The Best Location for You to Live

HABERT Paul, NEYPATRAIKY Yannick & RIMBOT Thomas
EPFL - Laboratoire d'Economie Urbaine et de l'Environnement
December 22, 2021

Abstract

***The Best Location for You to Live* is a tool that allows anyone to find their ideal residence place in a given city by specifying their personal level of importance for the location's visual appeal and environmental parameters. The underlying model is based on multiple layers of machine learning techniques applied to city streets images obtained with Google Street View's API. While environmental characteristics are strapped from multiple sources, images' appeals were derived from the results of an interactive survey. Each image was then associated to its vector of features by means of object recognition and image processing algorithms. Finally, regression and classification models were developed to find which features are the most significant for users and render possible the prediction of the most visually appealing neighborhoods in another city.**

1. Introduction

Where is your dream place to live? Most individuals have probably been confronted to such type of question during their lifetime. As simple as the question is, many factors come into account when anointing a given location as "perfect" for oneself. While environmental characteristics such as road noise or solar irradiation are easily quantifiable, a neighbourhood visual appeal can only be assessed qualitatively. This project aimed at providing a predictive model to quantify how appealing a given neighbourhood looks and combined this metric with environmental parameters to facilitate the search of the ideal residence place. To build this tool, the following steps were conducted: getting a dataset of streetview images, developing an interactive survey to label them, getting images' features to assimilate each view to a vector of features, carrying out regressions to analyse survey results and developing a predictive model.

As the title implies, the tool that was developed is modular and could be applied to any city worldwide. In this paper, it will be detailed how the model was defined for the city of Lausanne and how it could be adapted for another one.

2. Data

2.1. Getting the data

First, the assumption was made that a neighbourhood appeal can be captured by the view from its main street, hence why an algorithm that requires a Google Street View's API has been developed. Essentially, the API allows to get a streetview image from specified coordinates of the form (latitude, longitude) using a link. Other parameters such as the height and width in pixels of the image, the head (horizontal orientation) and pitch (vertical orientation) of the streetview have to be passed alongside the API key. Since the link is a simple string, methods were then derived to get a streetview by passing the required parameters as arguments, or its metadata to check if the view actually exists.

In order to get streetviews representative of a city's neighbourhoods, the road network was obtained through the python OSMnx library [1]. By getting the nodes and edges (streets) of the network as GeoDataFrames, it was possible to get a view at the middle of each edge. An edge is a LineString of two or more points, and the centroid where the view is taken lies between the two middle points of that line. For the sake of consistency, each view had to be aligned with the road. Hence, the bearing angle between the two middle points was computed to set the head of the streetview. Although this provided an overall clean and consistent dataset, outliers were bound to be present and required some cleaning process.

2.2. Cleaning

To clean the obtained dataset, a *Convolutional Neural Network*, more precisely the *VGG-16* algorithm [2], was used for image recognition. *VGG-16* contains a total of 24 layers: 16 with trainable parameters (13 convolutional and 3 fully connected), 1 input layer, 1 output layer, 5 pooling layers and 1 flattening layer. The algorithm takes as input an image and extracts its features by outputing a prediction vector $\hat{y}$, which gives the probabilities of the image belonging to the class labels (each class corresponding to a particular feature, e.g. "dog", "car").

After having extracted the features, a *Principal Component Analysis* (PCA) was carried out to reduce the dimension of the feature space. Finally, clustering was done with *k-means* to regroup similar images. This allowed to spot the unusable views and either remove or reorient them.

After preprocessing, 229 images were obtained. For each selected view, the image with the opposite head (+ 180 degrees) was also taken in order to have a fair assessment of each location since a view could be very appealing in one direction and displeasing in the other. This doubled the size of the dataset to 458 images on which the model could be trained, validated and tested.

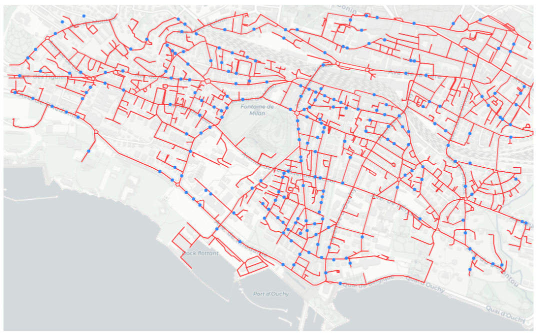


Figure 1: Final map with the selected views (blue dots) and road network (in red)

Here, the area was restricted to the bounding box of coordinates $(46.5065, 46.52, 6.605, 6.642)$ to reduce the dataset size. This allowed a trade-off between a precise ranking (many occurrence of each image in the survey) and a regression model which trains on a sufficient amount of images. This choice of location was also made due to its diversity (parks, railways, residential areas, etc.).

3. Features and ranking

3.1. Features to feed in the model

3.1.1 CNN for object recognition

Before training a model aiming to predict the images that people prefer, features had to be extracted. The first chosen feature was the objects spotted within each image. To do so, the so-called *YOLOv3* [3] algorithm was implemented. In a nutshell, a single neural network was applied to the full image to divide the image into regions and predict bounding boxes and probabilities for each region. These bounding boxes were then weighted by the predicted probabilities.

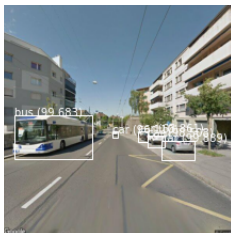
Features were then extracted by taking the type of detected object, as well as the box size, which reflects its proportion within the image. This box size approach can be biased in a way that the CNN only outlines rectangles, which is often far from the real shape of the detected object.

Furthermore, all images were very similar in terms of environment, which means that they all contained the same types of objects (trees, buildings, etc.), and the only objects variance among them were cars, people, or construction's signs. These can have a direct impact on appeal, if for example an individual had to choose between two exact same images where only one was cars' crowded. However object's impact can be considered limited relative to the environment the street is immersed in.

3.1.2 Images' properties

The remaining features were images' colors, which includes the brightness. This choice was made based on people's feedback, which corroborated the initial guess: green (trees) and light blue (sky) seem to be positively correlated with the choice, while grey (buildings), and darkness (narrow space) are negatively correlated. While these colors are pretty universal, others can be more ambiguous. For example, red can be very negative (construction works), or very positive (red trees, colorful houses, etc.). This can limit the model's performance and could be investigated further.

In order to extract these information, an *Image Segmentation* algorithm was implemented. Each pixel was associated to one of 12 color classes: black, light grey, grey, dark grey, white, light green, dark green, light blue, dark blue, light red, dark red, orange/brown. Then, these 12 classes were merged together to form 6 main classes: black, grey, white, green, blue, red. The corresponding features were then the number of pixels per main class.



(a) Detected objects on image

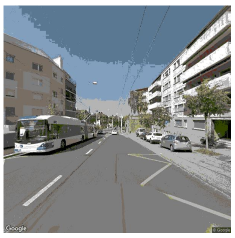

(b) Segmented image

Figure 2: Object recognition and image segmentation

3.1.3 Feature analysis

A PCA was carried out to check for potential correlations between features. This justified the removal of one color class. In fact, since the image is segmented, each pixel has to belong to a class. So if it is not in one of the first five color classes, it has to be in the sixth. As can be seen in Figure 3, the 6 final features (5 color classes + object size) are relatively independent.

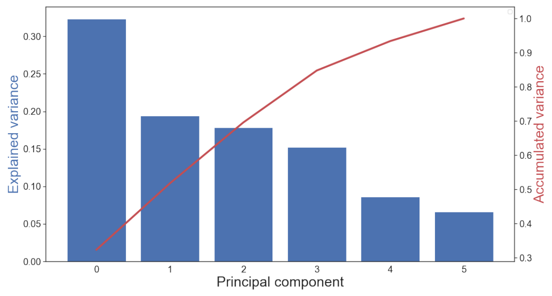


Figure 3: Explained variance of each PC and acc. variance

3.2. Survey for ranking

3.2.1 Insights

In order to gather data on people's preferences, an online survey was created, in which people can choose between two images. This discrete choice survey (accessible at *The Best Location for You to Live - Battle Survey*) was made with the Unity game engine and C#. This allowed to randomly generate battles for each person, and to make the survey more appealing by presenting it as a game, which led to more answers and thus a more accurate ranking.

Using these results, the goal was then to rank the images by giving them a score, and several possibilities were considered for that. The ranking model based on the win ratio was kept since the appeal score was built to create a continuous score range between 0 and 1. The accuracy of the ranking and the scoring system was important because it represented the output the model would train on. However, because the survey was put online and broadcasted to as many people as possible, some of them did not answer seriously, which led to outliers and fake data. For instance, they would purposely choose "wrong", or spam a side to quickly get to the end. To counter this, a unique ID was generated for each participant. Then, an algorithm was implemented to spot potential spammers by looking for repetitions, and intentional outliers by looking at the rank difference between two images. Above a certain threshold, it would delete all of these people's answers using the IDs. There were also unintentional outliers, i.e. people who misclicked, whose answers had to be removed.

3.2.2 Survey's results

After two weeks, the survey produced 16'688 answers (i.e. "image battles"). This corresponds to about 200 different people. After cleaning, 14'254 results were left to build the ranking upon. The score distribution is plotted in Figure 4.

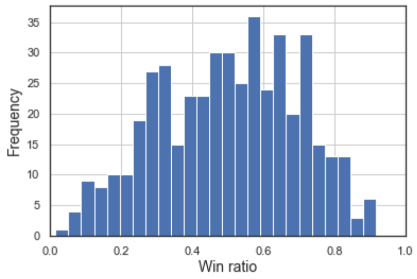


Figure 4: Distribution of win ratio on 458 images

Each image has been shown approximately 62 times which means that the win ratio could be precise up to 0.016.

4. Machine Learning models

4.1. Regressions models

After getting the data, cleaning it, gathering survey results and finding features from images, it was time to find correlations between these features and the appeal score. This was done with machine learning models. First of all, the data was divided into a training, validation, and test set with a 60/20/20 split. The appealing score of the test images was disregarded, and the objective was to get as close to the real appeal as possible with the predicted one. To do so, regression models were first used to try to find the exact value of the score.

4.1.1 Baseline

In order to compare the quality of the different models, a baseline model was computed to be the reference for further models. This model predicts the average outcome when talking about continuous values.

With this simple model, every score prediction is around 0.5, which is the average score. To assess the quality of these models, the Mean Squared Error (MSE) was inspected. The MSE is equal to 0.1798 for the baseline.

4.1.2 Models' training

Two models were tuned to get the best possible improvement against the baseline model: the *Linear Regression using LASSO regularization* and *Support Vector Regressor (SVR)*. Tuning was done with cross validation on the hyperparameters, respectively α and c.

α	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
MSE	0.158	0.158	0.160	0.192	0.201	0.201

(a) Linear Regression using LASSO regularization

c	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	10^{1}
MSE	0.192	0.188	0.166	0.146	0.154	0.185

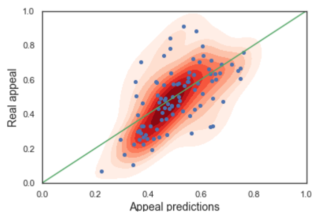
(b) Support Vector Regressor (SVR)

Table 1: MSE-mean on the validation set

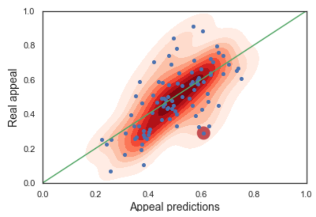
The best parameters for each model are $\alpha = 10^{-4}$ and $c = 0.1$, yielding a validation set's MSE of 0.158 for the linear regression model and 0.146 for the SVR model.

4.1.3 Appeal predictions

The test set represents 20% of 458 images, which leaves 92 appealing scores to predict. Plots of the predictions against real score are given in Figure 5. The green line represents the perfect regression that can be done, which exactly lies on the $x = y$ line where the actual and predicted score are equal.



(a) Lin. Reg. test, MSE = 0.1421



(b) SVR test, MSE = 0.1345

Figure 5: Regression predictions on the test set

Both models give a better prediction than the baseline when using the optimal hyperparameter. More precisely, the *Linear Regression using LASSO* led to a decrease of 21.0% of the MSE regarding the baseline, and the *Support Vector Regressor (SVR)* gave a decrease of 25.2%.

Therefore, the *SVR* slightly outperforms the *Linear Regression* on the test set, which was already the case on the training as discussed in subsubsection 4.1.2. The predicted appeal range is smaller than the actual one, which means that models tended to regress predictions to the mean to maximize the probability that they do not get far off the real appeal.

The most misclassified image in the optimal SVR model (red circle in Figure 5b) is shown in Figure 6.

It has appealing color features (coming from sky and bushes), so the model predicted a good appeal (0.61), but it also has a big building in the middle, which explains why people did not like it (real appeal of 0.29).

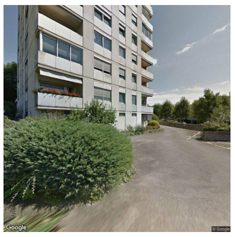


Figure 6: Worst SVR prediction

This image is also very different from the rest (it does not present a large road in the center), which might come from *VGG-16*'s initial clustering. A possible workaround would be to use more clusters for cleaning, but this would limit the model's generalization.

4.2. Classification models

After finding decent results with regression model, a step back was made. In fact, is it really necessary to find the exact appeal of each place in a city? Or could it be divided in few appealing categories? Dividing the continuous outcome into a discrete one would give a grade from 1 to 5 where 5 is the best, which seems to be a widely used grading system.

With that in mind, classification models were implemented. First, the discrete grading was built: appealing score within $[0, 0.2]$ corresponds to a 1, and so on until $[0.8, 1]$ which corresponds to a 5, as seen in Table 2. Also, the same 60/20/20 dataset split was used. The models' assessment measure was the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve. The AUC is an estimate of the probability that a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. For this reason, the AUC is widely thought to be a better measure than a classification error rate based upon a single prior probability.

Win ratio	0-0.2	0.2-0.4	0.4-0.6	0.6-0.8	0.8-1
Class	1	2	3	4	5
Freq.	32	109	160	131	26
[%]	7.0	23.8	34.9	28.6	5.7

Table 2: Classes' distribution for classification purpose

4.2.1 Baseline

Once again, a baseline model was set to assess the elaborated models' quality. In this new classification model, the AUC of the ROC curve was inspected to tune hyperparameters and improve it. The baseline predicted the most probable outcome, which is class 3 as seen in Table 2.

4.2.2 Models' training

Six classification models were implemented to try to get the highest AUC possible: a *Linear OLS model using LASSO*, a *Logit model*, a *SVM classifier*, a *K-nearest neighbors classifier*, a *Decision Tree classifier*, and a *Random Forest classifier*.

The optimal hyperparameters are summarized in subsubsection 4.2.3, where the AUC was measured with a *One-vs-Rest* target and with the *micro* averaging method. This treated the multiclass case in the same way as the multilabel case and calculated metrics globally by considering each element of the label indicator matrix as a label.

4.2.3 Appeal predictions

After testing models with the optimal hyperparameters, the AUC in Table 3 were found.

	base	lin.	logit	SVM	KNN	DT	RF
HPar.	-	α	c	c	n	d	d
Value	-	0.02	0.12	0.9	57	2	2
AUC	0.50	0.67	0.78	0.74	0.74	0.71	0.74

Table 3: Hyperpar. and Test AUC for classification models

The AUC of each model on the test set allowed to select the best classification model among all. In this case, and given the dataset cleaning's choices, the best model was the *Logit* one, with an AUC of 0.78.

Given the 5-class classification problem, the best model's confusion matrix found in Figure 7 was drawn on the test set, which contains 92 images.

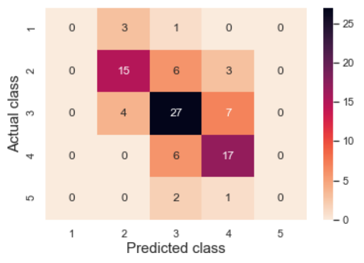


Figure 7: Confusion matrix for the Logit model

From the confusion matrix, it can be seen that the *Logit model* did not take a lot of risk in terms of farness to the baseline because it did not predict any image belonging to classes 1 or 5 (i.e. worst or best images). Therefore, it made mistakes for these actual classes. It would be interesting to study its behavior with a larger test set to see if it chooses to be conservative by predicting a class close to the most probable outcome (i.e. class 3), or if it predicts extreme classes this time.

Alternatively, lowering the number of classes causes an increase in the AUC, which was expected because less classes lead to a higher chance to get a prediction right.

5. User tuning interface

Since the training, testing and validation processes were carried out on the network area, it made sense to apply the obtained weights to assess the visual appeal of all the streets within the region. To do so, the Streetview-based application was reused to get all multiple views along each edge of the network. Overall, around 1300 images were obtained. The features were computed through image segmentation and object detection, thus allowing to predict each image's rank or class. It can be noted that the modularity of the procedure allows to make predictions on any other city with comparable environments.

The goal of the user interface is to represent how much a location is likable to a given individual. Data such as the solar irradiation, road and rail noise can be obtained through the *Vaud's Canton* data [4] [5]. The current model adds one information that is difficult to quantify, which is the appeal of the street below a building.

By combining those multiple layers, a map can be drawn. On *The Best Location for You to Live - User Interface*, anyone can get started with the visualisation made with SVR predictions. The color scheme goes from red to green (i.e. from worst to best location) and the user can tune the parameters according to its own preferences.

6. Discussion

A few points can be discussed. Firstly, about survey results. Because the study was realized as part of an EPFL project, the vast majority of answers came from students around 22 years old, most of them living in Lausanne. While some were outside of this category (some participants were above 70 years old, and some lived in other countries), a wider range of people could give a more generalizable model. Such model could then be expanded to have different appeal predictions with regards to the users age, preferences. Moreover, the outlier detection was subjective: what is the limit between actual outlier and personal preference? It was also next to impossible to detect spammers who would randomly pick a side. Also, it might be interesting to study the psychological pattern behind participant's answers. It is possible that they answered very seriously at the beginning, then got bored and wanted to quickly jump to the end. Alternatively, they might have been indecisive at the beginning, then understood what their preferences are and automatically go for the best image (trees' appeal for example).

Secondly, about features. Instead of image segmentation for extracting color values, it was first considered to get the pixels with RGB values above/below a certain threshold, which ended up being less precise. Something that was also tested was to augment the images' contrast in order to sharpen the differences between colors, as well as the brightness. While this proved efficient for some models, it did not yield better results overall. Further investigation could be carried out on this matter.

Thirdly, the very nature of the *best place to live* is extremely subjective, even in terms of visual appeal only. A good showcase of this is that, if instead of image segmentation, the more accurate VGG features (which are more than 400) are fed directly to the model, the performance does not improve. This could indicate that it is close to impossible to accurately predict a general people's preference.

7. Conclusion

In this project, different models were implemented to predict visual appeal based on image analysis. After tuning and comparison, an AUC score of 0.78 was attained, and new predictions on a fresh dataset were made. Together with data considering locations' environment, ***The Best Location for You to Live*** is now a tool that anyone can try to find the best location to live in given their own preferences. This tool can be used to help buildings' owners such as investments companies, or real estate agencies, to attract new house seekers by offering them the best accommodation fit. It is easy to use, and can be generalized to any region in the world.

Acknowledgements

The authors of this project would like to thank the Laboratoire d'Economie Urbaine et de l'Environnement, and more particularly Professor Philippe Thalmann and his doctoral student Adam Swietek for their precious advice. Many thanks to everyone who answered the discrete choices survey, and to Professors Jamila Sam and Barbara Jobstmann for allowing the authors to present this project to their class.

References

[1] OSMnx: Python for Street Networks, *Geoff Boeing*, 2016.
https://geoffboeing.com/2016/11/osmnx-python-street-networks/

[2] Convolutional Networks - VGG16, *Jason Osajima*, 2018.
https://www.jasonosajima.com/convnets_vgg.html

[3] You Only Look Once: Unified, Real-Time Object Detection, *Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi*, 2016.
https://arxiv.org/pdf/1506.02640v5.pdf

[4] Bruit: Cartes et géodonnées, *Office fédéral de l'environnement OFEV*, 2021.
https://www.bafu.admin.ch/bafu/fr/home/themes/bruit/etat/cartes.html

[5] Thèmes du guichet cartographique cantonal, *Canton de Vaud*, 2021.
https://www.geo.vd.ch/themes.html

[6] The Best Location for You to Live - Battle Survey, *Habert, Neypatraiky, Rimbot*, 2021.
https://toto1205.itch.io/ml2-project

[7] The Best Location for You to Live - User Interface, *Habert, Neypatraiky, Rimbot*, 2021.
https://the-best-location-to-live.github.io/the-best-location-to-live/