SE22MAID002
PARAM SHIV ASHISH
M.TECH. AI & DS

A retail chain in India wants to use data source to build an efficient forecasting model to predict the sales SKU wise in its portfolio at its 76 different stores using historical sales data for the past 3 years on a week-on-week basis. Sales and promotional seasonality is also available for each week- SKU and store wise. However, no other information regarding stores and products are available. Can you still forecast accurately ?

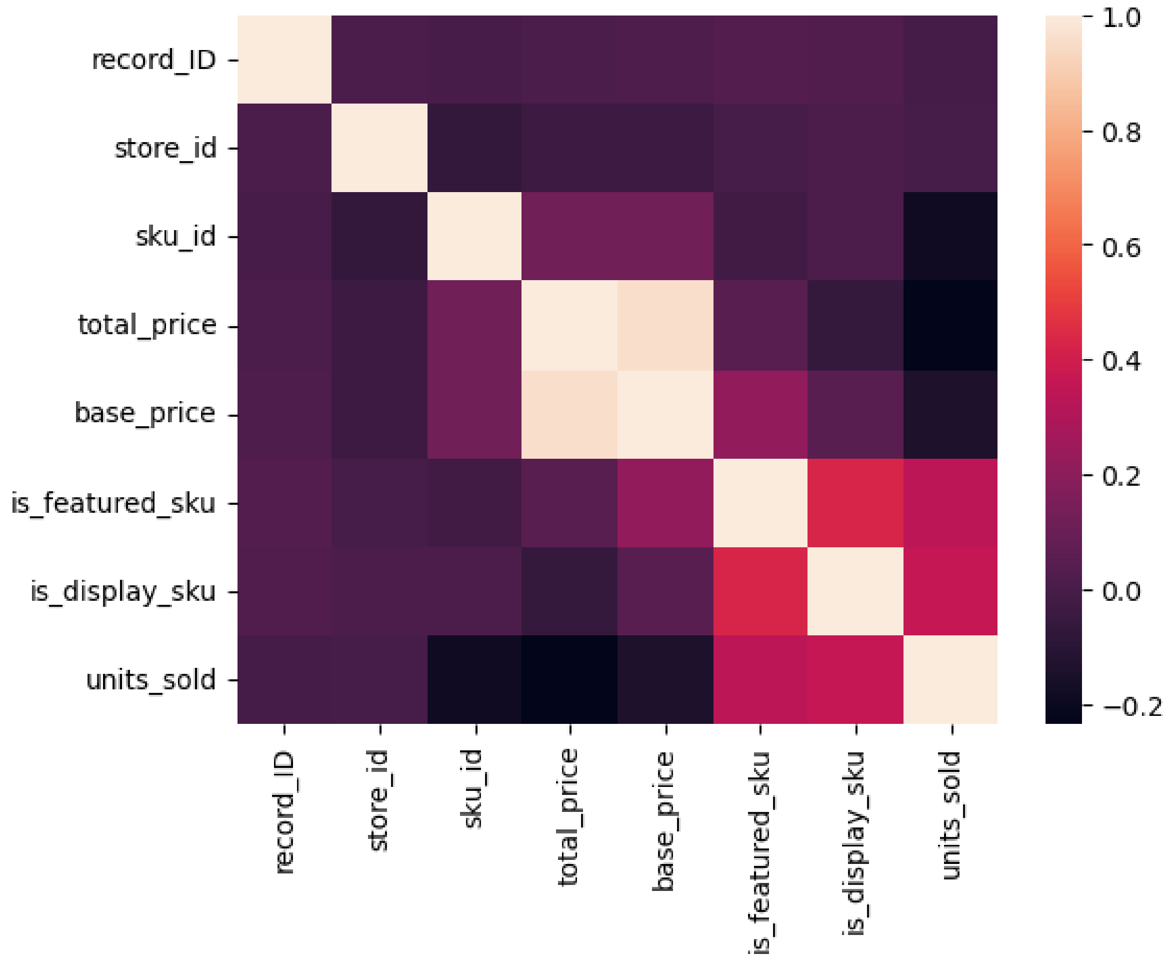
Yes, this problem can be solved by using regression approach and many more possible solutions can be used, for this assignment i am going to stick with LightGBM
LightGBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support of parallel, distributed, and GPU learning.
- Capable of handling large-scale data.

Challenges and Difficulties
Time series forecasting by regression, although widely used, can encounter several challenges. Some of the prominent issues include:
Autocorrelation: Time series information frequently displays examples of sequential connection, where the value of a variable at a specific time is reliant upon its past qualities. This violates the assumption of independence among the predictors in the regression model.
Non-stationarity: Many time series show patterns, irregularity, or different types of non-stationarity, which can violate the assumption of steady mean and variance
Seasonality and periodicity: Conventional regression models may not be prepared to deal with occasional variances in the information. This could lead to inaccurate modeling
Outliers and anomalies: Time series data can often have outliers or anomalies, which can skew the regression model's results. Failing to address these outliers can lead to inaccurate forecasts and biased parameter estimates.
Model selection and overfitting: Choosing an appropriate regression model for time series forecasting can be challenging. Overfitting can occur when the model is excessively complex or when it captures noise in the data rather than true patterns. This can lead to poor generalization and inaccurate predictions.
Handling missing data: Time series information frequently contains missing values, which can be a problem for regression techniques. Attributing missing values without accounting for the temporal characteristic of the data can introduce biases and affect the accuracy of the forecasts.

SE22MAID002
PARAM SHIV ASHISH
M.TECH. AI & DS

Autocorrelation Matrix using seaborn



From the above autocorrelation among the different features or variables in our dataframe we can model some new features based on the insights, here are a few new features i am building
Considered this as a regression problem with 'units_sold' as a target Generated following new features:

- Number of records per 'sku-id','store-id' and combination of both
- Number of Average units sold per 'sku-id','store-id' and combination of both
- Average base-price & total-price per 'sku-id','store-id' and combination of both
- Week of the year
- Week number from start of data
- Week of the month

These are some of the features i have used and the model can definitely be improved by building more new features

SE22MAID002
PARAM SHIV ASHISH
M.TECH. AI & DS

Model Training

First we split the data in training and validation sets as below

```
# SPLITING FOR TRAIN AND VALIDATION SETS
xTrain, xValid, yTrain, yValid = train_test_split(X, Y, test_size = 0.2,random_state=12)
```

Finally fit the model with parameters as below

```
# SE22MAID002
# PARAM SHIV ASHISH
# MTECH AI AND DS

#TRAINING THE MODEL UISING LGBM REGRESSOR
model = lgb.LGBMRegressor(bagging_fraction=0.8, bagging_frequency=4, boosting_type='gbdt',
              class_weight=None, colsample_bytree=1.0, feature_fraction=0.5,
              importance_type='split', learning_rate=0.1, max_depth=30,
              min_child_samples=20, min_child_weight=30, min_data_in_leaf=70,
              min_split_gain=0.0001, n_estimators=100, n_jobs=-1,
              num_leaves=1400, objective=None, random_state=None, reg_alpha=0.0,
              reg_lambda=0.0, silent=True, subsample=1.0,
              subsample_for_bin=200000, subsample_freq=0)

model.fit(X,Y)
```

Output can be displayed and saved with code below

```
# GETTING PREDICTIONS
pred = model.predict(X_test)
```

```
[LightGBM] [Warning] Unknown parameter: bagging_frequency
[LightGBM] [Warning] Unknown parameter: silent
[LightGBM] [Warning] min_data_in_leaf is set=70, min_child_samples=2
[LightGBM] [Warning] feature_fraction is set=0.5, colsample_bytree=1
[LightGBM] [Warning] bagging_fraction is set=0.8, subsample=1.0 will
```

```

preds=np.round(np.expm1(pred))
submissionDF['units_sold']=preds
```

```
submissionDF.head()
```

	record_ID	units_sold
0	212645	15.0
1	212646	20.0
2	212647	30.0
3	212648	27.0
4	212649	22.0

```
submissionDF.to_csv('DemandForecastPrediction.csv',index=False)
```