House Prices Prediction using TensorFlow Decision Forests

Mahla Entezari

March 5, 2025

1 Abstract

When we want to price a house, We can take a look at other houses' prices and make a good estimate of the specific house.

If we live a normal life, The price of the same houses will be the same. So, What does the price of houses depend on? For example, In equal terms, a house with more area is usually more expensive.

2 Introduction

Ask a home buyer to describe their dream house, and they probably won't begin with the height of the basement ceiling or the proximity to an east-west railroad. But this dataset proves that much more influences price negotiations than the number of bedrooms or a white-picket fence.

With 79 explanatory variables describing (almost) every aspect of residential homes in Ames, Iowa, we are challenged to predict the final price of each home.

It is our job to predict the sales price for each house. For each ID in the test set, I'm going to predict the value of the SalePrice variable.

In this notebook, we demonstrated the process of building a baseline Random Forest model using TensorFlow Decision Forests to predict house prices based on the House Prices dataset.

3 Dataset

The dataset used in this notebook is the House Prices dataset, which contains 1460 entries and 81 columns.

The dataset includes various features related to houses, such as the number of rooms, the size of the lot, the year built, and the sale price. The goal is to predict the sale price of a house based on these features. The dataset contains 79 feature columns and one target column, 'SalePrice'.

let us plot the distribution for all the numerical features:

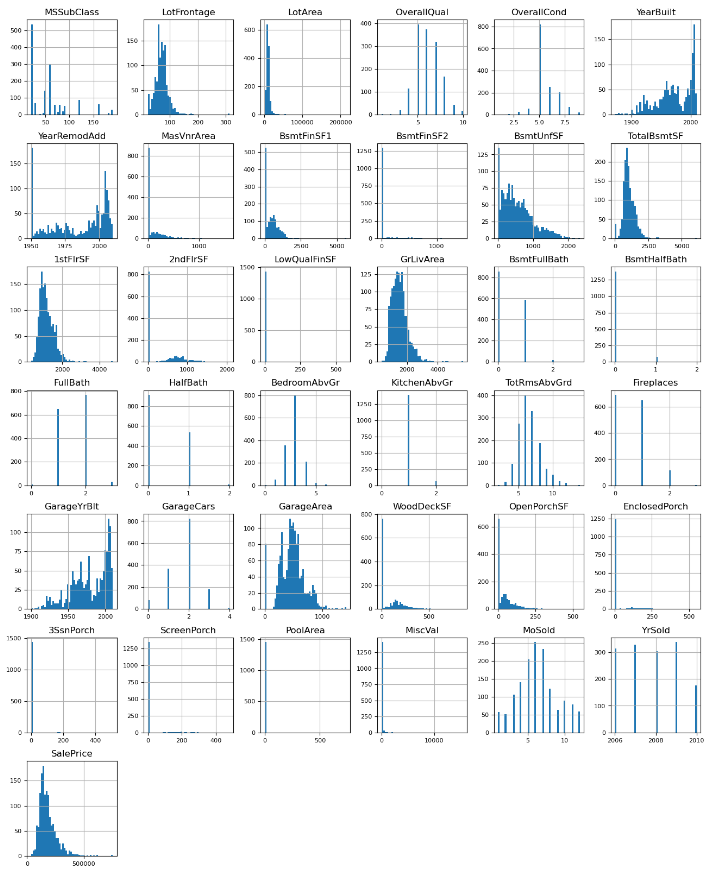

Figure 1: **Numerical Features Plot**.

4 Methods

Before building the model, I did some preprocessing steps:

4.1 Handling Missing Values

The dataset contains missing values in some columns. It does not explicitly handle missing values, but it is a crucial step that should be addressed in future iterations. Techniques such as imputation or removal of missing values could be employed.

4.2 Dropping Unnecessary Columns

The ‘Id‘ column is dropped from the dataset as it is not relevant for model training. This step simplifies the dataset and focuses on the features that contribute to the prediction.

4.3 Feature Engineering

It does not perform extensive feature engineering, but it does inspect the types of feature columns.

Future work could include creating new features or transforming existing ones to improve model performance.

4.4 Importing Libraries

The necessary libraries, including TensorFlow, TensorFlow Decision Forests, Pandas, Seaborn, and Matplotlib, are imported.
These libraries are essential for data manipulation, visualization, and model building.

4.5 Loading the Dataset

The dataset is loaded using Pandas, and the shape of the dataset is printed to confirm that it has been loaded correctly.
This dataset contains a mix of numeric, categorical, and missing features. TF-DF supports all these feature types natively, and no more preprocessing is required.
This is one advantage of tree-based models, making them a great entry point to Tensorflow and ML.

4.6 Data Visualization

I did several visualizations to understand the distribution of the target variable, ‘SalePrice‘, and the numerical features.
These visualizations help in identifying patterns and potential outliers in the data.

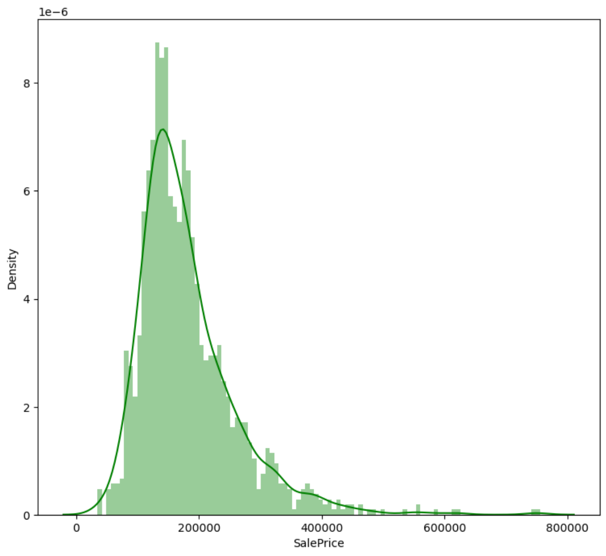

Figure 2: **Distribution of Sale Prices**. This figure shows the distribution of Sale Prices. Since the plot is skewed to the right, we can conclude that most homes are priced around $200,000.

5 Model

5.1 Model Selection

There were several tree-based models for us to choose from.

- RandomForestModel
- GradientBoostedTreesModel
- CartModel
- DistributedGradientBoostedTreesModel

To start, I started working with a Random Forest. This is the most well-known of the Decision Forest training algorithms.

A Random Forest is a collection of decision trees, each trained independently on a random subset of the training dataset (sampled with replacement). The algorithm is unique in that it is robust to overfitting, and easy to use.
The dataset is converted into a TensorFlow dataset format, and a Random Forest model is initialized. The model is then trained on the dataset.

One benefit of tree-based models is that we can easily visualize them. The default number of trees used in the Random Forests is 300. We can select a tree to display below.

5.2 Model Evaluation

Evaluate the model on the Out of bag (OOB) data and the validation dataset

Before training the dataset we manually separated 20
We can also use the Out of Bag (OOB) score to validate our RandomForestModel. To train a Random Forest Model, a set of random samples from the training set is choosen by the algorithm, and the rest of the samples are used to finetune the model. The subset of data that is not chosen is known as Out of bag data (OOB). OOB score is computed on the OOB data.

The training logs show the Root Mean Squared Error (RMSE) evaluated on the out-of-bag dataset according to the number of trees in the model.
Let us plot this:

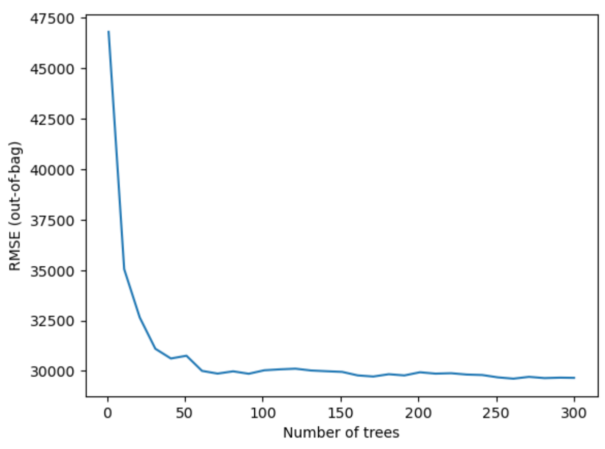

Figure 3: **RMSE Score according to the number of trees**.
Note: Smaller values are better for this hyperparameter.

5.3 Variable importances

Variable importance generally indicates how much a feature contributes to the model predictions or quality. There are several ways to identify important features using TensorFlow Decision Forests.
Let us list the available Variable Importances for Decision Trees:
Available variable importance:

1. INV_MEAN_MIN_DEPTH
2. NUM_AS_ROOT
3. NUM_NODES
4. SUM_SCORE

As an example, let us display the important features of the Variable Importance NUM_AS_ROOT.
The larger the importance score for NUM_AS_ROOT, the more impact it has on the outcome of the model.

By default, the list is sorted from the most important to the least. From the output, you can infer that the feature at the top of the list is used as the root node in the most number of trees in the random forest than any other feature.
Plot the variable importances from the inspector using Matplotlib:

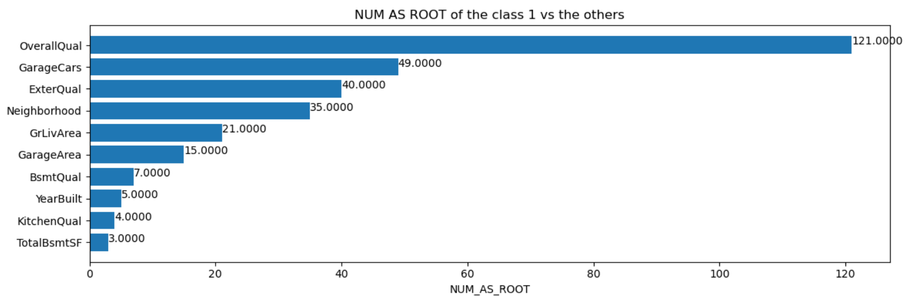

Figure 4: **Variable importance plot**.
The larger the importance score for NUM_AS_ROOT, the more impact it has on the outcome of the model.

6 Results

We successfully demonstrate the process of building a Random Forest model using TensorFlow Decision Forests to predict house prices. The model is trained on the House Prices dataset and It is provided as a good starting point for further exploration and improvement.

A visualization of feature importance showed that variables like lot area, overall quality, and year built had significant contributions to price prediction.

Additionally, the distribution of residuals indicated that the model performed well but had slight biases in extreme price ranges.

Future improvements may involve hyperparameter tuning and incorporating additional external data, such as neighborhood economic indicators.

This study demonstrated the effectiveness of Decision Forests in predicting house prices. The Random Forest model provided a strong baseline and showed promising results with relatively low prediction errors. Feature importance analysis revealed that property size, quality, and age are among the key factors influencing house prices.

Despite the model's strong performance, some limitations remain. The dataset may have missing or non-representative values, which could introduce biases.
Additionally, more complex models, such as gradient-boosted trees or deep learning approaches, could be explored to improve performance further.
Future work may focus on fine-tuning hyperparameters, incorporating external market data, and leveraging ensemble methods to enhance accuracy.

7 References

- TensorFlow Decision Forests Documentation: `https://www.tensorflow.org/decision_forests`
- House Prices Dataset: `https://www.kaggle.com/c/house-prices-advanced-regression-techniques`