COMP6208 Team Mai$on Machine Learning Report

Sameen Islam *si1u19@soton.ac.uk* Ian Simpson *ijs1c20@soton.ac.uk* Zuzanna Skorniewska *zas1g17@soton.ac.uk*

***Abstract*—This report outlines COMP6208 Team Mai$on's development and assay of machine learning models to predict house prices in the Ames, Iowa Housing Dataset.**

I. Introduction

HOUSE price prediction is the aim of the project, within the dataset of 1,465 observations and 80 variables on houses sold in Ames, Iowa between 2006 and 2010 (hereafter known as the 'Ames dataset'). This report focuses on the application of relevant machine learning models, and is laid out roughly in the order in which the group tried to approach the subject. First, the key findings from the prior Data Exploration Report [4] were reviewed. Secondly, various models were applied and diagnosed. Finally, the results and qualities of the various models were reflected on in terms of prediction error, ease of use, explainability and robustness.

II. Preparing to Model

A. Learning and Evolving from Data Exploration Report

Taking the top 5 features by importance across eight model types yielded a pooled list of 9 top features, which was a useful starting place for fitting models, especially linear models. More generally, the understanding of the data that the group built up in the data exploration stage paid dividends when it came to modelling. All group members could now feel confident in applying the models, and the 'situational awareness' of the data aided in diagnosing models. For example, linear regression with Lasso regularisation stochastically chooses which variables to push to zero weights - understanding whether the resultant set of features was 'sensible' in terms of covering the latent dimensions could only be achieved with the prior intuition that had been developed through data exploration.

B. Data Used

The set produced after feature encoding in [4] was used for all modelling. One-hot encoding of 14 variables which could the group determined could not otherwise be encoded without in-depth research yielded a 1465×195 set, of which for some models the categorical variables were excluded to make a 1465×66 subset. $SalePrice$ is the target variable.

A train/test split on observations was made by the group, with a 80/20 ratio, using $sklearn$'s $train\_test\_split$, with a specified random seed for reproduceability, with $n_{train} = 1168$, $n_{test} = 292$. For random forest and gradient boosting methods, cross-validation split took place on the training set during model selection and fitting. The test set was *never* used for fitting, and *only ever* used to produce fit-metrics on already trained models.

Plain text descriptions of labels used in the report are available at [2].

C. How to Compare Model Performance?

The group considered which metrics would be best to compare model performance. Root-mean-square error, $RMSE(\hat{y}) = E((\hat{y} - y)^2)^{0.5}$, was identified as an obvious metric, as it penalises models with large errors, whilst also providing a result that is readily relatable to the house price values. RMSE is a function of absolute error, scoring such errors the same on low and high $SalePrice$, so the group considered using log RMSE or RMSE on percentage error, but ultimately concluded plain RMSE on $SalePrice$ to be a suitable measure since 80% of the target lies in the range $(106k, 277k)$. R^2 is 1 minus the sum of squares of residuals divided by the total sum of squares, and represents the proportion of variance in the target variable accounted for by the model, and gives a measure of the goodness of the model given the variance present in the data.

These two metrics were settled upon as they are easily applied to all model types, provide two slightly different measures of fit, and are commonly understood. Results are summarised in Table 1. Other aspects of model performance, such as support and intepretability are more bespoke and considered on a per-model basis.

III. Model Application

A. Linear Regression

Several variants of linear regression models were fit. First of all, the regression was performed on all 195 features. This yielded a model with R^2 **0.90** and **0.88**, and $RMSE$ **25.8k** and **26.8k** for train and test respectively (Tab. 1 ref A). By these two metrics alone, this could be an attractive model, however being of dimension 195, given the sample size and what we know about the colinearity of features, this model is clearly overfit and unsuitable. Nevertheless, it serves as a baseline. Model fit is shown in Figure 1a and serves to show us that there are some 'general' outliers in our dataset, particularly around the $\$400 - 800k$ target range, but with a few at lower values too.

Reducing the number of dimensions, fitting the model this on only non-categorical features (i.e. on 66 dimensions), then on the pooled top 9 features, then on the top 5 features identified in the exploration phase, led to only an incremental decrease in performance (Tab. 1 ref B,C,D), with the latter model having R^2 **0.81** and $RMSE$ of **34.1k** on test. However when it is considered that this model is now comprised of only 5 features - it's strength is striking!

Examining this model further, when fitting linear models, five key assumptions must be verified to hold true: linear relationship, multivariate normality, negligble multicolinearity, no auto-correlation, and homoscedasticity [3]. For this model, these assumptions generally held true, and inspection of Figure

1B shows a model that is particularly strong in the central body of the data, from the origin up to around 375k. After this, the model can be seen to dislocate and underestimate the value of these more expensive houses. Furthermore, extreme outlier under and over estimates can be observed for prices greater than 500k. However, depsite these limitations of the model, this is a favoured model since it is adequately performant for the bulk of the data whilst being very parsimonious.

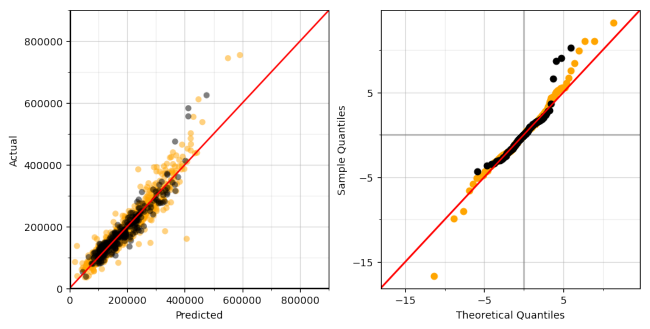
Lasso and Ridge regression apply L1 and L2 regularisation respectively. Applying Lasso led to models with 11 and 12 features (Tab. 1 ref E,F). Fit was comparable to the 5 term model (Tab. 1 ref D). Lasso is clearly a very easy way to quickly obtain a low-dimension model. However, running the algortihm with different random seeds yielded selection of different parameters (presumably colinear across runs). So, some data exploration is still important with Lasso to fully understand the dataset, but if in a hurry, Lasso clearly has benefits. Ridge regression does not promote such a sparse representation as Lasso and provided a reasonable score although remained of high-dimension. Elasticnet aims to provide the best of Lasso and Ridge, but with the $alpha\_scale$ set to an even balance, the score was poor (Tab. 1 ref H), with adjustment towards Lasso providing the best results.

With the analysis performed in the previous report on clustering having demonstrated the existence of such a nature in the data, in particular with examination of interia vs K showing between 3 and 7 clusters being optimal, the group was curious if this knowledge could be put to use. First, on 3 and 7 clusters of the top 9 features and $SalePrice$ ($K_{dim} = 10$), the RMSE was calculated with the cluster mean $SalePrice$ used as the prediction (Tab. 1 ref I,J). Remarkably, on 7 clusters this gives a very good score. Taking it one step, further a linear model using our original pooled top 9 features was now fit to each of the clusters (Tab. 1 ref K,L; Figure 1c). This gave the best score from this round of modelling.

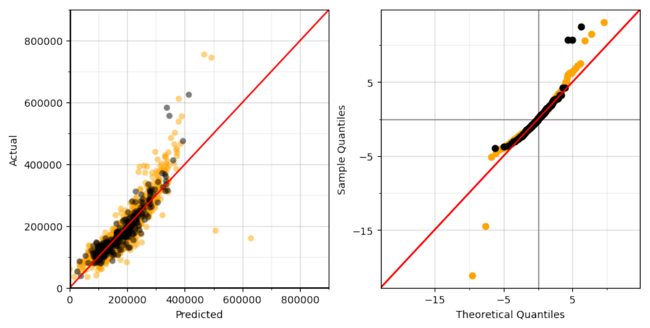
But, is this 'hybrid' a reasonable approach? Clearly as $K \to n$, $RMSE \to 0$. But this is only true for the train set, and on the test set the RMSE was good. Another potential issue is that for a small change in the data, the prediction will likely jump if a cluster boundary is crossed. Examination of the clustered showed high cross-correlation of the centroids, which was unexpected but actually meant the jumps will be less extreme and more intuitive, and this is perhaps also a result of using only our top 9 features. Depending on the use-case, in practice this hybrid approach could be practical.

B. Decision Tree

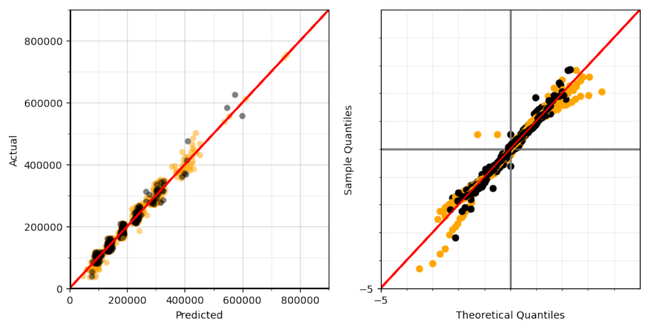
Decision trees make little assumptions about the training data, however, its non-parametric nature makes it prone to overfitting. We can control this by regularising the model through setting a maximum depth, as at the limit, the model perfectly fits the data. Further hyperparameters for controlling model fit are: the minimum number of samples a node must have before it can be split and the minimum number of samples a leaf node must have. We applied a grid search over the hyperparameter space, applied 10-fold cross validation to conclude that a tree of maximum depth = 6 with a minimum of 2 samples before split and a minimum of 1 sample in the leaf node had the best performance. On the test set, the best model had a $R^2 = \mathbf{0.809}$ with $RMSE = 34.4k$ and the average model on the 10-fold cross validated set had $R^2 = \mathbf{0.739}$. Figure 2 shows the predictions on test and validation set made by the optimal model (Tab. 1 ref M,N). Although decision trees are white box models which aid prediction interpretability, due to a depth, $d = 6$, we note that our model will contain $2^{d+1} = 128$ leaf nodes.



(a) Linear Regression: on all 195 features (Tab 1. ref A).



(b) Linear Regression: on top 5 features (Tab 1. ref D).



(c) Hybrid Model: Linear Regression on top 9 features, coefficients fit on K=7 clusters (Tab 1. ref L).

Fig. 1: Linear model diagnostics. Train set = orange; test set = black. Q-Q plot based on quantiles of Normal distribution.

C. Random Forest

Random forest is an ensemble algorithm based on 'forests' of decision trees, meaning prediction from an ensemble of trees are averaged to give the final random forest prediction. In doing so, we lose the interpretability we had with pure decision tree models. However, as shown in [4], SHAP features may be used to aid interpretability of this black box model.

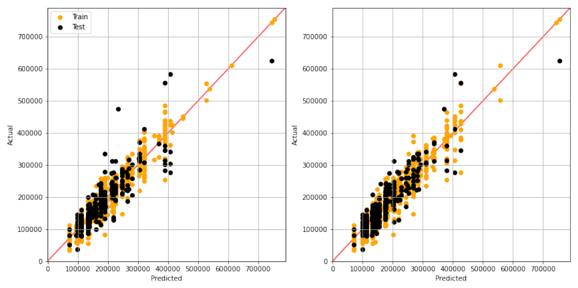


Fig. 2: Decision tree model predictions on all features (left) and on top 5 features (right).

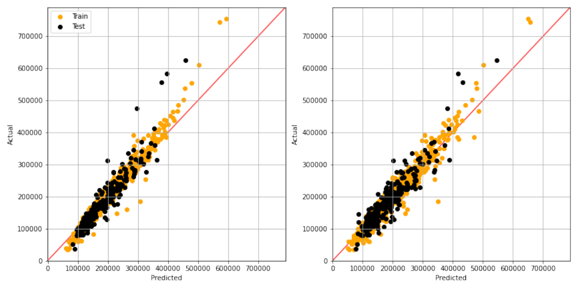


Fig. 3: Random forest model predictions on all features (left) and on top 5 features (right).

Random forests are suited to tabular and non-linear data. It is also robust to outliers. It also provides lower risk to overfitting compared with a decision tree.

We performed a grid search to optimise hyperparameters on a 3-fold cross validation set. We chose a lower k for k-fold as random forests are slower to train. We found the best performing model had a maximum depth of 10, with 600 trees and $\sqrt{n}$ features were considered when looking for best split, where n is the total number of features. However, for comparison, when we selected top 5 features from [4], to train a different model, it performed equally well (Tab. 1 ref O,P). Figure 3 shows the predictions made by both of these models.

D. Gradient Boosting

The Ames dataset spans across multiple dimensions with many of them being categorical. Datasets of this form are commonly thought to be handled well with ensembling methods such as *the Gradient Boosting*. Just like in the previous report, we trained both the sklearn Gradient Boosting model and the XGBoost, as Data Exploration study revealed that they predict significantly different sets of most important features and consequently they may yield different upon training.
As gradient boosting models rely on many hyperparameters, it is necessary to find one set of unique choices of hyperparameters that yields the highest results. This was found by the use of a grid search method, which for sklearn *Gradient Booster* was searching across: **the number of weak decision trees to include**, **learning rate**, **maximum depth of a single weak learner**, and the **minimum number of samples required to be at a leaf node**. The remaining hyperparameters were left with their default values. We found the most optimal choice of hyperparameters to be: $no-trees = 150$, $max-depth = 4$, $learning-rate = 0.1$ and $min-samples-leaf = 1$. We implemented cross-validation for the duration of training of *Gradient Booster*, with the corresponding learning curve shown in Figure 4.

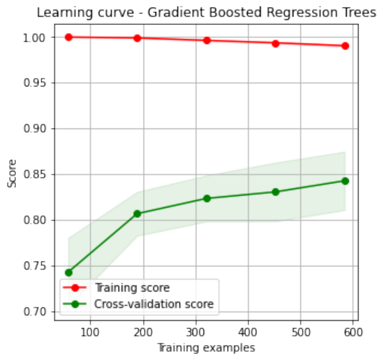


Fig. 4: Sklearn *Gradient Booster* cross-validation learning curve.

In the Figure 4, one can see that validation score improves steadily until the end of training, finally reaching a cross-validation score of $\approx 85\%$. However, one can see that the gap between the training and validation score remains significant. The *RMSE* values for the test set was found to be equal to **25.9k** (and **10.0k** for the training set), which is a relatively good score. The final test set R^2 was found to be equal to **0.89**, whilst for the training set that R^2 was found to be **0.98** (Tab. 1 ref Q).
The grid search for the best choice of hyperparameters was also applied to XGBoost model with the same set of hyperparameters optimised as for sklearn *Gradient Booster*. We found the most optimal value of $max-depth$ and $learning-rate$ to be the same as for *Gradient Booster*, whereas the most optimal number of weak learners for XGboost was found to be 60. Similarly, as for the first gradient boosting model, we performed cross-validation during training with a resulting learning curve shown in Figure 5.

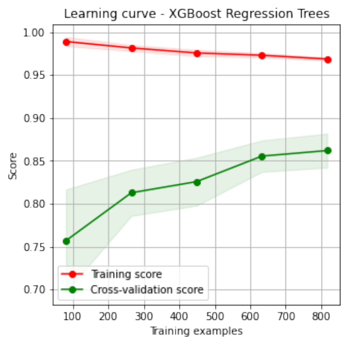


Fig. 5: XGboost cross-validation learning curve.

The cross-validation learning curve for XGBoost bears a very close resemblance to that of *Gradient Booster*, with perhaps slightly higher validation score for the former. In fact XGBoost converged to approximately equal RMSE value on the test score equal to **28.0k** (and **10.0k** for training set) and

TABLE I: Results of SalePrice prediction on the Ames dataset over different models.

Ref	Model	Train R^2	Train RMSE	Test R^2	Test RMSE	Rank by Test RMSE
A	Linear Regression (on all 195 feats.)	0.9	25784	0.88	26840	*(5)*
B	Linear Regression (on all 66 non-cat. feats.)	0.83	32734	0.86	29568	*(9)*
C	Linear Regression (on pooled top 9 feats.)	0.79	36588	0.82	33365	*(12)*
D	Linear Regression (on Exh. Srch. top 5 feats)	0.77	37879	0.81	34126	*(14)*
E	Linear Regression (LassoCV 195 to 12 feats.)	0.76	38793	0.81	34254	*(15)*
F	Linear Regression (LassoCV 66 non-cat. to 11 feats.)	0.81	34469	0.86	29839	*(10)*
G	Linear Regression (RidgeCV 66 non-cat. feats.)	0.80	35916	0.84	31289	*(11)*
H	Linear Regression (Elasticnet 66 non-cat. to 17 feats.)	0.59	50886	0.63	48100	*(19)*
I	K-Mean Cluster (K=3, on top 9 feats.)	-	63340	-	38373	*(17)*
J	K-Mean Cluster (K=7, on top 9 feats.)	-	17457	-	16704	*(2)*
K	Linear Regression (on ea. K-Mean cluster, K=3, top 9 feats.)	-	23268	-	24961	*(3)*
L	Linear Regression (on ea. K-Mean cluster, K=7, top 9 feats.)	-	13496	-	14281	*(1)*
M	Decision Tree (on all 195 feats.)	0.91	23667	0.76	38878	*(18)*
N	Decision Tree (on top 5 feats.)	0.89	34435	0.81	34435	*(16)*
O	Random Forest (on all 195 feats.)	0.94	17837	0.86	29475	*(8)*
P	Random Forest (on top 5 feats.)	0.94	29218	0.86	29218	*(7)*
Q	Sklearn Gradient Boosting Regressor	0.98	10028	0.89	25923	*(4)*
R	XGBoost	0.97	11807	0.89	28083	*(6)*
S	Multilayer Perceptron	0.83	30840	0.82	33610	*(13)*

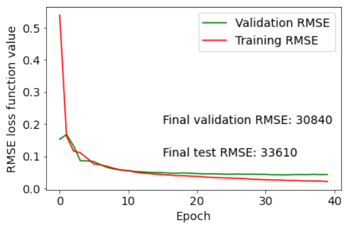


Fig. 6: MLP learning curve.

exactly equal test set R^2 value of **0.89**. The training set R^2 was found to be slightly smaller than that of *Gradient Booster* and equal to **0.97** (Tab. 1 ref R). We can hence see that despite significantly different results in Data Exploration study, both XGBoost and **Gradient Booster** achieved similar scores on the Ames dataset.

E. Multilayer Perceptron

MLP is a less obvious method for this prediction problem, but with a high-dimensional dataset the group was curious how it would perform. Architecture used was: one linear layer activated by leaky ReLU, followed by 3 hidden layers, activated by leaky ReLU and log of the softmax. Sizes of the layers were set to be (from top to bottom): 128 (input size), 500, 300, 30, 1 ($SalePrice$). We trained the model for 40 epochs with MSE set as the loss function and default parametrised *Adam* set as an optimiser. The resulting learning curve can be seen in Figure 6, and we can see that unlike e.g. learning curves for Gradient boosting models, here both results for validation and training data improve steadily and on par. The final validation RMSE score was found to be **30.8k**, whereas RMSE for the test set was found to be **33.6k**. Despite such low test set RMSE values, the network did not yield as high test set R^2 results as other models analysed in this report: test R^2 was **0.82** (**0.83** on validation set) (Tab. 1 ref S).

F. Areas for Further Research

Ensembling of divserse models has been empirically shown to lead to more robust models [5], and could be of interest. Gaussian processes could be applied to this problem in an online learning situation, and were identified as of potential use if new house sale samples were received continuously. Deploying models for non-scientific users is also an interesting practical problem; a prototype was deployed at [1].

IV. CONCLUSION

Several variations of *Linear Regression models*, *Decision Trees*, *Random Forests*, *Gradient Boosting*, *a MLP network* were applied to the house price prediction problem. Linear regression models can give respectable results, whilst being extremely parsimonious and explainable - although feature engineering is critical to this success. Decision trees can handle categorical data, but overfit on this dataset. Random Forest and Gradient Boosting models performed well, and an advantage is that are very easy to fit and work well with categorical data. Historically, a weakness has been lack of interpretability, but recent advances in SHAP mostly neutralise this weakness. MLP gave reasonable results but is overly complex for a task of this nature. Combining linear regression with clustering yielded the best results, showing the benefits of an open-mind when modelling; although shown to be low in severity on this dataset, an effect of this approach being discontinuities in prediction between clusters. Good data exploration and data encoding is necessary to leverage *all* models types to the fullest extent.

REFERENCES

[1] Cloud deployment of mai$on house price prediction model. https://soton-ml-houseprice-2021.herokuapp.com/.

[2] Kaggle competition website. https://www.kaggle.com/c/house-prices-advanced-regression-techniques.

[3] Robert J Casson and Lachlan DM Farmer. Understanding and checking the assumptions of linear regression: a primer for medical researchers. *Clinical & Experimental Ophthalmology*, 42(6):590–596, 2014.

[4] Sameen Islam, Ian Simpson, and Zuzanna Skorniewska. Team Mai$on Data Exploration Report.

[5] Ludmila I. Kuncheva and Christopher J. Whitaker. *Machine Learning*, 51(2):181–207, 2003.