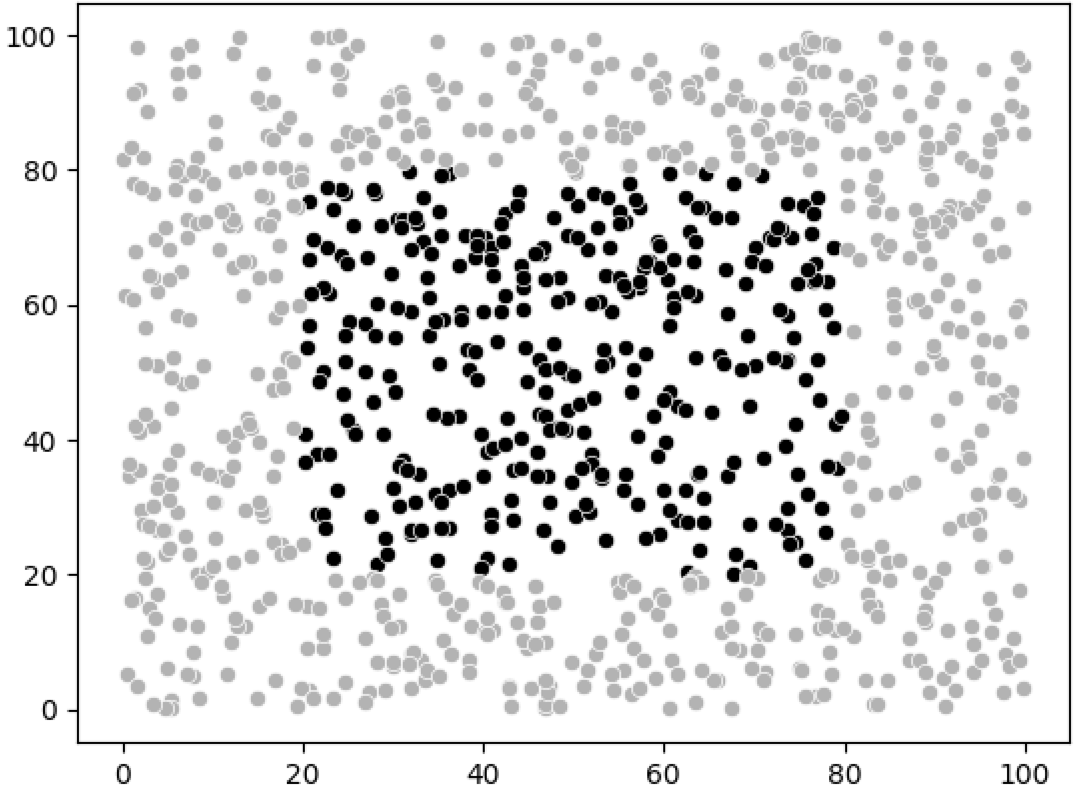
Calculate distance measure



LIME steps

- Permute data
- Calculate distance measure
- Make predictions with black box model

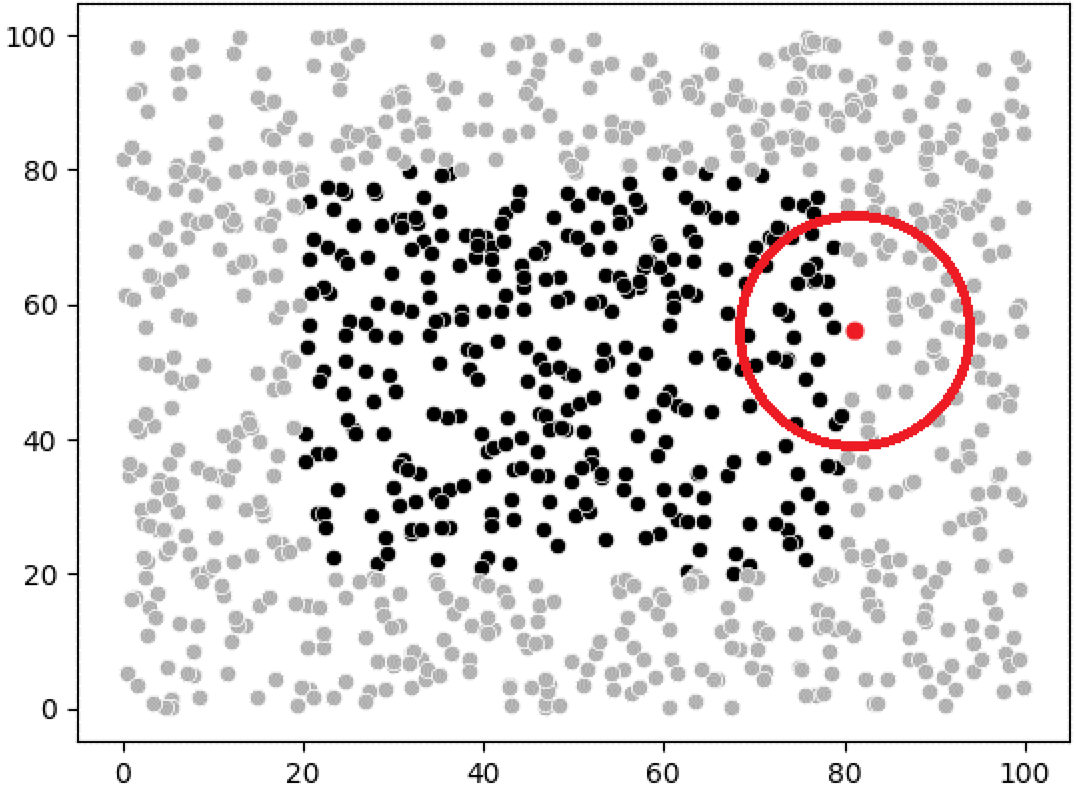
Make predictions with black box model



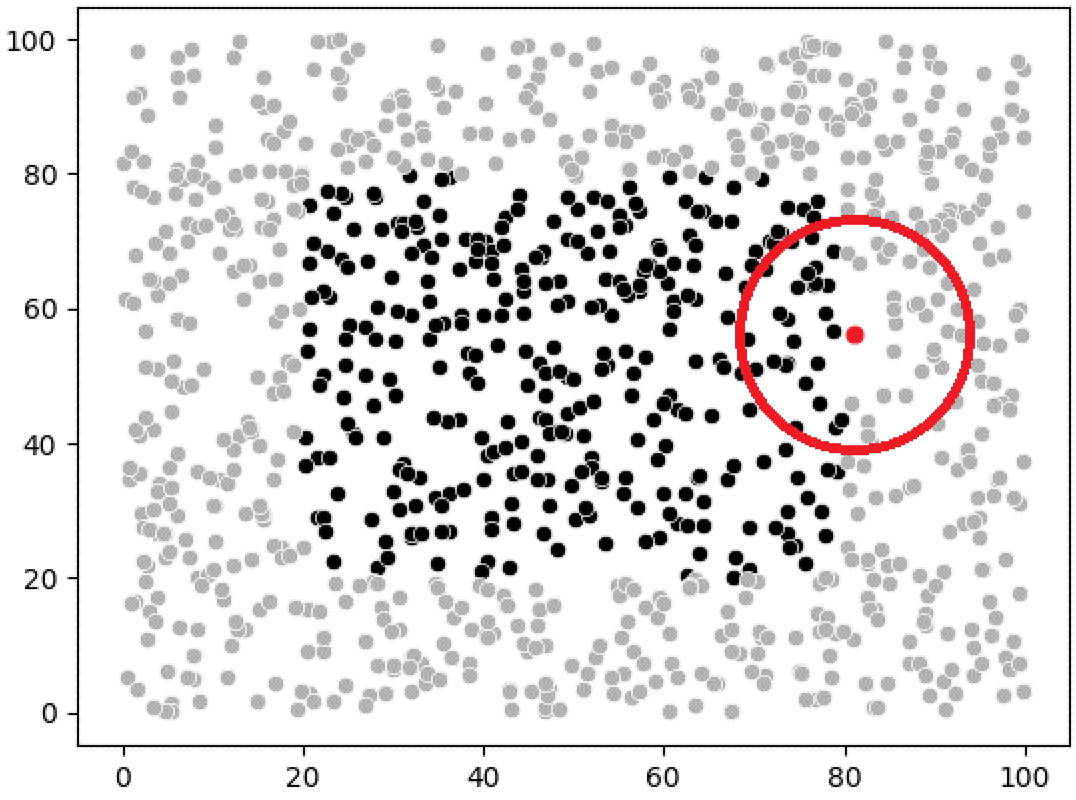
LIME steps

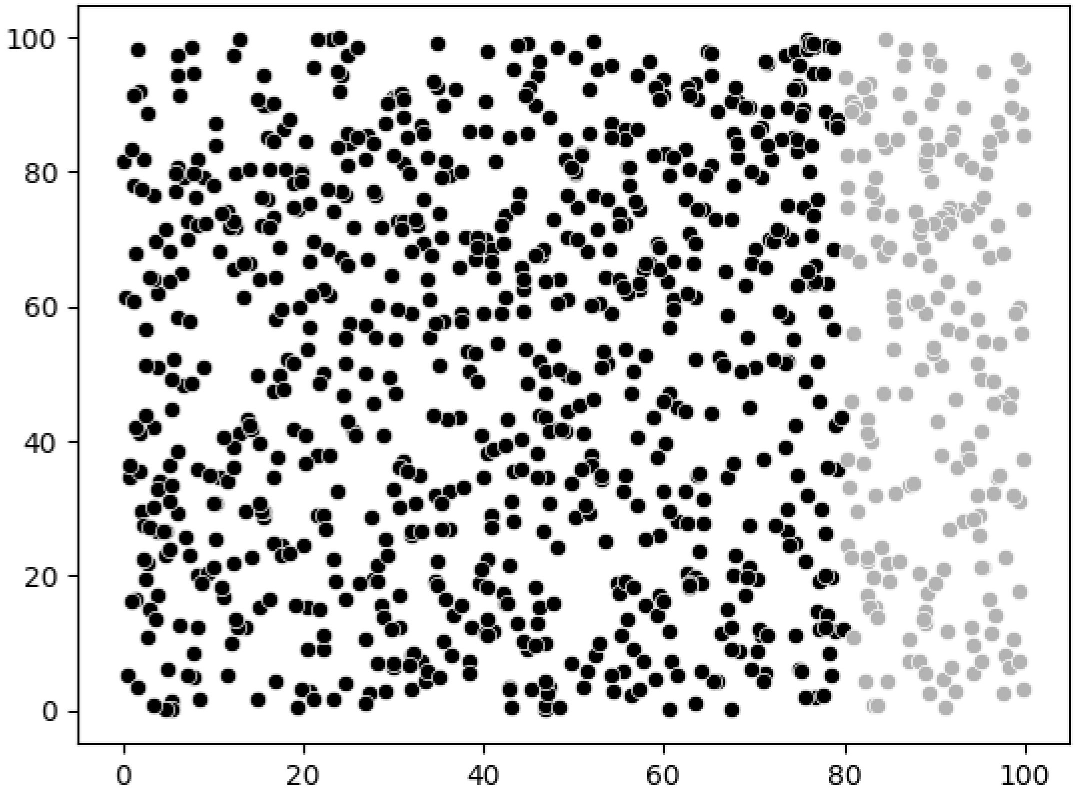
- Permute data
- Calculate distance measure
- Make predictions with black box model
- Choose features
- Fit a simple model (based on distance measure)

Fit a simple model (based on distance measure)

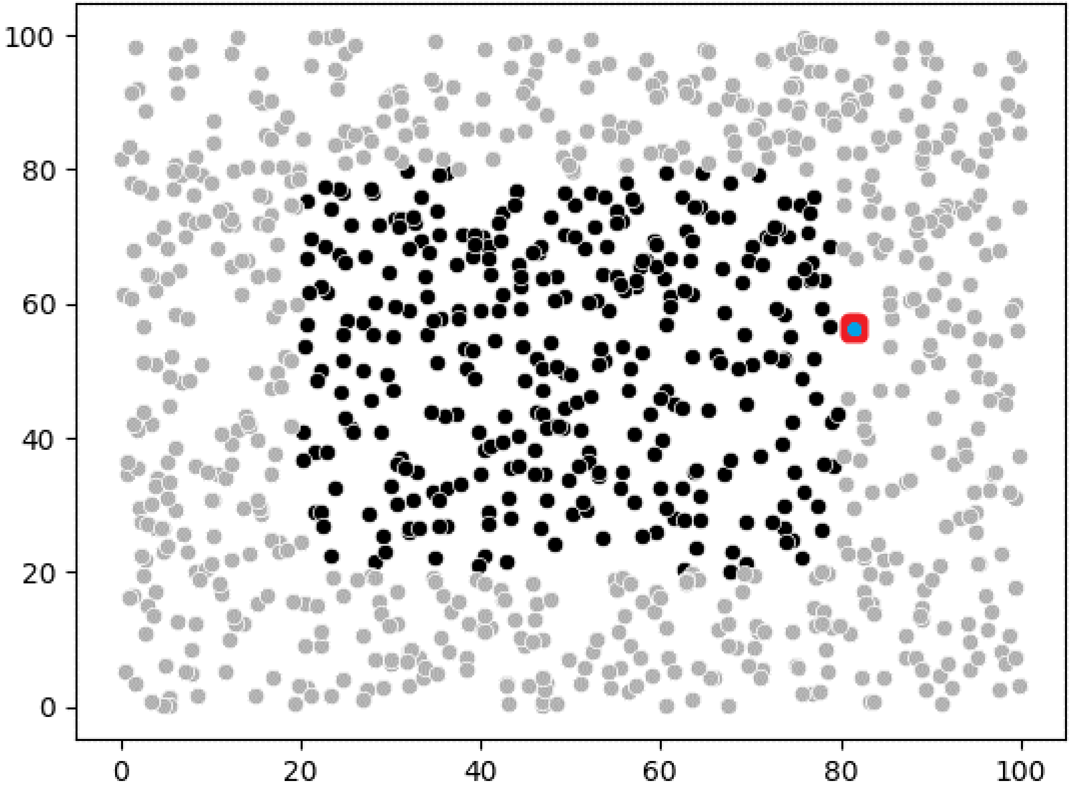


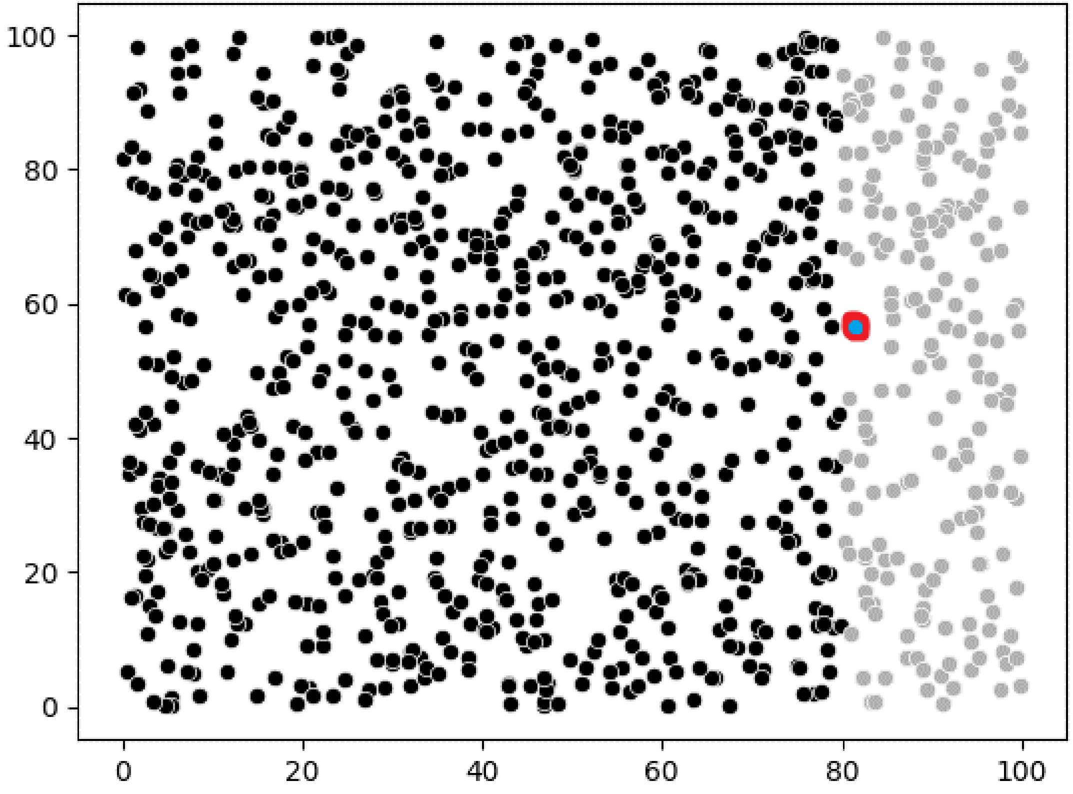
Fit a simple model (based on distance measure)





Explanation

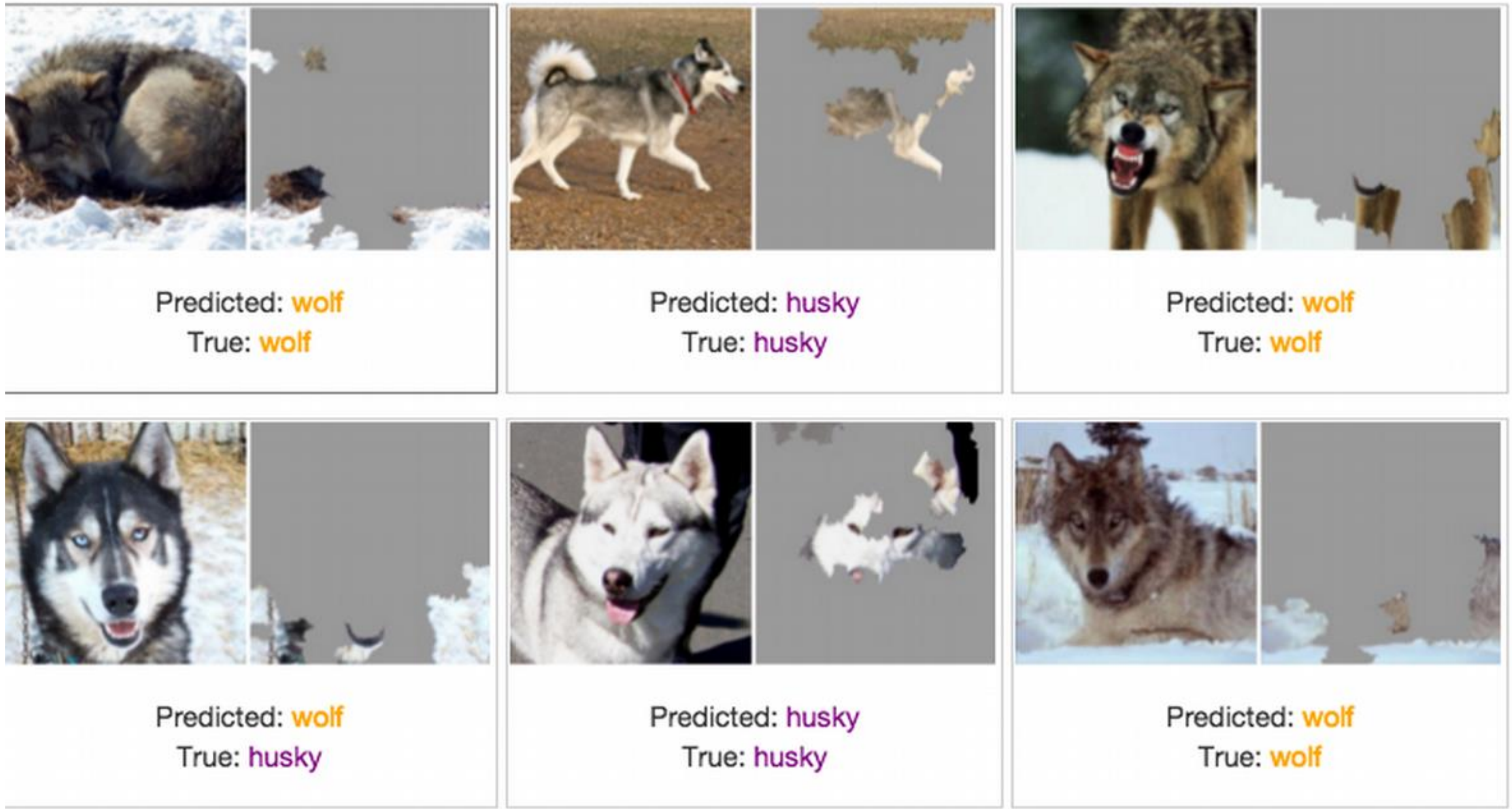




Explanation

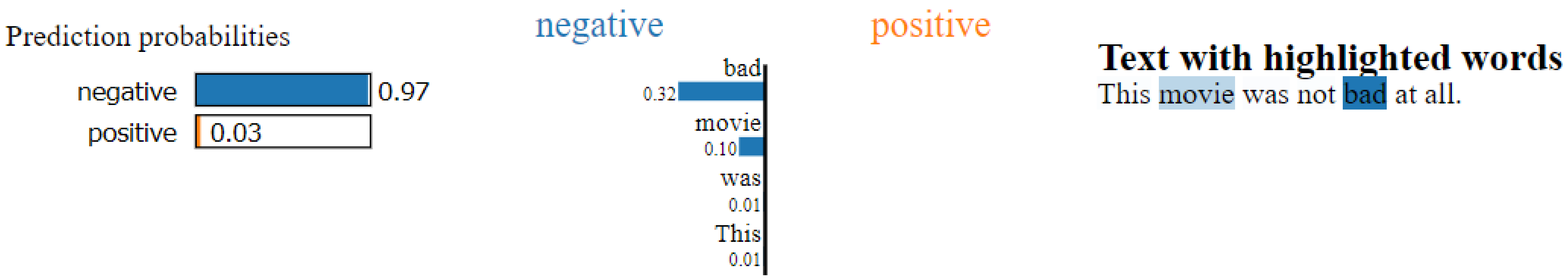
```python
from lime.lime_tabular import LimeTabularExplainer

explainer = LimeTabularExplainer(X_train,
                                 mode='regression',
                                 feature_names=X_train.columns,
                                 random_state=1)
exp_instance = explainer.explain_instance(X_test_sample.iloc[0, :], model.predict)
```

LIME for images



LIME for text

"This movie was not bad at all."



LIME

- **Pros**

  Readable/visual

  Configurable

  Applicable on tabular data, images, and text

- **Cons**

  Definition of neighbourhood is hard

  Instable explanations

Summary

- Many reasons to explain your models
- Many ways to explain your models
- Breakdown plots, Shapley values, LIME

Sometimes explainability is better than performance

Questions?

nino.halem@minbzk.nl

https://github.com/RIG-MYCELIA/XAI-workshop