Notes to Results

1 Predictive Performance

Figure 1 shows performance metrics for all fitted models on both the Training Set and the Validation Set. Whereas we used the Mean Squared Error Loss for *training* the Neural Net due to its convenient differentiability with regards to backpropagation, we focused on the Mean Absolute Error and the R^2 value for *evaluating* all regression models. The MAE measures absolute deviations between true and predicted price and is therefore interpretable on the same scale as the original data. While the R^2 certainly has some inherent flaws, it provides a scale-independent and highly interpretable measure of overall model fit.

The different colors in figure 1 indicate how many features were selected by the `RFE` algorithm for this particular fit with 59 total predictor variables. Unsurprisingly, one or two features (in case of the `RFE` the number of bedrooms and the (number of) accomodates) provide too little information to model the task appropriately. However, including merely 5 features (in this case adding the 30 day availability, the indicator variable for the *Frogner* neighbourhood and, notably, the price predictions from the Convolutional Net based on the image data) results in a very competitive performance for most models on the validation set.

The two subplots on the left displaying the MAE tell a very similar story to the subplots on the right that show the R^2: Generally, more features lead to better performance on training and validation set. In case of the flexible Histogram-based Gradient Boosting algorithm and, to some minor extent, for the Random Forest, a high number of input features lead to *overfitting* such that the performance on the training set is far superior to the values on

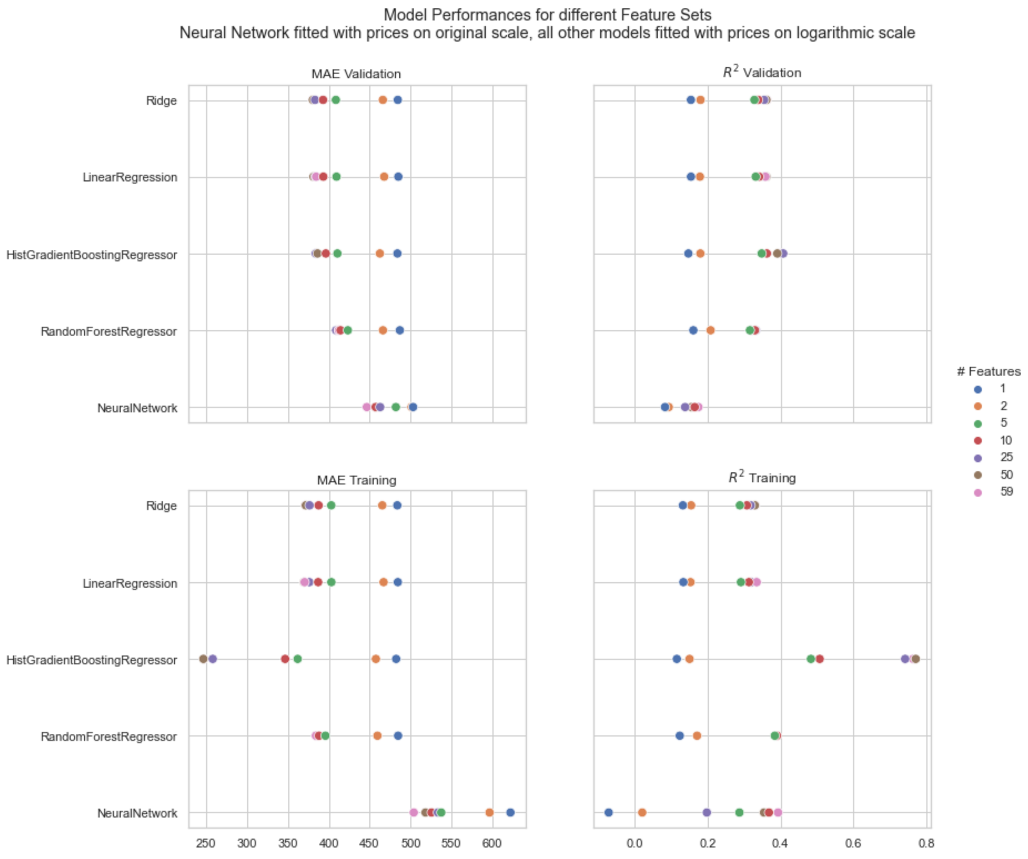

Figure 1: Performance Comparison of Classical Models with Neural Network

out-of-sample data. In contrast, models with few parameters such as Linear Regression or Ridge Regression generalize very well to the validation set with virtually no performance drop.

One exception to this rule is the highly complex Neural Network which appears to perform *better* on out-of-sample data. This phenomenon can be explained to some extent by the different behaviour of Dropout and Batchnorm during training and inference and was already discussed in the previous chapter.

When comparing the models within each subplot, all classical Machine Learning models perform similarly on the validation data. This finding emphasizes two aspects:

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

1. The prediction task does **not** require overly complex models and linear models perform just fine.
2. We can expect predictions within a distance of roughly 400 NOK (40 Euros) on average to the true price. Further, a R^2 value of around 0.4 is the best we can hope for. These statements hold for fitting on the *entire* training set, section 2.2 reveals how the performance radically improves when excluding some observations.

In comparison to all `scikit-learn` models our custom Neural Net appears to underperform. We have to keep in mind, though, that evaluation on the validation set is overly optimistic for those models whose hyperparameters were tuned on this data during cross validation. The best simulation to performance on truly unseen data is thus provided by the *test* set that was not used in any step up to this point.

Table 1 compares test set metrics for the best performing models on the validation set within each class (i.e. the Neural Net version with all 59 features and the Random Forest model with only 25 features were selected). In addition, we added some simple ensemble models that predict the average estimate from the top 2, 3, 4 or all five models, respectively.

Quite surpringly, the Neural Net shows a significant performance boost on the Test Set and is now very competitive with all other models. The Random Forest Model generalizes worst and suffers from overfitting given its selected hyperparameter configuration. Averaging predictions from multiple algorithms indicates promising results: The Top 3 Average achieves the lowest out-of-sample Mean Absolute Error of slightly below 400 NOK.

2 Explanations and Interpretation

Interpreting the results of high-dimensional and complex statistical models is notoriously difficult. This section approaches the challenge from two opposite angles.

First, we reduce the *complexity* by fitting a simple and well-understood Linear Regression Model with the same features that were selected for the best-performing Neural Network and analyze the coefficients.

Second, we reduce the *dimensionality* by leveraging a different Neural Network and visualize the results in two-dimensional space.

2.1 Feature Importance

In order to draw conclusions about which features were most important for the Neural Network one could imagine to compute gradients of the single output node with respect to each input node by backpropagating through the entire network. We chose to analyze the feature importance of an auxiliary Linear Regression model instead to provide some suggestions that could potentially apply to the Neural Network as well.

Therefore we preselected 25 of the network's input features with the `RFE` algorithm introduced in the previous chapter and compared the coefficient magnitudes. As noted before, all predictors were standardized such that a meaningful comparison is feasible.

The results are shown in figure 3. It is worth noting that the two most important features based on various different feature selectors of the `scikit-learn` library were always the number of *bedrooms* and the (number of) *accomodates* in this order, both measuring the apartment's *capacity*.

The coefficient plot, however, is dominated by categorical features: In agreement with human intuition the *room* type, the *property* type and the *neighbourhood* strongly influence the predicted price. Unsurprisingly, the property types *entire home* and *entire villa* are connected with high prices, whereas

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 2: Mean Absolute Error and R^2 value of the Neural Network on the validation set after removing the highest quantiles of the price distribution from the data set

the the room type *shared room* correlates with cheaper apartments. Notably, the price predictions from the Convolutional Network fitted on the image data indicates a significant positive effect, conditioned on all other selected features.

When analyzing the *marginal* effect of neighbourhood on price by e.g. ordering the neighbourhoods according to their median price, this order is nearly identical to the ranking in figure 3 with *Frogner* at the top and *Grorud* at the bottom.

Interestingly, apartments in the *Sentrum* (central area) have a large positive coefficient in the regression, yet the second to lowest median price. This finding indicates the presence of *confounders*: The city center might plausibly be connected to fewer rooms and smaller apartment sizes overall pulling the median price down. These confounding effects are controlled for in the regression model but not in the naive bivariate analysis.

2.2 Sensitivity to Outliers

Table 2 shows performance metrics for the Neural Network fitted on a subset of the data in the validation set. As indicated by the first column the dataset was reduced by successively cutting off observations from the top of the price distribution. More precisely, the first row refers to the entire data whereas the last row excludes the top 10% most expensive apartments.

By omitting just the top 1%, the MAE reduces by over 100 NOK (about 10 Euros) and the R^2 rapidly jumps to over 0.5. This sensitivity to outliers in the price distribution is **not** solved by log-transforming the price, all classical models from the previous chapter suffer from the same effect.

Clearly, the Neural Network lacks the ability to capture the entire price range accurately. There are two possible explanations for this phenomenon:

1. The model is flawed.
2. The model is faced with a nearly impossible task.

In order to discriminate the observations with the highest prices from all other listings, the corresponding feature combinations must be separable in the 59-dimensional feature space. Since this high-dimensional space cannot be visualized, we approximate it with an two-dimensional *embedding* or *latent space*. If the price outliers are clearly separated from the rest in this embedded space, the network is faced with a feasible task. If, however, the outliers are located in the *middle* of the latent distribution, there is little hope to discriminate the most expensive apartments from any of their potentially much lower priced embedded neighbours. In this case, the collected data might simply not be rich or expressive enough to capture all factors that contribute to very high prices.

Modern Machine Learning methods provide a large toolbox for low-dimensional embeddings. Since the project is mainly focused on Deep Learning we decided to use a *Variatonal Autoencoder*. In contrast to deterministic Autoencoders the VAE contains an additional loss term apart from the usual reconstruction loss that pulls the encoded latent space distribution towards an isotropic multivariate Gaussian distribution. For this reason, latent space visualizations of the VAE appear to be more spread out and output classes (that are not used for training the VAE) tend to be easier to identify.

Figure 2 visualizes the two-dimensional feature embedding. Already a *single* dimension seems so be sufficient for a general understanding of price segments. While some of the most expensive apartments are located on the far left, surrounded by highly-priced neighbours, the other half shares a similar feature representation to *medium to low* priced listings. Hence, the network likely maps this second half to comparably low price predictions resulting in large residuals with a high influence on the MAE and the R^2 value.

Finally, it can be argued that the entire task of predicting Airbnb prices is flawed in the first place. In order to draw connections from features to

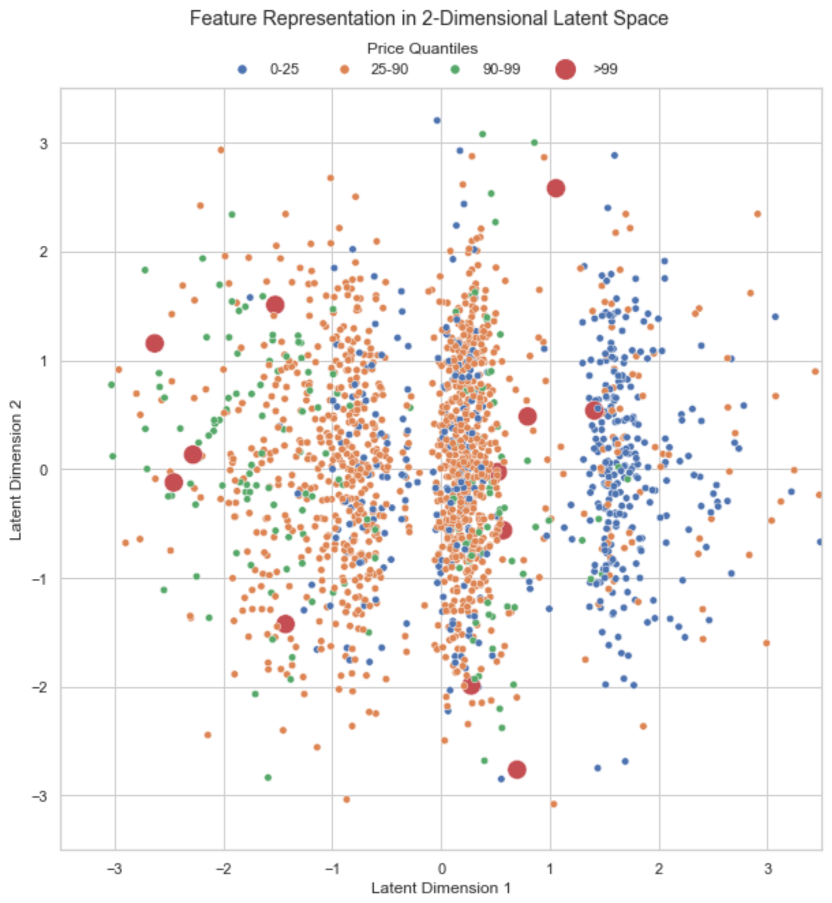

Figure 2: Feature Representation in a two-dimensional latent space embedding

outputs the process implicitly assumes observations whose listed price can be justified and explained by characteristics of the joint feature distribution.

In practice, however, some apartments might be drastically overpriced biasing the model to learn wrong feature-price mappings and inflating the error metrics. Such observations are extremely hard to detect: While some covariates might be able to accurately approximate low demand, the corresponding apartment could be undesirable for other reasons than its price. In fact, even a price outlier with zero guests can be well worth the money, yet nobody is willing to pay that high amount for an accomodation, regardless of its quality. As a consequence, a blind removal of such outliers is difficult to justify when obeying correct statistical methodology.

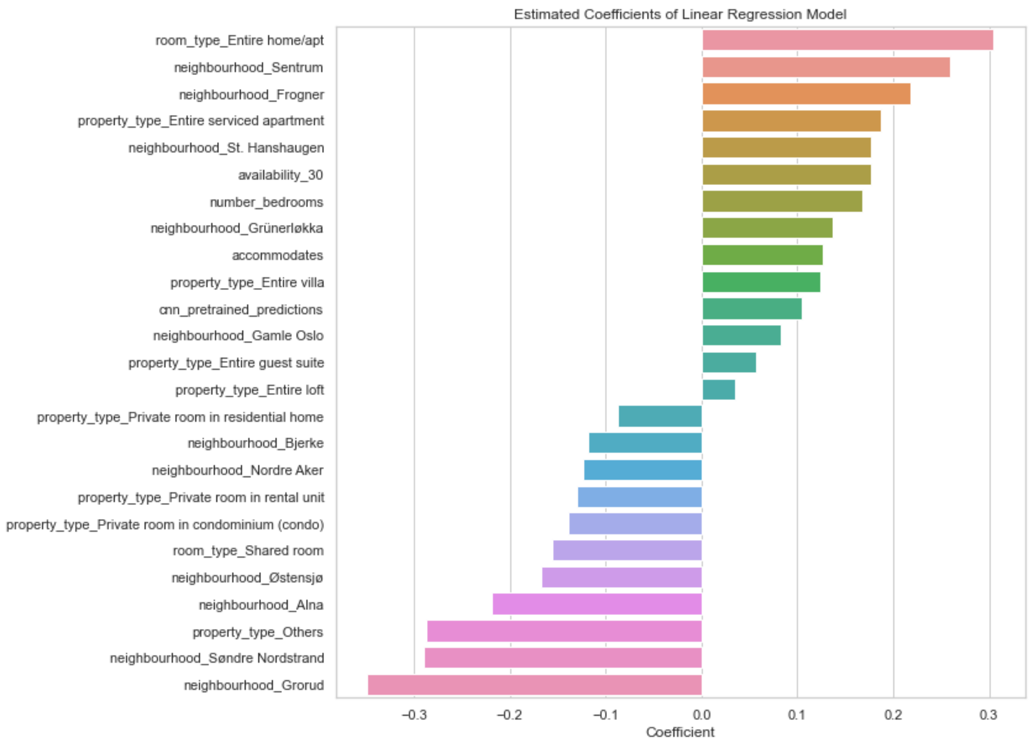

Figure 3: Estimated coefficients of a Linear Regression model for 25 preselected features