Annals of Operations Research (2025) 344:287–311
https://doi.org/10.1007/s10479-024-06273-1

ORIGINAL RESEARCH

Prototype-based learning for real estate valuation: a machine learning model that explains prices

Jose A. Rodriguez-Serrano[1]

Received: 22 December 2023 / Accepted: 3 September 2024 / Published online: 23 September 2024

Abstract
The systematic prediction of real estate prices is a foundational block in the operations of many firms and has individual, societal and policy implications. In the past, a vast amount of works have used common statistical models such as ordinary least squares or machine learning approaches. While these approaches yield good predictive accuracy, most models work very differently from the human intuition in understanding real estate prices. Usually, humans apply a criterion known as "direct comparison", whereby the property to be valued is explicitly compared with similar properties. This trait is frequently ignored when applying machine learning to real estate valuation. In this article, we propose a model based on a methodology called *prototype-based learning*, that to our knowledge has never been applied to real estate valuation. The model has four crucial characteristics: (a) it is able to capture non-linear relations between price and the input variables, (b) it is a parametric model able to optimize any loss function of interest, (c) it has some degree of explainability, and, more importantly, (d) it encodes the notion of direct comparison. None of the past approaches for real estate prediction comply with these four characteristics simultaneously. The experimental validation indicates that, in terms of predictive accuracy, the proposed model is better or on par to other machine learning based approaches. An interesting advantage of this method is the ability to summarize a dataset of real estate prices into a few "prototypes", a set of the most representative properties.

Keywords Real estate valuation · Non-linear regression · Machine learning · Prototype-based models

1 Introduction

Real estate valuation (Pagourtzi et al., 2003; d'Amato and Kauko, 2017) refers to estimating the price of a property with accuracy on the basis of the current market, and impacts a wide range of stakeholders across industry and society, such as individuals and organizations involved in real estate transactions, or policymakers seeking to manage urban development and housing policies effectively.

✉ Jose A. Rodriguez-Serrano
joseantonio.rodriguez15@esade.edu

[1] Department of Operations, Innovation and Data Sciences, Universitat Ramon Llull Esade, Barcelona, Spain

From an operations research standpoint, the ability of firms to predict real estate prices with precision is crucial for revenue management (Tchuente and Nyawa, 2022), portfolio optimization (Amédée-Manesme and Barthélémy, 2018; Kok et al., 2017), or risk management (Doumpos et al., 2021) and serves as a foundational step which impacts the operations of specific sectors, such as insurance or market research.

The progress in digitization and increased data accessibility has enabled the systematic and large-scale estimation of property prices, known as *mass valuation* (McCluskey et al., 2013; Kok et al., 2017), often involving the use of statistical or machine learning techniques. According to Alexandridis et al. (2019), "automatic mass valuation can reduce operational costs since it is inexpensive and can be performed in a regular basis".

A common approach to mass valuation of real estate properties is hedonic pricing (Malpezzi, 2003). Hedonic models decompose the utility of a good in the sum of utilities of its characteristics independently. An iconic example of a hedonic price model is OLS regression (James et al., 2013), wherein the unit value assigned to each characteristic corresponds to the regression coefficient. OLS has been widely applied to real estate valuation (Ottensmann et al., 2008) and possesses numerous benefits. First, it can be considered a *supervised learning* model: it can be adjusted with training data to optimize any given loss function, and later be used to *predict* the price of new properties. Second, it is easily interpretable: the price directly decomposes as a linear combination of the input variables. Third, OLS is a well-established, accessible and easily reproducible methodology.

Nevertheless, the OLS methodology for price prediction is not without common criticisms. These include its limited ability to handle intricate, non-linear associations or the risk for misleading coefficient interpretations when OLS assumptions are violated. Another drawback of OLS, which is crucial for real estate modeling, is that incorporating spatial coordinates is not straightforward. While technically feasible, it is unlikely that variables such as longitude and latitude have linear relationship with price. This is the reason why some hedonic models tend to incorporate manually constructed variables describing location (Tchuente and Nyawa, 2022) or other factors related to location, such as quality of the views (Potrawa and Tetereva, 2022) or employment potential (Ottensmann et al., 2008).

One step further in sophistication, state-of-the-art machine learning (ML) techniques, such as random forests (Breiman, 2001) or neural networks (Goodfellow et al., 2016) have attracted the attention of researchers in real estate valuation (Doumpos et al., 2021; Peterson and Flanagan, 2009). Their advantages include their capability to deal with non-linear relationships, which normally translates to superior performance. Importantly for real estate research, some models like decision trees, random forests or neural networks can accept location variables such as longitude and latitude without any further processing (Tchuente and Nyawa, 2022).

However, there is a prevalent consensus that while these systems excel in prediction capabilities, they often fall short in providing valuable insights due to their inherent lack of interpretability. Notably, the main criticism is the difficulty to assign a meaning to the model parameters. To counterbalance these limitations, some works in real estate valuation have successfully combined machine learning models with explainable AI (XAI) techniques (Lorenz et al., 2022; Potrawa and Tetereva, 2022). These techniques complement machine learning models by extracting metrics such as relevance scores for each variable. Still, an issue is that these techniques are applied *ex post* and remain detached from the model itself.

Furthermore, one additional limitation of both OLS and ML models is that they ignore a fundamental cue in valuation theory: direct comparison—perhaps the most common and intuitive way in which humans appraise real estate. In direct comparison, "the value of the property being appraised [...] is assumed to relate closely to the the selling prices of

selling properties within the same market area", as put by Pagourtzi et al. (2003), who name this concept the *comparables method*. The same idea has also been dubbed *direct capital comparison* (DCC) by Lins and L.F.d.L. Novaes, and L.F.L. Legey. (2005). In order to unify the terminology, in the following we refer to this concept as *direct comparison*.

If the models used for real estate valuation incorporated the notion of direct comparison, they would be better aligned with human expectations and market domain knowledge. Yet it can be clearly argued that OLS and, more broadly, machine learning approaches, do not implement explicitly the idea of direct comparison, because these models establish direct mappings from input variables to prices. In other words, most of these predictive models do not involve similarity computations between the input property and the nearby or similar properties.

There are notable exceptions to the previous statement, such as K-Nearest Neighbors (KNN) (Kramer and Kramer, 2013) and its variants. These are designed to offer predictions grounded in such comparative analyses and have exhibited good performance in the domain of real estate price prediction (Choy and Ho, 2023; Isakson, 1988). One technical drawback of KNN is that it is a non-parametric density estimation method; as such it cannot be adjusted to minimize a loss metric. The only mechanism to increase the quality of the estimation is by increasing K, which renders the method inefficient.

Based on this analysis, it is clear that none of the discussed approaches simultaneously complies with one of these characteristics: (i) capability to model complex and non-linear relationships, (ii) capability to optimize a predictive objective function, (iii) exhibit some degree of explainability, and (iv) implement the notion of direct comparison explicitly.

In this article, we propose a method that complies with these four characteristics simultaneously. The method is based on a methodology known as "prototype-based learning" (Biehl et al., 2016) which, to the author's knowledge, has not been previously applied for real estate valuation. Prototype-based learning is a predictive model that:

- can learn non-linear relations between the dependent and input variables,
- is a supervised learning method that can be trained to minimize any given loss function (unlike KNN),
- includes explainability as part of the model, and
- implements explicitly the notion of direct comparison: the model predicts the price of a property by contrasting the property of interest to a set of (automatically inferred) reference properties.

Table 1 summarizes these four characteristics and compares the aforementioned models and approaches in terms of which of them they fulfill.

The model works by minimizing a prediction error, based on comparing the variables of a property with the distributions of a set of references, and then optimizing these references. Therefore, as a by-product, the model produces a segmentation of the data into groups (or areas) with differentiated characteristics, similar to sub-market models (Goodman and Thibodeau, 2007; Rey-Blanco et al., 2024).

We study prototype-based models using a public dataset of real estate prices in Seattle. Our experiments conclude that a) prototype-based learning successfully distills the prototypes as theoretically expected, b) the method outperforms other strategies for finding prototypes, c) the prediction error observed is lower compared to other machine learning models, and d) the model can incorporate location-based information in an effective way.

The rest of the article describes the model (Sect. 3) in detail, reports the different experiments (Sect. 4), provides a detailed discussion (Sect. 5), reviews the practical and research implications (Sect. 6) and comments on conclusions and future perspectives (Sect. 7).

Table 1 The columns show the design desiderata for our real estate prediction model as discussed in the main text

	Non-linear	Parametric training	Explainable	Implements DCC
OLS		✓	✓	
ML	✓	✓		
ML + XAI	✓	✓	✓	
KNN	✓		✓	✓
Prototype-based	✓	✓	✓	✓

The rows specify which criteria are met by each family of approaches: ordinary least squares (OLS), state-of-the-art machine learning models (ML), ML with explainable AI (XAI) techniques, K-Nearest Neighbors (KNN) and prototype-based model, the method under study in this article. The latter is the only approach that fulfills all the desiderata

2 Related works

2.1 Hedonic models and OLS

The origins of the hedonic pricing theory have been a subject of debate between scholars (Goodman, 1998; Colwell and Dilmore, 1999). It is often acknowledged that the first use of "hedonic" is by Court (1939). However, Haas (1922) already proposed to predict land price per acre based on a linear model more than one century ago.

As of today, OLS is still a common method to model, and predict, land or real estate prices. We refer the readers to Hong et al. (2020) and Potrawa and Tetereva (2022) for comprehensive reviews of prior research that adhere to this methodological approach and delve into various categories of employed variables. A well-established consensus, substantiated by empirical evidence, underscores the positive correlation between property prices and factors such as floor area, the number of bedrooms, and bathrooms. In contrast, certain variables, such as property age, exhibit a negative association with property prices (Hong et al., 2020). It is important to consider factors other from those describing the physical property, such as "locational" or "neighborhood"-based factors (Potrawa and Tetereva, 2022) which tend to exhibit a strong influence, too. For instance, Ottensmann et al. (2008) and Hussain et al. (2019) study the impact of urban factors, such as proximity to employment locations, or impoverished neighborhoods, respectively, in real estate prices. Some works even go as far as using variables encode the quality of the view, as extracted by image analysis techniques (Potrawa and Tetereva, 2022) or quality scores extracted from user reviews (Gibbs et al., 2018).

OLS finds advantages in its simplicity and ability to interpret the coefficients as the average increase in price after a one-unit increase of the corresponding variable. However, some drawbacks include the limitation of the functional form, incapable to model non-linear relations between variables, interactions or colinearities (Zurada et al., 2011). Another common drawback is the usual assumption of linear distribution, rarely observed in practice (Potrawa and Tetereva, 2022). Despite these (well-known) technical limitations, OLS still finds many applications, as some practitioners find practical productivity increases in predictive applications.

2.2 ML models

As an alternative to OLS, previous studies have resorted to more advanced machine learning techniques in order to obtain models with more complex functional form.

This includes the use of neural networks (Peterson and Flanagan, 2009; McCluskey et al., 2013), which produce non-linear predictions by building a chain of linear operations and non-linear activation functions, and are considered universal approximators (Goodfellow et al., 2016). Other common approaches to produce robust non-linear characteristics are the random forests (Hong et al., 2020) or other forms of tree-based ensembles (McCluskey and W., D. Zulkarnain Daud, and N. Kamarudin., 2014; Kok et al., 2017). These models are based on regression trees, which produce predictions based on a sequence of variable tests, capturing non-linear relations and interactions between variables. Then they average many regression trees using different criteria to increase robustness.

Neural networks and tree-based ensembles are probably the two most common state-of-the-practice approaches in machine learning. We refer the reader to Doumpos et al. (2021) or Tchuente and Nyawa (2022) for an exhaustive review and comparison between different machine learning methods in real estate modeling.

Because of their added complexity, machine learning models tend to be, on the one hand, more accurate than OLS. On the other hand, an oft-cited issue of most ML approaches is their lack of interpretability. Interpretability denotes the capability to easily understand and describe how an output is produced given an input, and is an important feature for market regulators, or in general to augment the trust in decision-support systems. While some ML models are considered white-box, such as decision trees, most state-of-the-art ML models, such as those based on neural networks or tree ensembles are considered black-box (Tchuente et al., 2024). To overcome that limitation, there has been a recent emergence of explainable AI (XAI) techniques (Lundberg et al., 2020). In real estate prediction, Lorenz et al. (2022) have applied XAI methods such as *partial dependence plots* and *feature importance* to complement ML-based models, the latter also being used by Potrawa and Tetereva (2022). XAI methods are employed to diagnose black-box models, often are applied ex post, and are not part of the model. In contrast, as acknowledged by Li et al. (2018) (and further explained in Sect. 3), prototype-based models provide explanations *naturally* as part of the model.

2.3 Towards direct comparison: KNN and clustering

Among specific machine learning models used in real estate research, those that take into account direct comparisons *by design* include those based on KNN (Choy and Ho, 2023; Isakson, 1988; Tchuente and Nyawa, 2022). In KNN, the regression is taken as a weighted average of the outputs of the nearest data points of the training set, based on some distance (usually, Euclidean) between the vectors of input variables.

As such, KNN is capable of modeling complex, non-linear effects between price and the input variables. Its advantages also includes its methodological simplicity and interpretability. However, one drawback of standard KNN on the predictive performance side is that it is a non-parametric model and cannot be adjusted based on a target loss function.

In order to combine the benefits of KNN and other machine learning methods, some real estate studies perform spatial weighting schemes (Bitter et al., 2007; McCluskey et al., 2013), of which a well-known type is *geographically weighted regression* (Fotheringham et al., 2003). Spatial weighting assigns more importance during model training to those samples which are proximal to the example whose price is to be predicted.

The model proposed in this paper addresses both characteristics: first, it uses comparative-based measures, akin to KNN, while offering the flexibility to be fine-tuned based on a user-defined loss function. Experiments demonstrate that this crucially reduces the prediction error. Second, it incorporates spatial weighting as an intrinsic feature.

2.4 Prototype-based learning

In summary, the advantages of prototype-based models in terms of modeling non-linear relations, explainability, availability of comparison-based valuation, and ability to be optimized given a loss function make those models very promising for predicting real estate prices.

The term *prototype-based models* (Biehl et al., 2016) encompasses a family of approaches rather than a singular methodology, and within this family we focus on the subset of supervised learning approaches. These make predictions by leveraging latent "prototypes", which refer to a reduced set of data samples which summarize most of the patterns occurring the whole dataset. Prototypes can correspond to actual data points, or can consist of calculated variables and not strictly correspond to any real observed datum. Additionally, when the space is segmented based on proximity to prototypes, these models facilitate the identification of distinct regions or clusters within the data space, effectively acting as "cluster-aware supervised learning algorithms" (Chen and Xie, 2022).

The next section is devoted to describing the specific method employed in the paper and will introduce many technical aspects of prototype-based methods. This methodology finds applications in diverse areas such as computer vision (Li et al., 2018), natural sciences (Owomugisha et al., 2021), or healthcare (van Veen et al., 2022), including COVID detection (Kaden et al., 2022). To the best of the author's knowledge, however, this method has never been applied to real estate prediction, in spite of its appealing benefits.

3 Model

3.1 Notation

The proposed model requires a dataset of real estate properties with N samples, each with D independent or (in the following) *input variables* x_{nd} (thus $n \in \{1, \ldots, N\}$ and $D \in \{1, \ldots, D\}$) describing characteristics of the real estate properties, and a dependent or *output variable* y_n indicating the observed price. By *price* we mean either the direct price, or any transformation of the price, such as the price normalized by a constant or non-constant factor (e.g. price per square foot), or the logarithm of the price, just to mention some commonly used options. Our model is compatible with all these definitions.

We will use the boldface form $\mathbf{x}_n$, common in the ML literature, when we have to refer jointly to the set of all input variables $x_{n1}, \ldots, x_{nD}$ of sample n. We will omit the index n when it is not necessary for the discussion. In a slight abuse of language, occasionally the text may refer to $\mathbf{x}$ shortly as a "property" (instead of "the values of the input variables describing a property").

3.2 Basic model structure

The main design principle of our model is that it should implement the notion of direct comparison: if we observe a new property with input variables $\mathbf{x}$, we should be able to infer

its price by comparing it to a number of "references", denoted as prototypes, and interpolating from the price of similar references.

Next, we discuss how the model proposed in this article directly implements this principle.

We define a set of M *prototypes*, which represent a set of properties with similar input variables and with an attributed price. Formally, each reference consists of:

1. A prototype distribution, which models the probability distribution $f_m(\mathbf{x})$ of properties within group m. In the simplest case, we take a multivariate Normal distribution
$$f_m(\mathbf{x}) = \mathcal{N}(\mathbf{x}|\mathbf{p}_m, \boldsymbol{\sigma}_m) \tag{1}$$
where $\mathbf{p}_m = (p_{m1}, \ldots, p_{mD})$ specifies a vector of means and $\boldsymbol{\sigma}_m$ refers to the vector of standard deviations (assuming diagonal covariance).[1] Because $\mathbf{p}_m$ represents the mode of f_m, it therefore indicates the typical values of the input variables for each prototype.
2. A prototype value v_m, which models an price attributed to that group of properties.

To compare a property $\mathbf{x}$ to each of the M references, a principled method (Bishop, 2006) is determining the posterior probability that f_m has generated $\mathbf{x}$, i.e.
$$\gamma_m(\mathbf{x}) \equiv \frac{f_m(\mathbf{x})}{\sum_l f_l(\mathbf{x})}. \tag{2}$$
This comparison should yield, ideally, a small set of properties with non-negligible values of $\gamma_m(\mathbf{x})$ which represent the most similar references. Then the predicted value is a a weighted average[2] of the references, where the weighing coefficient is precisely the value of γ:
$$\hat{y} = \sum_{m=1}^{M} \gamma_m(\mathbf{x}) v_m. \tag{3}$$
As we will verify empirically later, it is common that most values of $\gamma_m(\mathbf{x})$ are negligible and only a few terms contribute to the sum, and therefore Eq. 3 can be understood as an "interpolation" of the attributed values of the most similar references.

Note that the proposed model is completely determined by p_{md} and v_m (and also σ_{md} in case these are optimized). We jointly denote these parameters as θ. Its optimal values will be *learned* from the data using the following loss function:
$$C = \frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2 + \eta R(\theta), \tag{4}$$
The first term is as mean squared error (MSE), which measures the fitness error of the model with respect to the data, and R is a regularization term, common in ML tasks to penalize model complexity or to impose reasonable restrictions. In our case, we use
$$R(\theta) = \frac{1}{M} \sum_{m} \min_{n} \|\mathbf{p}_m - \mathbf{x}_n\|. \tag{5}$$
This term averages the distance between each prototype and its closest data point, similar to Li et al. (2018), therefore adding a penalty if prototypes $\mathbf{p}_m$ are far from real properties $\mathbf{x}$(here, $\|\cdot\|$ stands for the Euclidean norm).

[1] A simplification used in some cases is to assume the same value of the standard deviation for all variables (isotropic covariance) and all prototypes, in which we will refer to the standard deviations as the scalar σ. Moreover, in that simplified case, σ can be fixed a priori and treated as hyperparameter (which sometimes improves the numerical stability of the model).

[2] Because, by definition, $\sum_m \gamma_m(\mathbf{x}) = 1$, Eq. 3 can be interpreted as a weighted average.

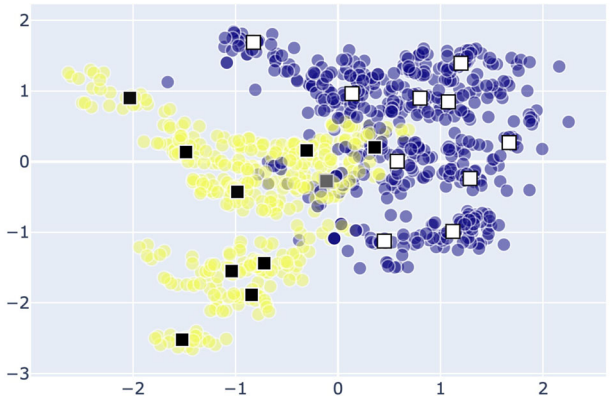

Fig. 1 (Best viewed in color) Illustrative example of the prototype-based method. (Color figure online)

It is straightforward to identify that Eq. 4 is differentiable with respect to θ. Therefore, we can use standard solvers that involve automatic differentiation (Baydin et al., 2018) such as `keras` to implement and this model and find its optimal parameters.

3.3 Model interpretation and simple example

The parameters p_{md}, v_m of the model have a natural interpretation. Specifically, as p_{md} is the mean of the distribution f_m, it can be interpreted as the variables of a representative property, and v_m as its representative price. Therefore, not only will the model be able to predict property prices, it also produces a list of M prototypical properties and suggests values for their expected variables and price.

To illustrate the model capabilities, for now we select a simple example on a controlled setting (experiments on real data are reported in the next section). Figure 1 shows a scatterplot corresponding to a small dataset with two input variables. The locations of the circles on the horizontal and vertical axes represent, respectively, the input variables x_{n1} and x_{n2}, and the color indicates the value of the output variable y (yellow for $y = 0.0$ and blue for $y = 1.0$).

In this case, we train the prototype-based model with $M = 20$, and using a loss function with $\eta = 0$. In this setting, the parameters can be directly translated to the plot: p_{m1} and p_{m2} represent the location of the prototype means on the horizontal and vertical axes. The color indicates the prototype values v_m (black for $v_m = 0$ and white for $v_m = 1$). The color scale is continuous, although perhaps that is not perceivable as most prototype values are close to $v_m = 0$ or $v_m = 1$.

By visual inspection, it is clear in this example that the locations p_{md} of the prototypes span the different "clusters" in the data and the values v_m match the values of y_n in each area. As a conclusion, we have quantitative evidence in a controlled example that the obtained prototypes can be interpreted as representative property characteristics and their price.

To shed light on how the model implements the intuition of *direct comparison*, consider the prediction for a given sample $\mathbf{x}$. To apply Eq. 3, first we need to determine the values of γ in Eq. 2. Continuing with our example, in the case of two variables, using a bivariate Normal distribution for f_m, we have:

$$\log f_m(\mathbf{x}) = -\log(2\pi) - 2\log\sigma - \frac{1}{2}\frac{(x_1 - p_{m1})^2 + (x_2 - p_{m2})^2}{\sigma^2}. \tag{6}$$

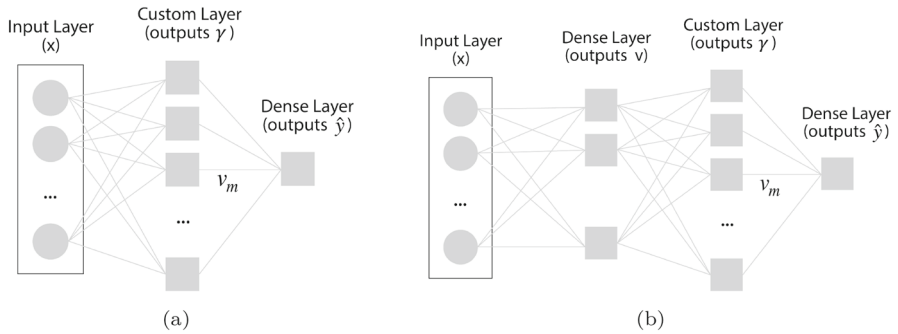

Fig. 2 **a** Prototype-based model interpreted as a neural network. **b** Example of extending the prototype-based model neural network with an additional hidden (embedding) layer

If we define $\lambda_m \equiv \log f_m(\mathbf{x})$, then Eq. 2 becomes

$$\gamma_m(\mathbf{x}) = \frac{e^{\lambda_m}}{\sum_k e^{\lambda_k}}, \tag{7}$$

sometimes known as the softmax function.

As we can verify from Eq. 6, as the Euclidean distance from (x_1, x_2) to (p_{m1}, p_{m2}) decreases, the value of γ_m increases and the contribution of the value v_m becomes higher. Crucially, this is precisely the expected behavior in a comparison-based model.

3.4 Model extension as neural network

The prototype distribution f_m in Eq. 1 takes as input a property $\mathbf{x}$. However, we note that our model is compatible with approaches that first compute a so-called embedding $\mathbf{h} = g(\mathbf{x})$, where g denotes the result of neural network layers applying some transformation to the input variables. In this section, we demonstrate how our model can be interpreted as a neural network and integrated into standard neural network frameworks. We utilize common neural network terminology throughout. Readers who are not familiar with neural network models can skip this section without impacting their understanding of the rest of the article. Alternatively, they may refer to foundational texts such as Goodfellow et al. (2016) for further information.

First, note the sequential computation of Eqs. 2 and 3 can be expressed in neural network vocabulary:

- Eq. 2 is equivalent to passing the inputs $\mathbf{x}$ through a probabilistic custom layer with M units, where the mth unit outputs $\log f_m(\mathbf{x})$, and then applying a standard "softmax" activation function.
- Eq. 3 is simply a dense layer, which takes as input the previous outputted γ's, multiplies them with some weights and outputs $\hat{y}$. Since there is only 1 output, there are m weights which are precisely the prototype values v_m.

A schematic representation of the prototype-based model interpreted as a neural network is depicted in Fig. 2a.

This interpretation as neural network has two main benefits. First, it facilitates the code implementation of our model, as the previously described network can be implemented

using commonly available libraries such as `tensorflow` or `keras` (see Sect. 4.2 for implementation details).

A second benefit is that we can add further hidden layers between the inputs and the custom probabilistic layer. This adds a (potentially non-linear) transformation of the input variables and thus increases the complexity of the predictive function. An example where a hidden layer (embedding) is added to our model is illustrated in Fig. 2b. With this architecture, now the input to the prototype-based model is the output of the hidden layer.[3]

In such a case, the parameters of the neural network g can be pre-trained, or jointly optimized during the training of the prototype-based model. Technically, one needs to replace $f_m(\mathbf{x})$ by $f_m(\mathbf{h})$ and this turns the loss function in Eq. 4 an implicit function of g and its weights. This change is straightforward to apply in the aforementioned automatic differentiation tools.

One drawback of this extension is that the prototype-based model becomes non-interpretable if additional layers are added. However, in some cases, prioritizing predictive accuracy over interpretability may justify this approach. Nonetheless, this extension is optional, and we simply emphasize that our model is compatible with it.

4 Experimental validation

4.1 Data

We use data from a record of properties in King County (Seattle),[4] which includes property prices and other variables collected from May 2014 to May 2015.

The dataset contains information about 21,613 properties. Table 2 indicates a summary of the variables available in the dataset that will be relevant for the analyses below, and some typical statistics. Fig. 3 displays a map of the location of the properties , where the color indicates the price per square foot.

4.2 Model implementation

The model is implemented in the Python programming language, using `keras` as the automatic differentiation engine. The code is publicly available.[5]

4.3 Model using location variables only

The first experiment will evaluate the model considering exclusively location and price per square foot. There are several reasons for this choice. First, it is widely acknowledged that "location is the primary factor in determining market prices" (Gabrielli et al., 2023) and that this primacy arises from its multifaceted influence on factors such as proximity to economic opportunities, appeal of certain leaving areas or transportation accessibility. Ultimately "virtually every attribute of a piece of land beyond its physical dimensions including any

[3] According to our previous definition $\mathbf{h} = g(\mathbf{x})$, in this simple case of 1 hidden layer, g is simply $g(\mathbf{x}) = \alpha(\mathbf{W}\mathbf{x})$, where $\mathbf{W}$ denotes a matrix of weights and α an activation function.

[4] Publicly available at https://geodacenter.github.io/data-and-lab/KingCounty-HouseSales2015/

[5] The code is structured as follows: model and examples are available in https://github.com/prof-jose/prototype_based_learning and the experimental benchmark in https://github.com/prof-jose/prototype_based_learning.

Table 2 Summary of the variables used for the case of the King County properties dataset

Variable	Data type	Statistics			
		Min	Max	Mean	Std
Date	Date	2014/05/02	2015/05/27	N/A	N/A
Latitude	Numeric	47.2	47.8	47.6	0.139
Longitude	Numeric	$-1.23 \cdot 10^2$	$-1.21 \cdot 10^2$	$-1.22 \cdot 10^2$	0.141
Number of bathrooms	Numeric	0	8	2.11	0.770
Square feet of living room	Numeric	$2.90 \cdot 10^2$	$1.35 \cdot 10^4$	$2.08 \cdot 10^4$	$9.18 \cdot 10^2$
Square feet of lot	Numeric	$5.20 \cdot 10^2$	$1.65 \cdot 10^6$	$1.51 \cdot 10^4$	$4.14 \cdot 10^4$
Number of floors	Numeric	1.0	3.5	1.49	0.540
Waterfront	Binary	0	1	$7.54 \cdot 10^{-3}$	$8.65 \cdot 10^{-2}$
View	Numeric	0	4	0.234	0.766
Square feet above	Numeric	$2.90 \cdot 10^2$	$9.41 \cdot 10^3$	$1.79 \cdot 10^3$	$8.28 \cdot 10^2$
Square feet of basement	Numeric	0.0	$4.82 \cdot 10^3$	$2.92 \cdot 10^2$	$4.43 \cdot 10^2$
Square feet of living room in 2015	Numeric	$3.99 \cdot 10^2$	$6.21 \cdot 10^3$	$1.99 \cdot 10^3$	$6.85 \cdot 10^2$
Square feet of lot in 2015	Numeric	$6.51 \cdot 10^2$	$8.71 \cdot 10^5$	$1.28 \cdot 10^4$	$2.73 \cdot 10^4$
Price	Numeric	$7.50 \cdot 10^4$	$7.70 \cdot 10^6$	$5.40 \cdot 10^5$	$3.67 \cdot 10^4$

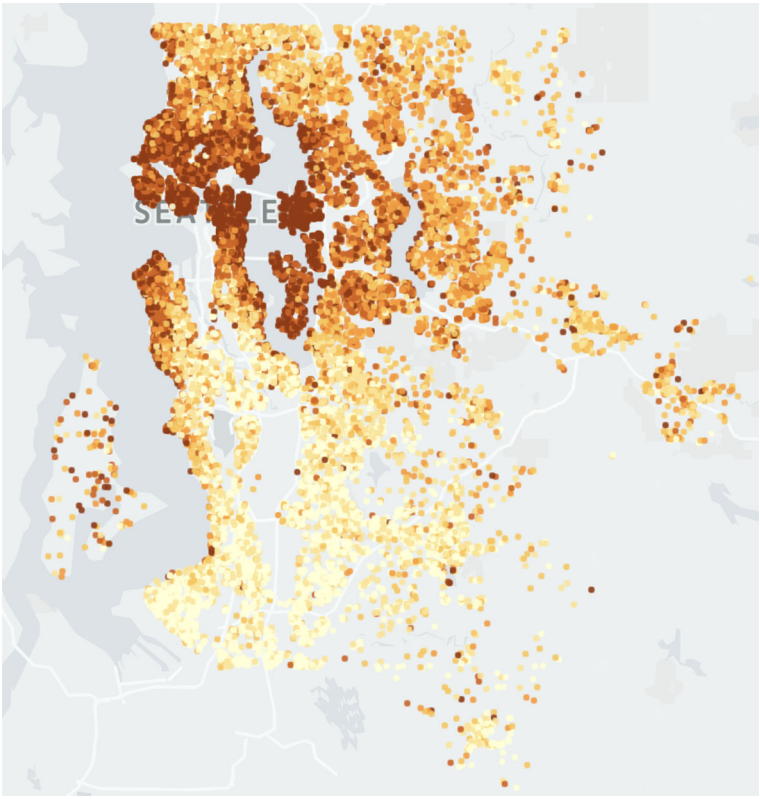

Fig. 3 (Best viewed in color) Locations of properties in out dataset on a real map. The color intensity indicates the quantile of the price per square foot (darker indicates higher). (Color figure online)

improvements can be reduced to location" (Atack and Margo, 1998). Therefore, we simulate a model that predicts real estate price solely based on location. Second, when the input variables are location, the prototypes $\mathbf{p}_m$ of our model can be directly interpreted as a list of typical locations, and the values as typical prices in these locations. While deliberately simplistic, this is still very relevant information for practitioners in real estate or seeking to obtain a geographical segmentation of the market and sub-market analyses (Goodman and Thibodeau, 2007).

Experimental protocol

To perform model training and evaluation, an initial step involves a linear partition of the dataset into training and testing subsets (with a training/test distribution of 60%/40%), structured in accordance with chronological order (given by the date column). This approach emulates real-world conditions where available information is time-bound, and the goal is to predict property values beyond a specific temporal point. This method effectively mitigates the potential leakage of temporal information into the predictive models. Notably, within the training subset, 20% of the data is reserved for parameter validation, ensuring that model parameters are optimized effectively.

Table 3 RMSE of the models using only location variables, averaged over 10 runs

Method	RMSE (Test) (ft^2)
Random prototypes	110
K-Means + Price Average	81.7
Prototypes	76.6

For the analysis, we take *latitude* and *longitude* as the input variables, and construct an output variable *price per square foot* as the quotient of the available columns *price* and *square feet*.

In the data pre-processing stage, each input variable is standardized (to zero mean and unit variance). The output variable y (price per square foot) is scaled by a factor of 1/1000, with the objective of rescaling the values of y to fall within the approximate range of (0.0, 1.0), which makes numerical optimization more favorable.

To build our model, we use $M = 50$ prototypes. The model is implemented and optimized using the `keras` library, with an Adam optimizer with learning rate l_r, batch size b and number of epochs E. To these hyperparameters, we add the regularization coefficient η of Eq. 4 and also use a constant preset σ for n Eq. 1. All these hyperparameters are systematically determined using a standard grid search strategy on the validation set. We highlight that we use a constant value of σ for both coordinates and for all prototypes, so that the computation of importances γ of Eq. 2 is explicitly a function of the *Euclidean* distance between each property and the prototype mean.

Our model is subjected to comparative evaluation against two alternatives. First, we conduct a comparison with a nearest neighbor algorithm using randomly selected neighbors and M=50 (in order to match the number of prototypes). This comparison offers insights into the efficacy of the prototypes in guiding the predictive process. Second, we compare with an alternative way of discovering prototypes, wherein we use a clustering algorithm. Clustering stands as an alternative for identifying groups of properties with similar characteristics and valuations. Therefore, we apply a K-Means clustering algorithm (Bishop, 2006) with $K = 50$ and take its means as the prototypes $\mathbf{p}_m$, and the average price per square foot in each cluster as the v_m. For all models, we measure the quality of fit with the root mean squared error (RMSE), in the original y-scale (before normalization).

To mitigate the impact of sources of variability such as random seeds used in parameter initialization, we repeat the experiment a total of 10 times, and report the RMSE averaged over the 10 runs.

Quantitative results

We first evaluate the predictive accuracy of the model. The RMSE values of the different models tested in experiments described above are shown in Table 3. We observe the prototype-based model yields significantly lower RSME than the two baseline methods (the standard deviation of the RMSE over the 10 runs is smaller than 1 $/ft^2), therefore indicating that it has accurate prediction capabilities.

Results: Interpretation of the prototypes

Now we turn to discussing the interpretation of the prototypes $\mathbf{p}_m$ obtained by the model. Because our input variables are longitude and latitude, the model parameters $\mathbf{p}_m$ and v_m can be interpreted directly as the prototype locations and prototype values. Figure 4 displays the locations $\mathbf{p}_m$ overlaid on the original data.

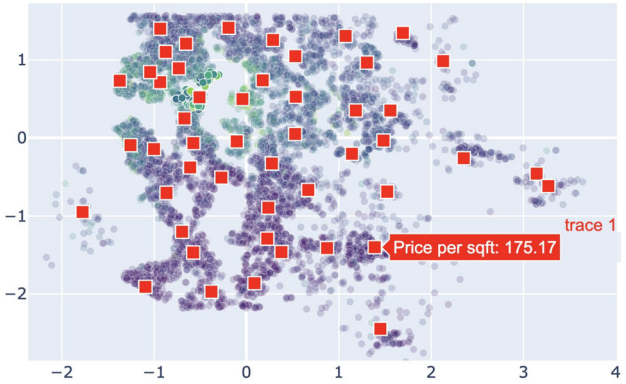

Fig. 4 (Best viewed in color) Locations of the prototypes $\mathbf{p}_m$ overlaid in the original data, and showing the corresponding v_m for a specific example. (Color figure online)

Fig. 5 Zoom into one of the prototypes. (Color figure online)

Through our analysis, it becomes evident that the prototypes p_m exhibit a wide distribution that effectively spans the different regions of the entire dataset. Furthermore, as depicted in Fig. 5, a closer examination is directed towards a specific prototype, shedding light on properties in close proximity to it or, in other words, all the properties that can be succinctly encapsulated or "summarized" by the respective prototype.

We verify by visual inspection that the values v_m and the nearby prices are consistent. Therefore, we qualitatively verify that the behavior of the model mimics the process of valuation-based comparison that is similar to the human intuition behind pricing.

Also, we quantitatively investigate how many prototypes typically contribute to each prediction when evaluating Eq. 3. To that end, we compute the prototype importances $\gamma_m(\mathbf{x})$ for all properties $\mathbf{x}$ of the test set, and count the non-negligible terms as those with $\gamma_m(\mathbf{x}) > 0.01$. Table 4 displays the resulting frequencies. Notably, in no case did we observe more

Table 4 Frequencies of the number of non-negligible prototypes contributing to the prediction in Eq. 3

Non-negligible prototypes	Frequency (%)
1	74.8
2	19.8
3	4.7
4	0.7
5	< 0.05
6 and more	0

All the predictions involve less than 5 non-negligible prototypes, and more than 94% involve only 1 or 2 prototypes

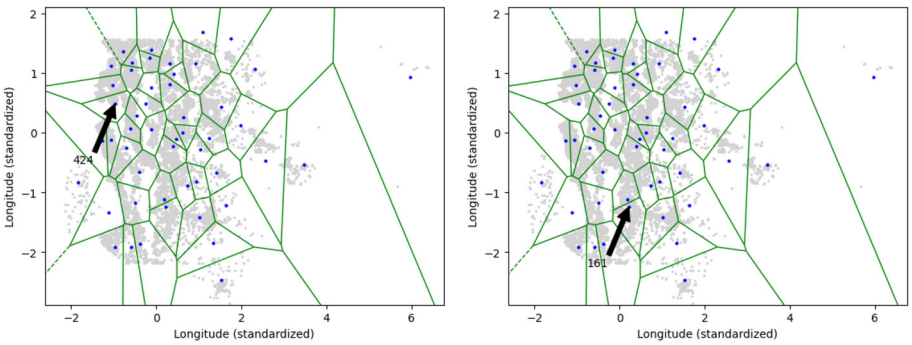

Fig. 6 Intepretation of the model as a zonal segmentation: each zone represents an area with approximately constant prices, where that constant is given by the obtained values v_m. Two examples are shown: **a** $v_m \approx 424$, **b** $v_m \approx 165$

than 5 prototypes with $\gamma_m(\mathbf{x}) > 0.01$. Furthermore, for over 94% of the test samples, only 1 or 2 prototypes significantly contributed to the prediction.

Interpretation as zonal segmentation

The model's output can alternatively be interpreted as a zonal segmentation: defining the mth "zone" as the area where all properties are closest to the mth prototype results in a Voronoi segmentation, as illustrated in Fig. 6. Each zone represents a geographical area with similar property prices, where the reference price is specified by the prototype value v_m. In Fig. 6, two examples of v_m are depicted to illustrate this concept.

This interpretation is connected to research in sub-market segmentation. According to Thibodeau (2003), "a housing [sub-]market defines a geographic area where the price of housing per unit of housing service [...] is constant". Therefore, our model could be interpreted also as a means to perform sub-market segmentation, since each prototype m roughly defines an area where, in its vicinity, the price is approximately a constant v_m.

It is noteworthy that many traditional approaches to sub-market segmentation rely on grouping predefined market-based or policy-based elemental areas, such as postcodes or school districts (see, for example, Table 1 in Borst and McCluskey (2008)). In contrast, our approach infers sub-markets automatically from raw data using a machine learning algorithm. This rather aligns us with recent works such as Rey-Blanco et al. (2024).

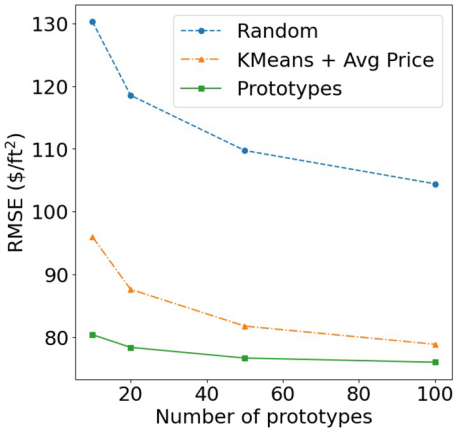

Fig. 7 RMSE of the different compared methods for different number of prototypes

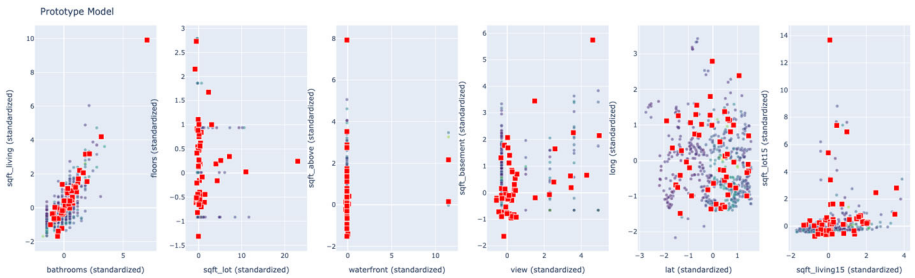

Fig. 8 Scatterplots for the 12-variable case, showing pairs of variables and the obtained prototypes

In any case, we emphasize this capability to produce market segments automatically as a practical by-product of the proposed model. An in-depth analysis of prototype-based models to the specific task of submarket segmentation would require a separate study, which we defer it to future work.

Results: On the effect of the number of prototypes

We now study the effect of the number of prototypes. We repeat the previous protocol using different number of prototypes and compare the three different methods with $M = 10, 20, 50, 100$. Results are reported graphically in Fig. 7.

As expected, the error decreases for all methods with more prototypes but the proposed approach gives the smallest errors across all values of M. Also, if we observe how the error increases by decreasing M, our model degrades with a smaller slope compared to the other models.

4.4 Model with more variables

Following the protocol established in the last section, we now incorporate a more comprehensive set of variables. Apart from latitude and longitude, we consider *number of bathrooms*, *square feet of living room*, *square feet of lot*, *number of floors*, *waterfront*, *view*, *square feet above*, *square feet of basement*, *square feet of living room in 2015*, *square feet of lot in 2015*. Thus, the new model uses 12 input variables in total.

Table 5 RMSE of the compared models using 12 input variables

Method	RMSE	Min RMSE	Max RMSE
Random prototypes	115	107	125
kmeans + Price Average	90.9	90.2	91.6
Prototypes (constant σ)	68.1	66.9	69.5
Prototypes (fitted σ's)	62.0	60.7	63.2

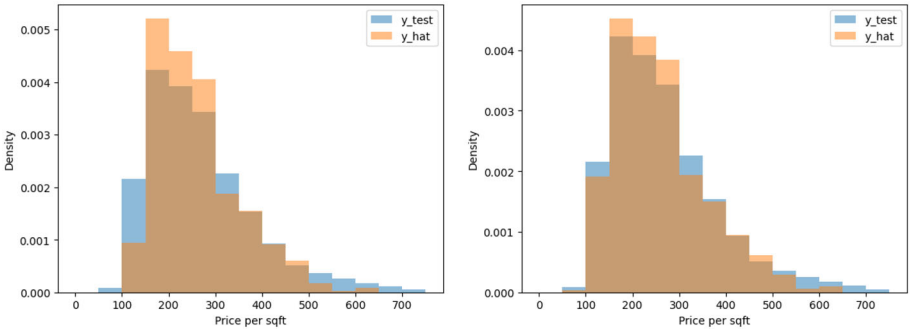

Fig. 9 Comparing the distribution of test prices with the distribution of estimated prices, for $M = 50$ (left)and $M = 100$ (right) prototypes

The experimental details are the same as in the last section, with one exception: because now we use several variables of different types, it is reasonable to expect that different variables need to be weighted differently for the importance computation of Eq. 2. Our model also accepts this design choice, which translates to fitting the values of σ_m of Eq. 1. For completeness, we will report the results using a constant hyperparameter σ for all prototypes (equivalently to the last section), and also by explicitly learning the σ_{md}'s as part of the optimization process.

The results are summarized in Table 5. Again, the table displays the values of RMSE for the same methods of the previous section. In this case, we found the random neighbor initialization of the KNN method yields a higher variance between the 10 runs. Therefore, we also report the minimum and maximum RMSE across the 10 runs to make clear there is no overlap in the error metrics.

There are some noteworthy observations when comparing to the results of the previous section, which leveraged location variables only. First, the RMSE of the prototype based method (using constant σ) has *improved* (down to 68.1 in Table 5 vs. 76.6 of Table 3). This underscores the positive impact of adding more variables in the prototype-based model. This impact is even larger when one considers the improvement brought by learning the parameters σ_m (down to 62.0).

We can still interpret the means $\mathbf{p}_m$ as the prototype modes—with the clarification that now these do not only specify the typical locations, but also the typical values of each variable considered. As an illustration, Fig. 8 displays six scatterplots, covering the 12 input variables, and also the projections of the prototypes in the corresponding subspace.

Second, when inspecting the results for the other 2 methods, we observe an opposite trend: in both cases the RMSE has not improved but *increased* as more variables are integrated (for K-Means, RMSE increases to 90.9 using the 12 input variables, up from a value of 81.7 with the model of the previous section; for KNN the increase is from 110 to 115).

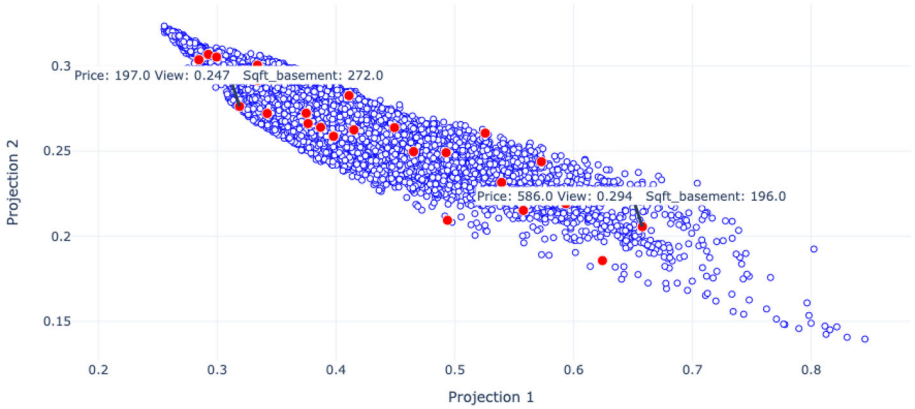

Fig. 10 Prototypes (red) obtained for the neural network embeddings of properties (blue). (Color figure online)

A possible explanation for this phenomenon is as follows. When we used latitude and longitude as the only input variables, the proximity to a prototype (both KNN and K-Means are based on Euclidean distance) is a good proxy for the similarity in price, as argued previously. However, when additional variables are introduced, the distance calculation between a property and the prototype becomes a composite measure, encompassing multiple components, some of which may not have a direct correlation with price differences. In essence, this expanded feature space introduces dimensions that are not necessarily indicative of variations in property prices. Consequently, similarity between variables does not translate to an effective similarity in price.[6] The reason why our prototype-based model exhibits the opposite behavior can be attributed to the model's ability to adjust the optimal locations of the prototypes (and, especially, the scales of each dimension). By doing so, it effectively hones in on the specific variables and combinations thereof that are most relevant for accurate price estimation.

Approximating the target distribution

Another research question we could ask is *do the model predictions* $\hat{y}$ *also approximate well the distribution of observed test prices* y? To that end, Fig. 9 displays the histogram of both test and estimated prices. For the case of $M = 50$ (the setting used in this section), indicated on the left, we see the distribution of estimated prices substantially overlaps that of real prices, albeit putting more mass in the center of the distribution and less in the extremes. However, this phenomenon is not surprising and, to some extent, expected, since prediction models can be interpreted as "smoothing the data" (and consequently reduce

[6] For instance, consider the K-Means method and property #4607 from the test set, which shows the largest error difference between the location-only model and the model with all features. This property has normalized coordinates $(-1.62, -1.05)$ and a normalized price of 0.118. The nearest prototype in the location-only model is at location $(-1.81, -1.05)$ with a value of v_9=0.149. In the model with all features, the closest prototype is at $(-0.55, -0.83)$, but the proximity has been driven by very similar values of the "waterfront", "floors" and "bathrooms" variables (and clearly not location). However, this prototype's normalized value is $v'_5 = 0.44$, which deviates significantly from the actual price. This is an example where similarity in specific variables does not necessarily imply a price similarity. The K-Means algorithm creates clusters (prototypes) without considering variable importance or property prices, leading to errors for properties like #4607 in the all-features model, which we verify is not an isolated case.

Table 6 Comparison of the RMSE of the different models (lower is better)

Method	RMSE	Num. parameters	Explainable
Regression tree	67.6	1748	YES
Random forest	65.8	1740	NO
Neural net	61.9	1321	NO
Prototypes	62.0	1250	YES

extreme observations). To verify this, if we increase M, as we use more prototypes we should expect less smoothing, which is clearly visible in the plot on the right for $M = 100$.

4.5 Comparison with other predictive models

So far, the prototype-based model has shown some interesting advantages in terms of predictive power and also in terms of explainability. And the previous sections have demonstrated that it outperforms other alternative ways of obtaining prototypes, both in the location-only case and when using multiple variables.

Now we compare the prototype-based model with three models discussed in Sect. 2: (i) regression trees, (ii) random forests and (iii) a standard neural network based on a multi-layer perceptron.[7]

First, we discuss the comparison settings. For all methods, we still use identical train/validation/test splits as in the case of the prototype-based model. We select the best hyperparameters by evaluating the RMSE of the trained models in the validation set. Then, we report the RMSE of the best hyperparameters on the test set. Again, we repeat this process 10 times and report the average RMSE, in order to mitigate factors such as randomness in parameter initialization.

For the regression tree, the only hyperparameter is the maximum number of leave nodes. The random forest also considers this parameter and also the number of trees. Finally, for the neural network we tune the following hyperparameters: the learning rate, the batch size, the number of epochs, and specific combinations of number of units and layers. Note that each setting of the hyperparameters determines the *size* of the model. For example, in a neural network, each unit has a number of coefficients equal to the number of units in the previous layer plus one. In the case of trees, each intermediate node has two parameters (the index of the variable and the threshold value), and the leaf nodes have one parameter (the value of the regression). We want to make sure the models are comparable to the prototype-based model in terms of the number of parameters. Therefore, when doing hyperparameter optimization, we restrict them to the ones that yield models (roughly) comparable in size to our prototype-based model (the model with $M = 50$ for the 12 input variables and learned scales σ_m has 1250 parameters).

Table 6 shows the RMSE of the different methods on the test set, the number of parameters of the models, and a column reminding whether the model is explainable.

In the case of the decision tree, by setting the maximum number of leaf nodes to 250, we get a tree with 1748 parameters, and the test RMSE is 67.6 \$/ft^2. In the case of the random forest, setting max leaf nodes to 50 and number of estimators to 5 yields 1740 parameters and a test RMSE of 65.8 \$/ft^2 (in the case of random forest, the error can be further reduced to 64.6

[7] Note this refers to a standard implementation of a neural network, not to the custom neural network discussed in 3.4.

$/ft^2, but using 50 estimators, which yields 17,400 parameters, many orders of magnitude higher than for our model). The best performing neural network has 3 hidden layers, with 22 units in each layer (amounting to 1321 parameters) and yields a test RMSE of 61.9 $/ft^2.

When we compare these values with the RMSE reported by the prototype-based model in the previous section (included in the table as a reminder), we verify that its error is significantly lower than for random forests and regression trees, and on par with the neural network. We conclude our model is comparable or better to the other alternatives, but with the added advantages, such as explainability.

4.6 Results with the neural network extension

Finally, we qualitatively evaluate the extension of the model as a neural network introduced in Sect. 3.4. Specifically, we illustrate the capability to apply the prototype based model to a so-called an *embedding* computed from a pre-existing neural network, instead from the input variables. As discussed in Sect. 3.4, this increases the complexity of the model but hinders explainability.

For the pre-existing neural network, we use the network trained in the last section, remove the output layer and add a 2-unit layer. This produces an intermediate representation which, consistently with Sect. 3.4, we denote as $\mathbf{h}$. Then, we feed $\mathbf{h}$ to the prototype-based model.

Implementation-wise, we proceed in two separate steps: first, we transform the properties $\mathbf{x}_n$ to embeddings $\mathbf{h}_n$ by computing a forward pass, and use these embeddings $\mathbf{h}_n$ to train the prototype-based model.

Because $\mathbf{h}_n$ has two dimensions, we can visualize the prototype distribution in this space. The result is in Fig. 10.

We visualize the prototype modes (red) on top of a sample of properties in the projected space $\mathbf{h}$. While it may not be possible to assign an interpretable meaning to each projection component, we identify some structure, e.g. top left is lower prices, bottom right higher prices. Also, by inspecting prototypes we can still assign a meaning to some of them, as indicated in the figure, based on the value (such as "expensive apartments with no basement and good views").

This experiment is to demonstrate qualitatively the potential extensions, but it suggests interesting directions for application and research.

5 Discussion

The main desideratum of the prototype-based model is to predict real estate prices mimicking the process of comparison-based valuation. The results in the previous section have validated that the model has the correct behaviour, in two aspects. First, the interpretation of the prototype distributions is as expected: qualitative analysis using data visualizations show that the prototypes are well-spread across all the space of input variables, as summarized in Fig. 8. As a matter of fact, when using spatial variables only (latitude and longitude) the prototypes spread well across all the different physical regions, as depicted in Figs. 4 and 5. Inspection of the prototype values also indicates a reasonable segmentation of the space and seemingly well-approximated price estimates.

Second, on top of this qualitative examination, the *quantitative* evaluation confirms that the prototypes and values obtained by the model, in different settings, are appropriate. The strongest argument for this is that the predictive errors obtained by the prototype-based model

are smaller than (i) alternative ways of constructing the prototypes, as summarized in Table 3, and (ii) alternative machine learning methods, as summarized in Table 6.

The experiments have focused on predicting the price per square foot, as this is a well-understood measure of unit of constructed land. We conducted experiments in two settings: predicting the price based only on location, and predicting based on location and other variables describing the property. The first setting is compatible well-acknowledged assumption in the real estate sector that location is the crucial factor of land price (Gabrielli et al., 2023; Atack and Margo, 1998). Even though this setting could be regarded as simplistic, a result like the one in Fig. 4 could be already interesting and actionable by firms with high stakes in the real estate market: it can be interpreted as a zonal segmentation of the real estate market, sometimes also referred to as a sub-market analysis (Goodman and Thibodeau, 2007) (as analyzed in Sect. 4.3). Effectively, this information alone already produces accurate predictions: by inspecting Tables 3 and 6, we actually note that the error of the prototype-based model using location-only variables is on par with those of machine learning models using all the variables. We conclude that location information alone, when coupled with a model that exploits this information adequately, is a very powerful signal.

Quantifying the location cliché

We could actually quantify the "location, location, location" cliche according to our data and model, which can be evaluated by comparing the $R^2 = 1 - RSS/TSS$ values of our regression models, where RSS denotes the residual sum-of-squares of the model under consideration, and TSS denotes the sum-of-squares with respect to the mean of the test data.

For the model using the location variables only, $R^2 = 0.56$, for the model using all the variables but constant σ, $R^2 = 0.64$, and for the model using all the variables, $R^2 = 0.73$. By inspecting the RSS values, the second model reduces the variance by about 20% compared to the first, and the third model reduces the variance by about 34% compared to the first. We see a big part of the variance is captured by the location variables, but adding more variables produces a significant reduction. According to this analysis, location is far from being the only factor that is significant.

However, these ratios also show an apparent limitation of the model, as the fraction of variance explained by the best model is $R^2 = 0.73$, indicating that there is a substantial portion of the variance not accounted for by the model, which can be attributed to unobserved variables or inherent irreducible noise within the dataset. This is not surprising given the complexity in the real estate market and is comparable to other studies (Tchuente and Nyawa, 2022). Some studies have indeed tried to cover this gap by capturing non-conventional variables such as the quality of the view from the window (Potrawa and Tetereva, 2022) using computer vision techniques.

In any case, this limitation points the need for caution when utilizing the model for individual sample predictions, as it may not adequately capture the intricacies of a specific data point. However, it is essential to recognize the utility of the model when used for group predictions or to summarize a real estate dataset with a reduced set of prototypes.

Modeling location in a principled way

Another aspect of the model is that it treats location in a general purpose way. Usually, location is difficult to model. Some previous studies added variables to the model such as the distance to different employment centers (Ottensmann et al., 2008). Those studies were focused on inference and precisely studied the effects of proximity to those key locations in the price. However, when the goal is price prediction with a general-purpose methodology that is valid across different markets, it will be cumbersome to specify the key locations for

each new study. The prototype-based model will capture the price changes in different areas regardless of the reason, without having to define specific variables.

Prototype distributions

The proposed model employs Normal distributions for the prototypes, for several reasons. First, we note previous prototype-based models, such as Li et al. (2018), work directly with the L2-norm between prototypes and samples. Therefore, using a distribution such as the Normal distribution represents a principled generalization. This has several advantages. First, as demonstrated in Table 5, using a distribution allows fitting additional parameters (σ's) and this has a beneficial impact in reducing the prediction error. Still, as discussed in Sect. 3.3, the significance of a prototype **p** in predicting a property **x** reduces to the Euclidean (or scaled Euclidean) distance between **p** and **x** when the covariance is diagonal. A second reason is that our `keras` implementation leverages the `tensorflow_probability` Python module, which supports Normal distributions. All in all, the choice of distribution aligns with common practice. However, it is important to note that our model is not limited to Normal distributions and can accommodate other types of unimodal distributions for the prototypes.

Explainability

Finally, it is also worth pointing this method could be considered a XAI method, because the prototypes and values can be thought of as a *summary* of the dataset and guide the explanation of predictions. This was already noted in a computer vision task (Li et al., 2018). While in real estate studies XAI techniques have also been used, we are not aware of previous prototype-based models—which are especially attractive for this scenario.

6 Practical and research implications

While the text has made reference to various practical implications of the model, they can be explicitly summarized as follows. First, the proposed model allows firms to perform a more accurate prediction of real estate properties. Consequently, firms can make better-informed investment decisions, optimize pricing strategies, and improve customer satisfaction by providing more accurate and understandable property valuations. However, where the model excels is in providing explanations in the forms of direct comparisons. Users of such a model can easily understand the rationale for a predicted price, on the basis of comparing it with a nearby "prototype", which could drastically change the relation in which real estate agents, and other stakeholders, interact with ML systems. Moreover, the prototypes can be used as "new data" to enrich operations of certain firms.

Beyond industry players, the model can also benefit governments and social actors by providing insights into the factors driving price dynamics. This understanding can aid in the development of social impact initiatives and policies aimed at addressing housing affordability and other real estate-related social issues. Thus, the model not only promises significant commercial benefits but also holds potential for broader societal impact.

From a research standpoint, this work introduces novel perspectives on interpreting machine learning models for real estate predictions. Traditionally, explainable artificial intelligence (XAI) approaches have concentrated on determining the relevance of each variable, as seen in studies by Lorenz et al. (2022) and Potrawa and Tetereva (2022). Even recent works, such as those by Bauer et al. (2023) and Holstein et al. (2023), have explored how real estate professionals utilize model information, still with a primary focus on feature importance but lacking other cues such as proximity or similarity, used traditionally by real estate agents. The

present work unveils this gap and demands for more intuitive and human-centric approaches in the real estate price prediction, with the aim of contributing to the transformation of this research field.

7 Conclusions and future perspectives

This article has introduced the application of prototype-based models in real estate valuation. Our research started with four desiderata that a model should ideally exhibit for real estate estimation: (a) non-linearity, (b) parametric learning using a loss function, (c) explainability, and (d) ability to implement the "direct comparison" criterion. Experimental results support that the proposed model has the four characteristics simultaneously, which was not the case of other machine learning models. Moreover, this is attained without sacrificing predictive accuracy: experimental results indicate that the prediction error obtained by the model is significantly lower than decision trees, random forests or KNN, and on par with neural networks (remarking, again, that the latter does not comply with the four characteristics discussed above). We also tried to highlight the usefulness of some of these characteristics, such as the capability to summarize a dataset with a reduced set of prototypes and the possibility to draw these prototypes on a map. We hope the method will be useful to operations research academics and practitioners which have shown previous interest in real estate valuation, as well as for the specific operations of some firms.

Future lines of research include using more complex form of the prototypes and values. In the present study, the prototypes are Normal distributions and the values are constant and we could explore other distributions or non-constant values. Perhaps if we could perform variable selection within the model we would obtain an even higher degree of explainability. Finally, we also acknowledge that there could be some use cases where researchers or practitioners are limited to the use of linear models (e.g. when they perform inference or work in an industry constrained by heavy regulation, such as banking or insurance). In such a situation, we could investigate if we could use the prototype-based model as a "feature engineering" method, where for example the obtained values of γ (Eq. 2) are used as additional location-related variables in the linear models.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

References

Alexandridis, A. K., Karlis, D., Papastamos, D., & Andritsos, D. (2019). Real estate valuation and forecasting in non-homogeneous markets: A case study in Greece during the financial crisis. *Journal of the Operational Research Society, 70*(10), 1769–1783.

Amédée-Manesme, C. O., & Barthélémy, F. (2018). Ex-ante real estate value at risk calculation method. *Annals of Operations Research, 262*, 257–285.

Atack, J., & Margo, R. A. (1998). Location, location, location! the price gradient for vacant urban land: New York, 1835 to 1900. *The Journal of Real Estate Finance and Economics, 16*(2), 151–172.
Bauer, K., von Zahn, M., & Hinz, O. (2023). Expl (AI) ned: The impact of explainable artificial intelligence on users' information processing. *Information systems research, 34*(4), 1582–1602.
Baydin, A. G., Pearlmutter, B. A., Radul, A. A., & Siskind, J. M. (2018). Automatic differentiation in machine learning: A survey. *Journal of Machine Learning Research, 18*, 1–43.
Biehl, M., Hammer, B., & Villmann, T. (2016). Prototype-based models in machine learning. *Wiley Interdisciplinary Reviews: Cognitive Science, 7*(2), 92–111.
Bishop, C. M. (2006). *Pattern recognition and machine learning*. Berlin: Springer.
Bitter, C., Mulligan, G. F., & Dall'erba, S. (2007). Incorporating spatial variation in housing attribute prices: A comparison of geographically weighted regression and the spatial expansion method. *Journal of Geographical Systems*, *9*, 7–27.
Borst, R. A., & McCluskey, W. J. (2008). Using geographically weighted regression to detect housing submarkets: Modeling large-scale spatial variations in value. *Journal of Property Tax Assessment & Administration, 5*(1), 21–54.
Breiman, L. (2001). Random forests. *Machine Learning, 45*, 5–32.
Chen, S., & Xie, W. (2022). On cluster-aware supervised learning: Frameworks, convergent algorithms, and applications. *INFORMS Journal on Computing, 34*(1), 481–502.
Choy, L. H., & Ho, W. K. (2023). The use of machine learning in real estate research. *Land, 12*(4), 740.
Colwell, P. F., & Dilmore, G. (1999). Who was first? An examination of an early hedonic study. *Land Economics*, 620–626.
Court, A. (1939). *The dynamics of automobile demand*. General Motors: Chapter Hedonic price indexes with automotive examples.
d'Amato, M., & Kauko, T. (2017). Advances in automated valuation modeling. *Springer International Publishing AG*. 10: 978-3.
Doumpos, M., Papastamos, D., Andritsos, D., & Zopounidis, C. (2021). Developing automated valuation models for estimating property values: A comparison of global and locally weighted approaches. *Annals of Operations Research, 306*, 415–433.
Fotheringham, A. S., Brunsdon, C., & Charlton, M. (2003). *Geographically weighted regression: The analysis of spatially varying relationships*. Wiley.
Gabrielli, L., Ruggeri, A. G., & Scarpa, M. (2023). Location, location, location: Fluctuations in real estate market values after covid-19 and the war in Ukraine based on econometric and spatial analysis, random forest, and multivariate regression. *Land, 12*(6), 1248.
Gibbs, C., Guttentag, D., Gretzel, U., Morton, J., & Goodwill, A. (2018). Pricing in the sharing economy: A hedonic pricing model applied to airbnb listings. *Journal of Travel & Tourism Marketing, 35*(1), 46–56.
Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.
Goodman, A. C. (1998). Andrew court and the invention of hedonic price analysis. *Journal of Urban Economics, 44*(2), 291–298.
Goodman, A. C., & Thibodeau, T. G. (2007). The spatial proximity of metropolitan area housing submarkets. *Real Estate Economics, 35*(2), 209–232.
Haas, G. C. (1922). A statistical analysis of farm sales in blue earth county, Minnesota, as a basis for farm land appraisal. Technical report.
Holstein, K., De-Arteaga, M., Tumati, L., & Cheng, Y. (2023). Toward supporting perceptual complementarity in human-ai collaboration via reflection on unobservables. *Proceedings of the ACM on Human-Computer Interaction, 7*(CSCW1), 1–20.
Hong, J., Choi, H., & Kim, W. S. (2020). A house price valuation based on the random forest approach: The mass appraisal of residential property in South Korea. *International Journal of Strategic Property Management*, *24*(3), 140–152.
Hussain, T., Abbas, J., Wei, Z., & Nurunnabi, M. (2019). The effect of sustainable urban planning and slum disamenity on the value of neighboring residential property: Application of the hedonic pricing model in rent price appraisal. *Sustainability 11*(4). https://doi.org/10.3390/su11041144 .
Isakson, H. R. (1988). Valuation analysis of commercial real estate using the nearest neighbors appraisal technique. *Growth and Change, 19*(2), 11–24.
James, G., Witten, D., Hastie, T., Tibshirani, R., et al. (2013). *An introduction to statistical learning*, (vol. 112). Springer.
Kaden, M., Bohnsack, K. S., Weber, M., Kudła, M., Gutowska, K., Blazewicz, J., & Villmann, T. (2022). Learning vector quantization as an interpretable classifier for the detection of sars-cov-2 types based on their rna sequences. *Neural Computing and Applications, 34*(1), 67–78.
Kok, N., Koponen, E. L., & Martínez-Barbosa, C. A. (2017). Big data in real estate? From manual appraisal to automated valuation. *The Journal of Portfolio Management, 43*(6), 202–211.

Kramer, O., & Kramer, O. (2013). K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, pp. 13–23 .

Li, O., Liu, H., Chen, C., & Rudin, C. (2018). Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI conference on artificial intelligence*, vol. 32.

Lins, M. P. E., Novaes, L. F. D. L., Legey, L. F. L. (2005). Real estate appraisal: A double perspective data envelopment analysis approach. *Annals of Operations Research*, *138*, 79–96.

Lorenz, F., Willwersch, J., Cajias, M., & Fuerst, F. (2022). Interpretable machine learning for real estate market analysis. *Real Estate Economics* .

Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., & Lee, S. I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence, 2*(1), 56–67.

Malpezzi, S., et al. (2003). Hedonic pricing models: A selective and applied review. *Housing Economics and Public Policy, 1*, 67–89.

McCluskey, W. J., Zulkarnain Daud, D., & Kamarudin, N. (2014). Boosted regression trees: An application for the mass appraisal of residential property in Malaysia. *Journal of Financial Management of Property and Construction*, *19*(2), 152–167.

McCluskey, W. J., McCord, M., Davis, P. T., Haran, M., & McIlhatton, D. (2013). Prediction accuracy in mass appraisal: A comparison of modern approaches. *Journal of Property Research, 30*(4), 239–265.

Ottensmann, J. R., Payton, S., & Man, J. (2008). Urban location and housing prices within a hedonic model. *Journal of Regional Analysis and Policy*, *38*(1) .

Owomugisha, G., Melchert, F., Mwebaze, E., Quinn, J. A., & Biehl, M. (2021). Matrix relevance learning from spectral data for diagnosing cassava diseases. *IEEE Access, 9*, 83355–83363.

Pagourtzi, E., Assimakopoulos, V., Hatzichristos, T., & French, N. (2003). Real estate appraisal: A review of valuation methods. *Journal of Property Investment & Finance, 21*(4), 383–401.

Peterson, S., & Flanagan, A. (2009). Neural network hedonic pricing models in mass real estate appraisal. *Journal of Real Estate Research, 31*(2), 147–164.

Potrawa, T., & Tetereva, A. (2022). How much is the view from the window worth? machine learning-driven hedonic pricing model of the real estate market. *Journal of Business Research, 144*, 50–65.

Rey-Blanco, D., Arbués, P., López, F. A., & Páez, A. (2024). Using machine learning to identify spatial market segments. A reproducible study of major Spanish markets. *Environment and Planning B: Urban Analytics and City Science*, *51*(1): 89–108 .

Tchuente, D., & Nyawa, S. (2022). Real estate price estimation in French cities using geocoding and machine learning. *Annals of Operations Research*, 1–38 .

Tchuente, D., Lonlac, J., & Kamsu-Foguem, B. (2024). A methodological and theoretical framework for implementing explainable artificial intelligence (xai) in business applications. *Computers in Industry, 155*, 104044.

Thibodeau, T. G. (2003). Marking single-family property values to market. *Real Estate Economics, 31*(1), 1–22.

van Veen, R., Meles, S. K., Renken, R. J., Reesink, F. E., Oertel, W. H., Janzen, A., De Vries, G. J., Leenders, K. L., & Biehl, M. (2022). Fdg-pet combined with learning vector quantization allows classification of neurodegenerative diseases and reveals the trajectory of idiopathic rem sleep behavior disorder. *Computer Methods and Programs in Biomedicine, 225*, 107042.

Zurada, J., Levitan, A., & Guan, J. (2011). A comparison of regression and artificial intelligence methods in a mass appraisal context. *Journal of Real Estate Research, 33*(3), 349–388.