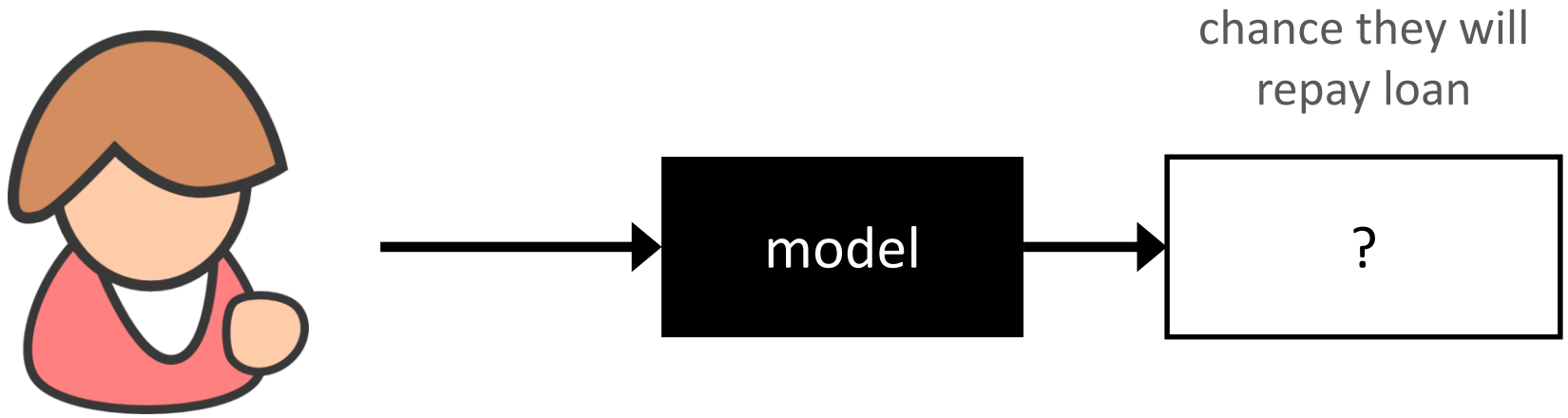
Consistent feature attribution for tree ensembles

Scott Lundberg and Su-In Lee

(presented by Nao Hiranuma)

Sample problem: Filtering loan applications



Susan, a bank customer

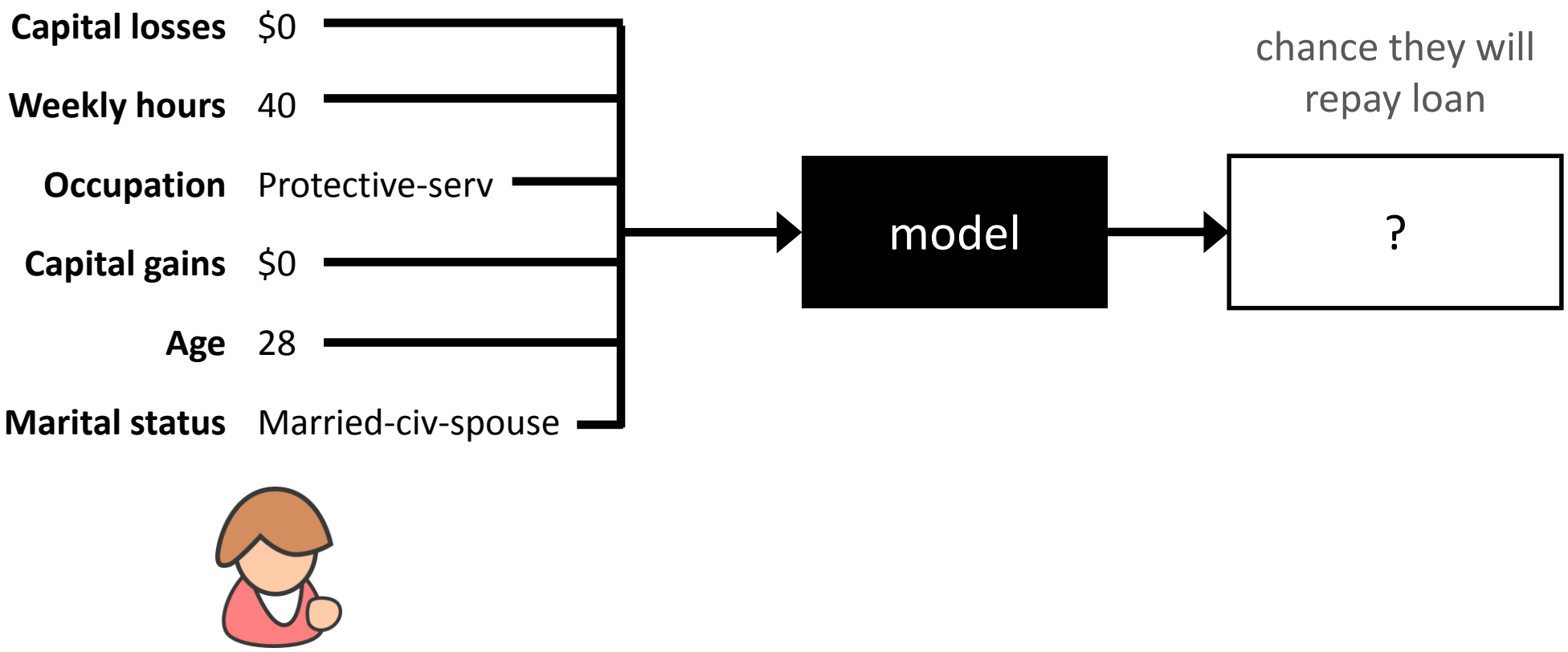
What kind of model would people actually use?

- 2 winning approaches in Kaggle:

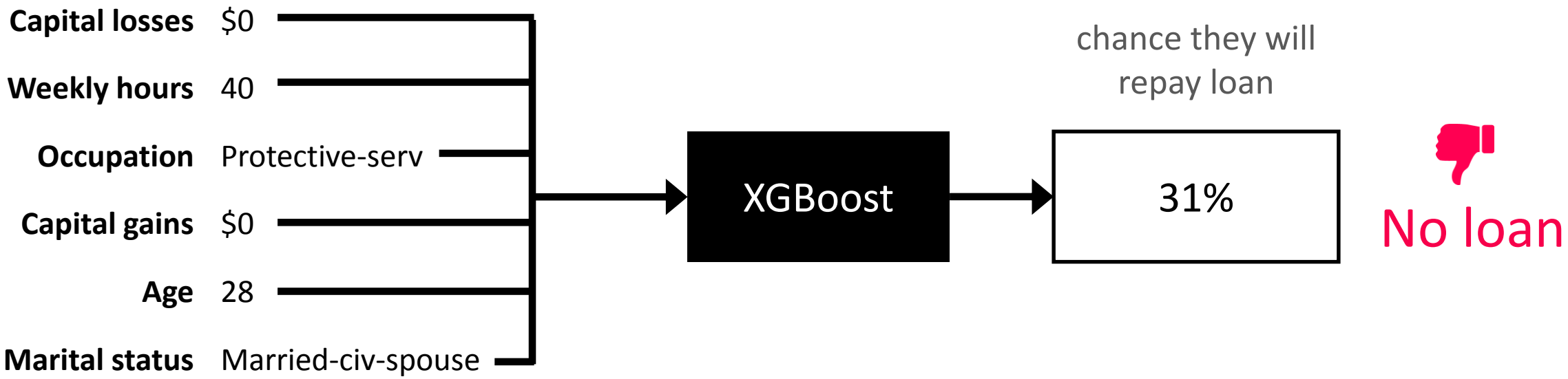
1. Tree ensembles for structured data (hand crafted features)

2. Neural networks for unstructured data (images, speech, etc.)

The bank has structured data



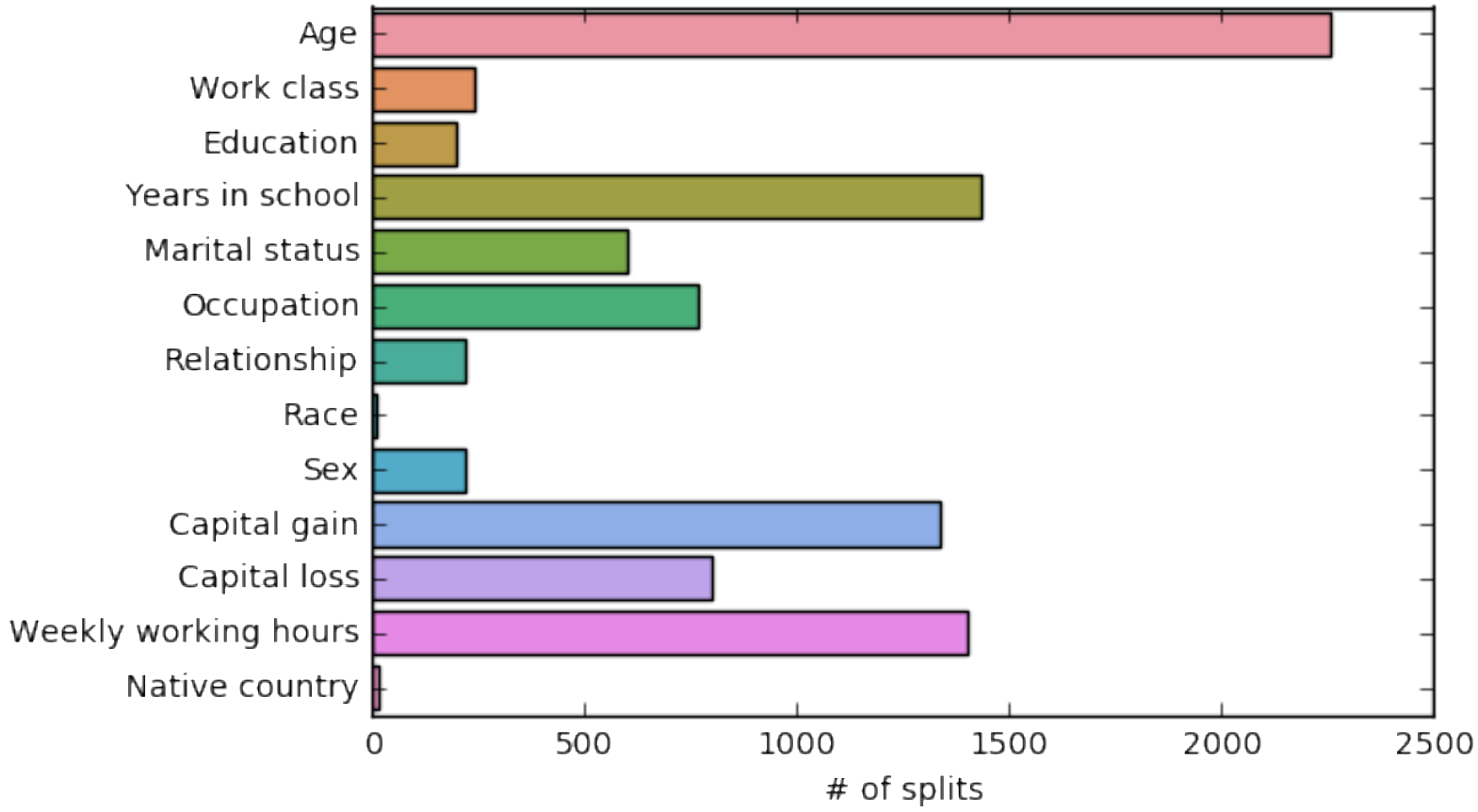
So we use a tree ensemble



Why did Susan's loan get denied?!

Let’s see what features are important

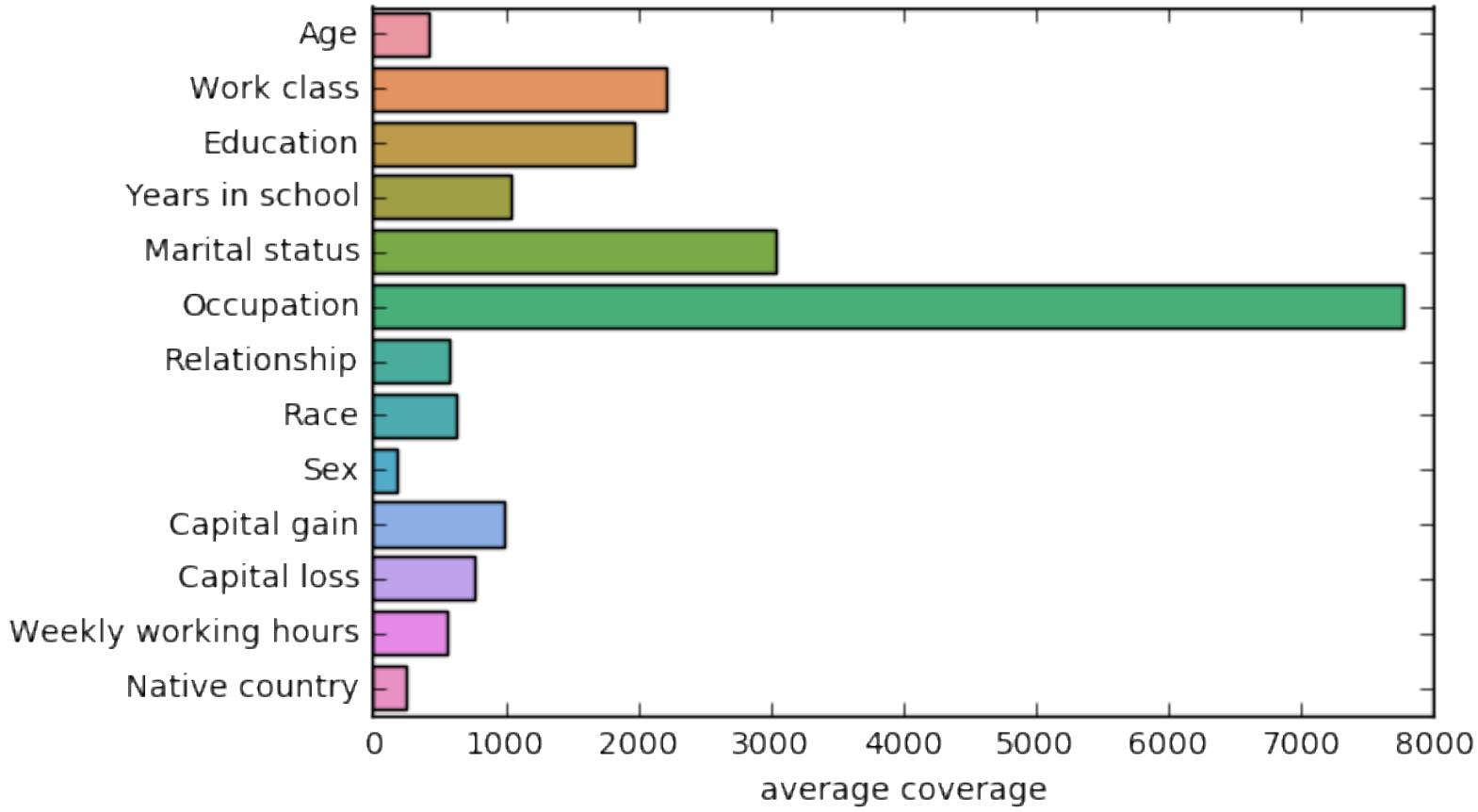
```
xgboost_model.get_score()
```



Age

What about coverage instead split counts?

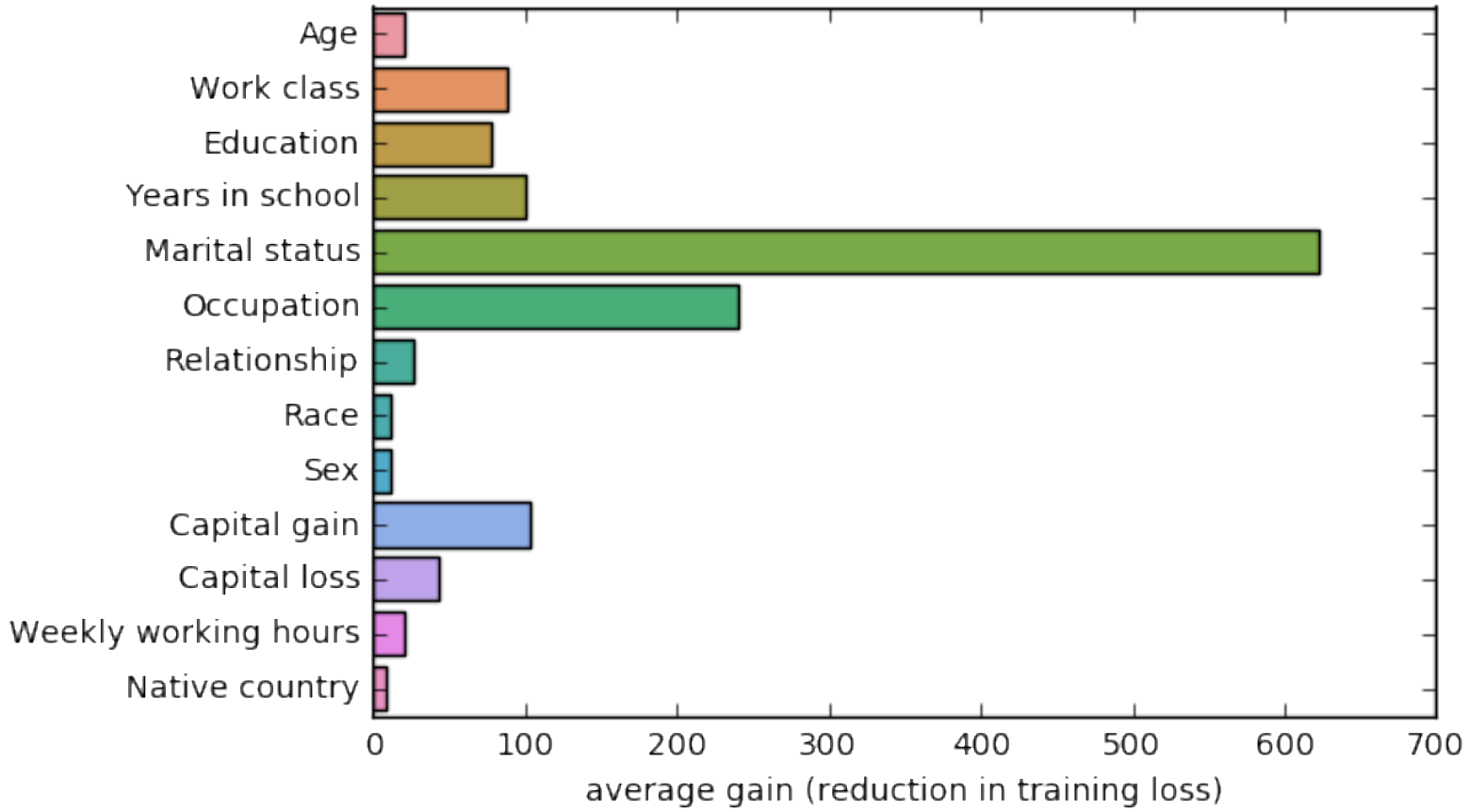
```
xgboost_model.get_score(importance_type="cover")
```



Occupation

What about ‘gain’ (reduction in training loss)?

```
xgboost_model.get_score(importance_type="gain")
```



Marital status

Two problems

1. Global feature importances don’t tell us specifically why Susan was denied a loan.

2. Current ways to measure feature importance are often based on heuristics.

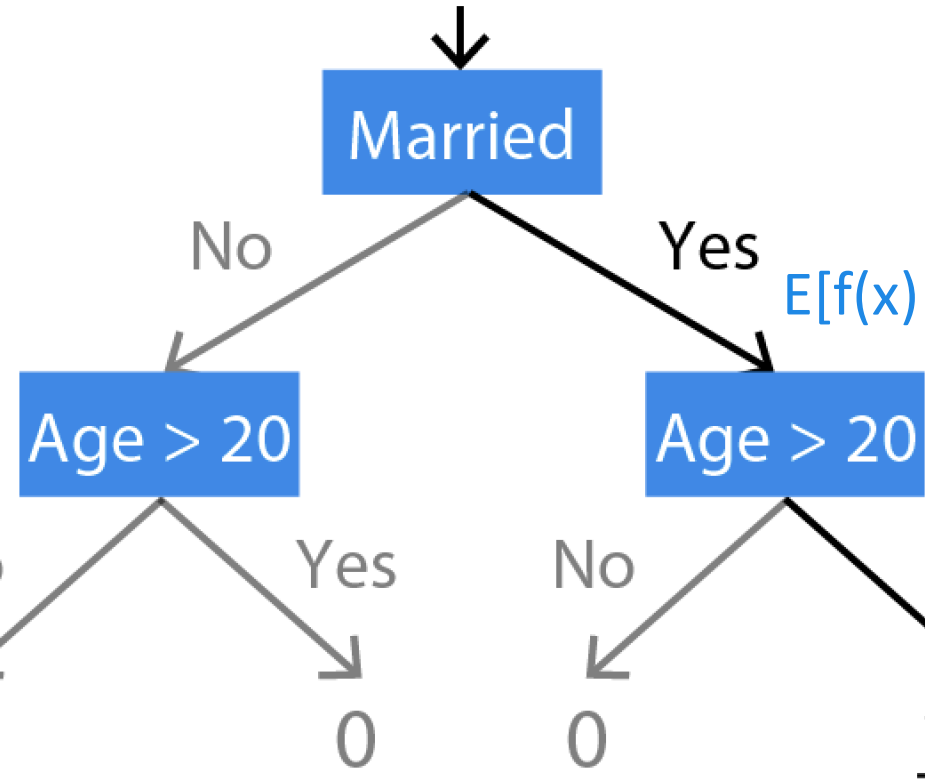
Addressing problem 1: Instance level feature importances

```
xgboost_model.predict(susan_data, pred_contribs=True)
```

Measures the impact of a feature as the change in expected model output, when splitting on that feature.

f(x) = [Married & Age > 20]*100

E[f(x)] = 25



Married: 50 - 25 = **25**

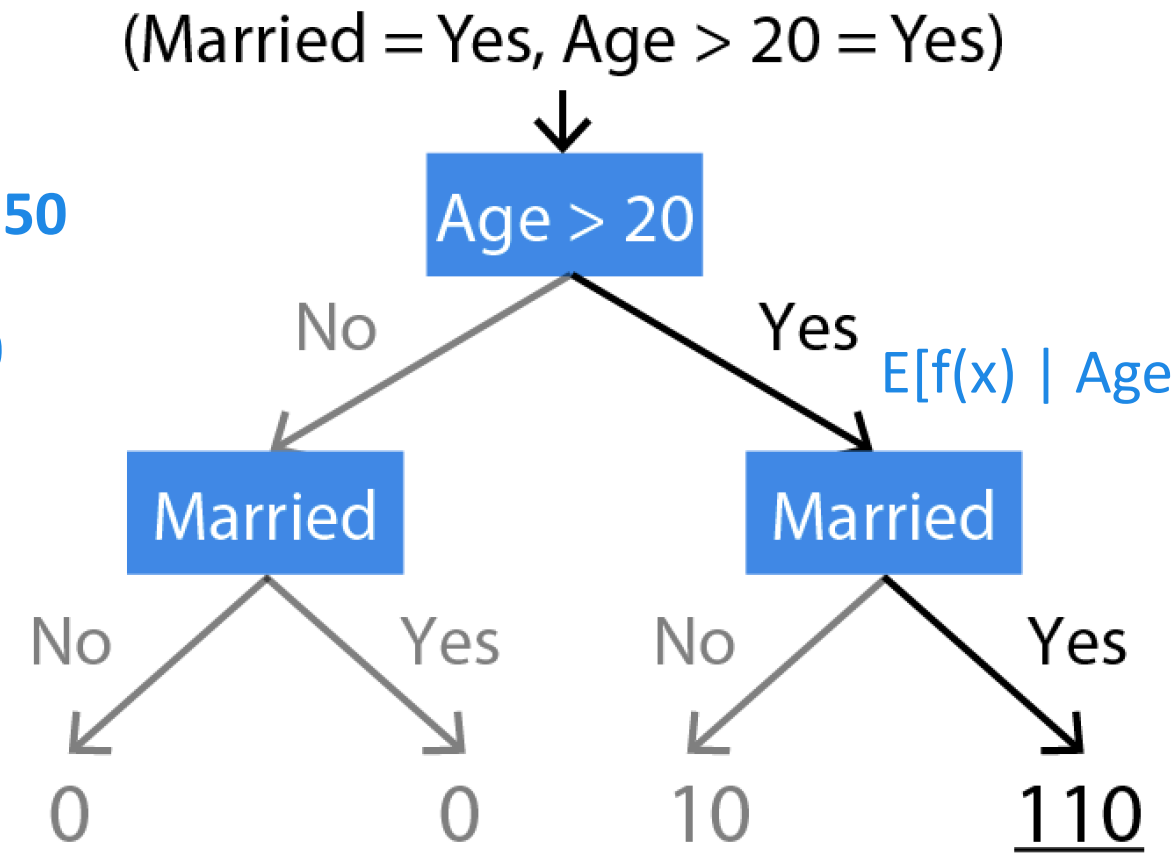
Age > 20: 100 - 50 = **50**

E[f(x) | Married] = 50

E[f(x) | Married, Age > 20] = 100

f(x) = [Married & Age > 20]*100 + [Age > 20]*10

E[f(x)] = 30



Married: 110 - 60 = **50**

Age > 20: 60 - 30 = **30**

E[f(x) | Age > 20] = 60

E[f(x) | Age > 20 , Married] = 110

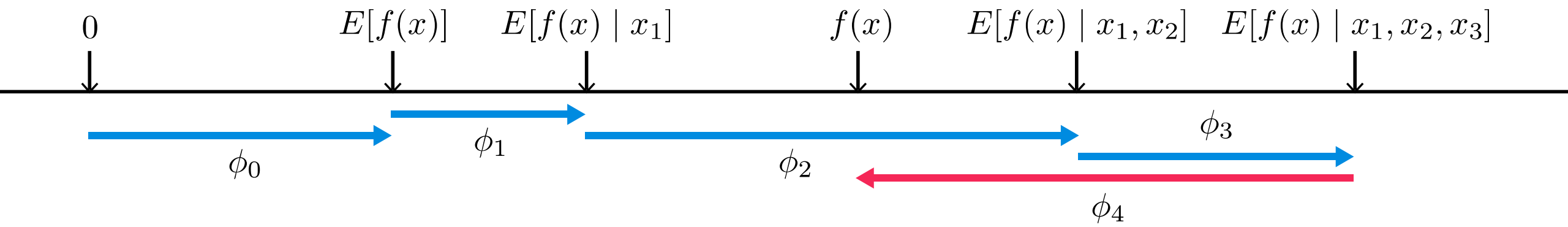
Addressing problem 2: SHapley Additive exPlanation (SHAP) values

- If we want to represent a function's output as a sum of feature attributions then there is **only one possible consistent allocation**.

- This uniqueness results comes from Shapley values in game theory, and when combined with conditional expectations of the function they give rise to **SHAP values**.

SHapley Additive exPlanation (SHAP) values

Use Shapley values to measure the impact of a feature as the change in expected model output, when conditioning on that feature.



The order matters! SHAP values average over all N! possible orderings.

Tree SHAP: Polynomial runtime

Current general SHAP methods require runtime O(2^M) for exact solutions with M features, even when approximating the expected values with a single sample.

We show how to compute SHAP values in $O(MD^2)$ for depth D trees.

This makes exact computation tractable for tree ensembles!

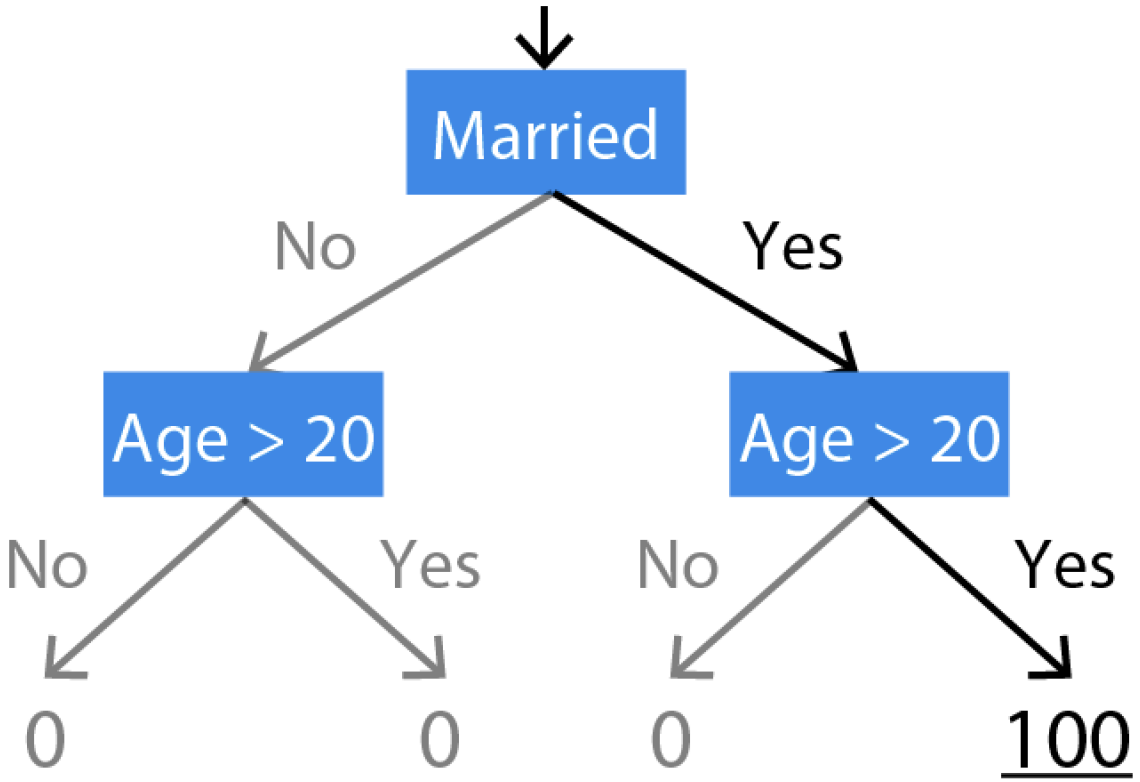
f(x) = [Married & Age > 20]*100

E[f(x)] = 25

(Married = Yes, Age > 20 = Yes)

Married: **37.5**

Age > 20: **37.5**

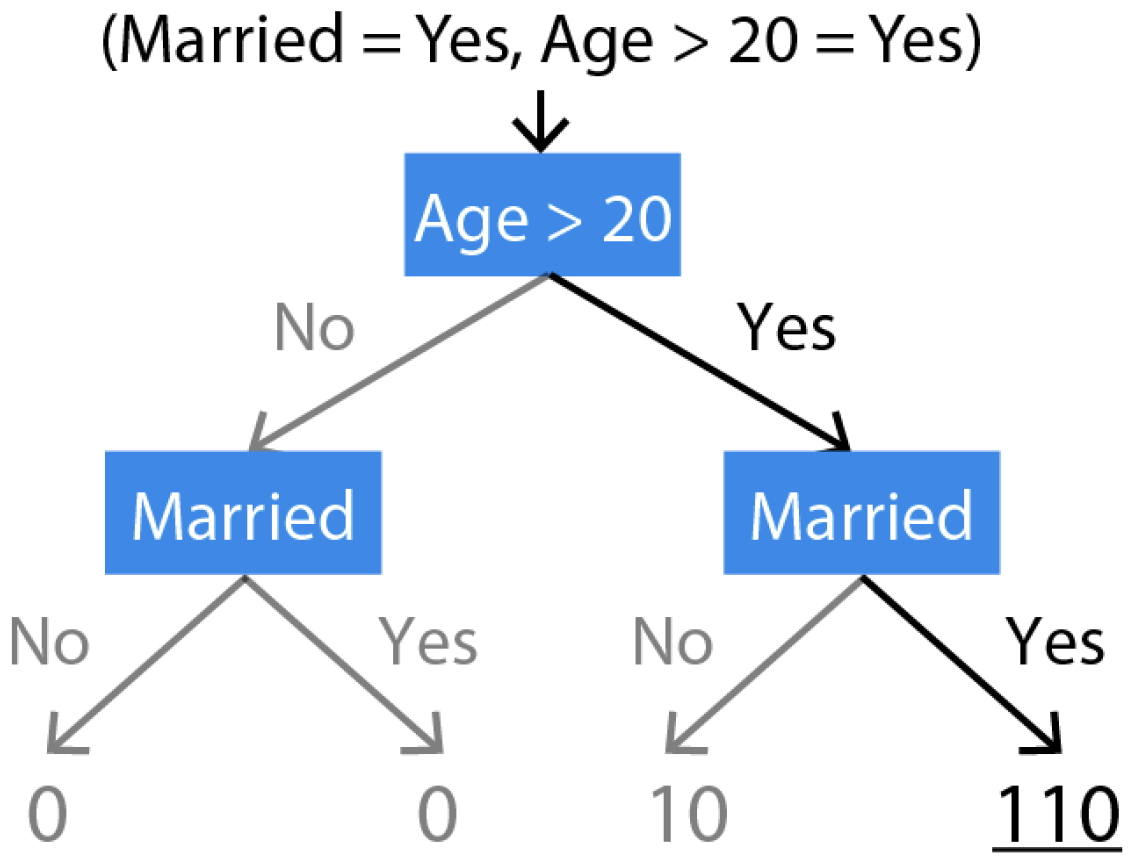


f(x) = [Married & Age > 20]*100 + [Age > 20]*10

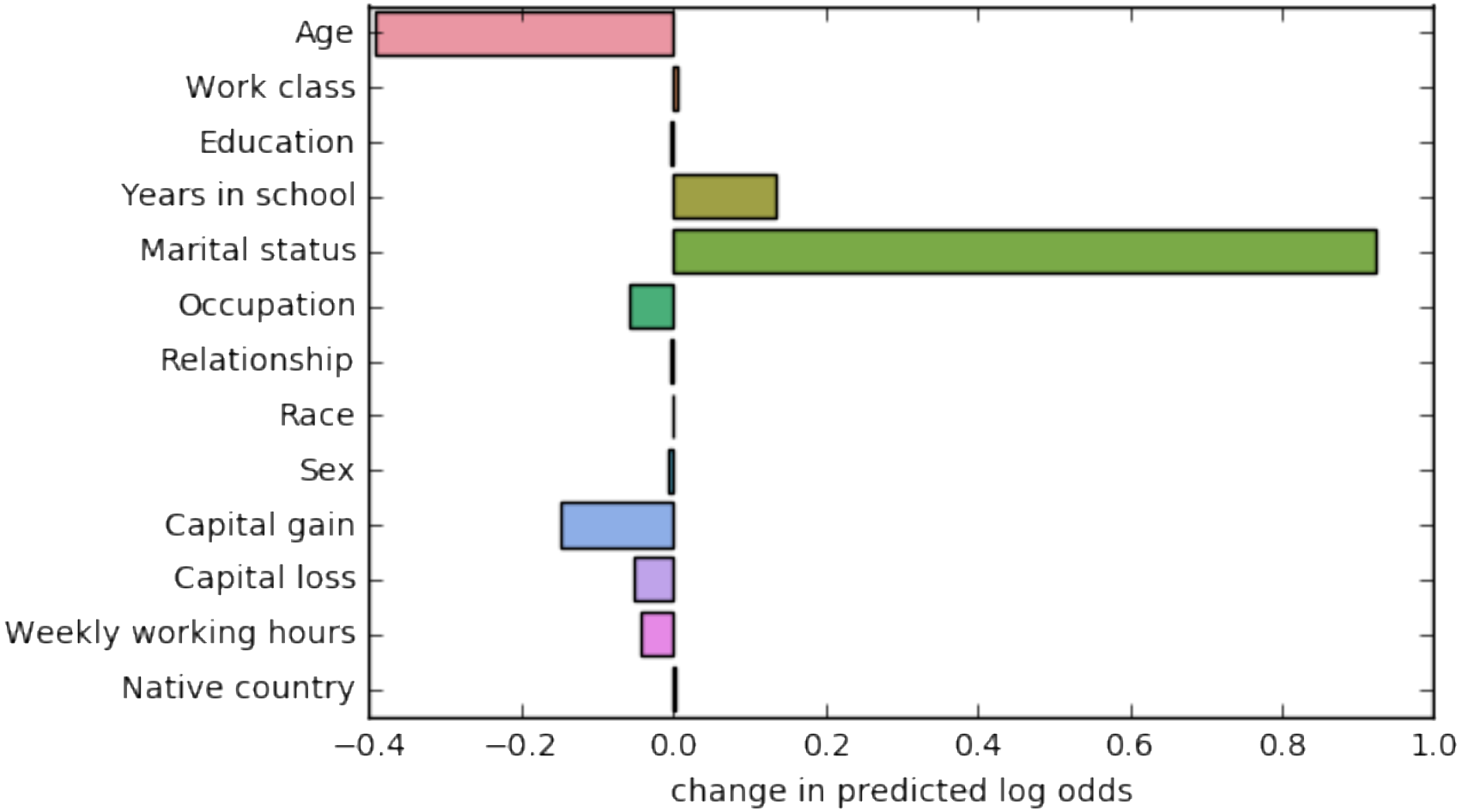
E[f(x)] = 30

Married: **37.5**

Age > 20: **42.5**



Why Susan's loan was denied



x 28 years old

✓ Married

x No capital gains

Tree SHAP

Exact theoretically justified feature attributions,
now very practical for tree models

Questions?

Superior supervised clustering

Superior supervised clustering