# Project 1: Predict the Housing Prices in Ames

### Fall 2023

## Contents

Ames Housing Data
Submission Guidelines
Code Evaluation
FAQ

# Ames Housing Data

#### **Dataset Overview**

The dataset for this project is available for download at proj1.zip. Once you unzip the file, you'll find **ten** folders. Within each folder, there are **three** files:

- train.csv: This file represents the training dataset and contains 2051 houses across 83 columns.
  - The first column is "PID", the Parcel identification number;
  - The last column is the response variable, Sale Price;
  - The remaining 81 columns are explanatory variables describing (almost) every aspect of residential homes.
- test.csv: This is the test dataset containing feature vectors for 879 houses.
- test\_y.csv: Complementary to the test.csv, this file contains the response column (Sale Price) alongside the "PID" column for the test set.

## **Dataset Origin**

The training and test splits are derived from the Ames Housing data. For more background on this dataset, you can refer to:

- De Cock, D. (2011). "Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project," Journal of Statistics Education, Volume 19, Number 3. [PDF]
- Check variable description [Here]
- This dataset also features in a Kaggle competition (https://www.kaggle.com/c/house-prices-advanced-regression-techniques). Our dataset, however, has two additional explanatory variables: "Longitude" and "Latitude". Exploring the Kaggle competition can offer insights on data analysis approaches and sample codes.

# **Project Objective**

Your task is to predict the price of homes, but importantly, **in log scale**. You need to build **TWO** prediction models selected from the following two categories:

- one based on linear regression models with Lasso or Ridge or Elasticnet penalty;
- one based on tree models, such as randomForest or boosting tree.

**Note**: The features selected for the two models can differ. Please refer to Campuswire to identify the packages that are permissible for use in this project.

### Submission Guidelines

Submit the following **two** items on Coursera/Canvas:

- Code: Your R/Python script should be in a singular file named either mymain.R or mymain.py. This script should:
  - Accept train.csv and test.csv as inputs.
  - Generate two submission files based on the specified format (detailed below).
  - Important: Do not submit ZIP files or markdown/notebook files.

The structure of your mymain.R (or mymain.py) should adhere to the following:

- Step 0: Load necessary R libraries or Python packages
- Step 1: Preprocess the training data, then fit the two models.
  - \* Note: At this step, you are strictly not allowed to access the test data.
- Step 2: Preprocess test data, then save predictions into two files: mysubmission1.txt and mysubmission2.txt. (The specific format for these files is detailed below.)
  - \* Note: At this step, you are strictly not allowed to access the training data.
- Report: Submit a concise report (maximum of 2 pages, in PDF format) which contains two sections:
  - Section 1: Technical Details: Discuss details such as data pre-processing and other non-trivial implementation aspects of your models. Do NOT paste your code in the report. Instead, explain the technical steps in clear English. Your description should be comprehensive enough for your fellow PSL classmates to replicate your results.

For instance, when documenting your pre-processing steps, provide specifics such as:

- \* Which variables did you exclude from the analysis?
- \* Identify the variables treated as categorical. How were these variables encoded, were any levels merged, etc?
- \* For numerical variables, were there any transformations applied?
- \* You're not required to justify these pre-processing decisions; just state what was done.

When documenting implementation, general statements like "We use lasso to fit a sparse regression model" are insufficient. Instead, aim for detailed descriptions, such as: "We utilized lasso for regression modeling. Specifically, we employed the glmnet function in R with the data standardized and with lambda set to lambda.min."

- Section 2: Performance Metrics: Report the accuracy of your models on the test data (refer to the provided evaluation metric below), the execution time of your code, and details of the computer system you used (e.g., Macbook Pro, 2.53 GHz, 4GB memory or AWS t2.large) for all 10 training/test splits.

## **Code Evaluation**

We will execute source(mymain.R) in RStudio starting from a clean environment (meaning, no pre-loaded libraries) or run python mymain.py from the command line, in a directory containing only two files:

- train.csv: 83 columns;
- test.csv: 82 columns without the column "Sale Price".

These files are sourced from one set of the 10 training/test data splits. It's crucial to note that test\_y.csv will **NOT** be present in this directory.

Upon successful execution of your code, we expect to find **two** new txt files in the same directory named mysubmission1.txt and mysubmission2.txt. Each of these files should contain predictions for the test data from one of your models. These two files should adhere to the following format, ensuring the comma is present between PID and Sale Price:

```
PID, Sale_Price
528221060, 169000.7
535152150, 14523.6
533130020, 195608.2
```

**Evaluation Metric.** Submission are evaluated on Root-Mean-Squared-Error (RMSE) between the logarithm of the predicted price and the logarithm of the observed sales price.

$$\sqrt{\frac{1}{\text{n.test}} \sum_{j=1}^{\text{n.test}} (\hat{y}_j - y_j)^2}$$

**Performance Target**. Your performance is based on RMSE from the two models. Full credit for submissions with RMSEs less than

- 0.125 for the initial 5 training/test splits and
- **0.135** for the subsequent 5 training/test splits.

# **FAQ**

• Do we need to download the training/test datasets from Kaggle and upload our predictions there for evaluation?

No, there's no need to download or upload any dataset from/to Kaggle.

• Is diagnostic work necessary? For this dataset, are diagnostics crucial? If so, do we eliminate any outliers?

Diagnostics, along with any pre-processing for missing or extreme values, are your responsibility. Detail your pre-processing steps in your code and describe them in your report. Remember, the primary objective isn't an extensive EDA but to create predictive models using the training data.

• The RMSE of the price's logarithm is either infinity or 'nan' for certain entries. What should I do?

Recall that  $\log(y)$  is defined only for y > 0. If you train a model to predict y without any constraints, it is possible that your model may return zero or negative values, leading to infinities and nans in the metric calculations. Since what matters in the evaluation metric is  $\log(y)$ , predict  $\log(y)$  or in this case  $\log(\text{Sales\_Price})$  instead of the raw target. In other words, build a model to predict the logarithm of the price.

Since you are asked to output the predicted Sales\_Price in mysubmission1.txt and mysubmission2.txt, remember to revert the target using the exponential function – i.e., exp(pred) – before finalizing your submission.

• How should we determine tuning parameters?

For algorithms like Lasso, which usually have one or two tuning parameters, leveraging their in-built CV procedures to select tuning parameters is recommended. On the other hand, some techniques, such

as boosting trees, come with multiple tuning parameters. Exhaustively tuning them for each training dataset can be time-consuming. For this assignment, it's acceptable to tune them based on the 10 training/test splits provided. Subsequently, in the code you submit, you can set some (or all) tuning parameters to specific values to save computation time.