# Spatially Inspired Price Prediction for Car Rentals

**Farhood Ensan**
fensan@ucsd.edu
University of California San Diego
La Jolla, CA

**Kaushik R. Ganapathy**
krganapa@ucsd.edu
University of California San Diego
La Jolla, CA

**Jiaxi Lei**
jil119@ucsd.edu
University of California San Diego
La Jolla, CA

## ABSTRACT

The presence of spatial dependencies has often been neglected during the development of modern day machine learning models. In this paper, we attempt to take into account inherent *spatial* dependencies, with an ultimate goal to build a novel machine learning system to predict car rental prices using data from *Turo*, a peer to peer car-sharing company.

## KEYWORDS

turo, spatial, machine learning, random-forest, geo-spatial

**ACM Reference Format:**
Farhood Ensan, Kaushik R. Ganapathy, and Jiaxi Lei. 2019. Spatially Inspired Price Prediction for Car Rentals. In *CSE-158: Final Project, Autumn 2019, Recommender Systems and Web Mining*. University of California San Digo, La Jolla, CA, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Prediction models have changed the way we estimate the value of everyday commodities. For instance, prediction models using data from AirBnB, has led to many insights on apartment rental pricing, which, in the past was predominantly determined by human subjectivity. These prediction models often incorporate features which are directly related to or a part of the system on which predictions are being made. For instance, while predicting nighly rental prices in AirBnb apartments, features such as the number of bedrooms, the size of the apartment, presence of a swimming pool, amongst others were considered.

Despite having seen widespread adoption and success, these classical prediction systems (e.g. AirBnb) ignored the presence of any spatial dependencies encoded implicitly or explicitly in the data. Encoding and using these spatial dependencies for predictive building prediction models, had led to the rise of spatially inspired approaches such as spatial regression, spatial clustering, and spatial random forest models, which inherently encode spatial dependencies present in data.

In the following paragraphs, we attempt to compare a spatially inspired Random Forest and Linear Regression models, with classical models, for predicting daily rental rates, using data from Turo.



*CSE-158, Dec. 03, 2019, La Jolla, CA*



## 2 DATA COLLECTION AND PRE-PROCESSING

(A) Data Collection

(a) **Car Rental Information**
The first dataset utilized was obtained from the github profile of *riley106*, and consisted of a 330MB JSON file with a heavily nested format. The inital goal for data processing was to convert the data to a flat file format to facilitate further processing. Upon initially reading in the datasets using the Python Package Pandas, it was found that the dataset consisted of 36000 datapoints, each of which corresponded to a single car rental event. Given the size of the dataset, and the heavily nested nature of the data source, it was soon found that simple iteration and Pandas apply functions were impractical to flatten the dataset. Hence, measures were found to parallelize the process of flattening the input data using frameworks such as Dask and Numpy Vectorization, which greatly reduced the time required to flatten the data source.

(b) **U.S Census and Zip Code Data**
On flattening the data source, we noticed that out primary data source (1), contained latitude longitude information for each one of the rental car locations. Using intuition, we decided to obtain the zip code from the latitude longitude coordinates using the Python Library *uszipcode*. Using the zip codes. Additionally this data was merged with the Zip code level median income levels obtained from the U.S Census Department data from *census.gov*. The intuitive notion is that the income levels of a place

could potentially dictate the rental car price in the surrounding area.

(c) **Webscraping Google Trends**
On a preliminary analysis of the primary dataset, it was found that there were no features which encoded for the popularity of each car. Hence, we decided to use webscraping, particularly using the *Google Trends* API to get the the relative popularity of a particular car over the last 1 year. The google trends API was queried using a combination of year, make, and model for each unique combination of the 3 attributes in the dataset (e.g. 2017 Toyota Camry). The popularity scores were averaged over the last 12 months, and the resultant value was used as the popularity of the particular car. A more detailed explanation of the interpretation and the intuitive notion of these values can be found in the following section. This was then added as a column to the primary dataset.

(B) Some Key Extracted and Engineered Features
Some of the key features [1] extracted and engineered from a combination of the three data sources were

(a) userID: The userID is a unique ID for a particular car host in the dataset.
(b) carID : The carID represents a unique ID for a particular car in the dataset.
(c) Rating: The Rating represents the rating given by a particular user for the car for a particular trip. The rating field had some missing values, which were imputed using relevant strategies as outlined below: For every rental in the event that a rating was not provided, the first choice was to impute the missing value with the average value for the particular carID. In case the car had never been rated before, the average rating for that renter's car's was chosen. Ultimately, if the renter's car ratings were not present, the global average of all ratings were chosen as a last resort.
(d) Response Time: The Response Time is the time taken by a user to respond to a particular user rent request. Upon un-wrangling the data from the data set, it was found that this column had many inconsistencies such as certain values being in hours, and some in minutes. A standardization operation was performed, and ultimately all response time values were in minutes. Missing values were encountered in the response times, so in case a particular response time was missing, it was imputed with 1440 minutes (a whole day).
(e) Response Rate: The Response Rate is the number of times on average a host replied to a rental request (in percent). In case no data was present, it was imputed with a 0 percent.
(f) Listing Difference [2]: This feature is an engineered feature, which represents the difference between the the year in which a car was listed on the website, and the year in which the car was released. Intuitively, if this difference is high, it should ideally mean that either this car is a classic (high rates), or is a really old car at the end of its lifetime.
(g) Income, and ZIP: This feature represents the median income of the area, and ZIP code of the latitude and longitude of the location of from where the car rental took place. The idea is that if the median income is lower, prices for a rental may be lower since it could mean that the area is poorer.
(h) Weekday, Month: Represents the day of the week and the month on which the rental took place respectively. These 2 features are meant to detect trends in weekly or seasonlity in car rental prices if any present.
(i) Car Popularity: This feature represents is overall car popularity as determined from Google Trends. The Input source being a year, model and make for a given car. The value of this feature ranges from 0 to 100, with 0 being extremely uncommonly searched on Google across the United States in a 12 month period. Thus, intuitively, a high value of popularity should indicate that the car is relatively popular, and a lower value should indicate a fairly uncommon or exotic car, which in effect has a direct impact on the price [3].
(j) Rate: The price for each particular car rental. This is our response variable.

(C) Key Pre-processing Steps In addition to using hardware acceleration, for unraveling (un-nesting the data), several pre-processing steps were undertaken. For instance, the time of listing of the car was provided in the form of an EPOCH, and required the usage of the datetime package in Python in order to render and extract useful features from it. Similarly, all the latitude longitude coordinate pairs were converted into Geomtric (GIS) points using the *shapely* framework in Python using a traditional EPSG:4326 projection space. This step would be critical for the development of a spatially realized model downstream.
Finally in order to build a spatial data model, use geopandas to use the newly created shapely points,

[1] A total list of feature descriptions can be found in the attached appendix.

[2] See appendix for referenced feature here

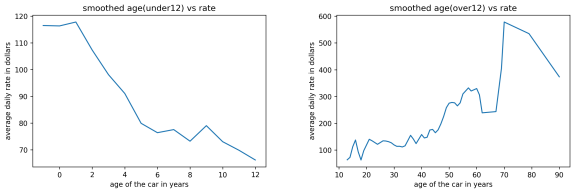
[3] This notion is proved in the exploratory segment onward

and then write the entire dataset, with all features, and the spatial information (Shapely Points) to a shapefile, which happens to be the propietry format used by Geo spatial analysis.
Additionally all of the boolean features were converted into decimal features, and all of the categorical features were encoded using Label Encoders in the sklearn library in Python in order to allow these to be used by machine learning models.

## 3 EXPLORATORY ANALYSIS

(A) Common Attribute Exploration

The data set consisted of 36279 individual rentals on the website Turo. Within the data set there were 58 different types of makes of cars, and 837 unique car models. The model years spanned from 1927 through 2019 for each car. The most expensive car in the data set was a Porsche 718 Cayman, costing $1999 to rent for a day. On the other hand, the cheapest car to rent was a 2006 Kia Rio, or a 2011 Toyota Camry costing $10 to rent for a day. Amongst the manufacturers, Toyota was the most common, having 4022 cars on Turo. The most uncommon car in the data set was a Yugo, having only 1 car listed. Amongst the cars listed, 94% of them were Automatic Transmission, and the rest had manual transmission. Within the listed vehicles, 66% of them were sedans whereas 25% of them were SUV's, and 4% being trucks. The rest 3% were vans. On average sedans costed $96 a day, SUV's costed $108 a day, trucks costed $91 a day, and vans costed $72.5 a day.
Specifically onto one feature, we analyze the daily rate of the cars vs the age of them.

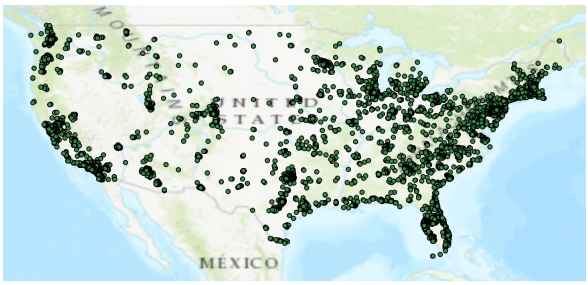



As shown in above figures, for newer cars, the average daily rate decreases as the cars get older and lose their values. However, as the cars get older past a certain point (about 12 years based on our exploration), the value on the cars starts increasing. intuitively, antique cars are a lot more expensive than new cars on average. Based on the smoothed figure above, an antique car that is at least 70 years old, is about 4 times more expensive than a relatively new car.

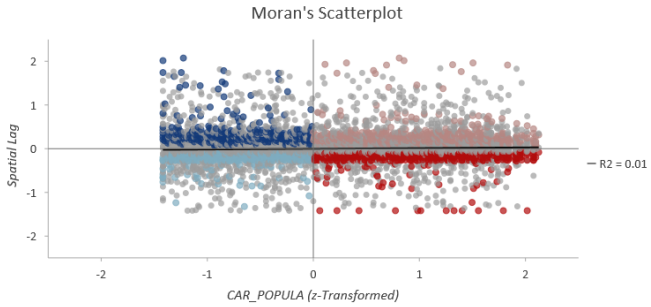
(B) Spacial Significance

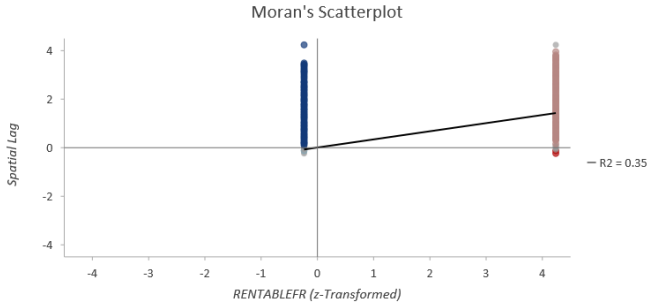
Taking the dataset's Latitude and Longtitude attributes and plotting them on a map, it is significant that Turo data points clusters major urban areas such as Los Angeles and New York City.

Geographical Distribution of Data Points




The above graph justifies that A spacial model would potentially be more ideal for our predictive task.
Furthermore, we analyze the spatial significance of certain features in the dataset. Here we present (in the below figures) the Moran Scatterplot of features *Car Popularity* (from Google Trends) and *Rentable From Airport*, which explores the similarity of data between its geospatial neighbors.




The Moran Scatterplot we generated shows that *Rentable From Airport* is reasonably correlated with the geospatial feature. On contrast, *Car Popularity* has no spatial correlation.

## 4 CLASSICAL MACHINE LEARNING MODEL

Having flattened the dataset, and extracted the relavant features, we can now begin to build a classical prediction model. It must be noted that any spatially dependent variables such as ZIP code, Income levels, and latitude or longitude were

removed from the feature-set $f$. On a preliminary glance of the feature set $f$, we can see that there are a number of categorical variables. Some of these categorical variables include the make of the car, and the model of the car. These categorical features can be dealt with 2 main ways

I. One Hot Encode the values of the Categorical Variables
II. Associate a number (categorize) each Categorical Variables.

Having seen in the exploratory segment that we have around 837 unique car models, and around 58 makes. Following strategy I. to deal with these many categories would lead to a *huge* number of columns being added to the feature set, and might cause extreme slowdowns while running a model such as a Linear Regression, or a SVM based Regressor. Dealing with this then would involve the usage of an appropriate Dimensionality reduction tool, which would involve the use of PCA. However, since the number of reduced dimensions is an arbitrary choice, and could determine the performance of the regression tool, we did not use this approach.

The second option would be to associate a number of each category (Ford = 1, Toyota = 2 etc.). However, the disadvantage is that it renders these features unusable by a model such as Linear Regression, of SVM based regression tool. On the other hand, having numerical categories is great for a decision tree based classifier! Hence, we decided use a Random Forest Regressor from sklearn to make our prediction model. We chose a Random Forest over a classical Decision Tree based regressor, since a Random Forest is inherently an Ensemble model, which is based on Bagging (Boostrap Aggregation), this means that the model would generate $k$ number of uncorrelated decision trees, and generate a output based on the majority value from these trees.

When we trained our model, we chose the number of trees to be 100, since we observed there was no improvements in validation set accuracy beyond this point for the hyperparameter.

Following a training of the Random Forest regression tool, we found the $R^2$ value obtained for the above dataset with a Random Forest Regression tool was 0.6115, and the Root Mean Squared Error for the classifier was 70.35.

This performance seems to be moderately good considering the relatively small size of our dataset.

Additionally here are the list of feature importances as determined by the classifier.
Feature Importances

| Feature | Importance |
|---|---|
| model | 0.163281202 |
| car popularity | 0.139191989 |
| make | 0.120859226 |
| rating | 0.084838697 |
| carID | 0.067868449 |
| renterID | 0.062844489 |
| renterTripsTaken | 0.050105886 |
| difference | 0.049427378 |
| year | 0.038869465 |
| distance | 0.032564916 |
| reviewCount | 0.031211299 |
| month | 0.029480026 |
| weekday | 0.026189532 |
| listHour | 0.022568032 |
| responseTime | 0.020364385 |
| type | 0.015641229 |
| responseRate | 0.013030761 |
| freeDelivery | 0.008148684 |
| instantBookDisplayed | 0.006886693 |
| automaticTransmission | 0.005941694 |
| listYear | 0.00448798 |
| businessClass | 0.003204402 |
| newListing | 0.001634248 |

As we can see here, the most important features happen to be model of the car, followed closely by the popularity of the car (an independently engineered feature), followed by the rating and the carID. This goes with our intuitive notion of what constitute high car prices. Hence, despite having mediocre performance, the important features do follow with Human intuition. However, the type of the car, which indicates whether a car is a car, truck, or SUV [4], has little to no importance, as is the year of listing the car. This seems to go against our intuition, which seems quite interesting to note indeed.

## 5 SPATIAL MODEL

Unlike a typical classical model, where we could directly go ahead and begin the process of transforming the dataset to be fit with a model, the first step in the development of the spatial model is to verify whether any spatial relationship exists between the response variable (price/rate), and spatial factors. This is done by constructing a weights matrix W, where the entries are weighted by a function $f$. In this case the function $f$ is chosen to be the inverse distance. Meaning that points close together are weighted more heavily than points away. Thus, the spatial metrics W consists of elements $W_{ij}$ where it represents the "spatial-influence" of element $W_i$ on $W_j$. Further, the influence on an element with itself

[4] See Appendix

$W_{ii}$ is assumed to be zero. The computation of the spatial matrix is done with the help of ArcPy, a derivative of Python developed by ESRI for the express purpose of geospatial analysis. We then compute the global Moran's I statistic for our response variable(rate), which is essentially determined by the formula:

$I = \frac{N}{W} \frac{\sum_i \sum_j w_{ij}(x_i - \bar{x})(x_j - \bar{x})}{\sum_i (x - \bar{x})^2}$ Given a computed weights matrix W, and x being the values of the response variable, and N being the number of spatial elements.

Upon computation and using the global moran's I function in ArcPy, we obtain the following results:

```
Global Moran's I Summary
Moran's Index:   0.277859
Expected Index:  -0.000032
Variance:        0.001920
z-score:         6.342526
p-value:         0.000000
```

The value of 0.27 suggests the presence of a statistically significant (p values = 0) degree of spatial auto-correlation (1, or -1 being perfectly auto-correlated). This gives us a motivation that spatial factors may be at play, and that spatial machine learning models may yield better outputs.

### Spatial Random Forest Model

Now that we have a motivation to build a spatial regression model, we once again decide to build a Random Forest Model by incorporating Spatial datapoints.

In order to do this, we utilize the Geographical Random Forest from the Analysis toolkit in ArcPy. For an example consider the Random Forest model which makes a prediction from Features $f$. $y_i = A\ x_i + \epsilon$ where x is the non linear prediction from the Random Forest using the feature set $f$ for each data-point $i$ in the dataset.

The Spatial Random Forest Model would then fit a model like so:

$y_i = A\ (u_i, v_i)\ x_i + \epsilon$ where $u_i, v_i$ is a point of interest, and smaller Random Forest model is fit to only a small cluster of data-points around the particular point of interest. This process takes place for each datapoint $i$.

Considering our Point features with Latitude and Longitude, we then fit a Random Forest Model with the help of ArcPy, as stated above. Attached below are the results obtained:

| | |
|---|---|
| Number of Trees | 100 |
| Leaf Size | 5 |
| Tree Depth Range | 28-40 |
| Mean Tree Depth | 32 |
| % of Training Available per Tree | 100 |
| Number of Randomly Sampled Variables | 7 |
| % of Training Data Excluded for Validation | 10 |

---------- Model Out of Bag Errors ----------

| | | |
|---|---|---|
| Number of Trees | 50 | 100 |
| MSE | 6680.612 | 6349.539 |
| % of variation explained | 48.356 | 50.915 |

---------- Top Variable Importance ----------

| Variable | Importance | % |
|---|---|---|
| car_popula | 52405997.09 | 16 |
| model | 43324133.94 | 13 |
| make | 38440602.33 | 12 |
| rating | 30412404.71 | 9 |
| month | 27743171.37 | 8 |
| year | 23100987.77 | 7 |

----- Training Data: Regression Diagnostics ------

| | |
|---|---|
| R-Squared | 0.929 |
| p-value | 0.000 |
| Standard Error | 0.001 |

---- Validation Data: Regression Diagnostics -----

| | |
|---|---|
| R-Squared | 0.514 |
| p-value | 0.000 |
| Standard Error | 0.007 |

As we can see, there is a statistically significant regression result with a strength $R^2$ of 0.514 on the validation set, and the Root Mean Squared Error was 79.68 This was constructed by taking a poll from 100 random trees constructed on the data-set. The degree of correlation $R^2$ is not extremely high, but however, does suggest that the model is moderately good at predicting the price.

A good note, is that the car popularity (an engineered feature), is the most important feature selected by the model. Additionally, model, make, month, and year of release of the car are the other determining factors in the Random Forest Regressor. This goes with our intuitive notion as well.

## 6 COMPARISONS BETWEEN MODELS

When we compare the $R^2$ value as obtained by the spatial Random Forest, and Classical Random forest we see quite clearly that the $R^2_{spatial} = 0.514$ is lesser than that of the $R^2_{classical} = 0.6115$. However, the magnitudes of these $R^2$ values are in fact close. However, the smaller value from the spatial model does not mean that it is necessarily worse. In fact, we attribute the decrease in the $R^2$ value to the presence of duplicate locations in the data set. This was because the location provided to us was that of the renter. Since there were multiple rentals for each renter, these led to the creation of duplicates from a spatial context. Hence, as a result, the duplicate records for each renter had to be dropped, with the values to be the mean of rental from that particular renter. This resulted in the loss of around 6000 data points, which

could have ultimately made an impact to help the predictive models. We anticipate that the $R^2$ should improve if the data set had more unique rental locations, thus leading to more unique spatial data points.

Another interesting observation, is the overall change in the list of feature importances from the regressor. In the classical model, the car popularity, model and make, and the car rating from the users, car and renter ID were the top features. However, while the top four feature remain the same, the spatial regression tool also placed much more greater weight on the year of release of the car, as well as the weekday and the distance driven during the ride. In particular the release year, and distance has nearly double the importance in the spatial model when compared to the classical model.

The difference in the feature importances clearly demonstrate how taking spatial relationships into account changes the fundamental nature of the prediction system, thus changing the output.

## 7 CONCLUSION

The development of spatial models has indeed been changing the face of classical machine learning systems. While the spatial Random Forest model which we developed failed to show any significant improvement from that of a classical system, it did show the change in relative feature importances when spatial features are taken into account. Additionally, the Spatial Regressor was able to attain similar degree of performance, despite having access to 6000 fewer data points, owing to the presence of spatial duplicates. Thus we anticipate that the potential for improving the performance of this model, when given a richer (less-spatially duplicate) data set is extremely high. Furthermore, trying a variety of spatial models such as Geographically Weighted Regression, or even Ordinary Least Squares Spatial Regression could potentially yield better results. Ultimately to conclude, having known that all prediction models are inherently wrong, the development of spatial approaches in machine learning systems yields for models, which encode important information, thus making them more useful!

## 8 REFERENCES

(1) Stefanos Georganos, Tais Grippa, Assane Niang Gadiaga, Catherine Linard, Moritz Lennert, Sabine Vanhuysse, Nicholus Mboga, Eléonore Wolff Stamatis Kalogirou (2019) Geographical random forests: a spatial extension of the random forest algorithm to address spatial heterogeneity in remote sensing and population modelling, Geocarto International, DOI:
(2) ESRI 2011. ArcGIS Desktop: Release 10. Redlands, CA: Environmental Systems Research Institute.
(3) Tang, Emily, and Kunal Sangani. Neighborhood and price prediction for San Francisco Airbnb listings. (2015).
(4) Li, Hongfei, Catherine A. Calder, and Noel Cressie. "Beyond Moran's I: testing for spatial dependence based on the spatial autoregressive model." Geographical Analysis 39, no. 4 (2007): 357-375.

## 9 APPENDIX

### Features

(1) userID: The userID is a unique ID for a particular car host in the dataset.
(2) carID : The carID represents a unique ID for a particular car in the dataset.
(3) Rating: The Rating represents the rating given by a particular user for the car for a particular trip. The rating field had some missing values, which were imputed using relevant strategies as outlined below: For every rental in the event that a rating was not provided, the first choice was to impute the missing value with the average value for the particular carID. In case the car had never been rated before, the average rating for that renter's car's was chosen. Ultimately, if the renter's car ratings were not present, the global average of all ratings were chosen as a last resort.
(4) Response Time: The Response Time is the time taken by a user to respond to a particular user rent request. Upon un-wrangling the data from the data set, it was found that this column had many inconsistencies such as certain values being in hours, and some in minutes. A standardization operation was performed, and ultimately all response time values were in minutes. Missing values were encountered in the response times, so in case a particular response time was missing, it was imputed with 1440 minutes (a whole day).
(5) Response Rate: The Response Rate is the number of times on average a host replied to a rental request (in percent). In case no data was present, it was imputed with a 0 percent.
(6) Listing Difference: This feature is an engineered feature, which represents the difference between the the year in which a car was listed on the website, and the year in which the car was released. Intuitively, if this difference is high, it should ideally mean that either this car is a classic (high rates), or is a really old car at the end of its lifetime.
(7) Income, and ZIP: This feature represents the median income of the area, and ZIP code of the latitude and longitude of the location of from where the car rental took place. The idea is that if the median income is lower, prices for a rental may be lower since it could mean that the area is poorer.
(8) Listing Weekday, Listing Month: Represents the day of the week and the month on which the rental took place respectively. These 2 features are meant to detect trends in weekly or seasonality in car rental prices if any present.
(9) Car Popularity: This feature represents is overall car popularity as determined from Google Trends. The Input source being a year, model and make for a given car. The value of this feature ranges from 0 to 100, with 0 being extremely uncommonly searched on Google across the United States in a 12 month period. Thus, intuitively, a high value of popularity should indicate that the car is relatively popular, and a lower value should indicate a fairly uncommon or exotic car, which in effect has a direct impact on the price
(10) Rate: The price for each particular car rental. This is our response variable.
(11) Make, Model, Year: Represents the Make of the Car, Model of the Car, and the year in which the car was released.
(12) Renter Trips Taken: This represents the number of times the car has been rented from that particular renter.
(13) Listing Year and Hour: The year in which the car was listed to rent, and the hour in which the car was listed on Turo.
(14) Review Count: The number of reviews for the given car.
(15) Distance: The distance driven during the rental.
(16) Type: The type of car being a car, van, SUV, or a truck
(17) Free Delivery: If the car rental had a free delivery promotion appliable if rented.
(18) Instant Book Displayed: Whether the car could be booked without any request processing time to the renter,
(19) Automatic Transmission: Whether the car had an automatic transmission or not.
(20) Business Class: Whether the renter had subscribed to Turo's expedited rental service (termed business class), high priority, lower wait times, better cars etc.
(21) New Listing: Whether the car was listed to rent the first time.
(22) Rentable From Airport: Whether the car was rentable from a nearby airport or not.

**Entire Features and Importances for Spatial Random Forest**

| Top Variable Importance | | |
|---|---|---|
| Variable | Importance | % |
| car popula | 52405997.09 | 16 |
| model | 43324133.94 | 13 |
| make | 38440602.33 | 12 |
| rating | 30412404.71 | 9 |
| month | 27743171.37 | 8 |
| year | 23100987.77 | 7 |
| weekday | 21399651.68 | 6 |
| distance | 15387596.35 | 5 |
| renterTrip | 15342267.19 | 5 |
| reviewCoun | 13640432.95 | 4 |
| listHour | 11995098.98 | 4 |
| responseTi | 8567079.92 | 3 |
| responseRa | 8498850.22 | 3 |
| listYear | 5921109.57 | 2 |
| type | 5612757.05 | 2 |
| freeDelive | 3204656.81 | 1 |
| instantBoo | 3032881.60 | 1 |
| newListing | 2263103.05 | 1 |
| automaticT | 2026867.01 | 1 |
| businessCl | 1025038.44 | 0 |