Introduction

This project entails the creation of a sophisticated machine learning model designed to simultaneously predict house prices and categorize house types. The model leverages the capabilities of the PyTorch Lightning framework to manage the complexities of multi-task learning, which involves concurrent regression and classification tasks. This report provides an in-depth exploration of each stage of the model development process, from initial data handling to the final evaluation of the model's performance.

Dataset Acquisition and Preliminary Processing

The dataset, sourced from the "House Prices - Advanced Regression Techniques" competition, comprises a variety of features related to residential houses. The initial phase of the project involved preparing this dataset for analysis and modeling:

1. *Data Loading:*

The data was loaded from three CSV files: train.csv for training data, test.csv for testing data, and sample_submission.csv which contains example submission formats with ID mappings. This step was crucial for merging and aligning the datasets appropriately.

2. *Data Merging:*

The test dataset was augmented by merging it with sample submission data based on the 'Id' field using an inner join operation. This approach ensured that every entry in the test set had a corresponding example target variable from the sample submission, which was necessary for consistent testing and validation.

3. *Consolidation and Cleaning:*

The training and augmented test data were concatenated to form a comprehensive dataset that facilitates uniform feature engineering and preprocessing. We removed the 'Id' column since it was merely an identifier and provided no intrinsic predictive value.

Detailed Data Cleaning and Feature Engineering

This stage was aimed at refining the data to enhance the model's learning efficacy:

1. *Handling Missing Data:*

Initial analysis revealed several features with a significant proportion of missing values. Columns like 'PoolQC', 'MiscFeature', 'Alley', and 'FireplaceQu' were dropped due to their high incompleteness, which could introduce bias or noise into the predictions.

2. *Imputation of Remaining Missing Values:*

For numerical columns, missing values were imputed with the mean, preserving the statistical characteristics of those features. Categorical columns were imputed with the mode, ensuring the most frequent category was used to fill gaps, thus maintaining the distributional properties of those features.

3. *Innovative Feature Engineering:*

New categorical features were constructed to encapsulate complex information into actionable insights:

- Age Category: Created by assessing the recency of construction and last renovation, categorizing houses into 'New' or 'Older'.

- Type and Style Categories: Derived from architectural style and building type to group homes into meaningful clusters like 'Residential', 'Multi-Unit', and others based on structural and stylistic characteristics.

These categories were further synthesized into a 'Composite House Category' through a custom function that combines the individual characteristics into a holistic categorical descriptor of each house.

4. *Outlier Identification and Removal:*

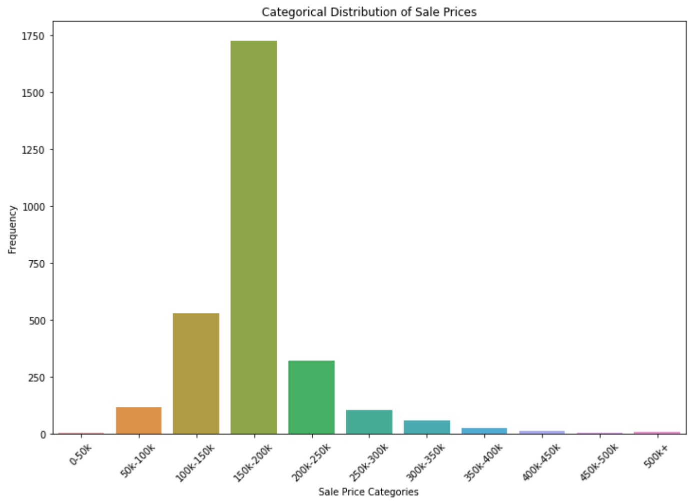
Z-score analysis was applied to the 'SalePrice' feature to identify and exclude extreme values that could distort the model training process.

Exploratory Data Analysis (EDA) and Visualization

Through various visualizations, we explored the underlying patterns and distributions in the data:

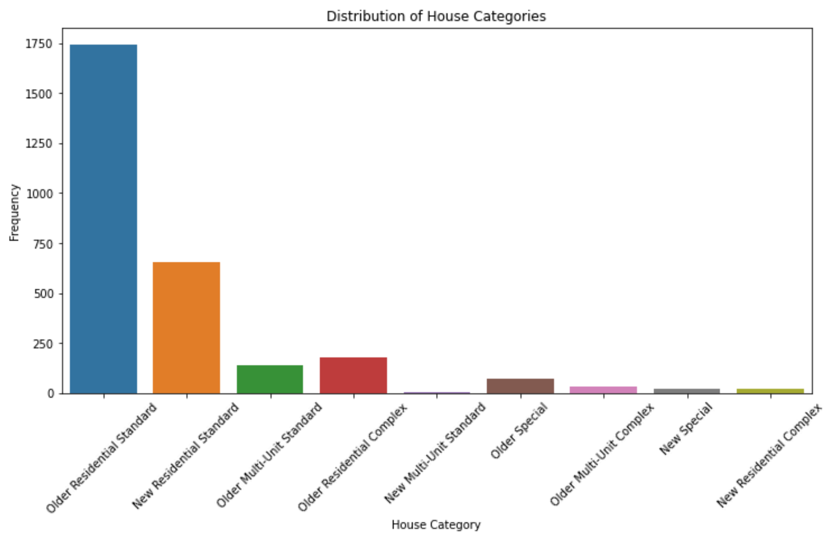
1. *Sale Price Distribution:*

The distribution of sale prices was visualized through histograms and pie charts, categorizing the prices into bins to analyze the frequency of different price ranges. This helped in understanding the skewness and kurtosis of the price data. We noticed that our data was left skewed which is why we decided to remove the outliers.



2. *Visual Analysis of Categorical Data:*

Count plots were used to examine the distribution across the newly engineered 'Composite House Category', providing insights into the commonality and rarity of different house types within the dataset. Older Residential Standard homes are the most common in our dataset.



Advanced Data Preprocessing for Modeling

Further data preprocessing steps were undertaken to optimize the data for the modeling process:

1. *Encoding of Categorical Variables:*

Textual data were transformed into a machine-readable format using label encoding. This step is crucial as it allows the model to process categorical data as numerical inputs.

2. *Normalization of Feature Values:*

To prevent any single numerical feature from dominating the model due to its scale, MinMax scaling was applied to all numerical features except the target variable, 'SalePrice'.

3. *Feature Selection via Statistical Techniques:*

The SelectKBest method, using the f_regression score function, was employed to identify and retain the top 20 features that have the most significant relationship with the target variable. This method effectively reduces dimensionality and focuses the model on the most informative attributes.

Model Architecture and Loss Function Design

The architecture of our model was meticulously designed to efficiently handle the dual objectives of predicting house prices (regression) and categorizing house types (classification).

1. ***Shared and Task-Specific Layers:***

Shared Layers: Our model architecture incorporated a series of shared linear layers. Specifically, these layers included three sequential blocks each consisting of a linear layer, followed by batch normalization and ReLU activation. This configuration ensures the model captures general features relevant to both tasks.

Residual Connections: To facilitate deeper learning without encountering the vanishing gradient problem, we implemented a residual connection that bypasses one or more layers by directly connecting the input of one layer to a deeper layer in the network. This connection was made from the input directly to the third block of shared layers.

Task-Specific Layers: Following the shared layers, the architecture diverges into two distinct paths:

- The regression path includes two linear layers tailored to refine features specifically for price prediction.
- The classification path similarly utilizes two linear layers, optimized to classify the house types.

2. ***Loss Functions:***

Heteroscedastic Loss for Regression: We chose a heteroscedastic loss function for the regression task to dynamically adjust the penalty based on the variance of each prediction, effectively dealing with heteroscedasticity in the house prices.
Cross-Entropy Loss for Classification: For categorizing house types, we used cross-entropy loss, which is well-suited for classification tasks as it measures the difference between the predicted probabilities and the actual class labels.

Hyperparameter Tuning with Optuna

Using Optuna, we conducted an extensive search for the best hyperparameters to optimize our model's performance. This included:
Learning Rate: We tested values ranging from 0.0001 to 0.01 to find the optimal speed at which the model learns without skipping over minima.
Hidden Units: Different configurations of hidden units in the layers (e.g., 64, 128, 256) were evaluated to determine the optimal capacity of the network to process features.
Activation Functions: We experimented with ReLU, Leaky ReLU, and Sigmoid to identify which activation function best facilitated non-linear learning for this specific task.
Optimizer Type: Choices between Adam and SGD were tested to decide which optimizer would provide the best convergence properties for our training process.

Model Training, Evaluation, and Metrics

The model underwent rigorous training, validation, and testing phases:

1. *Training Process:*

The model was trained using a split dataset approach where the data was divided into training, validation, and test sets. This separation helps in unbiased evaluation and prevents overfitting. Various callbacks like ModelCheckpoint and EarlyStopping were integrated to monitor the training process, save the best model, and halt training when the validation loss ceased to improve, ensuring optimal performance.

2. *Evaluation and Metrics:*

The model's performance was evaluated using comprehensive metrics. For regression, the Root Mean Squared Error (RMSE) was calculated to assess the accuracy of price predictions. For classification, metrics such as accuracy, precision, recall, and F1-score were computed to evaluate the effectiveness of house categorization.

Results and Performance Evaluation

The project's hyperparameter tuning phase using Optuna was instrumental in identifying optimal settings for our multi-task learning model. Despite extensive experimentation, including adding more layers, increasing the number of trials, and extending the number of epochs, the accuracy remained low. The best trial resulted in a value loss of 4,484,457.0, with optimal parameters comprising a learning rate of 0.0022298777573344524, hidden size of 64, activation function of leaky ReLU, and the Adam optimizer.

In our evaluation on the test set, the model recorded a total test loss of approximately 2391.61, primarily from the price prediction task (about 2390.80) with a smaller portion from category classification (about 0.81). The classification metrics demonstrated considerable challenges: accuracy, precision, and F1 score were around 42%, 41.6%, and 40.5%, respectively. The Root Mean Squared Error (RMSE) for price prediction stood at 180,194.89, indicating significant difficulties in accurately capturing price determinants.

Moreover, the model's performance exhibited notable variability in value loss across multiple runs, which suggests sensitivity to initial conditions or data splits. This variability points to a need for further refinements to boost the model's stability and performance, especially in improving the accuracy of house type classification and price prediction. These results underscore the complexity of adapting machine learning models to real-world data and highlight the ongoing challenge in achieving high accuracy in such sophisticated multi-task learning scenarios.

Future enhancements:

To advance the capabilities of our multi-task learning model for predicting house prices and categorizing house types, several future enhancements are recommended. These include broadening the dataset and refining feature engineering to capture more complex data relationships, exploring more sophisticated neural network architectures like convolutional or recurrent layers, and implementing advanced regularization techniques to improve model generalization. Additionally, stabilizing the training process, extending hyperparameter optimization, and integrating multi-modal data can further enhance performance. Deploying the model with real-time learning capabilities and incorporating explainability and fairness analyses will ensure the model remains dynamic and equitable, providing valuable insights and maintaining reliability in practical applications.

Conclusion:

This project highlighted the complexity involved in developing a multi-task learning model. Despite the final model achieving lower accuracy than expected, the project was instrumental in deepening understanding of multi-task learning principles and the practical challenges associated with implementing such models. Extensive data preprocessing, the employment of a shared-bottom architecture for simultaneous regression and classification, and the integration of specific activation functions, optimizers, and loss functions were critical components. Rigorous model evaluation and the use of advanced PyTorch Lightning features, including logging and hyperparameter tuning with Optuna, underscored the importance of a systematic approach in machine learning projects.