Project Presentation

Machine Learning 2

Artur Skowroński, Szymon Socha

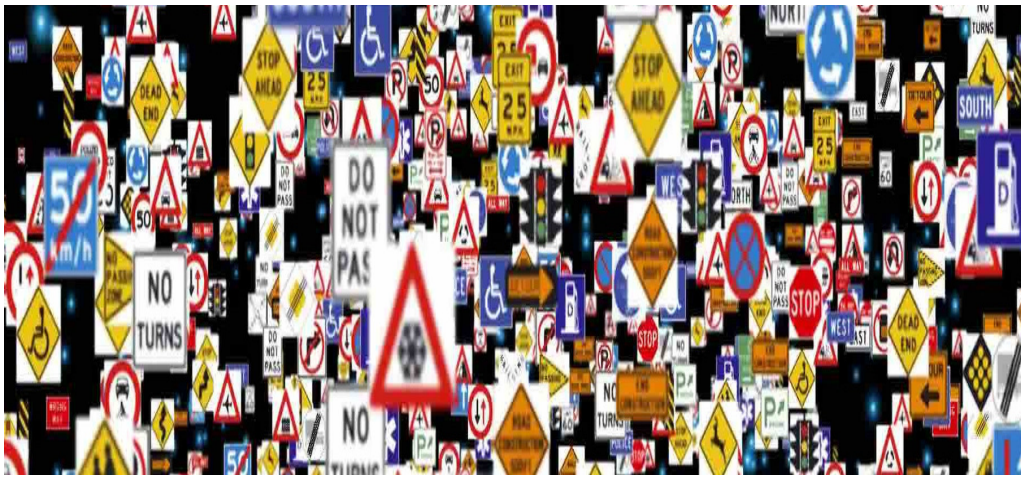
January 22, 2023

UNIVERSITY OF WARSAW | FACULTY OF ECONOMIC SCIENCES

Table of Contents

Dataset Description

The dataset contains 50,000 photos of 43 different German road signs.
Dataset is available on Kaggle (here).



Data preparation

At the beginning, we conducted an initial data exploration, which allowed us to conclude that we have an unbalanced dataset. In addition, we have presented sample photos available in the dataset to show that not all signs seem to be very obvious. As part of the image preprocessing, we took the following steps:

- photos were scaled to a resolution of 32x32
- converted to grayscale
- pixels were flattend and normalized to values between 0-1

Due to the lack of a validation set, it was decided to divide the training set in the proportion of 80%/20%. Thanks to this, we have obtained 3 independent harvests. Then, depending on the algorithm used, we additionally transformed the dependent variable.

Modelling part

We used 3 different classification algorithms for modelling:

- Random Forest
- LightGBM
- CNN (based on LeNet-5 architecture)

Accuracy was used as a performance metric, but a confusion matrix and a classification report were also built for each model.

In the case of the first two algorithms, Randomized Search Cross Validation was used to find the optimal hyperparameters.

For neural network, it was decided to use different convolution values and monitored the value of the loss function of the validation set.

Results

Both the selection of hyperparameters and the training itself took a long time, therefore, when checking, we recommend using the models already saved by us with the appropriate parameters.

- Random Forest
  Validation: 61%; Test: 51%
- LightGBM
  Validation: 86%; Test: 68%
- CNN (based on LeNet-5 architecture)
  Validation: 97%; Test: 89%

Possibilities to improve the model:

- Data augmentation
- Creating a larger CNN architecture
- Use of transfer learning*

Dataset Description

The dataset contains information about the specifications of laptops and their prices given in Euros. The dataset contains 1320 observations and 13 columns describing the specifications of the laptops:

- **Company** - *string* - Laptop Manufacturer
- **Product** - *string* - Brand and Model
- **TypeName** - *string* -Type (Notebook, Ultrabook, Gaming, etc.)
- **Inches** - *numeric*- Screen Size
- **ScreenResolution** - *string* - Screen Resolution
- **Cpu** - *string* - Central Processing Unit (CPU)
- **Ram** - *string* - Laptop RAM
- **Memory** - *string* - Hard Disk / SSD Memory
- **GPU** - *string* - Graphics Processing Units (GPU)
- **OpSys** - *string* - Operating System
- **Weight** - *string* - Laptop Weight
- **Price euros** - *numeric* - Price (Euro)

Dataset is available on Kaggle (here).

Data preparation

In order to prepare the data, we perform **EDA**. We also perform **feature engineering**:

- We extract the screen type and resolution from the `ScreenResolution` column
- From the `GPU` column, we extract the processor model and processor clocking
- From the `Memory` column, we extract whether the laptop has two drives or one, what type and what capacity

We note that the variable `Price euros` is right-skewed. We logarithmize this variable to give it a distribution closer to the normal distribution.

After feature engineering, we divide the dataset into a training set and a test set. We use the test dataset later **only** for performance comparison of finished models.

On the training set, we remove one **outlier**. Originally in the dataset there is one laptop with 64GB RAM, which looks like a very powerful, expensive, gaming laptop. We remove this observation.

Random Forest

We chose Random Forest as the first algorithm. Even when launching the algorithm with default parameters, the results are promising.
The result obtained on the validation set without hyperparameter tuning:
R^2: 90.89%

We use **Randomized Search Cross Validation** to hyperparameter tuning. We improve the model's performance to:
R^2: 92.38%

Final Random Forest Regressor model:

```
'n_estimators': 400, 'min_samples_split': 2, 'min_samples_leaf': 1,
'max_features': 'sqrt', 'max_depth': None, 'bootstrap': False
```

XGBoost

The next algorithm we use is Extreme Gradient Boosting. The result we get on the validation set is very similar to that obtained by Random Forest (slightly worse). The result obtained on the validation set without hyperparameter tuning:
R^2: 90.84%

We use **Randomized Search Cross Validation** to hyperparameter tuning. Thus, we improve the model and obtain a performance better than Random Forest. Obtained score:
R^2: 92.58%

Final XGBoost model:
`‘colsample_bytree‘ = 0.6993051668482431, ‘eta‘ = 0.109183012355693 73, ‘gamma‘ = 0.0596570836604370 94, ‘lambdaX‘ = 0.785792195852771 4, ‘max_depth‘ = 9, ‘min_child_weight‘ = 22.189028104694653, ‘subsample‘ = 0.4923455000631808`

Neural Networks

The last model we use is Neural Networks. We build a pipeline with `StandardScaler()` together with Random Search Cross Validation. In order to avoid **data leakage** we perform scaling on each fold. It is worth mentioning that training the Neural Networks among the three models took the most time - **more than 3 hours**. The result obtained on the validation set without hyperparameter tuning:
R^2: 79.47%

After tuning the hyperparameters, we get a very good model (even too good), which suggests that the model is overfitted (later compare how all models perform on the test set). Obtained score:
R^2: 96.6%

Final Neural Networks model:

```
'nn3': 300, 'nn2': 400, 'nn1': 1400, 'nl3': 2, 'nl2': 0, 'nl1': 0,
'momentum': 0.9, 'lr': 0.001, 'l2': 0.1, 'l1': 0, 'input_shape':
17, 'dropout': 0.3, 'decay': 0, 'act': 'relu'
```

Performance Comparison

Finally, we compare the performance of previously saved models by doing predictions on a test dataset, **unseen** by either model.

Test data results:
Random Forest R^2: **93.3%**
XGBoost R^2: **91.6%**
Neural Network R^2: **86.5%**

The model for which we get the best results is **Random Forest**.

Below is a table with a comparison of model performance with actual values for random observations.

Random Forest	XGBoost	Neural Network	Real Values
1247	1364	1264	1672
1144	1237	1132	1149
521	448	392	499
899	825	899	899
1484	1312	1222	1244

Summary