Zestimate Bazinga: Predicting Selling Price for Condos in Downtown Vancouver

Andrey Koch, Marina K Peremyslova, Lucas Lemanowicz

{andkoch, mpkoch, llemanowicz}@stanford.edu

Research Overview

- **Problem Statement:** Build a model to predict the selling price of a condo based on **48 features** pulled or derived from a dataset of downtown Vancouver condo listings under $2.5M between 2016 and 2018.
- **Data:** Pulled from an official database that contains all real estate listings in Canada, with the help of a local real estate agent:
  - Contains approximately **10,000 condo listings**
  - Contains **structured data** (ex: square footage)
  - Contains **semi-structured data** (ex: address)
  - Contains **unstructured data** (ex: "has view?")

Results

Model	R^2 (train)	R^2 (CV)	MSE (train)	MSE (CV)	MSE (CV) / MSE (train)	R^2 (test)	MSE (test)
Linear Regression	0.8154	0.8066	0.1839	0.1922	1.0452	0.8032	0.1998
LinReg + LASSO	0.8136	0.8033	0.1857	0.1955	1.0529	0.8023	0.2006
LinReg + Ridge	0.8150	0.8066	0.1843	0.1922	1.0426	0.8029	0.2000
Regression Tree	0.9512	0.8584	0.0486	0.1420	2.9212	0.8587	0.1434
Random Forest	0.9554	0.8950	0.0444	0.1042	2.3442	0.9033	0.0981
Gradient Boosting	**0.9200**	**0.8993**	**0.0797**	**0.1000**	**1.2545**	**0.9069**	**0.0945**
Neural Network	0.8470	0.8278	0.1525	0.1750	1.1480	0.8325	0.1699

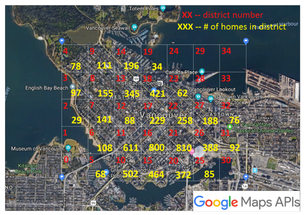
Discussion & Conclusions

- **Linear Regression:** the large observed bias is due to limitations of linear method
- **Regularization** with LASSO & Ridge only marginally improved the error and made us try nonlinear learning methods
- **Regression Trees & Random Forest**: overfitting was addressed with decreasing maximum tree depth and lifted prediction accuracy to a new level
- **Neural Networks** with optimal configuration gave prediction accuracy similar to the regression tree
- **Gradient Boosting Regression** finally provided most robust and accurate model leveraging non-linear nature of interaction between features and target variable

Fun Facts:

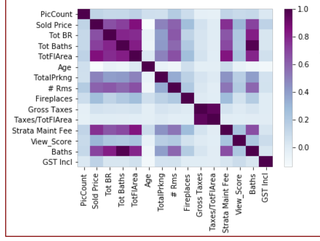
- As the plot shows, the more expensive the condo, the more subjective the price appears to be (and the worse our model performs)
- **Listing price**, the quintessence of a real estate agent's domain knowledge about current market situation, outperforms our best prediction with a 6x lower MSE

Data Exploration



Used **Google Maps API** to map condo addresses to (lat, lon) coordinates and further mapped to a rectangular grid to approximate neighborhood.

Total sales area is ~9 km².



Top features highly correlated with **Sold Price:**

- Number of bedrooms
- Number of bathrooms
- Total Floor Area
- Number of Parking Spaces
- Maintenance Fees
- Gross Taxes

Model: Gradient Boosting Implementation

Initialize $f(x_0) = argmin_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$

For $m = 1, .., M$:

(i) For $i = 1, .., N$:

$r_{im} = -[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}]_{f=f_{m-1}}$

(ii) Fit tree to r_{im} producing terminal regions $R_{jm}, j = 1, .., J_m$

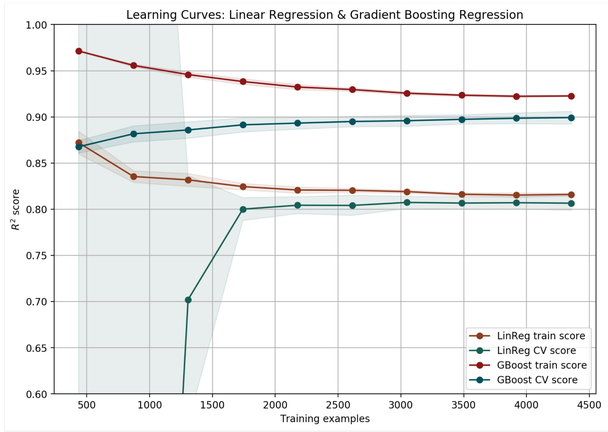
(iii) For $j = 1, .., J_m$:

$\gamma_{jm} = argmin_\gamma \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$

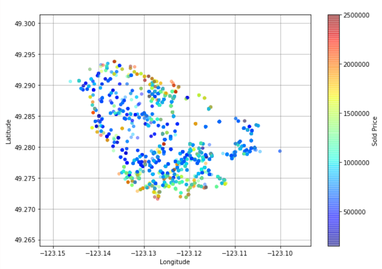
(iv) $f_m(x) := f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$

Output $\hat{f}(x) = f_M(x)$

- Hyper-parameters with **100 estimators** and **max tree depth 3** produced best results in accuracy/generalization trade-off
- Feature subset selected by LASSO gained no improvement
- Hyper-parameter optimization search improved CV error but rendered variance unacceptable
- Doubling estimator count and restricting max number of features used at split to log(# features) gave error and variance estimates close to default



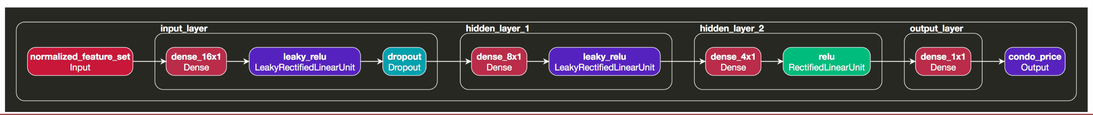
Feature Engineering



- **48 features** modelled
- **NLP** used to create "view score" feature
- Categorical features were **one-hot encoded**
- Numerical features **normalized** (mean + standard deviation)
- Missing data in some features was **imputed**

Model: Neural Network Implementation

- Experimented with various layer depths + sizes, and activation functions. In general deeper networks with smaller layers performed better.
- Overfitting was initially very high (test dataset had 10x MSE of train). Use of a dropout function after first layer, and running a 5-Fold CV reduced difference to less than 15%



Future Work

There is more work to be done around modeling the temporal aspect of condo prices. Further feature engineering (30 day sale volume) and data collection (new condo developments under construction) would likely further improve our model's accuracy.

References

[1] R. John, "Simple Housing Price Prediction Using Neural Networks with TensorFlow", Medium.com, May 29, 2018

[2] H. Yu and J. Wu, "Real Estate Price Prediction with Regression and Classification" (2016)

[3] TensorFlow, https://www.tensorflow.org/api_docs/python/tf

[4] scikit-learn, https://scikit-learn.org/stable/

[5] matplotlib, https://matplotlib.org/