Boston University

CS 699 A2 – Data Mining

Project Assignment

Price classification on Massachusetts housing data

Authors: Jan Allemann

Mike Zhong

Due: April 06th, 2020

Content

[1. Data mining goal 1](#_Toc36990605)

[2. Description of dataset 1](#_Toc36990606)

[3. Tools 2](#_Toc36990607)

[4. Attribute Selection 2](#_Toc36990608)

[5. Mining algorithms 2](#_Toc36990609)

[6. Conclusion 2](#_Toc36990610)

# Data mining goal

The class attribute is the PRICE, which is a continuous dollar value variable. To convert this into categorical ordinal data, we will bin the list prices using an equal width binning process. The widths of the bins is yet to be determined. The application of this classification will be suggest list prices for sellers.

The goal of the project is, to predict the PRICE category

# Description of dataset

The dataset is obtained from <https://www.redfin.com>. Redfin is a commercial real estate listing company that aggregates real estate data for consumer and business use. This data set was obtained by performing a search with no filters on the Boston, Massachusetts area and expanding the scope until a sufficient number of listings were included. The data set contains 350 tuples with 27 attributes each. The attributes are described in the following table.

|  |  |  |
| --- | --- | --- |
| Attribute | Format | Description |
| SALE.TYPE | Categorical | They are all MLS listing |
| SOLD.DATE | Datetime | N/A |
| PROPERTY.TYPE | Categorical | Categorical nominal |
| ADDRESS | String | House number and street name |
| CITY | String | Categorical nominal |
| STATE.OR.PROVINCE | String | Categorical nominal |
| ZIP.OR.POSTAL.CODE | String | Categorical nominal |
| PRICE | Integer | Continuous float |
| BEDS | Integer | Discrete integers |
| BATHS | Integer | Discrete ordinal |
| LOCATION | String | Categorical nominal |
| SQUARE.FEET | Integer | Discrete integers |
| LOT.SIZE | Integer | Discrete integers |
| YEAR.BUILT | Integer | Discrete integers |
| DAYS.ON.MARKET | Integer | Discrete integers |
| X..SQUARE.FEET | Integer | Continuous float |
| HOA.MONTH | Integer | Discrete integers |
| STATUS | Categorical | Categorical nominal |
| NEXT.OPEN.HOUSE.START.TIME | Datetime | Mostly empty |
| NEXT.OPEN.HOUSE.END.TIME | Datetime | Mostly empty |
| URL | String | URL to home on redfin |
| SOURCE | Categorical | Categorical nominal |
| MLS | Integer | Unique ID |
| FAVORITE | Binary | User specific variable |
| INTERESTED | Binary | User specific variable |
| LATITUTDE | Numerical | Continuous float |
| LONGITUDE | Numerical | Continuous float |

# Tools

The different steps of the data mining process are performed in multiple tools. Following table indicates the workflow:

|  |  |  |
| --- | --- | --- |
| **Step** | **Tool** | **Reason** |
| Data cleaning | Python | Easy and fast visualization of all attributes  Easy and fast replacement of missing values |
| Attribute Selection | WEKA | Integrated Attribute Selection Methods available |
| Data Mining | Python SciKit Learn | Scalability of the process. Looping over different algorithms and subsets. |
| Reporting | MS Word |  |

Description of SciKitLearn and algorithms

Python’s scikit-learn library provides a uniform interface for training and predicting with several different types of models. We have used the following models in our project:

* RandomForestClassifier:
  + A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement
  + All default model parameters were used
* MLPClassifier
  + Multi-layer Perceptron classifier. This model optimizes the log-loss function using LBFGS or stochastic gradient descent.
  + All default model parameters were used
* GradientBoostingClassifier
  + Gradient Boosting for classification. GB builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage n\_classes\_ regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. Binary classification is a special case where only a single regression tree is induced.
  + All default model parameters were used
* LogisticRegression
  + Logistic Regression (aka logit, MaxEnt) classifier. In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi\_class’ option is set to ‘ovr’, and uses the cross-entropy loss if the ‘multi\_class’ option is set to ‘multinomial’. (Currently the ‘multinomial’ option is supported only by the ‘lbfgs’, ‘sag’, ‘saga’ and ‘newton-cg’ solvers.)
  + All default model parameters were used
* DummyClassifier
  + DummyClassifier is a classifier that makes predictions using simple rules. This classifier is useful as a simple baseline to compare with other (real) classifiers. Do not use it for real problems.
  + All default model parameters were used

# Preprocessing

## Dropping columns

The distribution for each attribute is plotted. There are multiple columns (i.e. ‘Sale Type’ Figure 1) containing the same value in each tuple.

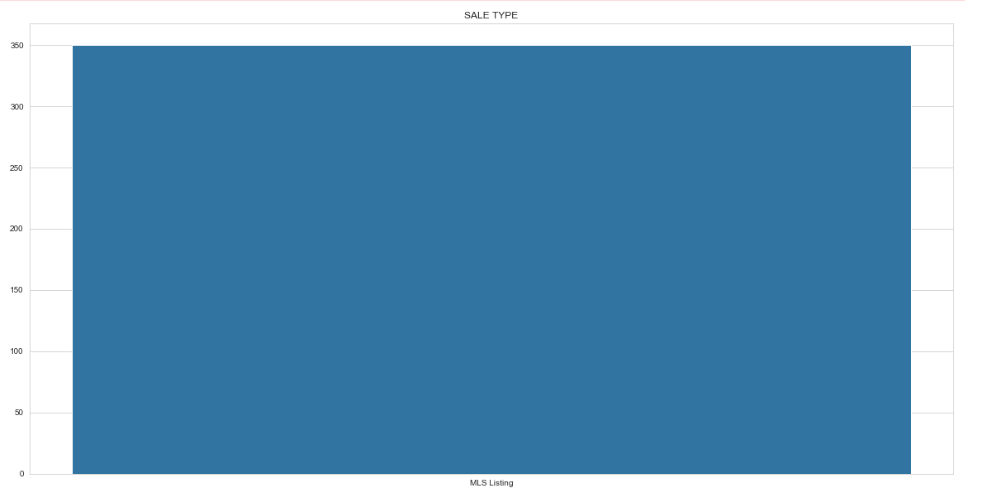


Figure 1: Distribution of the sale type column

Such columns don’t add any information for the classification and therefore the following columns are dropped:

* INTERESTED
* FAVORITE
* STATUS
* SALE TYPE

One column has a unique value in each row and would therefore lead to overfitting on the training set. The column is dropped.

* MLS#

Multiple columns are either mostly empty or are not related to the price and are therefore dropped.

* URL
* SOURCE
* SOLD DATE
* NEXT OPEN HOUSE START TIME
* NEXT OPEN HOUSE END TIME

## Missing Values and Outliers

The column HOA.MONTH describes the monthly HOA-fee for each listing. For listings without any HOA-fee the value should be set to 0 instead of NaN.

By plotting the LOT.SIZE against SQUARE.FOOT we can detect an outlier that exceeds the 1,5\*IQR limit and must be removed.

|  |  |
| --- | --- |
| LOT.SIZE | Figure 2: LOT.SIZE against SQUARE.FOOT including outliers |
|  | SQUARE.FOOT |

On the remaining LOT.SIZE, SQUARE.FOOT pairs, we can do a linear regression to interpolate missing values in both columns.

|  |  |
| --- | --- |
| LOT.SIZE | Figure 3: LOT.SIZE against SQUARE.FOOT linear regression |
|  | SQUARE.FOOT |

If the attributes BED or BATHS are not containing any value, the tuple can be dropped. These tuples most likely belong to empty lots and are therefore not of interest.

## Binning

The continuous, numeric target attribute PRICE must be converted into a nominal attribute to create a classification problem. As Figure 4 indicates, the PRICE attribute is highly skewed. A equal width binning would lead to many almost empty bins at the high end of the range.

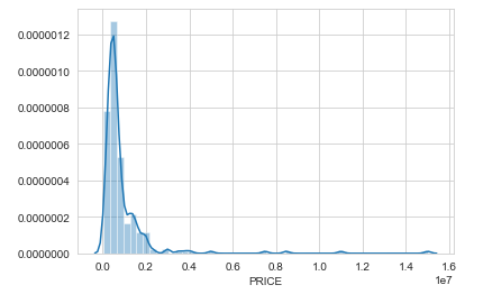


Figure 4: Distribution of PRICE

Therefore, an equal depth binning is performed on PRICE to transform it into a categorical attribute named PRICE\_BIN\_ALT.

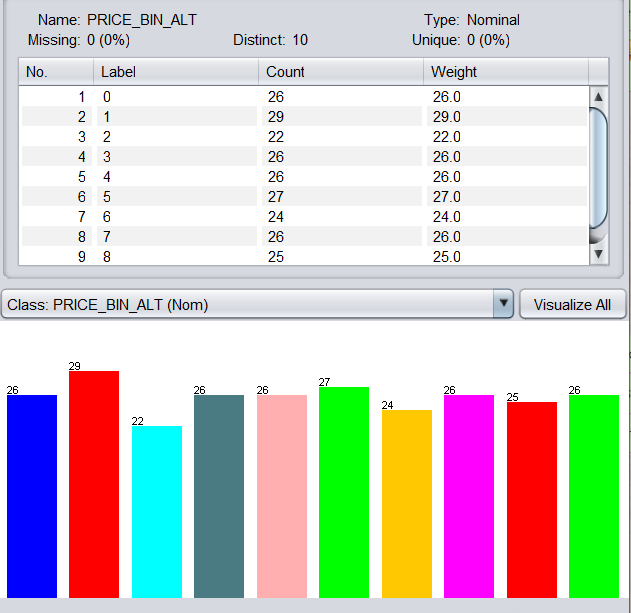


Figure 5: Equal size binning on PRICE

# Attribute Selection

4 different attribute selection methods are chosen from the WEKA library. The following screenshots show the results of each of the algorithms as well as the self-chosen subset.

|  |  |
| --- | --- |
| CfsSubset  Figure 6: Attribute Selection CfsSubset | Top five attributes:   * Property Type * Address * City * State or Province * ZIP or Postal Code |
| OneRAttributeEval  A screenshot of a cell phone  Description automatically generated  Figure 7: Attribute Selection OneRAttributeEval | Top five attributes:   * Square feet * Location * City * Baths * ZIP or postal code |
| ClassifierAttributeEval    Figure 8: Attribute Selection ClassifierAttributeEval | Top five attributes:   * Longitude * State or Province * ZIP or postal code * City * Baths |
| InfoGainAttributeEval    Figure 9: Attribute Selection InfoGainAttributeEval | Top five attributes:   * Address * Location * ZIP or postal code * City * Baths |
| Self-chosen subset  Based on data exploration and visualization in WEKA, a subset of attributes is selected. Mostly numeric continuous attributes are chosen.    Figure 10: Corrleation for the self-chosen subset | Top six attributes:   * Property Type * ZIP or postal code * Beds * Baths * Square feet * Lot size |

# Mining algorithms

5 data mining algorithms are chosen to model on each of the 5 subsets described in chapter 5.

* RandomForestClassifier
* MLPClassifier
* GradientBoostingClassifier
* LogisticRegression
* DummyClassifier

The combination of the data mining algorithm and attribute selection methods leads to 25 different models. The following pages show detailed results of all the 25 models.

The F-score takes both TP as well as FP into account and is considered as the most meaningful measure. The best model is therefore chosen based on the F-score.

The OneR Attribute Selection Method performs best in average.

The RandomForest Data Mining algorithm performs best in average

The combination from Classifier Attribute Selection and RandomForest performs best out of all models.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| F-Score Values | Random Forest | MLP Classifier | Gradient Boosting | Logistic Regression | Dummy Classifier | Mean |
| CfsSubset | 0.19 | 0.17 | 0.15 | 0.19 | 0.11 | 0.16 |
| OneR | 0.33 | 0.32 | 0.31 | 0.15 | 0.18 | 0.26 |
| Classifier | 0.35 | 0.18 | 0.30 | 0.23 | 0.08 | 0.23 |
| InfoGain | 0.30 | 0.19 | 0.21 | 0.25 | 0.11 | 0.21 |
| Self-chosen | 0.30 | 0.04 | 0.28 | 0.09 | 0.12 | 0.17 |
| Mean | 0.29 | 0.18 | 0.25 | 0.18 | 0.12 | 0.21 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| CfsSubset | RandomForest | 0.19 | 0.4, 0.42, 0.11, 0.38, 0.5, 0.11, 0.29, 0.0, 0.0, 0.57 | 0.23 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [0 3 1 2 1 0 0 0 0 0]  [3 2 1 1 1 0 0 0 0 1]  [1 0 0 3 2 1 1 0 0 0]  [1 0 0 0 2 0 1 0 0 0]  [0 0 0 2 6 1 0 0 0 0]  [0 2 0 1 1 1 2 0 0 0]  [1 1 1 0 3 3 1 0 1 0]  [0 0 0 0 2 1 3 1 0 4]  [1 0 0 0 1 0 1 0 0 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| CfsSubset | MLP Classifier | 0.17 | 0.4, 0.57, 0.11, 0.25, 0.0, 0.11, 0.29, 0.0, 0.0, 0.57 | 0.21 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [0 4 2 1 0 0 0 0 0 0]  [3 3 1 1 0 0 0 0 0 1]  [1 2 1 2 0 1 1 0 0 0]  [0 1 0 2 0 0 1 0 0 0]  [0 3 1 4 0 1 0 0 0 0]  [0 2 0 1 0 1 2 1 0 0]  [0 2 1 0 1 4 1 0 2 0]  [0 2 0 0 0 0 3 2 0 4]  [0 1 0 0 1 0 1 0 0 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| CfsSubset | Gradient Boosting | 0.15 | 0.0, 0.43, 0.0, 0.25, 0.5, 0.0, 0.29, 0.0, 0.0, 0.57 | 0.17 |
| Confusion matrix:  [[0 2 1 0 2 0 0 0 0 0]  [0 3 1 0 3 0 0 0 0 0]  [2 2 0 1 4 0 0 0 0 0]  [0 0 0 2 4 1 1 0 0 0]  [0 0 0 1 2 0 1 0 0 0]  [0 0 0 0 9 0 0 0 0 0]  [0 2 0 1 1 0 2 1 0 0]  [0 0 1 0 4 3 1 0 2 0]  [0 0 0 0 6 0 0 1 0 4]  [0 0 0 0 3 0 0 0 0 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| CfsSubset | LogisticRegression | 0.19 | 0.2, 0.43, 0.11, 0.38, 0.5, 0.11, 0.29, 0.0, 0.0, 0.71 | 0.23 |
| Confusion matrix:  [[1 2 1 0 1 0 0 0 0 0]  [0 3 1 2 1 0 0 0 0 0]  [2 2 1 1 2 0 0 0 0 1]  [0 0 0 3 3 1 1 0 0 0]  [1 0 0 0 2 0 1 0 0 0]  [0 0 1 1 6 1 0 0 0 0]  [0 3 0 0 1 1 2 0 0 0]  [1 1 0 0 2 4 1 0 1 1]  [0 0 0 0 2 1 0 1 0 7]  [1 0 0 0 1 0 0 0 0 5]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| CfsSubset | DummyClassifier | 0.11 | 0.2, 0.0, 0.0, 0.25, 0.25, 0.0, 0.29, 0.18, 0.09, 0.0 | 0.12 |
| Confusion matrix:  [[1 0 1 0 1 1 1 0 0 0]  [0 0 2 2 1 1 1 0 0 0]  [4 0 0 0 0 3 1 0 1 0]  [0 0 0 2 0 1 3 1 1 0]  [0 0 0 1 1 1 1 0 0 0]  [1 0 4 0 1 0 0 1 2 0]  [1 1 1 0 0 1 2 0 0 1]  [0 1 0 2 2 1 1 2 1 1]  [2 2 0 1 2 1 0 0 1 2]  [0 1 1 0 2 1 1 1 0 0]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| OneR | RandomForest | 0.33 | 0.6, 0.57, 0.0, 0.25, 0.25, 0.33, 0.43, 0.09, 0.64, 0.43 | 0.35 |
| Confusion matrix:  [[3 2 0 0 0 0 0 0 0 0]  [2 4 1 0 0 0 0 0 0 0]  [1 2 0 2 1 3 0 0 0 0]  [2 2 1 2 1 0 0 0 0 0]  [0 0 0 1 1 0 1 1 0 0]  [0 0 1 0 3 3 1 1 0 0]  [0 1 0 0 0 1 3 1 1 0]  [0 0 1 0 1 4 2 1 2 0]  [0 0 0 0 0 1 0 1 7 2]  [0 0 0 0 0 1 0 1 2 3]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| OneR | MLPClassifier | 0.32 | 0.2, 0.29, 0.22, 0.25, 0.25, 0.22, 0.29, 0.27, 0.46, 0.57 | 0.31 |
| Confusion matrix:  [[1 2 1 0 1 0 0 0 0 0]  [2 2 3 0 0 0 0 0 0 0]  [1 2 2 1 1 2 0 0 0 0]  [2 0 2 2 1 0 1 0 0 0]  [0 0 0 1 1 0 1 1 0 0]  [0 4 1 0 1 2 0 1 0 0]  [0 1 0 1 0 1 2 1 1 0]  [0 0 1 0 1 2 1 3 1 2]  [0 1 0 1 0 0 0 2 5 2]  [0 1 0 0 1 0 0 1 0 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| OneR | GradientBoosting | 0.31 | 0.4, 0.71, 0.11, 0.38, 0.25, 0.11, 0.29, 0.0, 0.55, 0.57 | 0.32 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [1 5 1 0 0 0 0 0 0 0]  [0 5 1 1 2 0 0 0 0 0]  [0 2 0 3 1 1 1 0 0 0]  [0 1 0 1 1 0 1 0 0 0]  [0 0 1 0 7 1 0 0 0 0]  [0 2 0 2 0 1 2 0 0 0]  [0 0 1 0 3 4 1 0 2 0]  [0 0 0 0 1 0 0 1 6 3]  [0 0 0 0 1 0 0 0 2 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| OneR | LogisticRegression | 0.15 | 0.2, 0.14, 0.0, 0.25, 0.0, 0.0, 0.43, 0.0, 0.09, 0.71 | 0.17 |
| Confusion matrix:  [[1 3 1 0 0 0 0 0 0 0]  [4 1 1 1 0 0 0 0 0 0]  [2 1 0 1 3 1 1 0 0 0]  [2 1 1 2 1 0 1 0 0 0]  [0 1 0 1 0 0 1 0 1 0]  [0 0 0 2 5 0 0 0 1 1]  [0 1 0 0 0 1 3 1 1 0]  [0 0 0 1 1 5 0 0 3 1]  [1 0 0 3 2 1 0 1 1 2]  [0 0 0 0 0 0 1 0 1 5]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| OneR | DummyClassifier | 0.18 | 0.0, 0.0, 0.33, 0.25, 0.25, 0.0, 0.29, 0.09, 0.09, 0.0 | 0.13 |
| Confusion matrix:  [[0 1 0 1 0 1 0 0 1 1]  [0 0 0 1 2 0 1 0 1 2]  [0 0 3 3 0 1 1 1 0 0]  [0 1 0 2 1 0 0 3 1 0]  [0 0 0 0 1 0 0 0 1 2]  [0 0 1 1 1 0 3 2 0 1]  [0 1 0 2 0 0 2 0 0 2]  [2 0 1 0 1 3 2 1 1 0]  [0 0 0 4 2 0 2 2 1 0]  [0 1 2 0 1 1 0 1 1 0]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Classifier | RandomForest | 0.35 | 0.4, 0.43, 0.11, 0.37, 0.25, 0.44, 0.28, 0.09, 0.45, 0.85 | 0.36 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [1 3 2 1 0 0 0 0 0 0]  [2 2 1 3 0 1 0 0 0 0]  [0 0 0 3 3 1 1 0 0 0]  [0 0 0 1 1 0 1 1 0 0]  [0 1 1 0 2 4 1 0 0 0]  [0 1 0 1 1 1 2 1 0 0]  [0 0 1 0 1 4 2 1 2 0]  [0 0 0 0 1 0 1 2 5 2]  [0 0 0 0 0 0 0 0 1 6]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Classifier | MLPClassifier | 0.18 | 0.4, 0.43, 0.11, 0.25, 0.0, 0.11, 0.29, 0.0, 0.0, 0.85 | 0.22 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [1 3 2 1 0 0 0 0 0 0]  [3 2 1 1 2 0 0 0 0 0]  [3 0 1 2 0 1 1 0 0 0]  [1 0 0 1 0 0 1 0 0 1]  [5 0 1 1 1 1 0 0 0 0]  [0 2 0 1 0 1 2 1 0 0]  [0 0 1 0 3 4 1 0 2 0]  [1 0 0 0 0 0 1 2 0 7]  [0 0 0 0 1 0 0 0 0 6]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Classifier | GradientBoosting | 0.3 | 0.4, 0.57, 0.11, 0.375, 0.5, 0.11, 0.29, 0.0, 0.45, 0.57 | 0.31 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [1 4 2 0 0 0 0 0 0 0]  [0 5 1 1 2 0 0 0 0 0]  [0 0 1 3 2 1 1 0 0 0]  [0 0 0 1 2 0 1 0 0 0]  [0 0 1 1 6 1 0 0 0 0]  [0 2 0 1 0 1 2 1 0 0]  [0 0 1 0 3 4 1 0 2 0]  [0 0 0 0 2 0 1 1 5 2]  [0 0 0 0 1 0 0 0 2 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Classifier | LogisticRegression | 0.23 | 0.4, 0.43, 0.11, 0.375, 0.0, 0.22, 0.28, 0.09, 0.09, 0.57 | 0.24 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [2 3 1 1 0 0 0 0 0 0]  [2 2 1 2 2 0 0 0 0 0]  [3 0 0 3 1 0 1 0 0 0]  [1 0 0 1 0 1 1 0 0 0]  [0 0 0 2 5 2 0 0 0 0]  [0 2 0 1 0 1 2 1 0 0]  [0 0 0 0 1 6 1 1 2 0]  [0 0 0 3 1 1 2 0 1 3]  [0 0 0 0 0 0 1 0 2 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Classifier | DummyClassifier | 0.08 | 0.0, 0.14, 0.11, 0.125, 0.0, 0.11, 0.0, 0.09, 0.0, 0.14 | 0.08 |
| Confusion matrix:  [[0 1 0 1 1 0 1 0 1 0]  [1 1 1 1 2 1 0 0 0 0]  [0 0 1 1 1 3 1 1 1 0]  [0 1 0 1 0 0 2 1 0 3]  [0 0 0 0 0 1 1 0 0 2]  [2 1 1 1 1 1 0 0 1 1]  [0 2 1 0 0 0 0 1 2 1]  [3 2 1 0 0 0 1 1 2 1]  [0 0 0 1 4 2 0 2 0 2]  [0 0 1 0 0 1 3 1 0 1]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| InfoGain | RandomForest | 0.3 | 0.4, 0.43, 0.11, 0.38, 0.0, 0.55, 0.28, 0.182, 0.27, 0.57 | 0.32 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [2 3 2 0 0 0 0 0 0 0]  [2 2 1 2 0 2 0 0 0 0]  [2 1 0 3 0 1 1 0 0 0]  [1 0 0 1 0 0 1 1 0 0]  [0 2 1 0 0 5 0 1 0 0]  [0 1 0 1 0 2 2 0 1 0]  [0 0 0 0 0 5 1 2 2 1]  [0 0 0 0 0 2 0 1 3 5]  [0 0 0 0 0 0 0 0 3 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| InfoGain | MLPClassifier | 0.19 | 0.4, 0.43, 0.0, 0.25, 0.0, 0.11, 0.29, 0.0, 0.18, 0.71 | 0.22 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [2 3 2 0 0 0 0 0 0 0]  [4 2 0 1 0 0 0 0 0 2]  [0 0 1 2 0 1 1 0 0 3]  [0 0 0 1 0 0 1 0 0 2]  [0 0 1 1 0 1 0 0 0 6]  [0 2 0 2 0 1 2 0 0 0]  [0 0 1 0 1 4 1 0 2 2]  [0 0 0 0 0 0 0 2 2 7]  [0 0 0 0 1 0 0 0 1 5]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| InfoGain | GradientBoosting | 0.21 | 0.4, 0.57, 0.0, 0.25, 0.25, 0.11, 0.29, 0.0, 0.27, 0.42 | 0.23 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [2 4 1 0 0 0 0 0 0 0]  [3 4 0 1 1 0 0 0 0 0]  [2 1 0 2 1 1 1 0 0 0]  [1 0 0 1 1 0 1 0 0 0]  [0 4 0 0 4 1 0 0 0 0]  [0 2 0 1 0 1 2 0 1 0]  [0 0 1 0 4 4 1 0 1 0]  [0 1 0 0 2 0 0 0 3 5]  [0 0 0 0 1 0 0 0 3 3]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| InfoGain | LogisticRegressoin | 0.25 | 0.4, 0.43, 0.11, 0.38, 0.0, 0.22, 0.29, 0.0, 0.18, 0.86 | 0.27 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [2 3 1 1 0 0 0 0 0 0]  [3 2 1 1 2 0 0 0 0 0]  [2 0 0 3 1 1 1 0 0 0]  [1 0 0 1 0 1 1 0 0 0]  [0 0 0 2 5 2 0 0 0 0]  [0 2 0 1 0 2 2 0 0 0]  [0 0 1 0 1 6 1 0 2 0]  [0 0 0 2 1 1 0 1 2 4]  [0 0 0 0 0 0 0 0 1 6]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| InfoGain | DummyClassifier | 0.11 | 0.0, 0.29, 0.11, 0.125, 0.25, 0.22, 0.29, 0.0, 0.0, 0.0 | 0.12 |
| Confusion matrix:  [[0 1 1 0 1 0 0 0 2 0]  [1 2 0 1 1 1 0 0 1 0]  [2 0 1 3 0 0 0 1 1 1]  [2 0 1 1 1 0 0 0 0 3]  [0 0 0 0 1 0 1 0 0 2]  [0 2 1 0 1 2 1 1 1 0]  [0 0 0 1 0 3 2 1 0 0]  [0 1 1 1 3 2 1 0 0 2]  [2 3 0 1 2 2 0 0 0 1]  [0 0 1 1 0 2 1 1 1 0]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Self-chosen | RandomForest | 0.30 | 0.4, 0.29, 0.0, 0.25, 0.0, 0.44, 0.29, 0.36, 0.36, 0.42 | 0.29 |
| Confusion matrix:  [[2 2 1 0 0 0 0 0 0 0]  [2 2 3 0 0 0 0 0 0 0]  [1 1 0 3 2 0 2 0 0 0]  [1 3 1 2 0 0 0 0 1 0]  [0 1 1 0 0 0 1 1 0 0]  [0 1 0 0 3 4 0 1 0 0]  [0 1 0 0 0 1 2 1 2 0]  [0 0 1 0 0 3 2 4 1 0]  [1 0 0 0 1 1 0 1 4 3]  [0 0 0 0 0 1 1 0 2 3]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Self-chosen | MLPClassifier | 0.04 | 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.27, 0.0, 0.14 | 0.10 |
| Confusion matrix:  [[0 0 0 0 3 0 0 2 0 0]  [0 0 0 0 4 0 0 3 0 0]  [0 0 0 0 5 0 0 4 0 0]  [0 0 0 0 4 0 0 3 0 1]  [0 0 0 0 4 0 0 0 0 0]  [0 0 0 0 4 0 0 2 0 3]  [0 1 0 0 3 0 0 3 0 0]  [0 1 0 0 6 0 0 3 0 1]  [0 0 0 0 6 0 0 5 0 0]  [0 0 0 0 3 0 0 3 0 1]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Self-chosen | GradientBoosting | 0.28 | 0.2, 0.43, 0.11, 0.375, 0.5, 0.11, 0.29, 0.0, 0.54, 0.57 | 0.29 |
| Confusion matrix:  [[1 2 1 0 1 0 0 0 0 0]  [2 3 2 0 0 0 0 0 0 0]  [1 2 1 3 1 0 1 0 0 0]  [1 0 1 3 1 1 1 0 0 0]  [0 0 0 1 2 0 1 0 0 0]  [0 2 1 0 5 1 0 0 0 0]  [0 2 0 1 0 1 2 1 0 0]  [0 0 1 0 3 4 1 0 2 0]  [1 1 0 0 1 0 0 1 6 1]  [0 0 0 0 0 0 1 0 2 4]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Self-chosen | LogisticRegression | 0.09 | 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 1.0 | 0.09 |
| Confusion matrix:  [[ 0 0 0 0 0 0 0 0 0 5]  [ 0 0 0 0 0 0 0 0 0 7]  [ 0 0 0 0 0 0 0 0 0 9]  [ 0 0 0 0 0 0 0 0 0 8]  [ 0 0 0 0 0 0 0 0 0 4]  [ 0 0 0 0 0 0 0 0 2 7]  [ 0 0 0 0 0 0 0 0 1 6]  [ 0 0 0 0 0 0 0 0 1 10]  [ 0 0 0 0 0 0 0 0 0 11]  [ 0 0 0 0 0 0 0 0 0 7]] | | | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attribute Selection** | **Data Mining algorithm** | **F-score** | **TPR for each bin 1-10** | **Accuracy** |
| Self-chosen | DummyClassifier | 0.12 | 0.2, 0.29, 0.22, 0.125, 0.0, 0.0, 0.0, 0.09, 0.09, 0.14 | 0.12 |
| Confusion matrix:  [[1 0 0 1 0 1 1 0 1 0]  [0 2 1 0 1 1 0 0 1 1]  [1 2 2 0 0 1 0 0 1 2]  [2 1 0 1 1 0 2 0 1 0]  [0 0 0 1 0 1 2 0 0 0]  [1 0 0 1 0 0 1 1 3 2]  [0 1 0 0 3 1 0 1 0 1]  [4 0 1 1 2 0 1 1 0 1]  [0 1 2 0 2 0 2 1 1 2]  [2 0 1 0 0 0 2 0 1 1]] | | | | |

# Conclusion

The overall F-score of only 0.21 indicates, that there is no pattern in the dataset that classifies the price reliably. Although a lot of different algorithms were tested, none of them could reach an F-score above 0.35.

What went well:

The group decided to switch from WEKA to Python in order to automate the process. Since 25 different models had to be tested, this decision lead to higher efficiency and scalability. Thanks to the python code, the group is able to test multiple models with multiple attribute selections for further analysis.

What did not go well:

The raw dataset contains a lot of columns that are not meaningful in terms of PRICE classification. Therefore, a lot of columns had to be dropped.

The PRICE class attribute is highly skewed which makes the binning process more difficult. A classification of the logarithmic PRICE values could lead to better results.

Appendix