Project 3: Tabular Regression

Approach 1: Simple preprocessing and Linear Regression

1.1 Pre-processing:

The less extensive preprocessing approach adopts a more straightforward strategy. Numeric data imputation relies on mean values, potentially introducing bias due to its sensitivity to outliers. Data errors are overlooked, 'saledate' remains unaltered, and categorical data is encoded using simple pandas categorical codes for space efficiency, at the expense of potentially introducing ordinal rank in non-ordinal data.

1.2 Regression Model

After a relatively simple preprocessing not catering to most of the factors, the dataset undergoes evaluation with four regression models: *Simple Linear, Lasso, Ridge, and ElasticNet*.

1.3 Results Analysis

All models exhibit similar R^2 and error metrics, suggesting comparable performance.
The preprocessing approach does not take into regard the nature of the data and hence somewhat serves as a control for comparison against other approaches. The particular issues with the simple preprocessing are detailed further in the notebook.

Approach 2: Extensive preprocessing and Linear Regression

2.1 Pre-processing:

The extensive preprocessing pipeline is characterized by a thorough examination and transformation of the dataset. Notable steps include median-based and most frequent by-year imputation for numeric data, rectification of errors such as impossible sale dates, and conversion of features like 'saledate' into 'saleyear,' 'salemonth,' and 'saleday' for streamlined model input. Categorical data undergoes label encoding, preserving rank, while nominal data is subject to target encoding, balancing dimensionality and relationship preservation with target variable. Redundant columns are systematically dropped, eliminating noise and enhancing model efficiency.

2.2 Regression Model

After the extensive preprocessing, the dataset undergoes evaluation with four regression models: *Simple Linear, Lasso, Ridge, and ElasticNet*.

2.3 Results Analysis

Simple Linear regression underperforms compared to the less extensive pre-processing approach, this can potentially be attributed to greater overfitting caused by target encoding, this overfitting could also explain why regularization (Lasso, Ridge, ElasticNet) performs well.
Extensive preprocessing enhances (compared to simple preprocessing) the performance of Lasso, Ridge, and ElasticNet, showing that preserving ordinal rank, eliminating noise, and utilizing features like 'saledate' effectively is beneficial.

Approach 3: K- nearest neighbors

3.1 Preprocessing

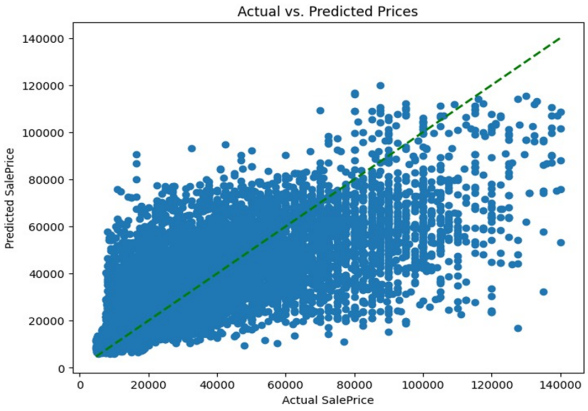
In the quest for optimal K-Nearest Neighbors (KNN) model performance, a thorough preprocessing pipeline is established to address the challenges posed by categorical variables, missing data, and high dimensionality. For handling categorical values, label encoding involving efficient conversion of categorical variables into numerical values and perseverance of dimensionality despite an extensive range of categories was being done. Iterative imputation models each feature with missing values as a function of other features and provides a sophisticated and accurate approach compared to mean or median imputation. For feature normalization and dimensionality reduction, standard scaling ensures equal contribution of all features in the distance-based KNN model and Principal Component Analysis (PCA) reduces dimensionality for improved computational efficiency and potential overfitting reduction.

3.2 KNN Model

In the parameter tuning phase, the optimal number of neighbors (n=9) for the K-Nearest Neighbors (KNN) model is determined through extensive experimentation aimed at minimizing prediction error. The subsequent model evaluation encompasses a comprehensive assessment using multiple metrics, including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R2 score. This thorough evaluation provides insights into the accuracy of the model and its ability to explain variance in the target variable.

3.3 Results Analysis

In the results analysis, the performance insights highlight the optimized model performance achieved through meticulous preprocessing, underscoring its contribution to the robustness of the K-Nearest Neighbors (KNN) model. A comparative analysis of evaluation metrics demonstrates the effectiveness of the preprocessing steps in enhancing model accuracy.



Approach 4: Feedforward Neural Network

4.1 Preprocessing/ Feature Engineering

In the pursuit of crafting an advanced machine learning model, a meticulous process was undertaken to enhance feature representation and optimize model performance. The journey began with a thorough exploration of missing values, visualized through distribution plots to identify features with the most gaps. Feature engineering played a pivotal role, encompassing various techniques to transform and preprocess the dataset.

Categorical labels were effectively encoded using label encoding, streamlining their representation for model input. The temporal aspect was addressed by extracting essential

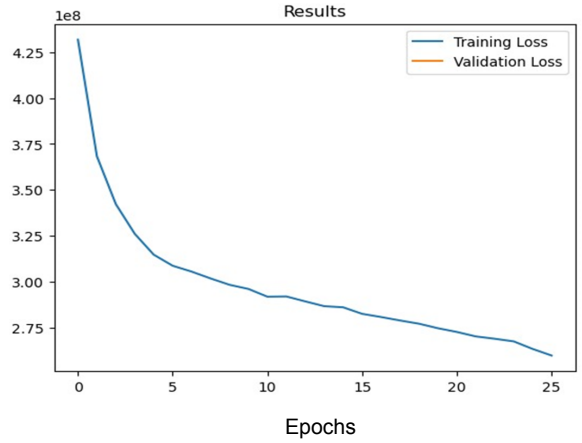
components such as sale year, month, and day from date features, accompanied by error handling for data anomalies. The 'ProductSize' and 'UsageBand' features underwent ordinal encoding, mapping specific labels to numeric values for improved representation.
To handle missing values, a median-based imputation strategy was employed, ensuring robustness in the face of data gaps. A nominal encoder was introduced to convert string-type columns into ordered categorical types, enriching the dataset's structure. The final touch in preprocessing involved feature scaling using the StandardScaler function, normalizing values to ensure uniform contribution across features.

4.2 FNN Model

Transitioning to model creation, a neural network architecture was devised with six hidden layers, strategically incorporating dropout layers to mitigate the risk of overfitting. The rectified linear unit (ReLU) activation function was employed to introduce non-linearity, enabling the model to capture complex mappings between inputs and outputs. Training the model extended over approximately 26 epochs, monitored by an early stopping mechanism that scrutinized validation loss and halted training if no improvement was observed after a predefined number of epochs.

4.3 Result Analysis

In the results analysis, the meticulous preprocessing steps played a crucial role in fostering the robust performance of the neural network model. The thoughtful integration of dropout layers and the ReLU activation function effectively addressed concerns related to overfitting, ensuring the model's generalizability. The implementation of early stopping further enhanced training efficiency by monitoring validation loss for optimal convergence.



<u>Approach 5: Random Forest Regression [BEST MODEL]</u>

5.1 Preprocessing

Similar extensive preprocessing as Approach 2 to allow for comparison between commonly used regression techniques.

5.2 Regression Model

The Random Forest weaves together the strength of multiple decision trees to enhance predictive accuracy and curb overfitting. Functioning adeptly in both regression and classification tasks, this ensemble technique leverages the bagging approach, merging insights from diverse decision trees to arrive at a robust final output. The essence lies in the synergy of

these decision trees, outperforming individual models and proving particularly resilient to noisy data, a challenge that standard regression methods often grapple with. The optimal forest parameters were also examined.

5.3 Result Analysis

Random forest is an ensemble of decision trees. This is to say that many trees, constructed in a certain “random” way form a Random Forest. Each tree is created from a different sample of rows and at each node, a different sample of features is selected for splitting. Each of the trees makes its own individual prediction. These predictions are then averaged to produce a single result, this averaging of many decisions can be considered the cause of the great performance shown by the technique.

R2 Score Comparison

Approach	R2 Score
Simple Linear Regression with less extensive preprocessing	0.318
Ridge with less extensive preprocessing	0.390
Lasso with less extensive preprocessing	0.386
ElasticNet with less extensive preprocessing	0.379
Simple Linear Regression with extensive preprocessing	0.184
Ridge with extensive preprocessing	0.470
Lasso with extensive preprocessing	0.735
ElasticNet with extensive preprocessing	0.716
KNN	0.530
FNN	0.737
Random Forest with extensive preprocessing	0.868

The r2 comparison can be used to come to the following conclusions:

- Preprocessing done in accordance to the nature of the data and its quirks is very useful and may allow even the simplest of regression techniques to perform well.
- Feature selection can be a very useful tool when it comes to regression on data with a whole lot of features, as shown by the generally good performance of Lasso.
- Ensemble techniques like the Random Forest are very useful and performant when combined with a good preprocessing approach owing to their robustness.
- Target encoding needs to be used carefully since it can result in overfitting (this potentially explains the lowered performance of simple linear regression in approach 2)