LV Datathon

Doug Corbin, Oli Deane, Mauro Camara Escudero, Jack Simons

January 2021

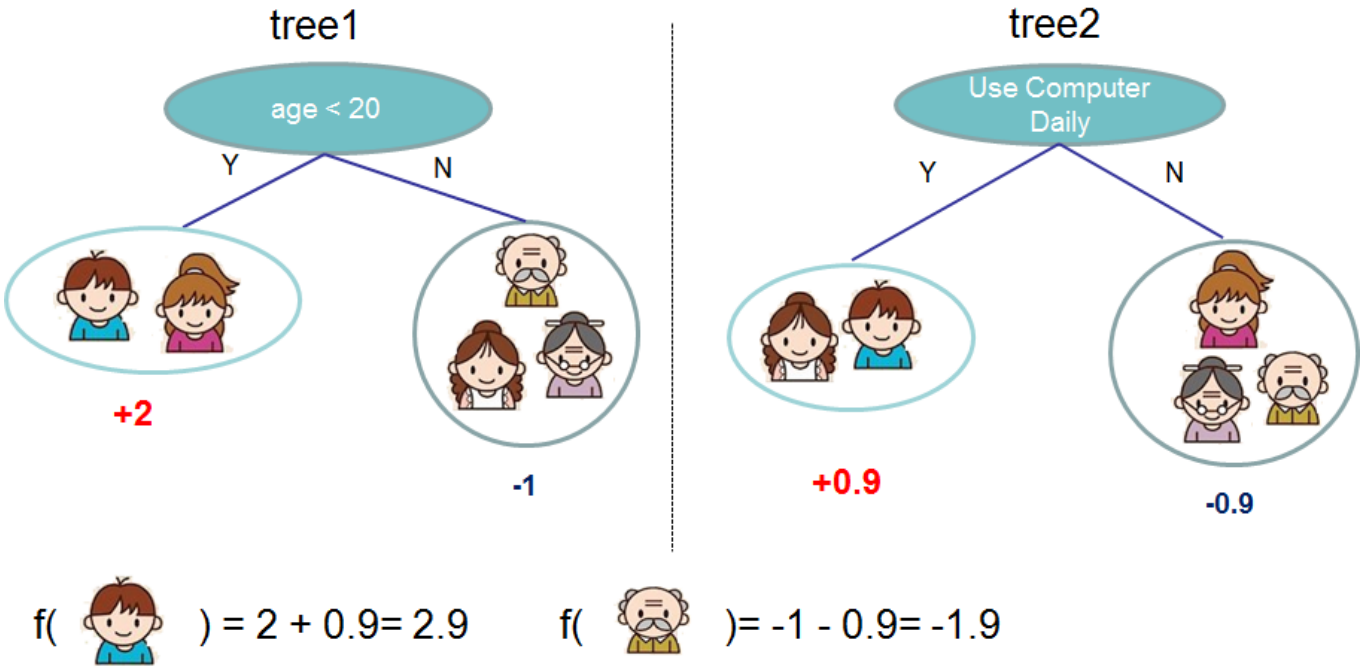
Table of Contents

Data Processing

- In it's raw form, the data has a number of categorical variables e.g. State $\in$ {Kansas, Nebraska, Oklahoma, Missouri, Iowa}. We one-hot-encode these variables. Furthermore we scale all the variables.
- Columns with repeated information (e.g. State) or only one level (e.g. Country) are dropped.

Feature Selection via Gradient Boosting

We find the feature importances computed by an XGBoost model to deliver the most effective features for optimising overall model performance.



(Image taken from: `https://xgboost.readthedocs.io/en/latest/tutorials/model.html`.)

Feature Selection via Gradient Boosting

- Compute feature importance for single trees based on gini impurity values at splitting nodes.
- Determine how much splitting on each feature reduced impurity across all splits in a tree.
- Sum up feature importance values of all trees and divide by total number of trees.
- This delivers a ranked list of features, ordered according to how useful each feature was in constructing trees within a model.

KNN-Regression

We opt to use a KNN-regression model for it's simplicity and explainability. KNN-regression works by finding the closest K data-points to a test-point and takes a weighted average of the output value of these closest K points. These weights are inversely proportional to the distance from the test-point.

We calculate K, the number of neighbours, via cross-validation. We get K=16

Pipeline

Our pipeline:

1. Removed features which aren't ethically sound
2. Utilised XGBoost to perform feature selection - this simplified the data and therefore our models
3. Find K via cross validation
4. Perform KNN-Regression with these features and K

A very simple, robust approach which gives a low MAE. It is highly interpretable.

MAE

We achieved an MAE of 75.2. We provide an overview of our analysis in the following Jupyter notebook...

Rejected/Failed Attempts

Feature selection:

- LASSO - promised to robustly detect complex behaviour between features. However, among other features, it suggested that vehicle class wasn't important.
- Permutation feature importance - seemed to output similar features to XGBoost's feature selection. However, a slow algorithm.
- MLP-VAE - not interpretable but promising. However, not enough data to learn properly. Might perform better on larger datasets and using CNN architectures for encoders/decoders.

Regression model:

- XGBoost - performed worse on the test set.