Towards Explainable Real Estate Valuation via Evolutionary Algorithms

Sebastian Angrick
Ben Bals
Niko Hastrich
Maximilian Kleissl
Jonas Schmidt
{firstname.lastname}@student.hpi.de
Hasso Plattner Institute, University of
Potsdam
Potsdam, Germany

Vanja Doskoč
Louise Molitor
Tobias Friedrich
{firstname.lastname}@hpi.de
Hasso Plattner Institute, University of
Potsdam
Potsdam, Germany

Maximilian Katzmann
maximilian.katzmann@kit.edu
Karlsruhe Institute of Technology
Karlsruhe, Germany

ABSTRACT

Human lives are increasingly influenced by algorithms, which therefore need to meet higher standards not only in accuracy but also with respect to explainability. This is especially true for high-stakes areas such as real estate valuation. Unfortunately, the methods applied there often exhibit a trade-off between accuracy and explainability.

One explainable approach is *case-based reasoning (CBR)*, where each decision is supported by specific previous cases. However, such methods can be wanting in accuracy. The unexplainable machine learning approaches are often observed to provide higher accuracy but are not scrutable in their decision-making.

In this paper, we apply *evolutionary algorithms (EAs)* to CBR predictors in order to improve their performance. In particular, we deploy EAs to the similarity functions (used in CBR to find comparable cases), which are fitted to the data set at hand. As a consequence, we achieve higher accuracy than state-of-the-art deep neural networks (DNNs), while keeping interpretability and explainability.

These results stem from our empirical evaluation on a large data set of real estate offers where we compare known similarity functions, their EA-improved counterparts, and DNNs. Surprisingly, DNNs are only on par with standard CBR techniques. However, using EA-learned similarity functions does yield an improved performance.

CCS CONCEPTS

• **Theory of computation** → **Evolutionary algorithms**; • **Computing methodologies** → *Neural networks*; *Probabilistic reasoning*.

KEYWORDS

Evolutionary algorithms, case-based reasoning, neural networks, real estate valuation

1 INTRODUCTION

Algorithms have been guiding human decision-making for quite some time, affecting the way we shop or how we select a movie to watch. Increasingly, they are employed in sensitive areas such as finance [10], medicine [31], and the legal system [18] as well as real estate valuations [7, 29]. In such delicate areas algorithms need to fulfill a plethora of requirements: They need to be accurate, trustworthy, and therefore scrutable to the user. Furthermore, they must not exacerbate existing biases or prejudices. In the context of real estate, these requirements are necessary as setting the price of a house or apartment correctly is of crucial importance to buyer, seller, and the bank providing financing. However, the algorithms applied there typically fall short of the posed requirements.

On the one hand, *explainable* algorithms outline their reasoning or provide evidence for the decisions they make and are therefore scrutable. A commonly used explainable approach is built on the following intuition: A human, who is about to make a decision, naturally compares the current situation to similar past experiences and their outcomes. This process is heavily used by professional real estate appraisers and can be formalized into the *case-based reasoning (CBR)* framework. There, real estate valuations are estimated as an average of past sales of comparable properties, which then serve as witnesses for the obtained prediction. This enables humans to check the predictions by looking at the basis on which the algorithm picked the price, allowing them to decide whether the prediction is trustworthy or if it exacerbates certain biases. In algorithmic applications, the central part of CBR is a *similarity function* that defines how similar the property to be valued is to a property whose value is known. Similarity functions that have been applied before include simple metrics on attribute vectors [4]. The similarity functions can also be learned from the data set at hand so they are fitted to it [14].

On the other hand, *unexplainable* approaches like *deep neural networks (DNNs)* have been used to valuate real estate properties [29], but they fail the trust requirement as they only provide an opaque output without any reasoning. With the inherent complexity of a DNN, one is not able to explain in retrospect how a certain output value is produced. Such methods are therefore liable to exacerbate existing biases [13, 20, 27, 28], which can only be discovered by statistical analysis of many predictions. A user cannot check an individual prediction for them. Nevertheless, DNNs have several advantages. Standard architectures can be applied to a variety of areas without relying on domain experts, which can save cost and time resources, and at the same time they are observed to provide high accuracy [29].

In this paper, we improve upon previous explainable methods by using *evolutionary algorithms (EAs)*, which are getting more and more popular in the field of artificial intelligence [35]. In particular, for CBR predictors we apply EAs to find good similarity functions that are fitted to the considered data set, allowing us to improve the accuracy of CBR while maintaining its explainability. Moreover, while previous approaches based on similarity functions do not scale to large data sets as they rely on maximum likelihood estimation and numerical methods [26], our approach reduces the complexity for a single prediction from linear to logarithmic. This improvement can be attributed to the use of geometric data structures that allows us to find similar objects efficiently.

To evaluate the resulting adaptation, we perform an empirical evaluation on a large data set of Japanese real estate offers [21], where we compare the different approaches. The results show that, surprisingly, the unexplainable DNNs do not beat known CBR approaches. However, our EA-assisted CBR method outperforms all others, yielding an explainable approach with higher accuracy.

In our analysis, we further identify an advantage that CBR has over DNNs. One of the fundamental rules in real estate valuation is that the value of a property is heavily influenced by its location. Hence, its surrounding properties are an important indicator for its price. While DNNs are a general framework that can be used to achieve good results without relying on domain knowledge, they seem to be unable to make these local relationships, as this information is only implicitly present in the weights learned during training. On the other hand, CBR makes explicit connections between properties, which turns out to intrinsically capture this important aspect. When providing DNNs with information about surrounding properties, they become much more accurate.

Structure of the Paper. This work is structured as follows. In Section 2, we introduce preliminary notation and used metrics. We discuss various methods to evaluate real estate properties in Section 3. In particular, this includes the CBR and DNN approach, as well as our proposed method to utilize EAs to improve the underlying similarity functions. We evaluate our work in Section 4.2 and conclude it in Section 5.

1.1 Related Work

Early work in the computerized assessment of residential properties was mainly conducted using linear regression techniques, e.g. [6]. They were superseded by *hedonic price models*, which determine the price by estimating the effect of each characteristic a property has. These well-researched models have remained the dominant technique for decades [17]. However, since hedonic models assume that a property's price is the sum of its desirable attributes, they are limited. With the advent of more general machine learning methods like *Support Vector Regression*, *Random Forests*, and *Deep Neural Networks*, these have also been applied to real estate valuations. While these methods are more accurate and outperform the classical hedonic models [1, 7, 9, 23, 34, 37], they lack explainability due to their black box character.

Moreover, it has been observed that the explainable approach of taking the average of the k-nearest neighbors beats Support Vector Regression and *Multi-layer Perceptrons* (simple neural networks), while *Regression Trees* perform even better [4].

Further explainable approaches were obtained by introducing the concept of *case-based reasoning* and *rule-based reasoning* (a hedonic regression model) to real estate valuations [14]. There, it was observed that the CBR approach works better on a database of rental prices, while the rule-based approach was superior on a sales database. We extend their work by learning a more complex similarity function for case-based reasoning using EAs, which are applicable to large data sets.

EAs have been previously applied to real estate valuation [16], where they are utilized to learn a hedonic model. To the best of our knowledge, EAs have not been used to learn similarity functions for real estate valuation before.

We work with a data set of real estate offers from the Japanese "LIFULL HOME'S Data Set" [21], which has been used before to compare DNNs with *Kriging*, a generalization of Gaussian modeling to spatial data [34]. In particular, they look at the *nearest neighbor Gaussian processes model*, which enables the application of Kriging to big data sets. In contrast to our contribution, they examine rent price predictions on data set samples with sizes from 10^4 to 10^6. They find that for the largest sample size 10^6 Kriging and DNNs perform similarly but DNNs are superior on the extreme ends of the price range. While they only consider simple architectures, like previously used neural networks with up to 5 layers [32], we examine more complex architectures. In particular, we use the state-of-the-art TabNet [3], a high-performance and interpretable canonical deep tabular data learning architecture. In the context of real estate, two further architectures are proposed on Kaggle[1], a data science community offering machine learning competitions: A DNN published in a competition on real estate prices [11] and one published in another real estate price prediction challenge with the intent to serve as a baseline for DNNs with tabular data [24]. Throughout this work, we denote the former architecture by *Kaggle Housing* and the latter by *Kaggle Baseline*, both of which are also considered in our evaluation.

2 PRELIMINARIES

Let $\mathbb{N}$ and $\mathbb{R}$ denote the set of natural and real numbers, respectively. For $n \in \mathbb{N}$, we define $[n] = [1, n] \cap \mathbb{N}$. Let $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ be the extended real numbers. For a vector $\boldsymbol{v}$ we denote by v_i the i-th entry of $\boldsymbol{v}$.

2.1 Error Measures

In order to evaluate the quality of a predictor, we take a set of n objects, whose values $\boldsymbol{y} \in \mathbb{R}^n$ are known. We then give the objects (but not the values) to the predictor, which then generates a set of predictions $\hat{\boldsymbol{y}} \in \mathbb{R}^n$. Now we want to measure how well the predictions match the actual values of the objects. One such measure is given by the *mean percentage error (MPE)*, which is defined as

$$\text{MPE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{\sum_{i \in [n]} 1 - \hat{y}_i / y_i}{n}.$$

The MPE can be used to determine whether the predictor has a bias, i.e., tends to produce predictions that are larger or smaller than the ground truth. One disadvantage, however, is that prediction errors may cancel each other. This can be avoided by using the

---

[1]www.kaggle.com

well-known *mean absolute percentage error (MAPE)*, that is,

$$\text{MAPE}(\boldsymbol{y}, \hat{\boldsymbol{y}}) = \frac{\sum_{i \in [n]} |1 - \hat{y}_i/y_i|}{n}.$$

Both measures are intuitively understandable, and in contrast to other loss functions allow us to compare the success on different price ranges, since the errors are given in percentage and are therefore not relative to the size of the numbers in the data itself. In particular, the MAPE is standard in machine learning and was used in the context of predicting real estate prices before [7, 34].

3 REAL ESTATE VALUATION

In this section, we introduce a well-known prediction scheme that is already being applied to real estate valuation [4, 14]. We propose extensions to make the method work on large data sets, and explain how EAs can be used to improve its performance. Moreover, we briefly explain how DNNs are used to predict real estate prices [1, 29, 34].

3.1 Valuation with Case-Based Reasoning

Case-Based Reasoning. When applying CBR, we gain knowledge about a new property by considering similar properties whose valuations are known. Formally, this means that we have a set of properties P, divided into two sets P_u and P_v, denoting *unvalued* and *valued* properties, respectively. The valuations of properties in P_v are given by a function $f_{\text{value}} \colon P_v \to \mathbb{R}$, while a symmetric similarity function $s \colon P \times P \to \overline{\mathbb{R}}_{\geq 0}$ indicates how similar two properties in P are. Now, we predict the value of an unvalued property $p_u \in P_u$ as

$$f_{\text{pred}}(p_u) = \frac{\sum_{p_v \in P_v} s(p_u, p_v) \cdot f_{\text{value}}(p_v)}{\sum_{p_v \in P_v} s(p_u, p_v)}. \tag{1}$$

This definition is well known and often referred to as *weighted average prediction* [15].

The similarity function s can be used to represent a natural notion of closeness. In the real estate case an intuitive choice is the squared inverse geographical distance, which we denote as *location-based similarity (LBS)*. Another commonly used approach is to develop the similarity function manually based on the attributes of the properties. One example is the unweighted Euclidean distance between attribute vectors [4]. This function can be generalized by assigning weights to the attributes and allowing the use of different metrics. The resulting parameters can then be adjusted to fit a given data set using machine learning techniques [14]. However, this method is not computationally feasible for large data sets, since the expensive objective function has to be computed often [26].

Extending the CBR Approach. We propose finding similarity functions via EAs. This offers a number of advantages: (1) the similarity function is still fitted to the data set at hand and therefore promises better performance, (2) EAs are very flexible and do not rely on special properties of the function to learn, such as differentiability, and (3) since EAs need to evaluate the objective function less often than comparable optimization techniques, they are better suited to large data sets. We refer to this method as the *CBR+EA* approach.

Formally, we encode a property $p \in P$ as an *attribute vector* $a(p) \in \mathbb{R}^n$, where, for $i \in [n]$, the entry $a_i(p)$ corresponds to the i-th attribute of the property. We propose to learn the inverse of a weighted quasi-norm. That is, we learn $q \in \mathbb{R}^+$ and a weight vector $\mathbf{w} \in \mathbb{R}^n$ and then define the similarity for two properties $p_1, p_2 \in P$ to be

$$s^{q,\mathbf{w}}(p_1, p_2) = \left(\sum_{i \in [n]} w_i (|a_i(p_1) - a_i(p_2)|)^q \right)^{-1/q}.$$

This measure has four key advantages: (1) it has previously demonstrated good performance as a distance measure in high dimensional spaces [2], (2) it can represent a wide variety of similarity functions due to weighting and a variable exponent, (3) it only exposes a reasonable number of parameters (namely the number of attributes plus one), and, (4) by introducing q, we extend the function learned by [14] while by allowing weighting, we extend the Euclidean norm employed by [4].

Application to Large Data Sets. Note that, by Equation (1), determining a price for a single unvalued property p_u involves computing the similarity function for all objects in the data set. This may become infeasible when using large data sets. Moreover, even if all computations are performed, the resulting amount of data may be too large for a human to grasp in order to verify the decision of the algorithm. To mitigate both problems, we extend upon our model by introducing *filtering* as well as *pre-* and *post-selection*.

In filtering, we learn a vector $f \in \overline{\mathbb{R}}_{\geq 0}^n$ that we can wrap around a given similarity function s such that we get a filtered similarity function s_f. For two properties $p_1, p_2 \in P$ we define

$$s_f(p_1, p_2) = \begin{cases} 0, & \text{if } \exists i \in [n] \colon |a_i(p_1) - a_i(p_2)| \geq f_i \text{ and} \\ s(p_1, p_2), & \text{otherwise.} \end{cases}$$

Intuitively, the filtering ensures that two properties are considered not similar at all when their attribute vectors differ by too much in at least one component.

Our second extension of the CBR approach is motivated by the fact that the value of a property $p_u \in P_u$ is mostly influenced by its location [19], making it unlikely that properties with a large geographical distance to p_u contribute reasonable values to the prediction. Therefore, we introduce *pre-selection*, which works as follows. When predicting the value of p_u, we want to compute the function in Equation (1), which involves computing the similarity between p_u and *all* valued properties $p_v \in P_v$. To avoid this, we approximate the weighted average prediction f_{pred} by only considering the subset of properties in P_v containing the $k \in \mathbb{N}_{>0}$ properties that are geographically closest to p_u (among those whose geographical distance to p_u is below a certain threshold $r \in \mathbb{R}_{\geq 0}$). Pre-selection can be implemented efficiently using geometric data structures like R*-trees [5]. This reduces the number of required computations of the similarity function, which in turn further reduces the computational cost of the fitness function.

Finally, in order to reduce the number of objects that influence the price, therefore improving human verifiability, we introduce *post-selection*. There, the weighted average (Equation (1)) is only taken over the $m \in \mathbb{N}_{>0}$ valued properties (which also pass pre-selection) that are most similar to p_u.

In total, the resulting prediction function is given by

$$f_{\text{pred}}^{q,w,f,m,k,r}(p_u) = \frac{\sum_{p_v \in P'_v} s_f^{q,w}(p_u, p_v) \cdot f_{\text{value}}(p_v)}{\sum_{p_v \in P'_v} s_f^{q,w}(p_u, p_v)},$$

where P'_v is the subset of properties of P_v, that fulfill the requirements of pre- and post-selection with parameters m, k, and r, and $s_f^{q,w}$ is the above mention weighted $s^{q,w}$ similarity function after applying the filter f.

3.2 Searching Similarity Functions with EAs

We now use an EA to find good values for the parameters of this similarity function. In particular, we consider a tuple (q, w, f, m, k, r) of the parameters as an *individual*. Throughout several *generations*, a *population* of multiple individuals is altered using *mutation* and *crossover*. All parameters are mutated separately with a fixed probability; if selected for mutation, the new value is sampled from a normal distribution centered at the current value. We employ *uniform crossover*, which means that each parameter is taken from either parent with the same probability. For parameters that are vectors (w and f), the crossover is applied element-wise. The *fitness* of an individual, i.e., the performance of the prediction function obtained when using the corresponding parameters, is evaluated as follows. The given data set is split into two sets containing valued and unvalued properties, respectively. We then predict the values of all unvalued properties by using the similarity function defined by the parameters of the individual on the valued properties. The MAPE (see Section 2.1) of these predictions then represents the fitness value. Among the individuals encountered throughout all generations, the one with the highest fitness value is then used to obtain the final similarity function.

3.3 Valuation with Deep Neural Networks

Deep neural networks (DNNs) are algorithmic computing systems that are designed after biological neural networks [30], e.g., the human brain. A *feed forward neural network* consists of *neurons*, which are arranged in *layers* that are typically densely connected to each other. A neuron processes the linear combination of the *weighted* output of *all* (hence the term *dense*) neurons in the previous layer. In turn, its output is obtained by applying an *activation function* to the combination of the resulting value and a *bias* value. In the end, multiple such layers form a DNN.

The weights and biases are then adjusted to fit the data as follows. In the *forward-propagation* step, part of the data (a *batch*) is passed through the DNN to obtain the predictions thereof. These predictions are then evaluated according to a *loss function*. To minimize the loss function, the weights and biases of the DNN are adapted using gradient descent in order to improve the quality of the predictions in the *back-propagation* step. After the whole data set is passed through the DNN once, we say that one *epoch* passed. After sufficiently many epochs, the DNN is trained. Predictions for a new object are then obtained by passing it through the DNN.

In a *recurrent neural network* the current layer additionally receives its own last value as an input. TabNet [3], a neural network designed especially for tabular data, follows such a recurrent structure. In particular, each recurrent block consists of a feature transformer (a network on its own), an attentive transformer (which aggregates how much each feature has been used in the current decision step) and a mask (which ensures that the model focuses on the most important features).

3.4 Using Location Data for DNNs

The location of a property is known to highly determine its price [19]. Hence, to make useful predictions about a property, information about the prices of the properties in its surrounding area is very useful for predictions.

Making the connections between nearby properties is usually very difficult for DNNs, which learn a direct relationship between the attributes of a property and its value. On the other hand, CBR is all about the relationship of properties among each other, and only indirectly makes the connection to the value of a property with the help of the comparison objects.

In the case of the location-based CBR approach LBS (see Section 3.1), these relationships are captured in a single value that represents the average price in the neighborhood of a property. Since this value can be computed easily in advance, we take it as a standard input for the considered DNNs.

We note that, instead of using the LBS values as input for the DNN, one can instead utilize the predictions made by a more sophisticated CBR approach. Therefore, we also consider combinations of the explainable CBR+EA method and the DNNs, in order to see whether this ensemble leads to an improved performance.

4 EMPIRICAL COMPARISON

We empirically evaluate the above mentioned approaches in order to answer the following questions.

(1) Does the explainable EA-assisted CBR approach improve the valuation performance compared to previous state-of-the-art methods?
(2) How does the geography of a property affect the different approaches? Is there a specific kind of area, such as cities, in which a particular approach is dominant?
(3) How resilient are the approaches to unclean data?
(4) How does the location data provided by CBR approaches affect the performance of the DNNs?

In the following, we first describe our experimental setup and present our results afterwards.

4.1 Setup

Data Set. We evaluate the different approaches on a large real estate data set from Japan. The "LIFULL HOME'S Data Set" [21] is available to computer science researchers worldwide upon request through the Japanese National Institute of Informatics. It contains both rental and sales as well as residential and commercial properties. Here, we focus on residential sales properties in the data set to test the approaches in a specific, real-world use case.

Besides standard property attributes, such as *asking price*, or *living area*, we use additional data included in the set, like the *distance to the nearest schools and bus station*, and an *urbanity score*. Additionally, we use 12 attributes that further increase the information
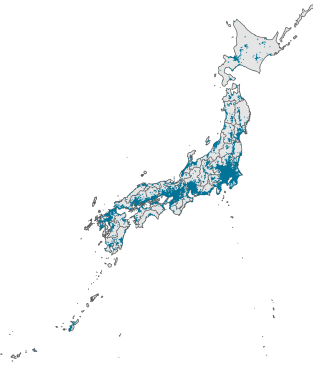
Figure 1: Locations of properties in the considered data set. Each blue point represents one property.

Parameter	Value
Number of generations	200
Restart threshold	10
Population size	20
Sample size	10 000
Mutation rate	0.2
Type switch probability	0.05
Offsprings per parent	5

about a property, without reducing the size of the data set due to missing values. The full list of attributes can be found in Table 3 in Appendix A.

Furthermore, we apply basic cleaning rules. For properties that occur multiple times in the data set, we only consider the latest entry to prevent using the same properties in training and test set, leaving us with 723 680 entries. We remove outliers, i.e., data with unrealistic prices ($>$ 300 000 000 ¥) or location outside of Japan. We are then left with 723 115 entries, whose distribution over Japan can be observed in Figure 1.

Experimental Setup. The standard approach to evaluating the performance of a method when solving a regression task is to split the available data into two sets: a *training set* on which the method can learn and a *test set* on which it is evaluated. Commonly, this split is done randomly, where 80% of the data are used for training, while the remaining 20% are used for testing. Here, we consider another split instead, which better reflects the practical applications in real estate. In particular, the current or future price of a property is predicted only knowing prices of past properties. We model this use case by splitting our data set into training and test sets at 2017-03-01, which results roughly in the common 80/20 ratio. All prices posted before that date are treated as known and used for training and all others are used as test data. We now search for a similarity function by running the EA and train the DNNs, on the same training sets.

The hyperparameters are mostly selected from existing best practices. The ones chosen for the EA are described in Table 1. We picked the number of generations in the EAs such that we did not observe or expect further improvements in fitness beyond that point. All CBR experiments are written in Rust 1.50.0.

Regarding the DNNs, the architecture of Kaggle Housing and Kaggle Baseline consist of 5 and 4 densely connected layers with 200, 100, 50, 25, 1 and 128, 128, 64, 1 neurons, respectively. All layers use *relu* as activation function. For the architecture of TabNet, we refer the reader to the original source [3]. We chose the number of training epochs analogous to how we determined the number of generations for the EAs. Kaggle Baseline and Kaggle Housing yielded almost no improvements after 50 epochs, while TabNet

seems to require much longer training. Due to time constraints, we introduced a time-out by allowing all networks to train for 200 epochs. For deep neural networks we employ TensorFlow[2] 1.7.0 and Python 3.8. A complete list of dependencies is available alongside the source code.[3]

To measure the performance, we now ask all predictors to make predictions about all properties in the test set and calculate the MAPE (Section 2.1) among these. To this end, the known similarity functions as well as the EA-produced ones are given the training data as valued properties P_v. Exemplary EA-produced similarity function parameters (i.e., the fittest individual generated in the run) can be found in Table 2 in Appendix A.

In total we consider seven methods in our experiments. Among them the explainable ones are

- Location-based similarity (LBS),
- Unweighted similarity (Unweighted[4]),
- EA-learned similarity (CBR+EA).

For pre- and post-filtering (see Section 3.1), we set $r = 10$ km for LBS and Unweighted, whereas the CBR+EA method can decide between using k and r, which are constrained to $k \leq 2000$ and $r \leq 50$ km, respectively. Moreover, in preliminary experiments we determined values for m that yield a good trade-off between run time and prediction quality. We set $m = \infty$ for LBS and $m = 50$ for Unweighted, to obtain a good such trade-off. For CBR+EA we consider two variants, one where $m \leq \infty$ and one where $m \leq 10$. The latter is particularly important, as in that case the number of considered comparison objects is small enough to be evaluated by a human.
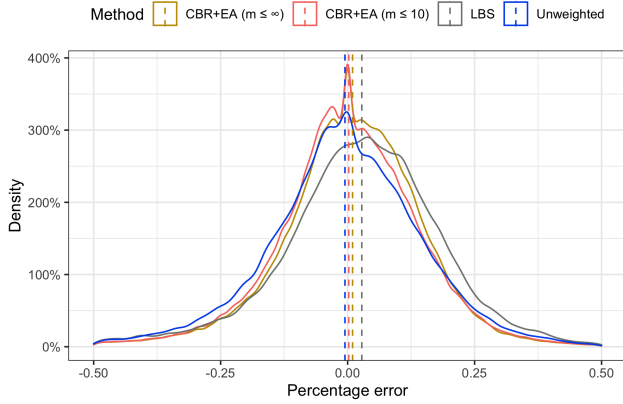
Among the unexplainable methods, we consider Kaggle Baseline, Kaggle Housing, and TabNet.

We note that all methods (except LBS and Unweighted) are inherently random, which is why each run may yield different outcomes. To obtain meaningful statements, we perform each experiment 10 times, and report the average MAPE and the standard deviation.
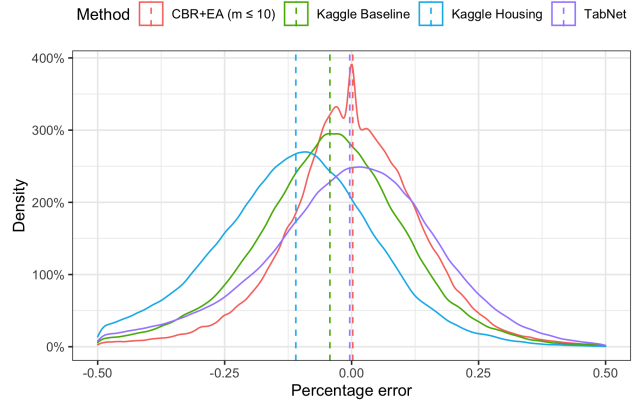
---

[2]https://tensorflow.com

[3]A full copy of the code will be released upon acceptance under a license that enables free use for scientific purposes.

[4]This configuration uses the Euclidean distance, which performed best on our clean data set out of those considered by [4].

Figure 2: Density plot of percentage errors (see Section 2). Dashed lines show the corresponding mean percentage error (Section 2.1). For improved readability, the plot is cut off at 50% percentage error (making up for 3.7 % of predictions).

4.2 Results

First, we evaluate the quality of predictions generally and per prefecture before we turn our focus on analyzing the performance on unclean data, in order to determine how much the methods are affected by faulty values in the training data. Lastly, we compare the DNNs when using different kinds of location data as input, as explained in Section 3.4.

General Predictive Performance. Figure 3 lists performance metrics of different DNN and CBR approaches, which we can use to answer Question 1. As can be seen, the LBS and Unweighted methods already yield viable results with a MAPE of 16% and 13.7%, deterministically. To our surprise, the neural networks did not provide more accuracy, only one of them getting close to the performance of the Unweighted method, which is Kaggle Baseline achieving a MAPE of 14.3% over the 10 runs on average with a standard deviation of 0.2%. There, only one run reached a MAPE of 13.8%, which is on par with the performance of the Unweighted method. The other two DNNs were farther off, with a MAPE of up to 20.4% on average. Despite not being as bad, the results of TabNet were rather unreliable with a standard deviation of 5.3% over the 10 runs.

On the other hand, the EA-improved CBR approaches consistently yielded the best results. The CBR+EA ($m \leq \infty$) method obtained an average MAPE of 12.1% with a standard deviation of 0.1%. Moreover, the CBR+EA ($m \leq 10$) variant, which uses at most 10 comparison objects to compute the final prediction obtained the same results with only a slight increase in the standard deviation (0.2%). Thus, we obtain solid human verifiability (as less than 10 objects are easy to comprehend) with practically no loss in performance.

Looking at the percentage errors in Figure 2, we can get a clearer picture of the prediction performances. The LBS method tends to overshoot the predictions, whereas the other CBR methods, which are based on the same technique feature similar error densities. Notably, tuning the parameters of the similarity functions with

EAs instead of using an unweighted one, further concentrated the density around 0, as can be seen by the peaks in the distribution (Figure 2a).

In comparison, Figure 2b reveals that the DNNs tend to predict values that are too small, except for TabNet whose MPE is very close to 0. However, its distribution is less concentrated around 0 than the CBR+EA method, which results in a worse MAPE.

Local Variations in Predictive Performance. Since property prices are highly determined by location, local variations in the available data are very likely to influence accuracy. To measure this, we consider the best performing CBR and DNN methods, which are CBR+EA and Kaggle Baseline, respectively, and compute the error measures for each of the prefectures of Japan. To obtain reliable results, we only consider the 30 out of 47 prefectures in which we have at least
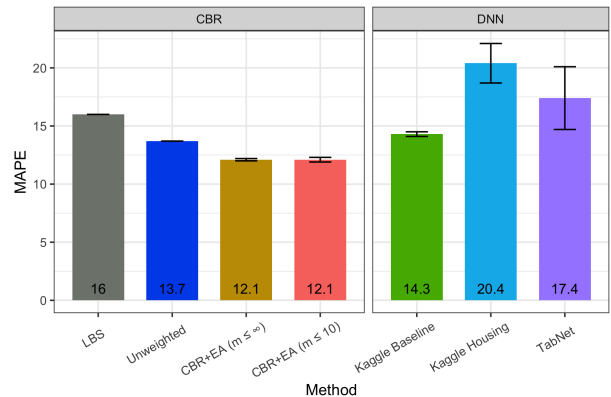

Figure 3: Predictive performance of CBR and DNN approaches over 10 runs. The bars denote the average MAPE in percent and the whiskers show the standard deviation of the MAPE over the ten runs.
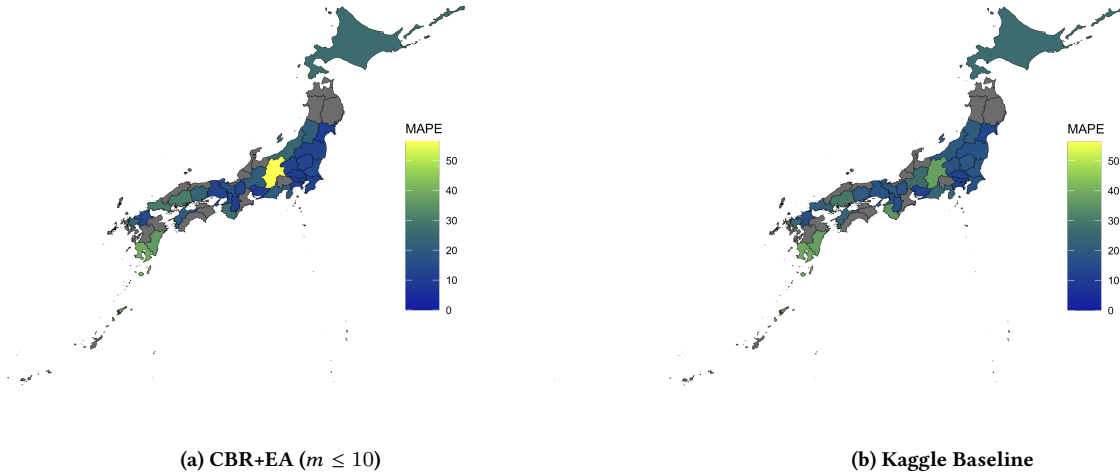
(a) CBR+EA ($m \leq 10$)

(b) Kaggle Baseline

Figure 4: MAPE in percent per prefecture. Prefectures with less than 100 properties in the test set are shown in gray.

100 properties in the test set. The results are shown in Figure 4. In the following, we use this figure to answer Question 2.

While both plots look similar in many locations, there are areas where they diverge visibly. Most notably, for the CBR+EA predictions there is one prefecture (Nagano, shown yellow in Figure 4a), which features a MAPE of 56%. On the other hand, the predictions of Kaggle Baseline in the same prefecture yield a smaller MAPE of 37%. As can be seen in Figure 1, this prefecture is rather sparsely populated. In fact it only contains 165 properties, which is way below the average number of properties per prefecture, which is 3749. For CBR, this is an issue as it bases its valuations purely on the properties in the vicinity of a property. While this can be advantageous, as these properties are likely especially relevant, it also makes the approach more vulnerable to local quality differences in the data or the number of training data close by. On the contrary, while DNNs take local variables into account, they are trained on the whole data set and use that information for every prediction. Consequently, the DNNs can be more resilient to such local variations than CBR, which leads to a more homogeneous picture in Figure 4b.

If, however, the surrounding area features enough comparison objects, CBR can utilize this local information better than DNNs, which leads to stronger blue coloring in the densely populated prefectures surrounding the yellow one in Figure 4a.

Unclean Data. Since data cleaning requires lots of time and domain knowledge, the resilience of a method to not properly cleaned data is of high importance. To answer Question 3, we re-ran all our experiments without applying our cleaning steps outlined in Section 4.1, which adds only 565 additional properties as input data.

Despite this small change in the considered data, the effects are severe, as can be observed in Figure 5. For the CBR-based approaches, performance deteriorates by at least 24% (LBS), which is now the best method among the explainable ones. In fact, for the previously well performing unweighted method the MAPE increases by 76.5% compared to the clean data, which yields a

value of 92.2%. Still, improving the similarity functions with EAs yields a strong performance increase, reducing the MAPE by more than 30% compared to the unweighted similarity function.

Similar issues with uncleaned data have been observed before [14]. Weighted average prediction is particularly sensitive to outliers in price, due to the nature of the average [15]. A single misvalued property can therefore heavily influence the predictions of many similar properties. This effect might be mitigated by using a median instead of an average, as explained in [8].

In stark contrast to the observations on the clean data, here the DNNs can outperform all CBR approaches. While the performance also decreases for the DNNs, the two methods Kaggle Baseline and TabNet only suffer a loss of 0.8% and 5.2% MAPE, respectively. The effects are stronger for Kaggle Housing, where the MAPE increased
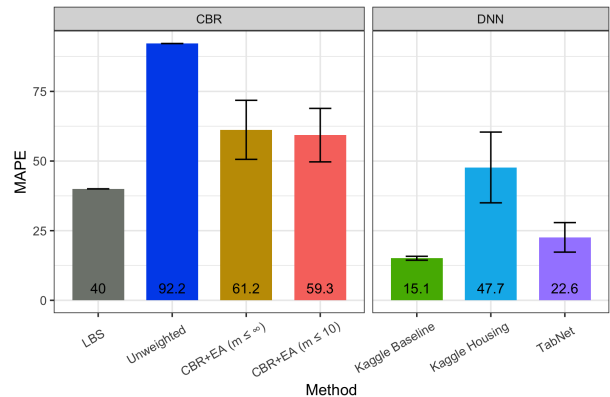

Figure 5: Predictive performance of CBR and DNN approaches over 10 runs on *unclean data*. The bars denote the average MAPE in percent and the whiskers show the standard deviation of the MAPE over the ten runs.
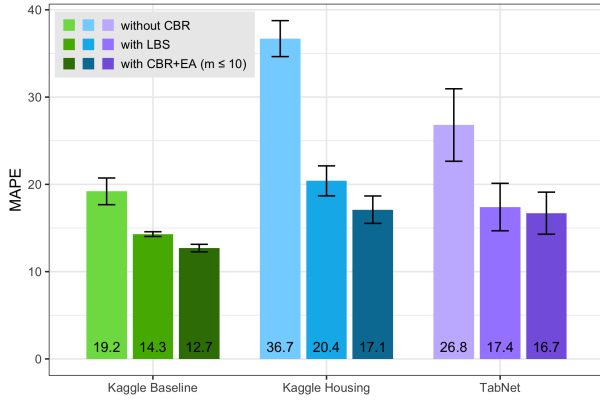
Figure 6: Predictive performance of the DNN approaches for different inputs of location data. Results using LBS predictions are equal to those on clean data, which are shown in Figure 3.

by 27.3% when using unclean data. Still, we can conclude that the DNN approaches tend to be more resilient to unclean data than the CBR methods.

Location Data. As previously explained in Section 3.4 the location of a property has a large impact on its value. Therefore, we consider the easily computable information provided by LBS to be part of the standard input for the DNNs. That is, we supply the DNNs with the predictions made using LBS in all above-mentioned experiments.

To answer Question 4, we examine how much of an advantage this yields for the DNNs. To this end, we performed the experiments on clean data again, without using the LBS predictions as input for the DNNs. Figure 6 shows the results.

Non-surprisingly, the performance of all DNNs decreases when the location data is withheld. For the best-performing DNN, Kaggle Baseline, the increase in MAPE is smallest with 4.9%, yielding a value of 19.2%. As a consequence, and somewhat surprising, all DNNs perform worse than LBS on its own, which obtained a MAPE of 16%. This highlights how reliant neural networks are on the relationships between close-by data points captured by the LBS information that was provided before.

As mentioned before, it is also interesting to examine whether the performance of the DNNs increases if more sophisticated location data is provided as input. Therefore, we ran the same experiments but provided the predictions of the CBR+EA ($m \leq 10$) method as input for the DNNs. As expected, this improves the performance of the DNNs. Kaggle Baseline now achieves a MAPE of 12.7%, which is better than the standard CBR-methods. Surprisingly, however, it does not improve below the CBR+EA methods, whose data was provided as input. We can conclude that the DNN approaches are very reliant on the information provided by the CBR predictions.

5 CONCLUSION & FUTURE WORK

In this paper, we introduce an EA-learned CBR approach for predicting real estate prices which, outperforms not only other CBR approaches but also deep neural networks. The most important factor to this approach is the synergy between the explainable CBR

method and the optimum seeking EA. Even though the EA itself is non-explainable, the resulting similarity function can be interpreted and predictions made using it can be explained with the usual CBR witness system. Thereby, we marry an unexplainable machine learning technique and an explainable approach to receive both accuracy and explainability. Through its comparison properties, our approach also enables an interactive human-algorithm process, which allows rapid integration into existing, human-performed real estate valuation.

Additionally, we investigate how reliant the DNNs are on location data, which provides them with semantic information regarding surrounding properties and greatly improves their performance. However, some DNNs perform worse than the LBS method, even when providing the LBS prediction as an input. Similarly, while the performances of the DNNs improves when allowing them to utilize the CBR+EA predictions, they do not match the quality obtained when using these predictions on their own. One potential explanation is that the data set encompassing all properties in Japan is too heterogeneous for the DNNs to recognize and utilize local patterns. Future work could explore whether training different networks for the different prefectures improves the performance in each one of them, in order to analyze this conjecture. Another aspect that may be investigated is the architecture (layers, activation functions, etc.) of the considered neural networks. While the above considered architectures are standard in machine learning, there are techniques to further improve the performance of DNNs, e.g., by utilizing skip connections or performing *Neural Architecture Search (NAS)* [33]. An interesting direction would be *evolutionary NAS*, which has been used to generate more sophisticated neural networks before [22]. There, EAs have previously been applied in the context of image recognition or classification [12, 25, 36]. To the best of our knowledge, they have not been applied to tabular data before. So far, our preliminary experiments have not shown improvements here.

We note that our combination of CBR+EA with DNNs can be seen as a so-called *ensemble method*, which refers to the combinations of two or more machine learning algorithms that have the potential to achieve better performance than any of them could alone [38]. While this was not the case in our experiments, it would be interesting to investigate other ensembles.

While the neural networks perform worse in most of our experiments, they were not affected as much when working with unclean data. There, the best performing DNN is almost not affected by the change in data quality. On the other hand, the performance of the CBR methods deteriorates to a point where the predictions are unusable. As explained above, one explanation may be that CBR methods rely on taking the average of determined comparison objects, which tends to be affected heavily by faulty data. Future work may explore how this performance drop can be mitigated.

While considering other data like rental prices [34], or a comparison with other techniques like hedonic price or decision tree based models [1, 17], is beyond the scope of this paper, future work might extend our evaluation and validate our approach using different data sets and methods. Since we see great potential in machine learned but explainable methods, we also look forward to further applications in other domains, especially those where the stakes are high.

REFERENCES

[1] Bruno Afonso, Luckeciano Melo, Willian Oliveira, Samuel Sousa, and Lilian Berton. 2019. Housing Prices Prediction with a Deep Learning and Random Forest Ensemble. In *ENIAC*. 389–400.

[2] Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. 2001. On the Surprising Behavior of Distance Metrics in High Dimensional Space. In *ICDT*. 420–434.

[3] Sercan Ö Arık and Tomas Pfister. 2021. TabNet: Attentive Interpretable Tabular Learning. In *AAAI*. 6679–6687.

[4] Alejandro Baldominos, Iván Blanco, Antonio José Moreno, Rubén Iturrarte, Óscar Bernárdez, and Carlos Afonso. 2018. Identifying Real Estate Opportunities Using Machine Learning. *Appl. Sci.* 8, 11 (2018). https://doi.org/10.3390/app8112321

[5] Norbert Beckmann, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger. 1990. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. In *ICDEM*. 322–331. https://doi.org/10.1145/93605.98741

[6] Robert Carbone and Richard L Longini. 1977. A Feedback Model for Automated Real Estate Assessment. *Manage. Sci.* 24, 3 (1977), 241–248.

[7] Marjan Čeh, Milan Kilibarda, Anka Lisec, and Branislav Bajat. 2018. Estimating the Performance of Random Forest versus Multiple Regression for Predicting Prices of the Apartments. *ISPRS Int. J. Geo-Inf.* 7, 5 (2018), 168. https://doi.org/10.3390/ijgi7050168

[8] Joe Celko. 2015. Basic Aggregate Functions. In *SQL for Smarties*. Elsevier. https://doi.org/10.1016/B978-0-12-800761-7.00023-1

[9] T. Dimopoulos, H. Tyralis, N. P Bakas, and D. Hadjimitsis. 2018. Accuracy measurement of Random Forests and Linear Regression for mass appraisal models that estimate the prices of residential apartments in Nicosia, Cyprus. *Adv. Geosci.* 45 (2018), 377–382. https://doi.org/10.5194/adgeo-45-377-2018

[10] Matthew F Dixon, Igor Halperin, and Paul Bilokon. 2020. *Machine Learning in Finance*. Springer.

[11] Diego S Ferreira. 2017. Neural Network Model for House Prices (Keras). *Kaggle* (2017). https://www.kaggle.com/diegosiebra/neural-network-model-for-house-prices-keras.

[12] Saya Fujino, Naoki Mori, and Keinosuke Matsumoto. 2017. Deep convolutional networks for human sketches by means of the evolutionary deep learning. In *IFSA-SCIS*. 1–5.

[13] Megan Garcia. 2016. Racist in the Machine: The Disturbing Implications of Algorithmic Bias. *World Policy J.* 33, 4 (2016), 111–117. muse.jhu.edu/article/645268

[14] Gabrielle Gayer, Itzhak Gilboa, and Offer Lieberman. 2007. Rule-Based and Case-Based Reasoning in Housing Prices. *B. E. J. Theor. Econ.* 7 (2007).

[15] Itzhak Gilboa, Offer Lieberman, and David Scheidler. 2011. A similarity-based approach to prediction. *J. Econom.* 162, 1 (2011), 124–131. https://doi.org/10.1016/j.jeconom.2009.10.015

[16] Vincenzo Del Giudice, Pierfrancesco De Paola, and Fabiana Forte. 2017. Using Genetic Algorithms for Real Estate Appraisals. *Buildings* 7, 2 (2017). https://doi.org/10.3390/buildings7020031

[17] Brano Glumac and François Des Rosiers. 2018. Real Estate and Land Property Automated Valuation Systems: A Taxonomy and Conceptual Model. *LISER Working Paper Series* 9 (2018).

[18] Sharad Goel, Justin M. Rao, and Ravi Shroff. 2016. Personalized Risk Assessments in the Criminal Justice System. *Am. Econ. Rev.* 106, 5 (2016), 119–23. https://doi.org/10.1257/aer.p20161028

[19] Marko Kryvobokov. 2007. What location attributes are the most important for market value? *Prop. Manag.* 25, 3 (2007), 257–286. https://doi.org/10.1108/02637470710753639

[20] Tomo Lazovich. 2020. Does Deep Learning Have Politics?. In *Neurips Resistance AI Workshop*.

[21] LIFULL Co., Ltd. 2019. *LIFULL HOME'S Data Set*. https://www.nii.ac.jp/dsc/idr/en/lifull/.

[22] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, and Gary G Yen. 2021. A Survey on Evolutionary Neural Architecture Search. *IEEE Trans. Neural Netw. Learn. Syst.* PP (2021).

[23] Roy Lowrance. 2015. Predicting the Market Value of Single-Family Residential Real Estate.

[24] Luckeciano Melo. 2019. NN With Tabular Data - Baseline. *Kaggle* (2019). https://www.kaggle.com/luckeciano/nn-with-tabular-data-baseline.

[25] Erfan Miahi, Seyed A Mirroshandel, and Alexis Nasr. 2019. Genetic Neural Architecture Search for automatic assessment of human sperm images. *CoRR* (2019).

[26] Russell B. Millar. 2011. *Maximum Likelihood Estimation and Inference: With Examples in R, SAS and ADMB*. John Wiley & Sons.

[27] Poppy Noor. 2020. Can we trust AI not to further embed racial bias and prejudice? *Br. Med. J.* 368 (2020).

[28] Cathy O'Neil. 2016. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Crown.

[29] Nkolika J Peter, Hilary I Okagbue, Emmanuela C. M Obasi, and Adedotun O Akinola. 2020. Review on the Application of Artificial Neural Networks in Real Estate Valuation. *Int. J. Adv. Trends Comput. Sci. Eng.* 9, 3 (2020), 2918–2925. https://doi.org/10.30534/ijatcse/2020/66932020

[30] D T Pham. 1970. Neural networks in engineering. *WIT Trans. Ecol. Environ.* 6 (1970).

[31] Alvin Rajkomar, Jeffrey Dean, and Isaac Kohane. 2019. Machine Learning in Medicine. *N. Engl. J. Med.* 380, 14 (2019), 1347–1358.

[32] M Mohan Raju, RK Srivastava, Dinesh Bisht, HC Sharma, and Anil Kumar. 2011. Development of Artificial Neural-Network-Based Models for the Simulation of Spring Discharge. *Adv. Artif. Intell.* 2011 (2011), 1–11.

[33] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. 2021. A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions. *Comput. Surv.* 54, 4 (2021), 1–34.

[34] Hajime Seya and Daiki Shiroi. 2021. A Comparison of Residential Apartment Rent Price Predictions Using a Large Data Set: Kriging Versus Deep Neural Network. *Geogr. Anal.* (2021). https://doi.org/10.1111/gean.12283

[35] Moshe Sipper, Randal Olson, and Jason Moore. 2017. Evolutionary computation: The next major transition of artificial intelligence? *BioData Min.* 10 (2017), 26. https://doi.org/10.1186/s13040-017-0147-3

[36] Yanan Sun, Bing Xue, Mengjie Zhang, Gary G Yen, and Jiancheng Lv. 2020. Automatically Designing CNN Architectures Using the Genetic algorithm for Image Classification. *IEEE Trans. Cybern.* 50, 9 (2020), 3840–3854.

[37] Agostino Valier. 2020. Who performs better? AVMs vs hedonic models. *J. Prop. Invest. Finance* 38, 3 (2020), 213–225. https://doi.org/10.1108/JPIF-12-2019-0157

[38] Cha Zhang and Yunqian Ma. 2012. *Ensemble Machine Learning: Methods and Applications*. Springer.

A APPENDIX

We include additional tables to mainly give more context and help with the traceability of our results.

Table 2: Parameters in a the weighted average prediction $f_{\text{pred}}^{q,\text{w},\text{f},m,k,r}$ using filters as well as pre- and post-selection. These correspond to the fittest individual in one run of the respective EAs in Figure 3. Weights with value 0.0 or disabled filters are marked with an empty cell. Attributes which are left out have fixed weight of 0.0 or cannot be chosen as filters, respectively. Before applying weights, all attributes are scaled linearly to to the interval $[0,1]$.

Parameter	$m \leq \infty$	$m \leq 10$
Exponent q	0.283	0.526
Weights w		
Balcony area	0.637	0.629
Floor	1.000	0.564
Living area	0.877	0.069
Location (x)	1.000	1.000
Location (y)	0.998	0.345
Lot area	0.762	0.492
Urbanity score	1.000	0.995
Year of construction		0.122
Distance to closest		
Public transport	0.038	0.036
Convenience store	0.470	
Elementary school		
Junior highschool	0.165	0.009
Parking	0.434	0.105

Filters	$m \leq \infty$	$m \leq 10$
Balcony area		
Date of valuation		
Floor		
Object type	1	
Urbanity score		
Year of construction (in years)	9.601	10.754
Distance to closest		
Bus station	4109.871	
Convenience store		
Elementary school		
Junior high school		
Parking		
Pre- and post-selection		
m	65	10
k	625	1304
r	∞	∞

Table 3: Attributes from the data set used by the predictors.

Name	Description
Asking price	Price that is asked for the property
Average local price	(For DNNs only) predicted price of the LBS approach
Balcony area	The area of a balcony (if present)
Offer date	Date on which the offer was posted
Distance public transport	Distance to the nearest bus or train station in m
Distance convenience store	Distance to the nearest convenience store in m
Distance elementary school	Distance to the nearest elementary school in m
Distance junior high school	Distance to the nearest junior high school in m
Distance parking	Distance to the nearest parking spot in m
Floor	Floor of the property
Living area	Living area in m^2
Location	Longitude and latitude
Lot area	Area of land belonging to the property
Object type	A number denoting the type of object (e.g., house, apartment)
Property size	The area of the property (including garden, garage, etc.)
Urbanity score	A score from 1 to 4 specifying the urbanization of the property's environment
Year of construction	Year in which the property has originally been build