In this project, we will be analyzing the "daily_offers.xlsx" dataset which contains information on steel items offered to customers. Each row represents an item, and has an offered/selling price of that item. We will be performing data cleaning, data summarization, correlation analysis, feature engineering, and regression modeling to predict the price of an item. The objective of this project is to develop a regression model that can predict the price of an item based on the given features.

**A possible brief description of the columns in the dataset:**

- id: A unique identifier for each item.
- item_date: The date the item was offered/sold.
- quantity tons: The quantity of the item in tons.
- customer: The name of the customer to whom the item was offered/sold.
- country: The country where the customer is located.
- status: The status of the item, i.e., whether it was offered or sold.
- item type: The type of steel item.
- application: The application of the steel item.
- thickness: The thickness of the steel item in mm.
- width: The width of the steel item in mm.
- material_ref: A reference code for the material of the item.
- product_ref: A reference code for the product type of the item.
- delivery date: The date the item was/will be delivered.
- selling_price: The price at which the item was sold or offered. This is the target variable we want to predict using regression models.

The steps I have followed to complete the assignment are as follows:

## 1: Initial Data Analysis

First, I started by importing the necessary libraries and loading the dataset. Then did some descriptive statistics to take a look at the data to gain some insight into its structure.

The dataset initially has 181673 rows and 14 columns. I.e.., 181673 items are sold.

The Statistical description of the data is given below:

| | **item_date** | **customer** | **country** | **application** | **thickness** | **width** | **product_ref** | **delivery date** | **selling_price** |
|---|---|---|---|---|---|---|---|---|---|
| **count** | 181672 | 181672 | 181645 | 181649 | 181672 | 181673 | 181673 | 181672 | 181672 |
| **mean** | 2.020459e+07 | 3.051221e+07 | 44.89 | 25.61 | 2.56 | 1295.28 | 4.739679e+08 | 2.020738e+07 | 1.918036e+03 |
| **std** | 4.551119e+03 | 2.433382e+07 | 24.40 | 17.75 | 6.57 | 261.63 | 7.175101e+08 | 2.411059e+04 | 3.317956e+05 |
| **min** | 1.995000e+07 | 1.245800e+04 | 25 | 2 | 0.180 | 1 | 6.117280e+05 | 2.019040e+07 | 1.160000e+03 |
| **25%** | 2.020093e+07 | 3.019688e+07 | 26 | 10 | 0.70 | 1180 | 6.119930e+05 | 2.020110e+07 | 6.690000e+02 |
| **50%** | 2.020113e+07 | 3.020524e+07 | 30 | 15 | 1.5 | 1250 | 6.406650e+05 | 2.021010e+07 | 8.120000e+02 |
| **75%** | 2.021020e+07 | 3.028042e+07 | 78 | 41 | 3 | 1500 | 1.332077e+09 | 2.021040e+07 | 9.530000e+02 |
| **max** | 2.021040e+07 | 2.147484e+09 | 113 | 99 | 2500 | 2990 | 1.722208e+09 | 3.031010e+07 | 1.000010e+08 |

The descriptive statistics of the dataset show that the mean selling price is 1918.03 with a standard deviation of 331795.6

Upon checking the data type, we found that the quantity tons column should be of float data type but it is marked as object so we changed that.

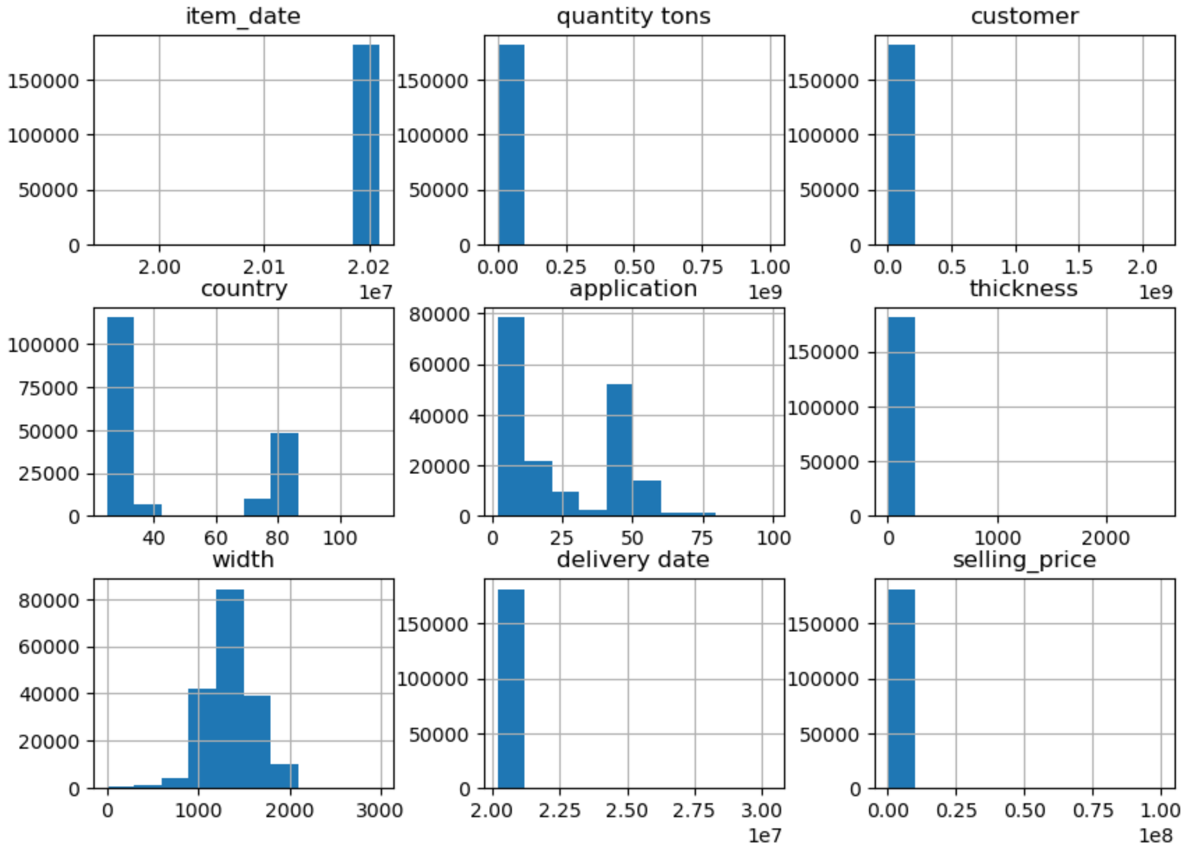
Now the Categorical Features are : 'status', 'item type', 'material_ref'. And Numerical Features are : 'item_date', 'quantity tons', 'customer', 'country', 'application', 'thickness', 'width', 'product_ref', 'delivery date', 'selling_price'.

## 2: Data Cleaning

- **Handling Missing Values**

There are some missing values in the dataset. Since the missing values in the feature 'material_ref ' is 77918, which is quite a lot I decided to drop that feature.

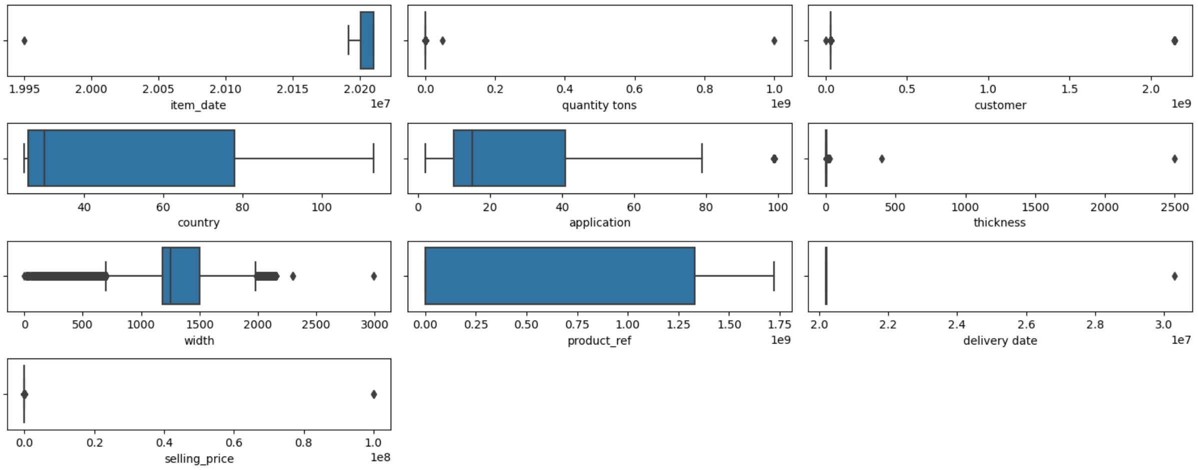
Plotted the histogram:



Here, mean is affected by extreme values thus we use median which is not affected by extreme values to fill the missing values of columns with int or float data type. Filled missing values in col 'status' with mode since its categorical.

We could also observe that only feature 'width' follows a normal distribution.

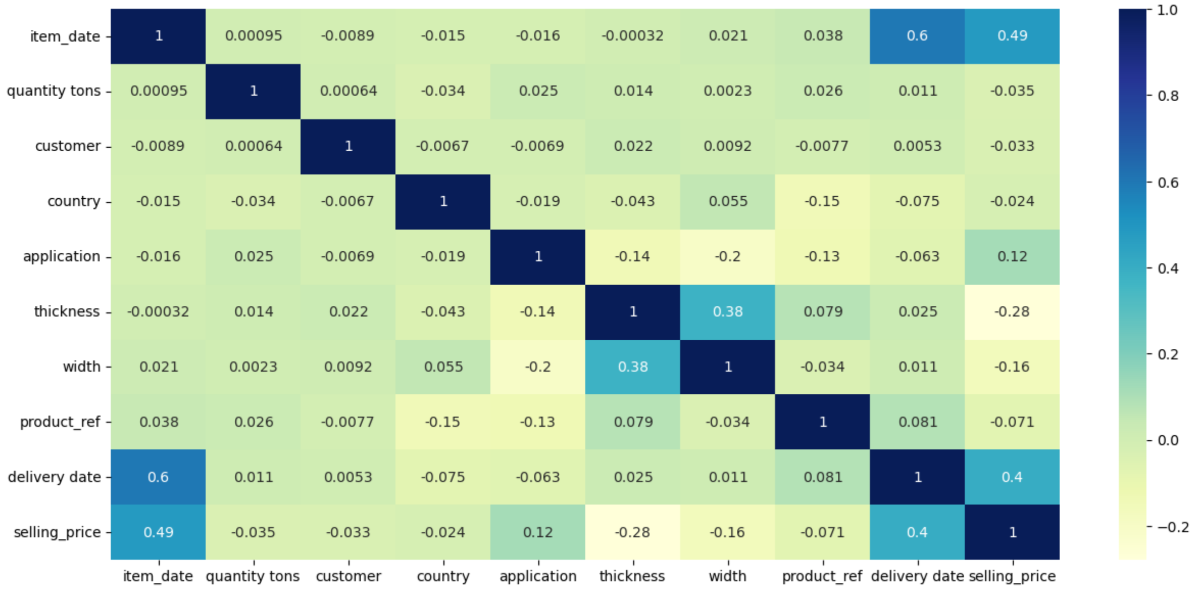
- **Handling Outliers**



There are some outliers in few columns that we handled. The values 19950000.0 and 30310101.0 in features item_date and delivery_date respectively are outliers. So, dropped those rows. These values that seems to be outlier in feature ‘application’ is 99 which cannot be considered as outliers , so we are not removing them. In the features ‘width’, ‘thickness’, ‘quantity tons’ and ‘selling price’ we found some outliers we needed to be removed because or else it would overfit our model. So, we dropped those outliers.

- **Handling duplicate values**

There are no duplicated values that needs to be removed.



| | item_date | quantity tons | customer | country | application | thickness | width | product_ref | delivery date | selling_price |
|---|---|---|---|---|---|---|---|---|---|---|
| item_date | 1 | 0.00095 | -0.0089 | -0.015 | -0.016 | -0.00032 | 0.021 | 0.038 | 0.6 | 0.49 |
| quantity tons | 0.00095 | 1 | 0.00064 | -0.034 | 0.025 | 0.014 | 0.0023 | 0.026 | 0.011 | -0.035 |
| customer | -0.0089 | 0.00064 | 1 | -0.0067 | -0.0069 | 0.022 | 0.0092 | -0.0077 | 0.0053 | -0.033 |
| country | -0.015 | -0.034 | -0.0067 | 1 | -0.019 | -0.043 | 0.055 | -0.15 | -0.075 | -0.024 |
| application | -0.016 | 0.025 | -0.0069 | -0.019 | 1 | -0.14 | -0.2 | -0.13 | -0.063 | 0.12 |
| thickness | -0.00032 | 0.014 | 0.022 | -0.043 | -0.14 | 1 | 0.38 | 0.079 | 0.025 | -0.28 |
| width | 0.021 | 0.0023 | 0.0092 | 0.055 | -0.2 | 0.38 | 1 | -0.034 | 0.011 | -0.16 |
| product_ref | 0.038 | 0.026 | -0.0077 | -0.15 | -0.13 | 0.079 | -0.034 | 1 | 0.081 | -0.071 |
| delivery date | 0.6 | 0.011 | 0.0053 | -0.075 | -0.063 | 0.025 | 0.011 | 0.081 | 1 | 0.4 |
| selling_price | 0.49 | -0.035 | -0.033 | -0.024 | 0.12 | -0.28 | -0.16 | -0.071 | 0.4 | 1 |

- there is no highly positively or negatively correlated features.
- item_date and delivery_date is somewhat positively correlated.

## 3: Feature Engineering

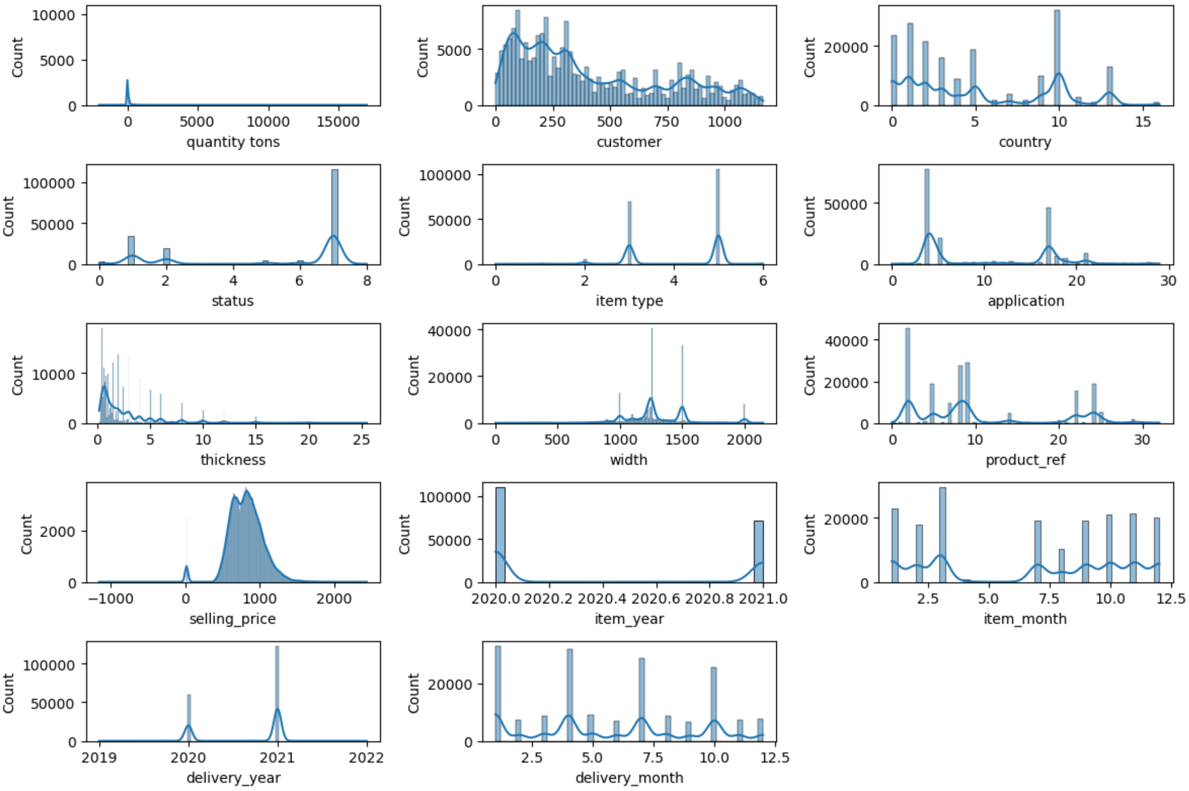
New features were made from cols ‘item_date’ and ‘delivery date’ because it is correlated to our target variable ‘selling price’. New cols made are : ‘item_year’, ’item_month‘ , ‘delivery_year’ and ‘delivery_month’. Then dropped features 'item_date' and ‘delivery date' as they have been already extracted.

## 4: Encoding

Label encoded features such as: 'customer', 'country', 'status', 'item type', 'application' and 'product _ref'.
After all pre-processing steps we have our cleaned data. The new data has 181640 rows and 14 features. Initially we had 181673 rows and 14 columns.

## 5: Exploratory data analysis (EDA)

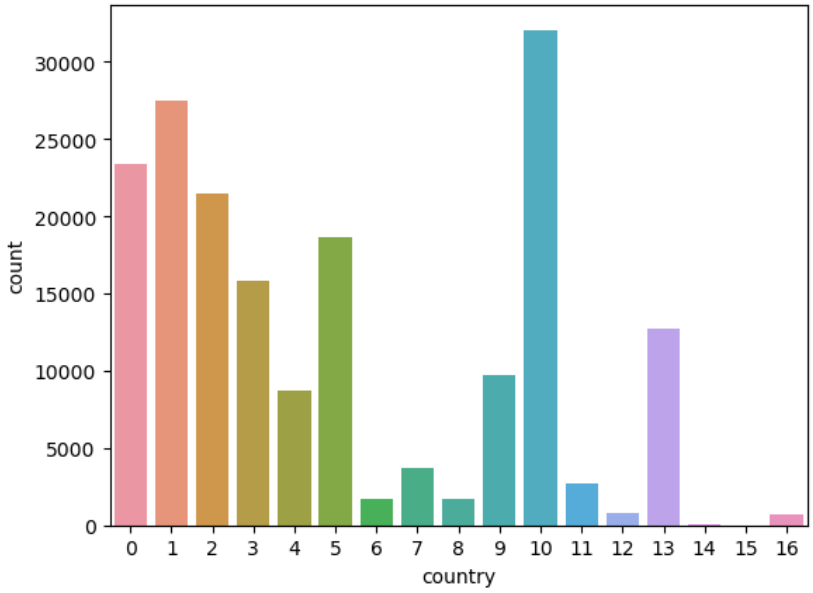
Let us now visualize the relationships between the features and the target variable of cleaned data using some hist plots, count plots, line plots and box plots:



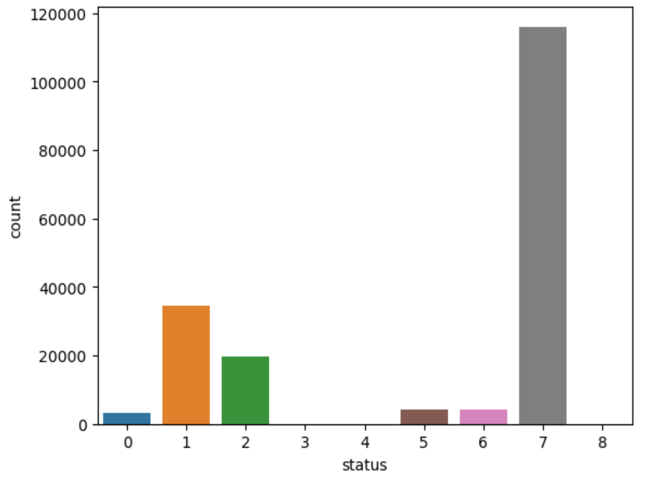
From above histplot it is evident that no features follow normal distribution

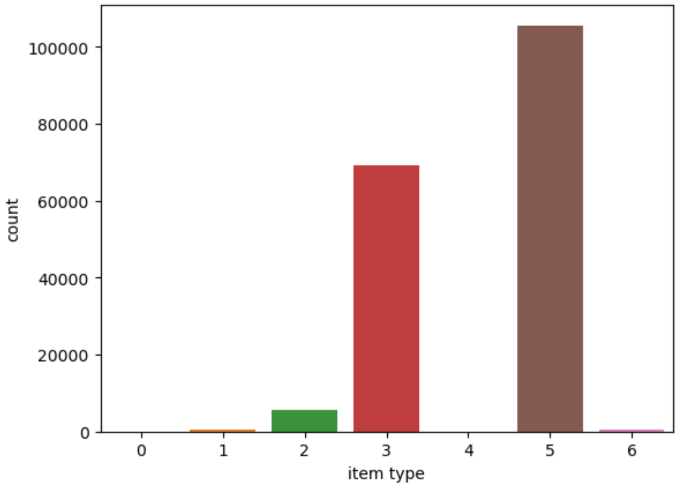
## **Count plots for features:**

From the given plot we could tell that country no.10 has occurred the greatest number of times.

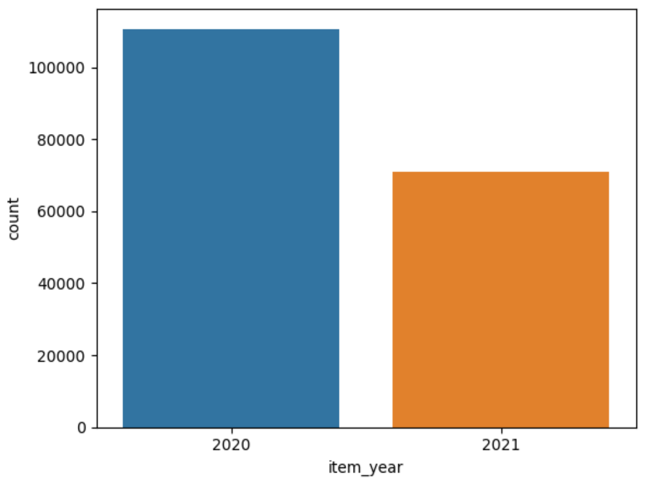


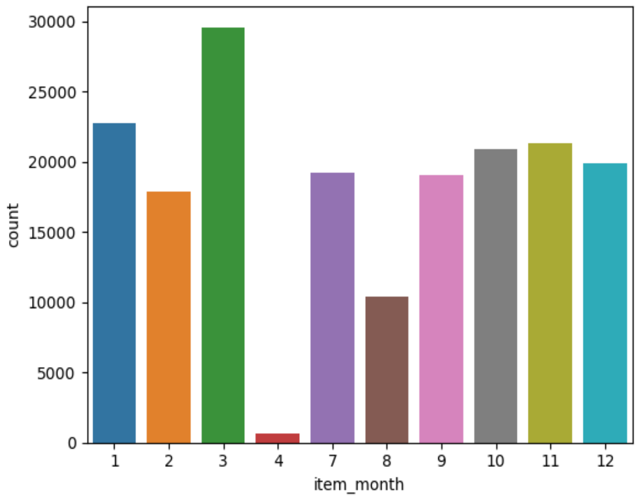
Item status 7 and item type 5 is most frequent ones.



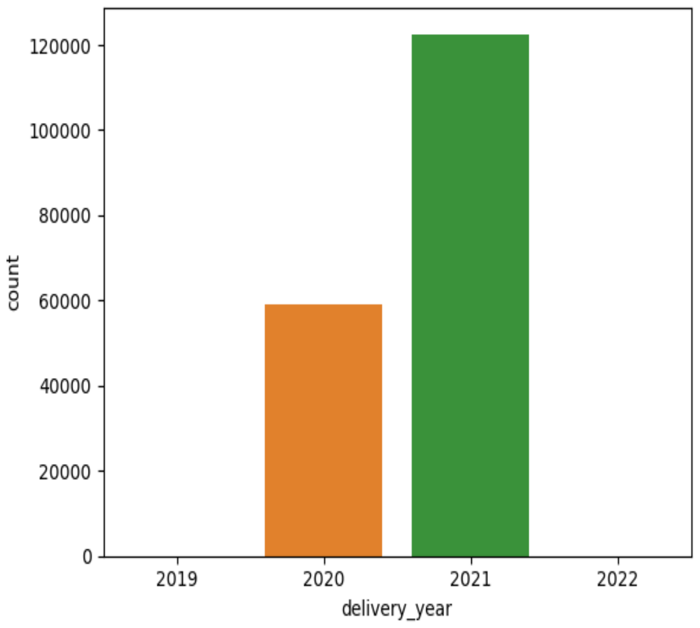


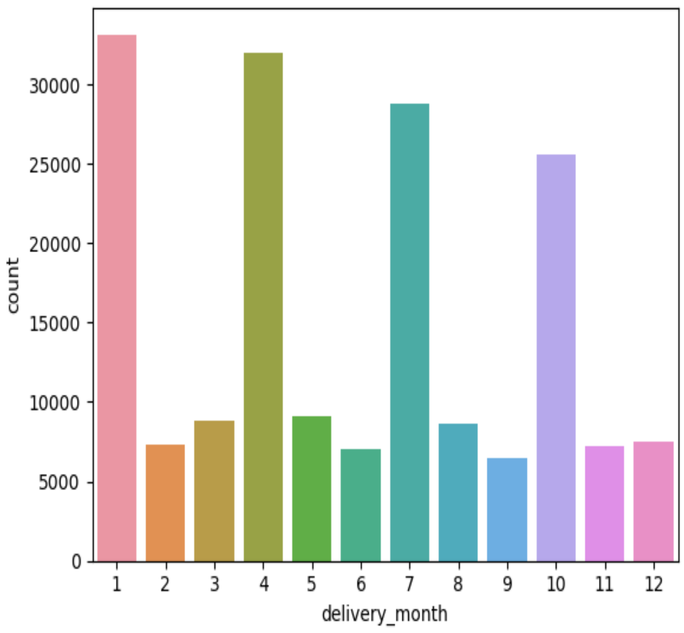
Item_year 2020 is most frequent, also march is the most frequent item_month.



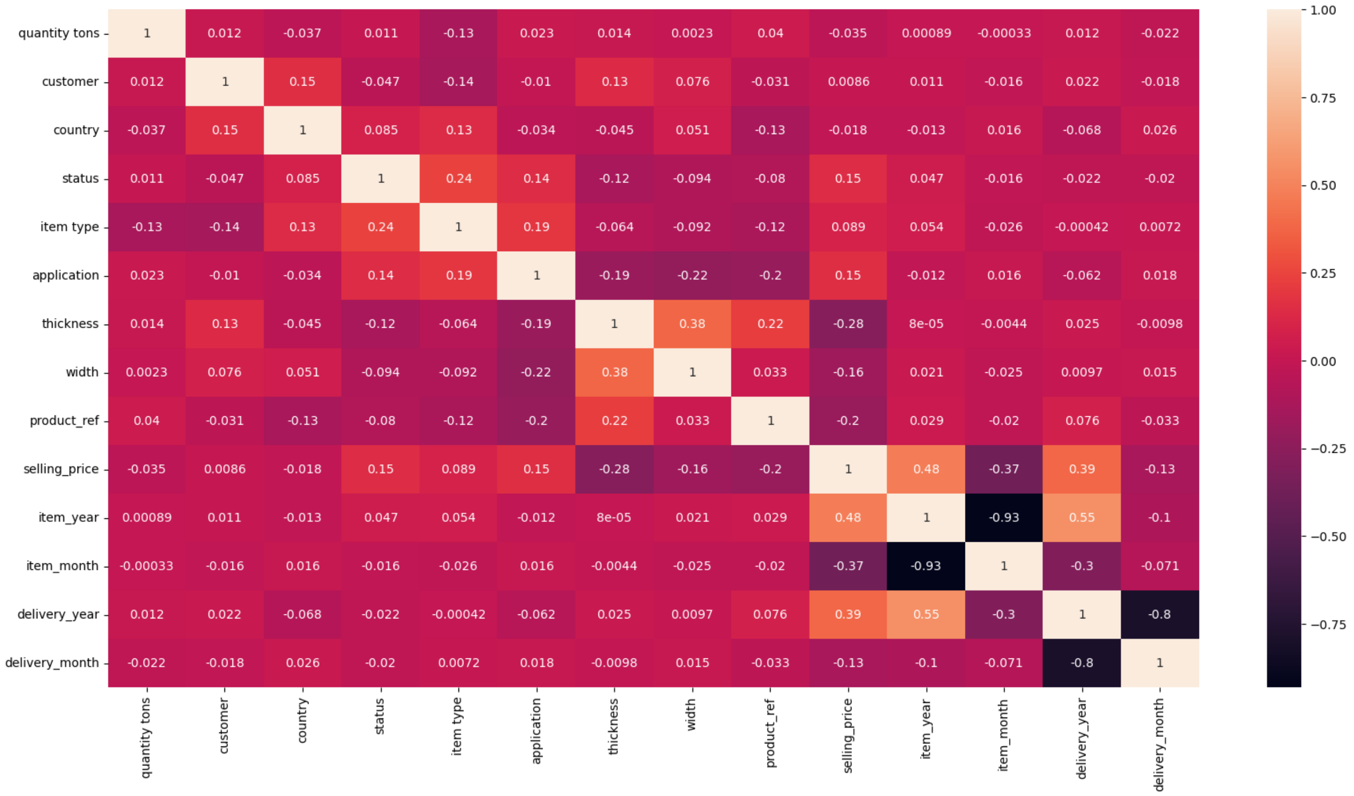


2021 is the most frequent delivery year and January is the most frequent delivery month.

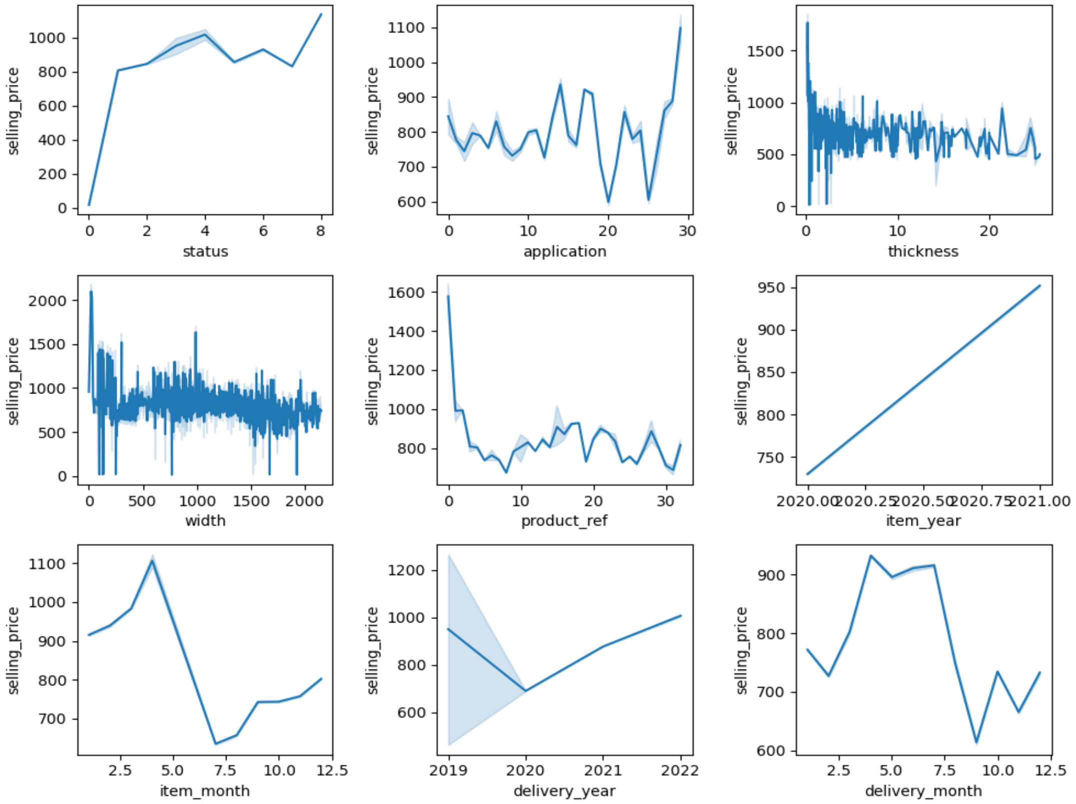




The heat map representing the correlation between features is given below;



According to heatmap features like 'status', 'application', 'thickness', 'width', 'product_ref' , 'item_year', 'item_month', 'delivery_year', and 'delivery_month' are more related to our target variable selling price. A line plot is plotted usimg these features and it is given below:



We can find that item year and selling price are directly correlated.

## 6: Splitting independent and dependent features

Our target variable or dependent feature is selling price and other features are the independent features. Split the data into training and testing sets in the ratio of 90:10 respectively.

## 7: Normalization (min-max Scaling)

Normalization is a technique used to scale the data to a standard range of values. It is a pre-processing step used in machine learning and data analysis to improve the performance and accuracy of the models. Normalization helps in making the data consistent and comparable, which is essential when we have variables measured in different units or scales.

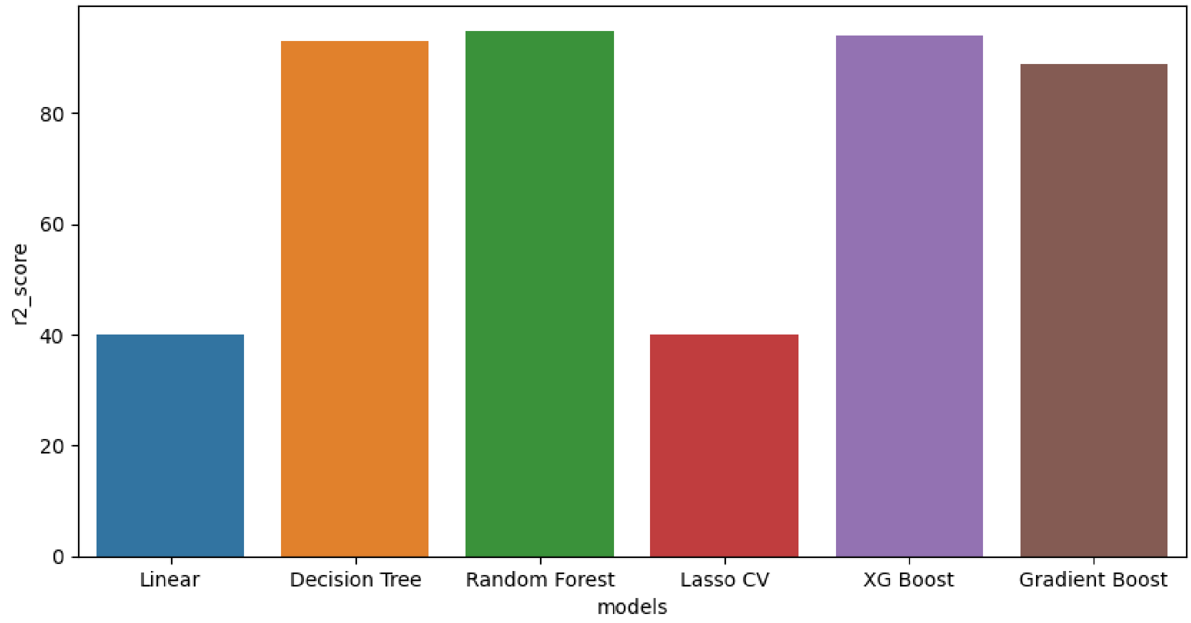
There are several methods of normalization, including Min-Max scaling, Z-score normalization, and log transformation. Here we used min-max scaling. Min-Max scaling is a simple normalization technique that scales the data to a range between 0 and 1. It is calculated as follows:

normalized = (x - min(x)) / (max(x) - min(x))

## 8: Model Building and Evaluation

Fitted models such as Linear regression, Decision Tree regression, Random Forest regression , Lasso CV, XG Boost regression, and Gradient Boosting regression.

The graph plotted for the comparison of r2 Score is as follows;



From graph it is evident that best model is random forest with r2 Score of 94.75%.

## 9:  Hyperparameter Tuning

Using Random Search CV did the hyper parameter tuning and found the best parameter for random forest regression.

The best parameters are:    'n_estimators': 10, 'min_samples_split': 2, 'min_samples_leaf': 1, 'max_samples': 1.0, 'max_features': 1.0, 'max_depth': None, 'bootstrap': True

With hyper parameter tuning got the coefficient of determination (r2 score) as : 95.03%