Price Predictions on Airbnb Accomodations in Oslo, Norway

Marei Freitag, Joel Beck

Georg-August-University of Göttingen

21.02.2022

Table of Contents

Introduction

Aims of this work:

- Establish a deep learning approach to predict the price of an Airbnb accomodation per night in Oslo, Norway
- Focus on explainability and interpretability

$\rightarrow$ Underlying data: provided by Airbnb, contains various information about the listings in Oslo

Table of Contents

Feature Engineering: Images

- Use transfer learning on a pretrained CNN (`ResNet18`) with the first 5 images per listing as input data
- Added Fully Connected Network at the end containing three layers and `ReLU` activation functions to be sure the CNN is able to generalize
- Also implemented CNN manually as a benchmark model to compare the results

Feature Engineering: Images

- Use transfer learning on a pretrained CNN (`ResNet18`) with the first 5 images per listing as input data
- Added Fully Connected Network at the end containing three layers and `ReLU` activation functions to be sure the CNN is able to generalize
- Also implemented CNN manually as a benchmark model to compare the results

Results:

- pretrained `ResNet18` achieved a Mean Absolute Error of 579 NOK (approx. 58 Euros) on the validation set
- But correlation of the CNN predictions with the true price is 0.41

Image Predictions

Figure: CNN example predictions

Feature Engineering: Reviews

- Language: Detect language of each review
- Sentiment analysis: Get the sentiment of each review

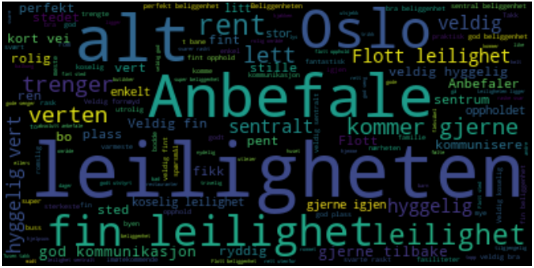
Feature Engineering: Reviews

- Language: Detect language of each review
- Sentiment analysis: Get the sentiment of each review

New features per listing:

1. Number of reviews
2. Median review length
3. Number of different languages of the reviews as well as a list of the different languages
4. Fraction of Norwegian and English reviews
5. Ratio of negative reviews to the total number of reviews

Wordclouds of the Reviews



Feature Selection & Data Cleaning

Feature Selection:

1. Manually selected features based on background knowledge, correlation analysis and the number of missing values
2. Adjusted these features by analyzing the results of different feature selection algorithms and fitted auxiliary linear regression

Feature Selection & Data Cleaning

Feature Selection:

1. Manually selected features based on background knowledge, correlation analysis and the number of missing values
2. Adjusted these features by analyzing the results of different feature selection algorithms and fitted auxiliary linear regression

Data Cleaning:

- Converting data types
- Splitting text-based variables into more convenient numeric or boolean features
- Aggregating rare categories of categorical variables into one larger *Other* group to stabilize estimation
- One-Hot encoding of categorial variables and standardization of numerical variables

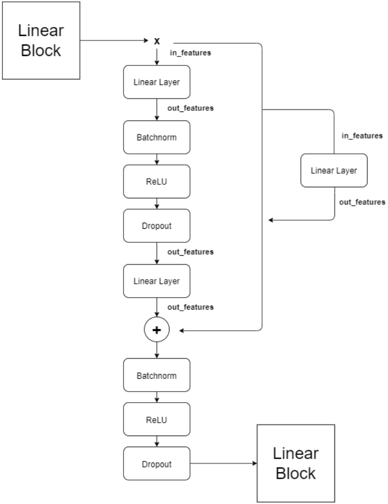
Table of Contents

Classical Models

1. **Linear Regression**: simple, well understood in terms of underlying theory and highly interpretable.
2. **Ridge Regression**: still very interpretable with a closed form analytical solution
3. **Random Forest**: very flexible model, can be applied to many contexts and often works 'out of the box'
4. **Histogram-Based Gradient Boosting**: modern and fast tree-based gradient boosting algorithm; similar to Random Forest, but uses Boosting instead of Bagging

Neural Network: Model Architecture

- Linear input layer (about 60 features)
- 6 intermediary **blocks** with 64, 128, 256, 128, 64 and 8 output features:
  - Residual connection
  - Linear layer with BatchNorm, `ReLU` activation function and dropout
- 1 output neuron



Neural Network: Model Training

- Optimizer: `Adam` with learning rate set to 0.01
- Loss function: *Mean Squared Error* Loss
- Epochs: Number of epochs vary; stopped training if Loss stagnated or model began to overfit

Neural Network: Model Training

- Optimizer: `Adam` with learning rate set to 0.01
- Loss function: *Mean Squared Error* Loss
- Epochs: Number of epochs vary; stopped training if Loss stagnated or model began to overfit

Most impactful hyperparameter: **Dropout rate**

- high influence of the network's generalizatioin availability
- model overfitted significantly by setting dropout rate to zero $\rightarrow$ that shows the current model structure is flexible enough to model the task properly
- increasing the rate leads to higher training MAE but also improves the model's performance on the validation set

Table of Contents

Evaluation Metrics

- Mean Squared Error for *Training*, Mean Absolute Error and R^2 for *Evaluation*

Evaluation Metrics

- Mean Squared Error for *Training*, Mean Absolute Error and R^2 for *Evaluation*
- When using **log-price** for model fitting, MAE and R^2 are computed on the **original** price scale for better interpretability

Evaluation Metrics

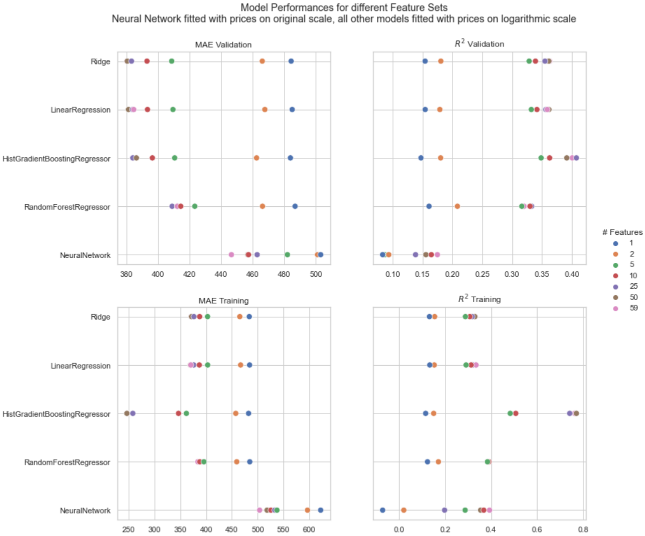
- Mean Squared Error for *Training*, Mean Absolute Error and R^2 for *Evaluation*
- When using **log-price** for model fitting, MAE and R^2 are computed on the **original** price scale for better interpretability
- Metrics highly depend on the exact evaluation procedure:

$$MAE\left(\mathbf{y}, \exp\left(\widehat{\log(\mathbf{y})}\right)\right) \neq \exp\left(MAE\left(\log(\mathbf{y}), \widehat{\log(\mathbf{y})}\right)\right)$$
$$R^2\left(\mathbf{y}, \exp\left(\widehat{\log(\mathbf{y})}\right)\right) \neq R^2\left(\log(\mathbf{y}), \widehat{\log(\mathbf{y})}\right)$$

- Direct comparisons across groups have to be taken with care

Table of Contents

Training & Validation Performance



Main Findings

- More features tend to work better with diminishing returns (5 vs. all 59 features)

Main Findings

- More features tend to work better with diminishing returns (5 vs. all 59 features)
- Price prediction task with small tabular data does not require overly complex models
  $\Rightarrow$ similar performance of linear and nonlinear classical models, linear models generalize better to validation set

Main Findings

- More features tend to work better with diminishing returns (5 vs. all 59 features)
- Price prediction task with small tabular data does not require overly complex models
  $\Rightarrow$ similar performance of linear and nonlinear classical models, linear models generalize better to validation set
- Neural Net: At first sight worst performance but best generalization
  $\Rightarrow$ Differing behaviour of *Dropout* and *Batchnorm* layers during training and inference + presence of outliers

Main Findings

- Without outlier removal we can expect a MAE of 400 NOK (40 Euros) and a R^2 value of around 0.4.

Main Findings

- Without outlier removal we can expect a MAE of 400 NOK (40 Euros) and a R^2 value of around 0.4.
- Validation performance is biased towards models whose hyperparameters were tuned during cross validation

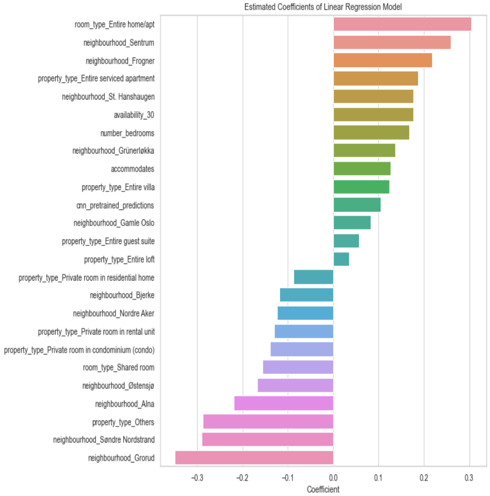
Test Set Performance

Model	MAE	R^2
Linear Regression	404.709	0.298
Ridge	405.932	0.294
Random Forest	444.166	0.268
HistGradientBoosting	412.243	0.387
Neural Network	402.24	0.333
Top2 Average	404.848	0.296
Top3 Average	399.315	0.343
Top4 Average	404.206	0.332
Top5 Average	408.116	0.27

Table: Test Set Performance of Classical Machine Learning Models, our custom Neural Network and Ensemble Predictions

Table of Contents

Feature Importance



- Largest (absolute) Coefficients: *room type*, *property type* and *neighbourhood*
- Top 2 by Feature Selector: *number of bedrooms* and *accomodates*
- Marginal vs. Conditional Impact (*Sentrum* neighbourhood)

Impact of Outliers

Quantile Threshold	MAE	R^2
0.0	443.35	0.16
1.0	337.59	0.51
2.5	282.17	0.53
5.0	240.57	0.54
10.0	214.76	0.49

Table: MAE and R^2 value of the Neural Network on the validation set

Impact of Outliers

Quantile Threshold	MAE	R^2
0.0	443.35	0.16
1.0	337.59	0.51
2.5	282.17	0.53
5.0	240.57	0.54
10.0	214.76	0.49

Table: MAE and R^2 value of the Neural Network on the validation set

- **Not** fixed by log-transformation!
- Explains performance boost of Neural Network from training to validation to test set

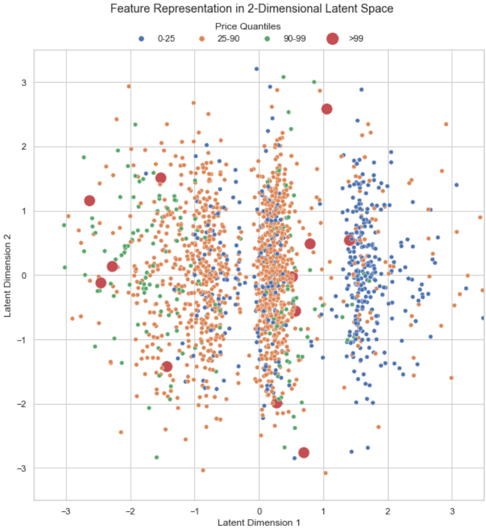
Impact of Outliers

- In theory the Neural Net should be flexible enough to approximate any arbitrary function reasonably well
- Why is it not able to capture entire price range?

Impact of Outliers

- In theory the Neural Net should be flexible enough to approximate any arbitrary function reasonably well
- Why is it not able to capture entire price range?
- Model only uses features during prediction
  $\Rightarrow$ To **discriminate** between outliers and non-outliers, the corresponding feature combinations must be **separable** in high-dimensional feature space
- Idea: approximate full feature space by two-dimensional **embedding**

Feature Space Embedding



Takeaways

- Why is there no separability?
  1. Data is not rich/expressive enough to capture all factors that contribute to very high prices
  2. Some apartments are listed at a price that does not represent their **true** value

Takeaways

- Why is there no separability?
  1. Data is not rich/expressive enough to capture all factors that contribute to very high prices
  2. Some apartments are listed at a price that does not represent their **true** value
- Can't we just remove those outliers?
  - Expensive, but **fairly priced** observations should be kept
    $\Rightarrow$ Might be contained in prediction tasks on new data, thus model should learn to handle those cases during training
  - Detecting (and removing) truly **overpriced** observations is difficult (many reasons for e.g. low demand)

Table of Contents

Munich - Predictive Performance

- About twice as large as Oslo Data
  $\Rightarrow$ Flexible Models like Gradient Boosting and the Neural Network benefit most

Munich - Predictive Performance

- About twice as large as Oslo Data
  $\Rightarrow$ Flexible Models like Gradient Boosting and the Neural Network benefit most
- However: Overall Performance *not* significantly better than for Oslo Data
- MAE of Neural Net on validation set from 44.3 Euros for Oslo to 42.5 Euros for Munich
- Gradient Boosting model with best test set performance: MAE of 32.8 (Oslo: 41.2) and R^2 of 0.453

Munich - Predictive Performance

- About twice as large as Oslo Data
  $\Rightarrow$ Flexible Models like Gradient Boosting and the Neural Network benefit most
- However: Overall Performance *not* significantly better than for Oslo Data
- MAE of Neural Net on validation set from 44.3 Euros for Oslo to 42.5 Euros for Munich
- Gradient Boosting model with best test set performance: MAE of 32.8 (Oslo: 41.2) and R^2 of 0.453
- Deeper/Wider architecture of neural net that is specifically designed for larger data set might lead to performance boost

Munich - Understanding & Interpretation

- Most important features: *Property Type*, *Neighbourhood* and *Accomodates*

Munich - Understanding & Interpretation

- Most important features: *Property Type*, *Neighbourhood* and *Accomodates*
- Munich Data again contains large price outliers with high impact on the evaluation metrics

Quantile Threshold	MAE	R2
0.0	42.49	0.31
1.0	35.32	0.44
2.5	30.26	0.42
5.0	25.2	0.45
10.0	22.94	0.4

Table of Contents

Conclusion

- Linear Models show competitive predictive performance for small Oslo data
- Top 3 Ensemble leads to lowest test set error
- Large impact of outliers

Conclusion

- Linear Models show competitive predictive performance for small Oslo data
- Top 3 Ensemble leads to lowest test set error
- Large impact of outliers
- Extension to gain further understanding of the network's behaviour: *Adversarial Examples* in regression context
  $\Rightarrow$ Find small input **perturbations** which lead to spike in price predictions

References

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015).
Explaining and Harnessing Adversarial Examples.
arXiv:1412.6572 [cs, stat].

He, K., Zhang, X., Ren, S., and Sun, J. (2015).
Deep Residual Learning for Image Recognition.
arXiv:1512.03385 [cs].

Ioffe, S. and Szegedy, C. (2015).
Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.
arXiv:1502.03167 [cs].

Kingma, D. P. and Ba, J. (2017).
Adam: A Method for Stochastic Optimization.
arXiv:1412.6980 [cs].

Kingma, D. P. and Welling, M. (2014).
Auto-Encoding Variational Bayes.
arXiv:1312.6114 [cs, stat].

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014).
Dropout: A Simple Way to Prevent Neural Networks from Overfitting.
Journal of Machine Learning Research, 15(56):1929–1958.

Thanks for listening!

Questions?