**MGSC 673 – Assignment 3**

Deep Learning Applications in Management Analytics

Konstantin Volodin – 261083570

Table of Contents

[1. Introduction 2](#_Toc137409937)

[2. Data Review 2](#_Toc137409938)

[2.1. Feature Review 2](#_Toc137409939)

[2.2. Data Cleaning 2](#_Toc137409940)

[2.3. Data Preprocessing 2](#_Toc137409941)

[2.4. Data Splitting 3](#_Toc137409942)

[3. Network Architecture 3](#_Toc137409943)

[3.1. Multi-Class Architecture 3](#_Toc137409944)

[3.2. Hyper Parameter Optimization 3](#_Toc137409945)

[3.3. Experimentation Results 4](#_Toc137409946)

[4. Final Model Results 4](#_Toc137409947)

[5. Conclusion 5](#_Toc137409948)

[Appendix 5](#_Toc137409949)

# Introduction

In this project, our goal is to develop a multi-task neural network that predicts house prices and categorizes houses simultaneously. We will follow a comprehensive machine learning pipeline, including data exploration, model building using PyTorch Lightning, hyperparameter optimization with Ray, and final model evaluation. Additionally, we will leverage advanced features of PyTorch Lightning, such as logging, callback system, and Trainer API, so we can efficiently manage this complex project.

# Data Review

The dataset consists of 1460 rows, and our objective is to predict five target variables: Sale Price, Year Built, Year Remodeled, House Style, and Building Type. Among these target variables, Sale Price, Year Built, and Year Remodeled are numerical variables, while House Style and Building Type are categorical variables. To ensure compatibility with our multi-task learning model, we will appropriately adjust the categorical variables using suitable encoding techniques. In the subsequent sections, we will explore the dataset further and perform necessary preprocessing steps to prepare the data for model training and evaluation.

## Feature Review

The dataset comprises 75 predictor features that provide comprehensive information about various aspects of the houses. Out of these, 33 features are numerical variables, while 42 features are categorical variables. As most of the variables are categorical, appropriate variable encoding techniques will be employed. These features can be grouped into several categories: (1) features related to the property itself, such as its size, shape, and overall condition, (2) features related to the construction and structure of the house, such as the type of dwelling, style of the house, and the presence of amenities like fireplaces and pool, (3) features related to the location and neighborhood, such as the proximity to main roads or railroads (4) features that capture details about the basement, garage, heating, electrical systems, and more, (5) lastly, there are miscellaneous features.

## Data Cleaning

During the analysis of the dataset, it was observed that several features had a significant number of null values. However, these null values represented a separate category, for all categorical variables, the null values were replaced with a "none" category. Out of all the features, only 7 had less than 95 percent null values: *LotFrontage*, *Alley*, *MasVnrType*, *FireplaceQu*, *PoolQC*, *Fence*, and *MiscFeature*. Since *PoolQC* has a very strong class imbalance (1433 vs 7 other), it was decided to drop this feature entirely. Additionally, *MSSubClass*, appears to have similar information to the target variables related to house type, thus it was dropped to prevent data leakage.

To handle numerical data, a median imputer was used to fill in the missing values. This approach helps maintain the integrity of the data and ensures that no important information is lost during the cleaning process.

## Data Preprocessing

The data preprocessing stage involves distinct steps for handling categorical and numerical variables. Numerical variables undergo scaling to achieve unit variance and a mean of 0, ensuring comparability and interpretability. Categorical variables are one-hot encoded, expanding them into binary columns. After preprocessing, the transformed dataset comprises 247 features, consisting of the original variables and newly created one-hot encoded columns.

## Data Splitting

To ensure proper evaluation of the model's performance, the dataset is split into three subsets. 1000 data points are allocated for training, 400 for validation, and 60 for testing. During the training phase, the model's weights are calculated through iterative training epochs using the training set. The validation set serves as an independent evaluation set, allowing us to assess the model's performance after each training epoch. Finally, the test set is reserved exclusively for the end of the training process to provide an unbiased assessment of the final model's predictive accuracy.

# Network Architecture

For the purposes of this project, we employed a multi-class dense neural network. The network consists of a shared baseline layer and a custom output layer for each predictor variable. In this project the baseline model was skipped in favor of performing hyper parameter tuning for finding a good network architecture. During the process of hyper parameter tuning, the shared layer will be adjusted while the custom output layers remain fixed. The final network architecture will be determined based on the model with the lowest overall validation loss.

## Multi-Class Architecture

One of the challenges in this predictive task is formulating the custom layers and combining losses for different target variables. In this report, a custom output layer is designed for each target variable. For numerical targets, a linear layer with 1 output is used, while categorical variables are one-hot encoded and fed into a final SoftMax layer with an appropriate number of outputs for multi-label classification.

To ensure a balanced approach, the overall loss function is calculated as the sum of individual loss values for each target variable. To avoid bias towards any specific variable, all numerical target variables are scaled to have a mean of 0 and a unit variance of 1. In future experiments, it would be interesting to explore the possibility of assigning different weights to the loss of each target variable, especially if certain targets are deemed more important than others. This could further enhance the model's performance and adaptability to specific requirements.

## Hyper Parameter Optimization

To fine-tune the model's performance, a comprehensive hyperparameter tuning process is conducted using the Optuna algorithm implementation in the Ray package. The following hyperparameters are optimized within specified ranges:

* Number of hidden layers: Ranging from 1 to 3.
* Number of neurons in each shared layer: Varying from 30 to 500.
* Dropout rate after each hidden layer: Ranging from 0.0001 to 0.9.
* Optimizers: Adam, SGD, and LBFGS.
* Learning rates for each optimizer: Ranging from 0.0001 to 0.1.
* Activation functions: ReLU, Leaky ReLU, and Tanh.
* Batch size: Ranging from 10 to 500.

A total of 50 combinations of hyperparameters are attempted, and each combination is trained for 300 epochs with the specified batch size. The model with the lowest validation loss is selected as the final model for further evaluation.

## Experimentation Results

The performance of various network architectures is analyzed by evaluating the validation error over the course of 300 epochs. The model with the best validation error achieves a loss of 2.997 at the end of the 300 epochs, as illustrated in Figure 1. It is worth noting that there is no significant overfitting observed, as the validation loss does not exhibit any increasing trend. Consequently, this model is selected as the final architecture.

Figure 2 presents the individual validation loss for each metric in the best performing model. Categorical variables employ cross-entropy loss, while numerical values utilize mean squared error (MSE) loss. The model demonstrates early generalizability, but to further explore its potential, the number of epochs can be increased. Furthermore, considering the difference in magnitude between MSE and cross-entropy errors, it is advisable to assign different weights to ensure a more balanced focus on both categorical and numerical variables.

The final model has the following hyper parameters:

* Number of hidden layers: 2
* Number of neurons in each shared layer: 385
* Dropout rate after each hidden layer: 0.0003
* Optimizers: Adam
* Learning rates for each optimizer: 0.013
* Activation functions: ReLU
* Batch size: 69

# Final Model Results

# The final model was trained for 1000 epochs using the full dataset of 1400 data points for training and 60 for testing. The overall validation loss for the model was 3.394. The results can be further analyzed using more appropriate evaluation metrics:

# Sale Price RMSE: $40,113.64

# Year Remodeled RMSE: 14.31 years

# Year Built RMSE: 13.35 years

# Building Type Accuracy: 86.67%. The largest category corresponds to approximately 83.56%, indicating that the accuracy is higher than a basic heuristic.

# House Style Accuracy: 85%. The largest category corresponds to about 49.73%.

# The final RMSE values and accuracy rates demonstrate a high level of performance compared to a basic guess for both numerical and categorical variables. The accuracy for categorical variables is notably higher than the largest category, indicating that the classification aspect of the model possesses reasonable predictive power. The RMSE values represent only a small percentage of both the mean and the overall range of the corresponding variables, indicating strong predictive capability.

# 5. Conclusion

In conclusion, our multi-task learning model demonstrates strong predictive power and generalizability. It can be a valuable tool for predicting house prices and categorizing houses based on their attributes. Further improvements and optimizations can be explored in future experiments to enhance the model's performance and adaptability to specific requirements.

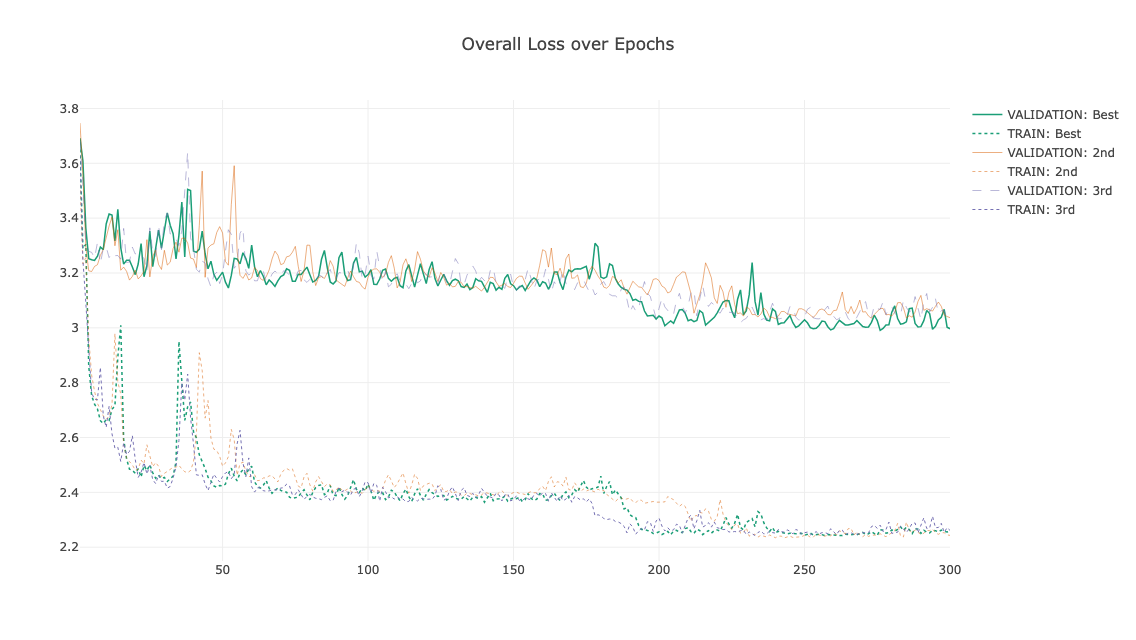
Appendix

Figure 1: Evolution of train vs validation loss for top 3 performers

A picture containing text, line, plot, diagram

Description automatically generatedFigure 2: Evolution of individual validation loss for top performer