Applications of Data Science and Machine Learning in Supply Chain Optimization

Anduri Roshan

9-12-2022

Abstract

The food and beverage industry is one of the most important sectors of every countries economy. Forecasting the demand of meals is rather a challenging task, especially nowadays where integration of online and physical store orders creates an abundance of data that has to be efficiently stored, analyzed, understood and finally, become ready to be acted upon in a very short time frame. This project addresses the demand forecasting of various meals in different meal delivery stores of different regions with the help of various Machine Learning algorithms where a user can predict demand of meals for upcoming weeks.

1. Introduction

A supply chain is a network of facilities that include customers, retailers, distributors, manufacturers, and transporters. As a result, it is essential to comprehend the significant needs and preferences of customers, who are the primary node of every supply chain and force numerous entities to supply and distribute. Machine Learning has overtaken the normal way of knowing and forecasting demand. With a vavariousctions like purchasing, contracts, procurement, warehousing, production, packaging, transportation, or distribution—and consequently consumption—supply chains are extrehighlylex and challenging. Because each function is complicated, bringing them all togetogether requires muchffort, time, and money. It is essential to include strategies that provide prompt responses to complex situations. Previously, small decisions like when to deliver a product to a customer took a long time, but with the development of AI and machine learning techniques; The product can now be delivered within 24 hours, making this easier. In this proposal, we'll look at the use case of ML/DS in one of the areas of supply chain management known as Demand Forecasting.

2. Problem Statement

In supply chain management, demand forecasting is the process of planning or predicting material demand to ensure that you can deliver the right products in the right quantities to meet customer demand without creating a surplus. Demand forecasting estimates a probable future demand for a product or service. Demand Forecasting uses data from the past to estimate what will happen in the future. Here, it's a Data Science problem. Using Machine Learning, we can create a market-like situation with the highest level of detail possible, which reduces the difference between the predicted value of the purchase and the actual value of the purchase. Using ML models, it is possible to extract what factors behave like pros and cons to the sales figures.

3. Business Need Assessment

A McKinsey Global Institute study found that new machine learning and deep learning technologies have a significant impact on marketing and sales, and these industries stand to gain the most. Forbes published one of the reports saying "61% of organizations picked machine learning as their company's most vital data initiative for next year."

- In the data-rich retail environment, machine learning is a very powerful tool. Any situation in which data can be used to anticipate or explain changes in demand should make use of it. It may even be able to fill in any data omissions in some cases.
- Retail forecasts can now take into account the numerous factors and relationships that influence demand on a daily basis thanks to machine learning. Since weather data can include hundreds of different factors that could affect demand, this is extremely valuable.
- Using only the data you provide, whether, from your company or external data streams, machine learning algorithms automatically generate models that are continuously improved.
- The ability of such a system to process retail-scale data sets from a variety of sources without the need for human labor is the primary advantage.

4. Target Specifications

- To Forecast future demand on various levels with the help of current and past sales.
- Forecasting in a detailed manner at various sales points like retailers, stores, distributors, etc.
- To understand how holidays, weather forecasts, and promotional events affect the demand.
- Demand behavior is constantly being changed by external factors like product innovation, trends, tariffs, and new laws.
- ML model usage in demand forecasting will help to understand the market and demand in case of a crisis like COVID-19.

5. External Search

5.1 Application of machine learning in Food Demand Forecasting:

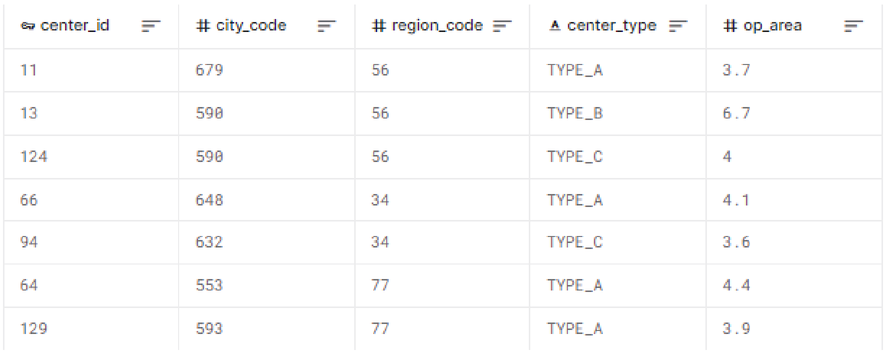
A subfield of artificial intelligence known as "machine learning" makes use of a variety of statistical, probabilistic, and optimization methods to enable computers to "learn" from previous examples and to identify challenging patterns in large, noisy, or complex data sets. This tech is well-performing in the supply chain management sector, mainly in Demand Forecasting. I’ve conducted a comprehensive survey of the various machine learning methods that are utilized, the types of data that are integrated, and the performance of these methods in order to compile this review of machine learning in food demand forecasting.

The conclusions were made based on the data available on Kaggle:

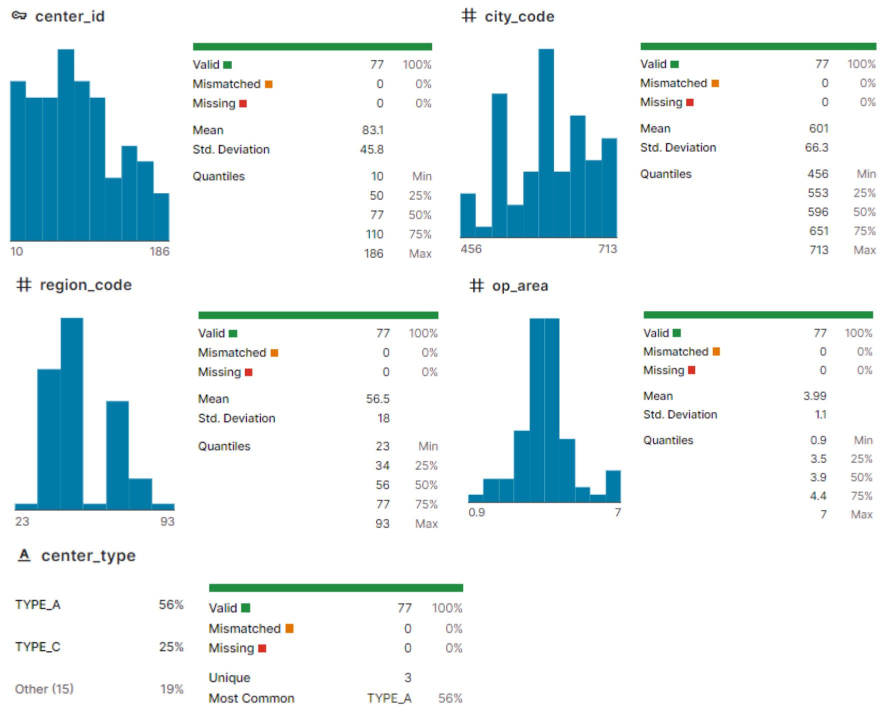
Link to dataset: Food_Demand_Dataset

5.2 Dataset review:

5.2.1 fulfilment_center_info.csv:



center_id	# city_code	# region_code	center_type	# op_area
11	679	56	TYPE_A	3.7
13	590	56	TYPE_B	6.7
124	590	56	TYPE_C	4
66	648	34	TYPE_A	4.1
94	632	34	TYPE_C	3.6
64	553	77	TYPE_A	4.4
129	593	77	TYPE_A	3.9



center_id

Valid 🟩	77	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	83.1	
Std. Deviation	45.8	
Quantiles	10	Min
	50	25%
	77	50%
	110	75%
	186	Max

city_code

Valid 🟩	77	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	601	
Std. Deviation	66.3	
Quantiles	456	Min
	553	25%
	596	50%
	651	75%
	713	Max

region_code

Valid 🟩	77	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	56.5	
Std. Deviation	18	
Quantiles	23	Min
	34	25%
	56	50%
	77	75%
	93	Max

op_area

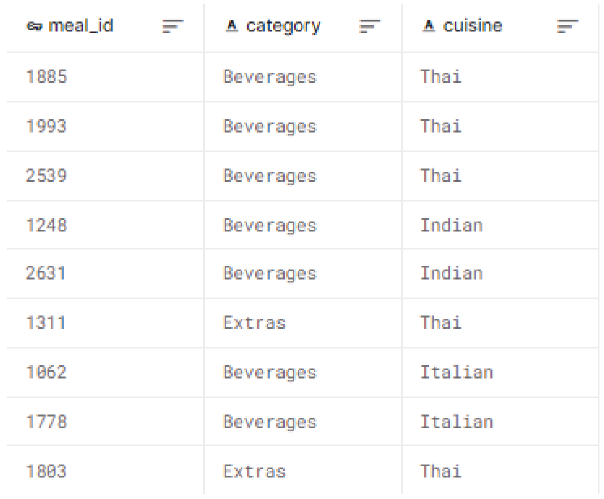
Valid 🟩	77	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	3.99	
Std. Deviation	1.1	
Quantiles	0.9	Min
	3.5	25%
	3.9	50%
	4.4	75%
	7	Max

center_type

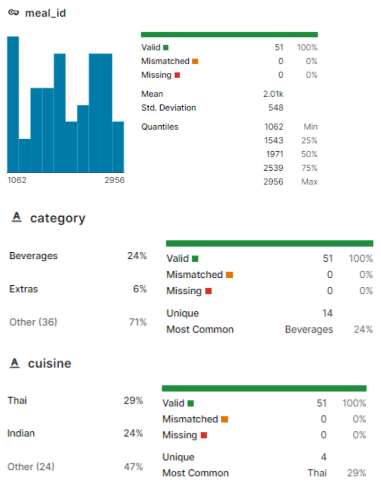
TYPE_A	56%
TYPE_C	25%
Other (15)	19%

Valid 🟩	77	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Unique	3	
Most Common	TYPE_A	56%

5.2.2 meal_info.csv:



meal_id	category	cuisine
1885	Beverages	Thai
1993	Beverages	Thai
2539	Beverages	Thai
1248	Beverages	Indian
2631	Beverages	Indian
1311	Extras	Thai
1062	Beverages	Italian
1778	Beverages	Italian
1803	Extras	Thai



meal_id

Valid 🟩	51	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	2.01k	
Std. Deviation	548	
Quantiles	1062	Min
	1543	25%
	1971	50%
	2539	75%
	2956	Max

category

Beverages	24%
Extras	6%
Other (36)	71%

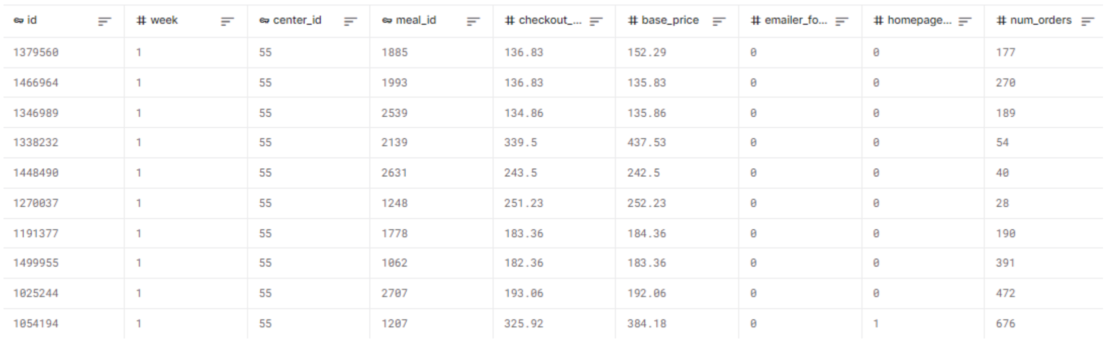
Valid 🟩	51	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Unique	14	
Most Common	Beverages	24%

cuisine

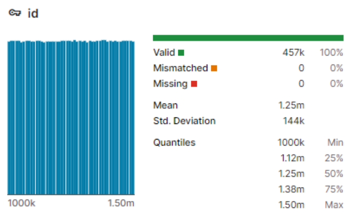
Thai	29%
Indian	24%
Other (24)	47%

Valid 🟩	51	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Unique	4	
Most Common	Thai	29%

5.2.3 train.csv:

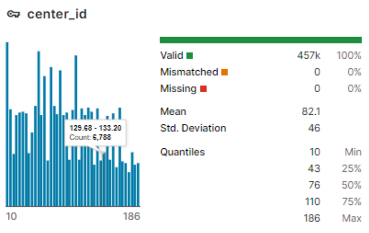


id	# week	center_id	meal_id	# checkout_...	# base_price	# emailer_fo...	# homepage...	# num_orders
1379560	1	55	1885	136.83	152.29	0	0	177
1466964	1	55	1993	136.83	135.83	0	0	270
1346989	1	55	2539	134.86	135.86	0	0	189
1338232	1	55	2139	339.5	437.53	0	0	54
1448490	1	55	2631	243.5	242.5	0	0	40
1270037	1	55	1248	251.23	252.23	0	0	28
1191377	1	55	1778	183.36	184.36	0	0	190
1499955	1	55	1062	182.36	183.36	0	0	391
1025244	1	55	2707	193.06	192.06	0	0	472
1054194	1	55	1207	325.92	384.18	0	1	676



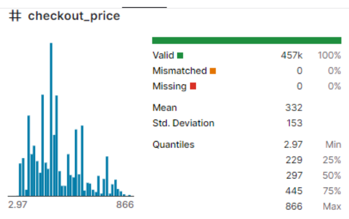
id

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	1.25m	
Std. Deviation	144k	
Quantiles	1000k	Min
	1.12m	25%
	1.25m	50%
	1.38m	75%
	1.50m	Max



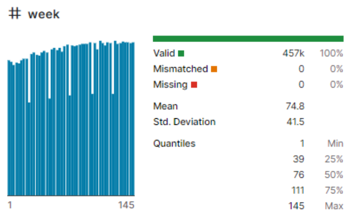
center_id

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	82.1	
Std. Deviation	46	
Quantiles	10	Min
	43	25%
	76	50%
	110	75%
	186	Max



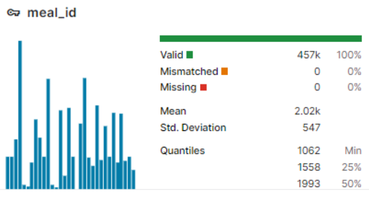
checkout_price

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	332	
Std. Deviation	153	
Quantiles	2.97	Min
	229	25%
	297	50%
	445	75%
	866	Max



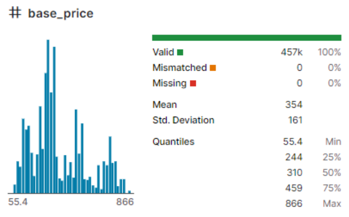
week

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	74.8	
Std. Deviation	41.5	
Quantiles	1	Min
	39	25%
	76	50%
	111	75%
	145	Max



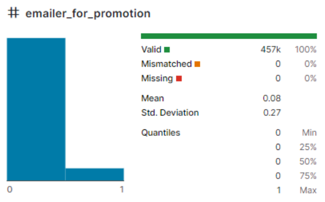
meal_id

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	2.02k	
Std. Deviation	547	
Quantiles	1062	Min
	1558	25%
	1993	50%



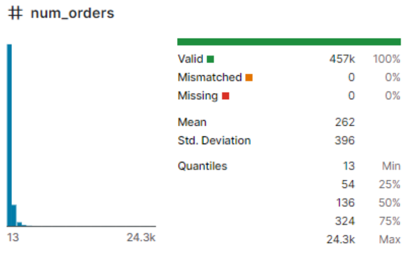
base_price

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	354	
Std. Deviation	161	
Quantiles	55.4	Min
	244	25%
	310	50%
	459	75%
	866	Max



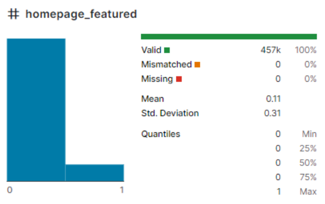
emailer_for_promotion

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	0.08	
Std. Deviation	0.27	
Quantiles	0	Min
	0	25%
	0	50%
	0	75%
	1	Max



num_orders

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	262	
Std. Deviation	396	
Quantiles	13	Min
	54	25%
	136	50%
	324	75%
	24.3k	Max

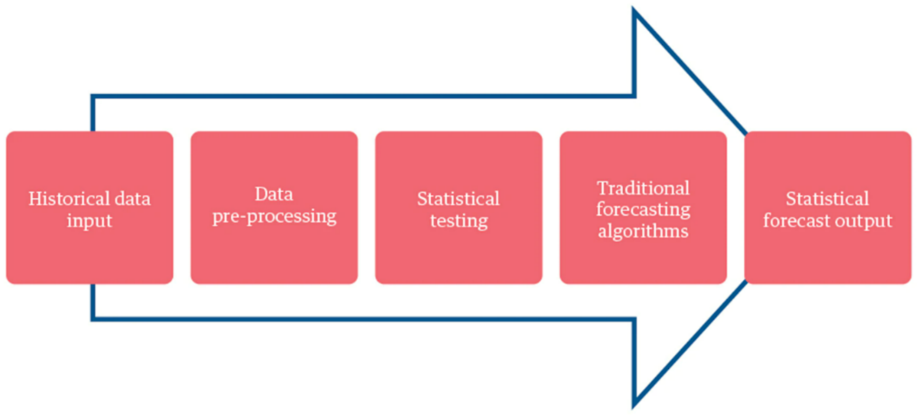


homepage_featured

Valid 🟩	457k	100%
Mismatched 🟧	0	0%
Missing 🟥	0	0%
Mean	0.11	
Std. Deviation	0.31	
Quantiles	0	Min
	0	25%
	0	50%
	0	75%
	1	Max

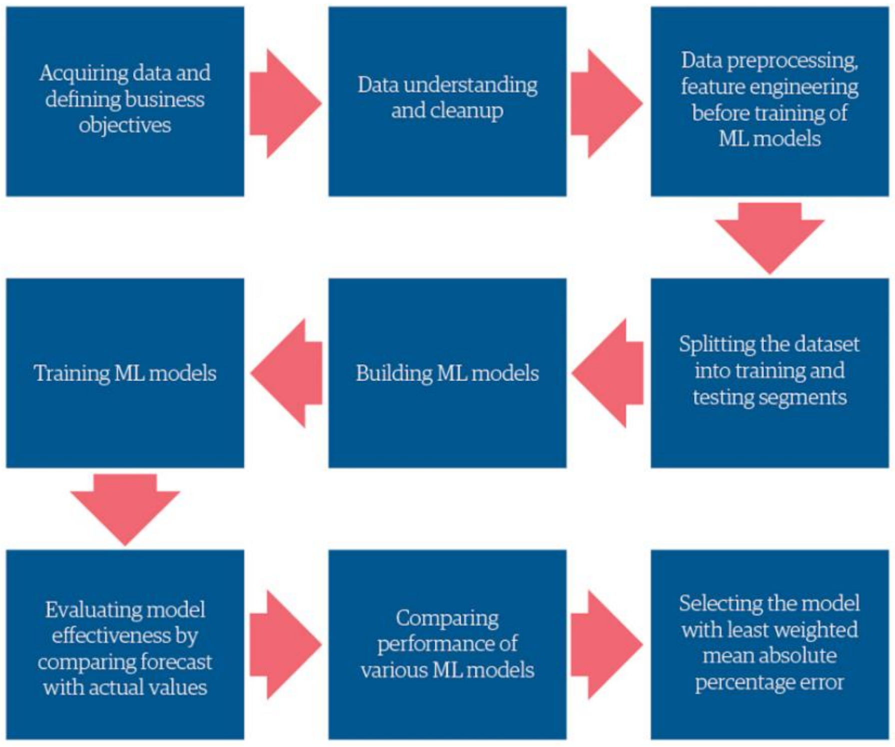
6. Benchmarking:

- In current market scenarios, traditional forecasting methods are used.
- In general, traditional algorithms tend to use predefined techniques and statistical models such as linear regression, autoregressive integrated moving average (ARIMA), and autoregressive integrated moving average with explanatory variable (ARIMAX). The goal of traditional forecasting methods is largely descriptive in nature, aimed at analyzing a univariate dataset or a multivariate dataset with finite, countable, and explainable predictors.
- Some of the following classical models are used effectively when dealing with univariate data with a high degree of accuracy:
    - Moving average
    - Simple exponential smoothing (SES)
    - Holt-Winters
    - Damped exponential smoothing (DES)
    - Average of SES, Holt, and DES
    - Linear regression
    - ARIMA, ARIMAX
    - Unobserved component modeling


Historical data input
Data pre-processing
Statistical testing
Traditional forecasting algorithms
Statistical forecast output

- ML forecasting algorithms often use techniques that involve more complex features and predictive methods, but the objective of ML forecasting methods is the same as that of traditional methods – to improve the accuracy of forecasts while minimizing a loss function. The loss function is usually taken as the sum of squares due to errors in prediction/forecasting.

- Here are some of the existing machines learning methodologies:
    - Linear regression approach
    - Featured Engineering
    - Decision tree
    - Classification and regression trees
    - LSTM neural networks
    - SVR


Acquiring data and defining business objectives
Data understanding and cleanup
Data preprocessing, feature engineering before training of ML models
Splitting the dataset into training and testing segments
Building ML models
Training ML models
Evaluating model effectiveness by comparing forecast with actual values
Comparing performance of various ML models
Selecting the model with least weighted mean absolute percentage error

- In this study I've used XGBoost as my ML algorithm for the problem statement.
- XGBoost is intensively used in business intelligence modeling because of its ability to process scattered data efficiently, resulting in overall excellent model performance. By optimizing CPU memory use with parallel computing, XGBoost becomes the appropriate model for extensive data models, classification, and feature recognition data.
- XGBoost can handle sparse and weighted data. It has an inbuilt capability to handle missing values. It has better accuracy than single decision tree models.
- Better prediction performance as it uses boosting ensemble learning algorithm.

7. Applicable Patents:

- WO2014075108A2 – Forecasting system using machine learning and ensemble methods
- US603215A – Demand forecasting method, demand forecasting system, and recording medium.

8. Applicable Regulations:

The patents mentioned above might claim the technology used if the algorithms
are not developed and optimized individually and for our requirements. Using a
pre-existing model is off the table if it incurs a patent claim.

- Must provide access to the 3rd party website to audit and monitor the authenticity and behavior of the service
- Enabling open-source, academic and research community to audit the algorithms and research on the efficacy of the product.
- Laws controlling data collection: Some websites might have a policy against collecting customer data in form of reviews and ratings.
- Must be responsible with the scraped data: It is quite essential to protect the privacy and intention with which the data was extracted.

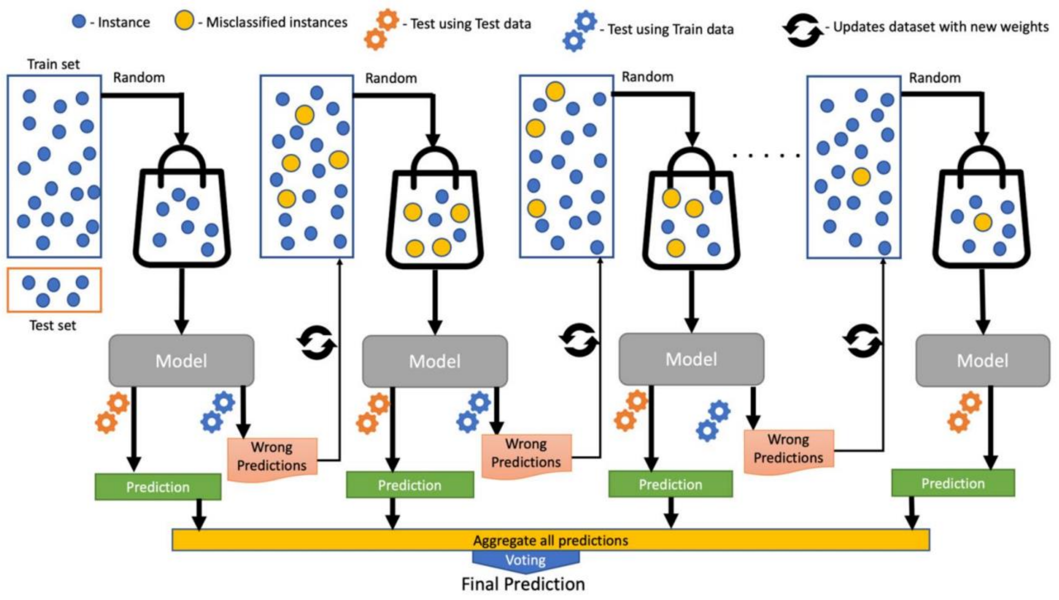
9. Business Opportunity:

- Demand forecasts are used by all retailers, from the largest omnichannel giants to the smallest brick-and-mortar boutiques, to forecast their best estimate of how much they will sell.
- Modern demand forecasting is a sophisticated statistical analysis that optimizes that prediction by considering numerous variables.
- While certain retailers depend on bookkeeping sheets and manual computations, the such powerful measurable examination is best executed by particular programming intended to process gigantic, retail-scale informational collections.
- This method is advantageous because it clarifies to users what data is used to construct forecasts and how estimates are calculated. Machine learning improves predictions in modern demand forecasting software, which automates difficult and time-consuming decisions.

- Forecasting demand can help reduce production lead times, increase operational efficiencies, save money, launch new products, and provide a better overall customer experience, even though it will never be 100% accurate.

10. Concept Generation:

- XGBoost (eXtreme Gradient Boosting), an ensemble learning method that offers a systematic solution to combine the predictive power of multiple learners, is used for better performance and execution speed.
- Regardless of the type of prediction task at hand, XGBoost is one of the machine learning algorithms that enjoys the highest level of popularity today; either classification or regression. An ensemble learning approach is XGBoost. We can see that relying solely on the results of a single machine learning model may not be sufficient, as XGBoost outperforms all others in this comparison. A methodical approach to combining the predictive abilities of multiple learners is offered by ensemble learning.
- In boosting, the base learners are weak learners with a high bias and slightly better predictive power than random guessing. By effectively combining these weak learners, the boosting technique is able to produce a strong learner by incorporating important prediction information from each of these weak learners. The variance and bias are reduced by the final strong learner. This powerful algorithm's beauty lies in its scalability, which enables efficient memory use and accelerates learning through parallel and distributed computing. The method is similar to a neural network.
- The diagram below shows how XGBoost works.

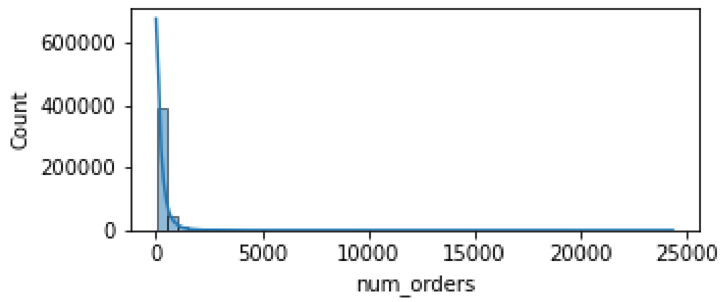


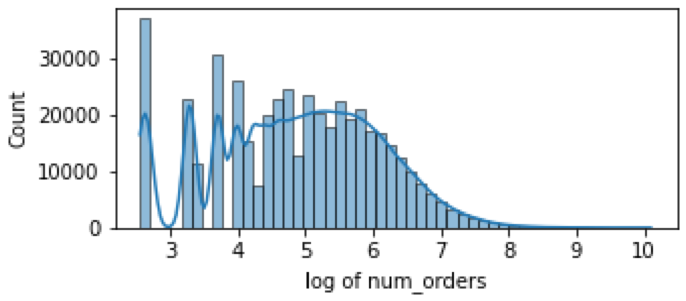
- **Univariate and Bivariate Feature Analysis:**

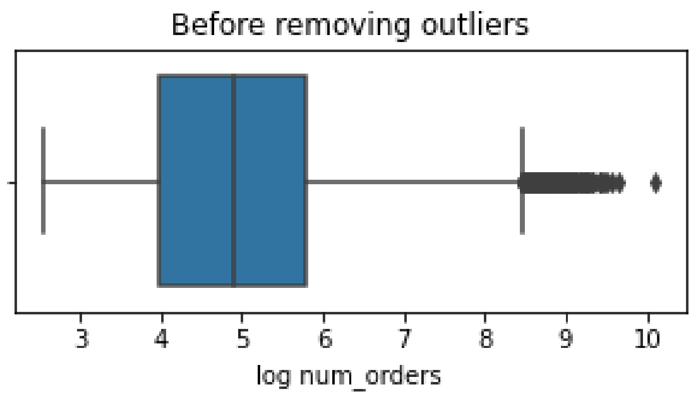
  Histograms, distribution plots, scatter plots, line plots, and boxplots are some of the analyses of the features used to understand insights into the data. These are the most powerful strategies to come up with beautiful insights into the data, especially when it comes to the relation between two variables.

  However, it fails to give complete information on which variable has the most important feature to give the highest accurate prediction.
  I've plotted some of these to know more about the behavior of our data.
  The pattern of number of orders follows exponential distribution and can be seen from the graph below along with its histogram of "log of num_orders" with its kernel density estimation (kde).

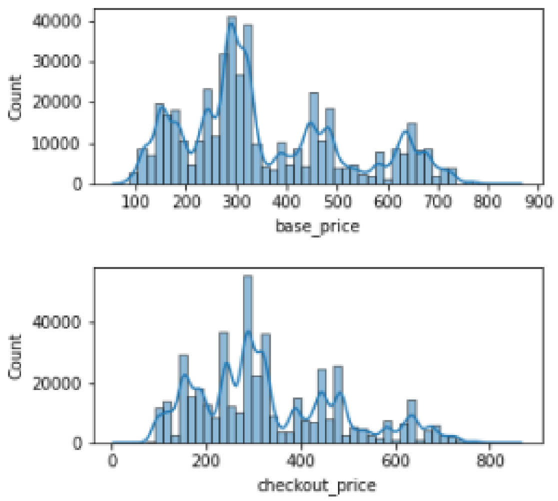
  Log is taken to handle the wide range of data (13 to 24299). It is multimodal as it has more than one peak point. It is right skewed with a value of 6.929966065.
  The higher number of orders is concentrated in between 1 to 6. The plots below depict the difference on the histogram of num_orders with and without log.
  The log of num_orders follows log gamma distribution and from the boxplot of the log of num_orders, it is seen that there are some outliers that needs to be removed.

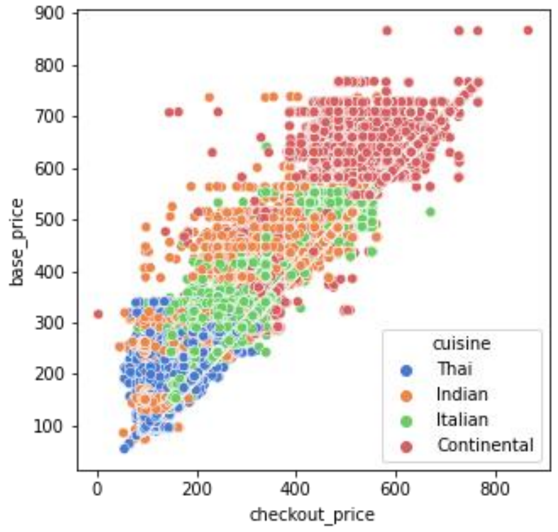




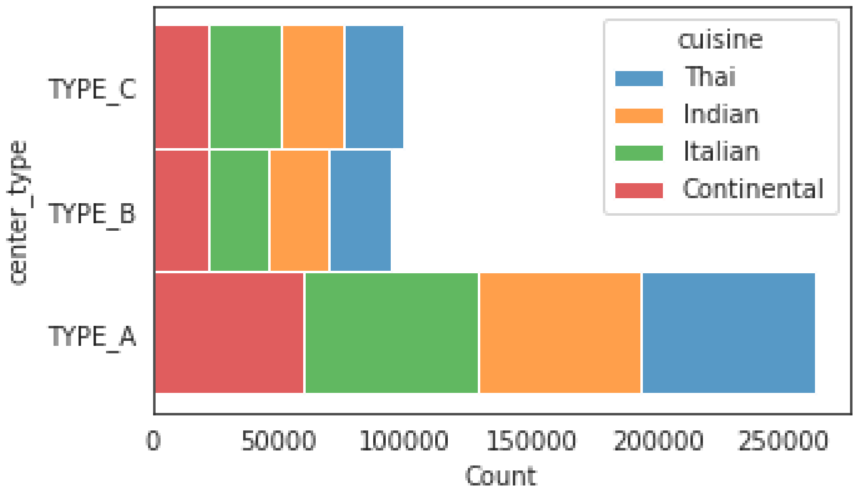


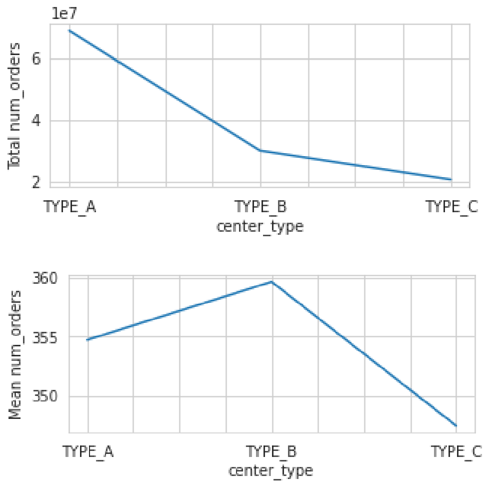
From the histogram of base_price and checkout_price below, it is found that they both are in the range of 100 to 800 with 300 as the highly concentrated range.
As the scatter plot between base_price and checkout_price depicts they have linear relation where majority of the base_price are on the higher side although there are also some points with higher checkout_price than its base_price. This means that there is more number of discounts.





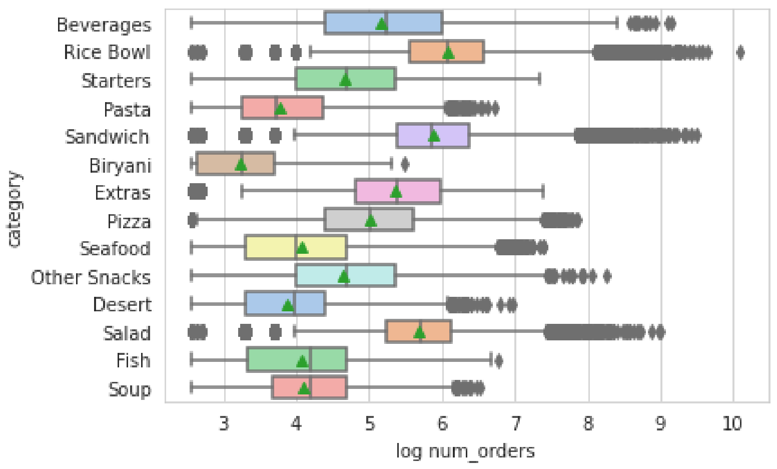
There are 14 meals and 4 cuisines in the given dataset. From the below histogram of variable "category", "Beverages" can be seen in all the "cuisine", consequently having the highest number of orders compare to other meals.



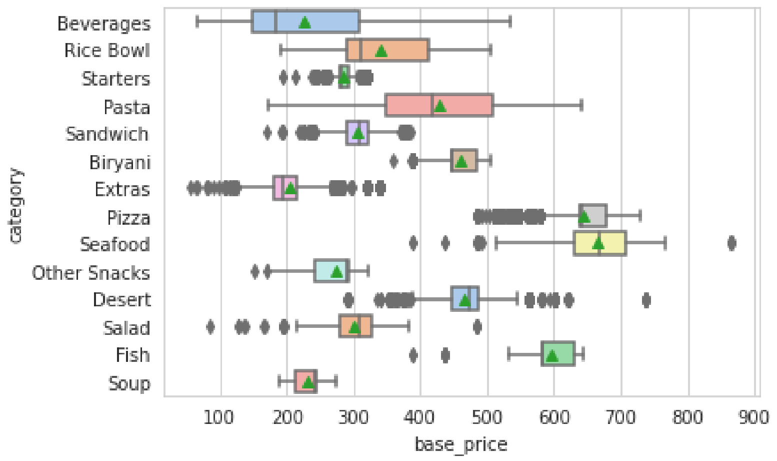


There are 3 center types given in the dataset. From the above histogram of variable "center_type" color encoded with "cuisine", it is seen that there is equal distributions of all cuisines in all types of center.

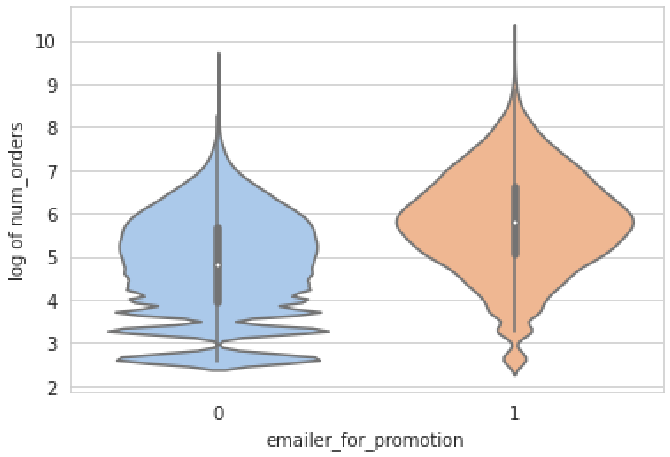
From the below boxplot of variable "category" with the number of orders, it is found that "Rice Bowl", "Sandwich" and "Salad" has higher outliers and mean number of orders compare to other meals, while "Biryani" has the lowest mean order value

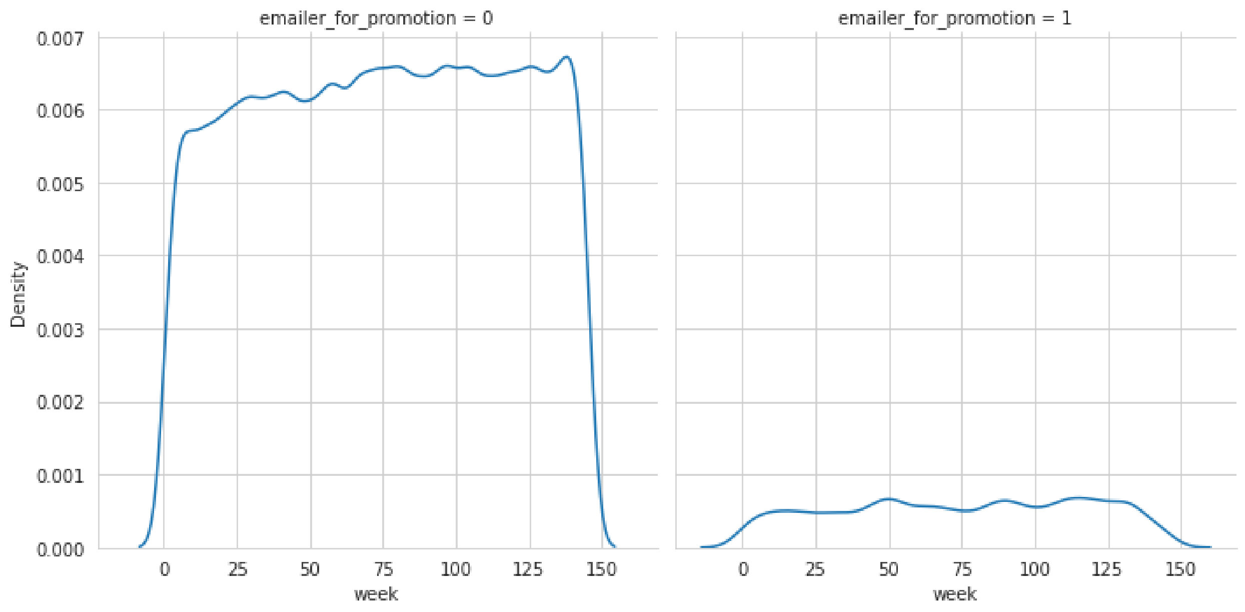


"Extras", "Beverages" and "Soup" has lower base_price while "Pizza", "Sandwich" and "Fish" are on the higher range as shown in the boxplot below



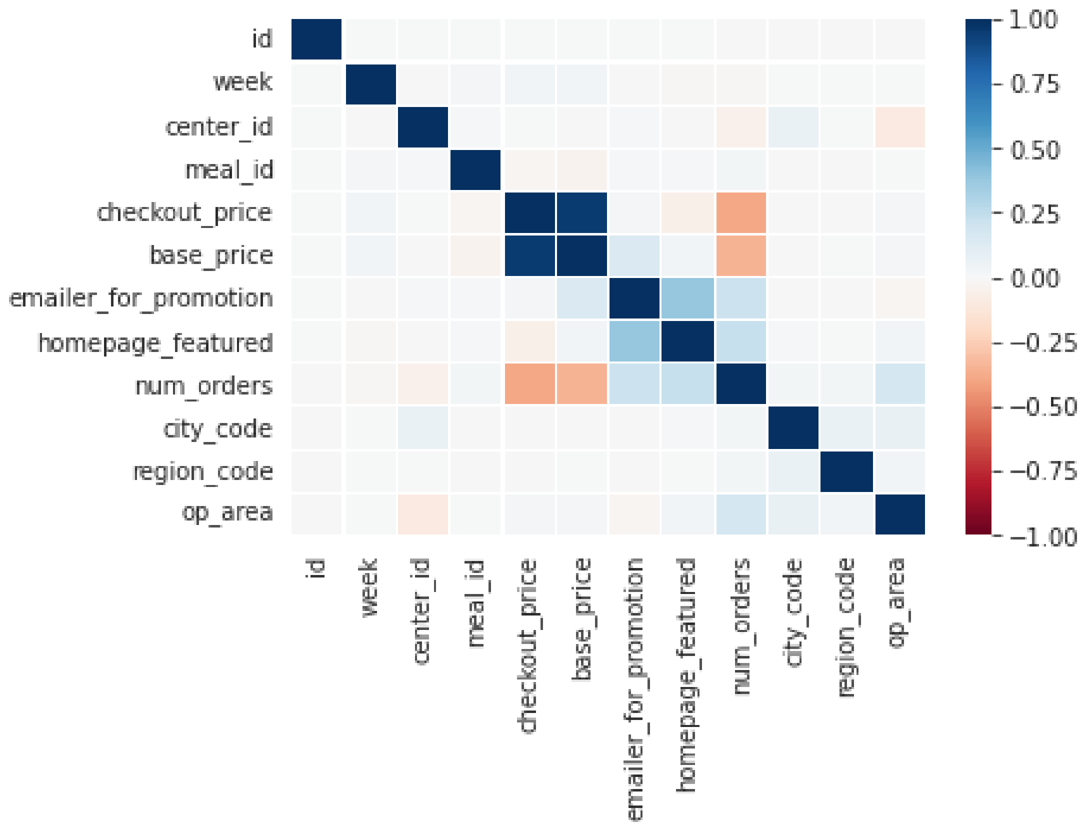
From the distribution and violin plots below, it can be seen that "emailer_for_promotion" with value "1" has a smooth distribution curve compare to the one with value "0", even though it was lasted for very less number of weeks. Likewise, variable "homepage_featured" with value "1" has same effect on the number of orders.



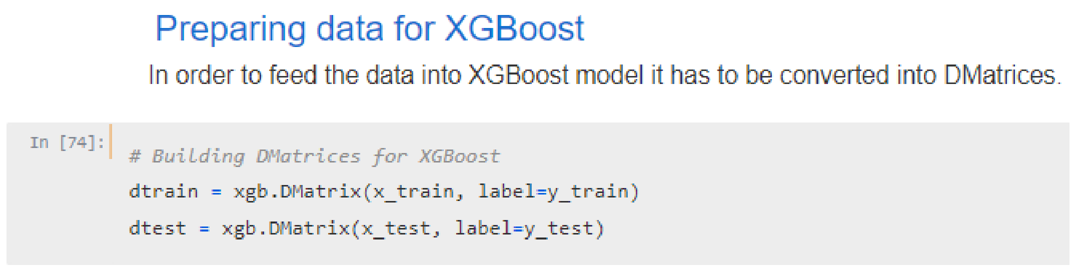


- **Multivariate Feature Analysis:**
  Finding correlation coefficients is one of the most effective multivariate analyses. Pearson correlation is the most common one, which measuresof how numerical variables are correlated to each other. So as Spearman and Kendall coefficients.

  However, the Spearman correlation coefficient is more robust to outliers and handles sparse matrices more effectively. Below is a heatmap plot for the spearman coefficient. It gives values between -1 to 1. 1 being linearly related, -1 being inversely related, and 0 being no relations. Checkout_price and base_price are closely related and inversely associated with the num_orders. Emailer_for_promotion and homepage_featured are also closely related to num_orders as well. Op_area is also somehow associated with num_orders.



11. Concept Deployment:



Preparing data for XGBoost

In order to feed the data into XGBoost model it has to be converted into DMatrices.

```
In [74]:
# Building DMatrices for XGBoost
dtrain = xgb.DMatrix(x_train, label=y_train)
dtest = xgb.DMatrix(x_test, label=y_test)
```

Parameters

After extensive hyperparameter tuning we have come with the parameters that give the lowest RMSE value.

```
In [75]:
# Assigning parameter values
# These parameters are set after extensive hyperparameter tuning
params = {"max_depth":10,"min_child_weight": 7,"eta":0.1,"subsample": 1,"colsample_bytree": 0.7,
          "eval_metric": "rmse","objective":"reg:squarederror"}
params
```

```
{'max_depth': 10,
 'min_child_weight': 7,
 'eta': 0.1,
 'subsample': 1,
 'colsample_bytree': 0.7,
 'eval_metric': 'rmse',
 'objective': 'reg:squarederror'}
```

Training

```
In [76]:
# Noting the start time of execution
s = time.time()
# Training on dtrain and evaluation is done on dtest,
# Iterating 10 more rounds after getting the minimum rmse
model = xgb.train(params,dtrain,num_boost_round=1000,evals=[(dtest, "Test")],early_stopping_rounds=10)
# Printing the time taken to execute the above code
print("Time taken:",round(time.time()-s,3),"sec")
```

```
[90]	Test-rmse:129.09473
[91]	Test-rmse:129.05078
[92]	Test-rmse:129.06374
[93]	Test-rmse:129.05747
[94]	Test-rmse:129.09470
[95]	Test-rmse:129.09460
[96]	Test-rmse:129.05034
[97]	Test-rmse:129.03139
[98]	Test-rmse:128.98842
[99]	Test-rmse:129.01183
[100]	Test-rmse:128.99223
[101]	Test-rmse:128.97459
[102]	Test-rmse:129.09937
[103]	Test-rmse:129.06796
[104]	Test-rmse:129.06293
[105]	Test-rmse:129.23036
[106]	Test-rmse:129.19930
[107]	Test-rmse:129.15303
[108]	Test-rmse:129.15530
[109]	Test-rmse:129.14887
[110]	Test-rmse:129.12695
[111]	Test-rmse:129.20375
Time taken: 14.772 sec
```

We now get the lowest RMSE value with the number of rounds needed.

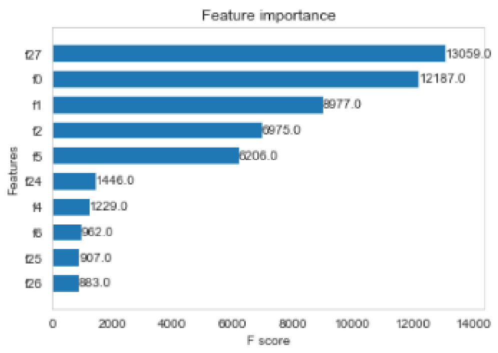
```
In [77]:
# Printing the minimun RMSE with its iteration
print("Best RMSE: {:.2f} in {} rounds".format(model.best_score, model.best_iteration+1))
```

```
Best RMSE: 128.97 in 102 rounds
```

Feature importance

From the 33 features we have trained the model, we want to know which feature contributes the highest weight in predicting the number of orders for a meal. This can be done with the following codes

In [79]:
```
# Plotting feature importance plot
# Displaying only first top 10 features
xgb.plot_importance(best_model, height=0.7, grid=False, max_num_features=10)
plt.show()
```



Prediction

After training the data, we now measure the error and accuracy by predicting train and test data. It is needed to compare train and test error to check whether the model is overfitting or not. As the error difference is not large, we see that the model is performing well with an accurary od 80.4% on unseen data.

In [80]:
```
# Calculating rmse and r2 value for train and test data
RMSE_train = round(np.sqrt(mean_squared_error(y_train,best_model.predict(dtrain))),3)
R2_train = round(r2_score(y_train,best_model.predict(dtrain)),3)

RMSE_test = round(np.sqrt(mean_squared_error(y_test,best_model.predict(dtest))),3)
R2_test = round(r2_score(y_test,best_model.predict(dtest)),3)

print("         Train         Test")
print("________________________________")
print(" RMSE:  {}      {}".format(RMSE_train, RMSE_test))
print("   R2:   {}        {}".format(R2_train, R2_test))
```

```
          Train         Test
________________________________
 RMSE:  87.542      128.975
   R2:   0.904        0.803
```

Test data

It is now time to feed the actual test.csv data into the model and predict them.

Loading data

We are now reading the test.csv file, merge meal and center file with test.csv. Pre-processed, feature engineered, dropped columns and convert into DMtarix like the way we did for train data. Then predict them with the model.

In [45]:
```python
# Reading test.csv file
test_data = pd.read_csv("food_demand_test.csv")
# Merging meal and center file to test_data
test_data = test_data.merge(meal, on="meal_id")
test_data = test_data.merge(center, on="center_id")
# Sorting test data by week and category
test_data = test_data.sort_values(by=["week","category"])
# Perfoming one hot encoding
test_data = pd.get_dummies(test_data)
# Calculating discount percentage and forming a column
test_data["discount_percentage"] = (test_data["base_price"]-test_data["checkout_price"])*100/test_data["bas
# Dropping some of the unnecessary columns
test_data.drop(["id","checkout_price","base_price","city_code","region_code"],axis=1,inplace=True)
test_data
```

In [46]:
```python
s = time.time()
# Converting into xgboost dmatrix
test_matrix = xgb.DMatrix(test_data.values)# Predicting with our best model
predicted_value = best_model.predict(test_matrix)
# Calculating time taken to to predict test data
t = round(time.time()-s,6)
print("Time taken to predict test data: {} sec".format(t))
```

```
Time taken to predict test data: 0.054834 sec
```

Sample submission

The predicted values are then saved in the sample submission as csv file.

In [48]:
```python
# Inserting the predicted output to sample submission file
sample_submission = pd.read_csv("sample_submission.csv")
sample_submission["num_orders"] = predicted_value
sample_submission
```

	id	num_orders
0	1028232	207.194901
1	1127204	204.598083
2	1212707	128.009064
3	1082698	62.903759
4	1400926	39.255215
...	...	...
32568	1250239	204.236954
32569	1039516	126.166000
32570	1158107	121.962502
32571	1444235	242.046097
32572	1291286	194.152878

32573 rows × 2 columns

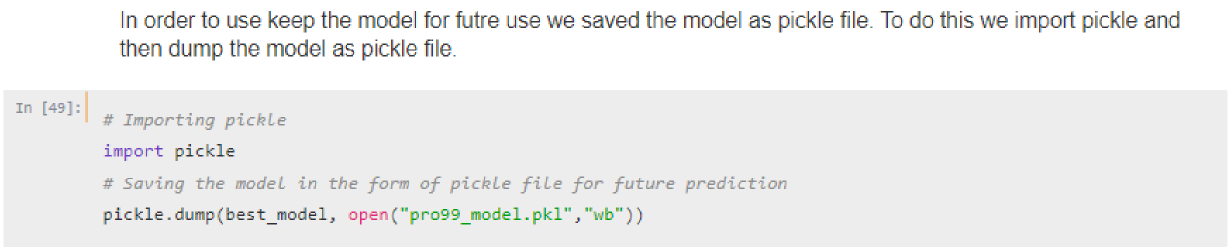
12. Final Report Prototype:

The product takes the following functions to perfect and provide a good result.

Back-end

Model Development: This must be done before releasing the service. A lot of manual supervised machine learning must be performed to optimize the automated tasks.

- An application of the system developed with is built with Streamlit and is deployed on Amazon Web Service (AWS).
- To deploy a system on this platform one has to sign up with AWS and create an EC2 instance, then connect with local host through "ssh" commands.
- After successfully connecting with the remote host on AWS, data from the local host can be transferred to remote host by various methods like "scp" commands, PuTTy or WinSCP.
- Python notebook of the system is converted into pickle file, then transfer to the remote host along with other required files of the system. Pickle converts a python object into a byte stream to store it in a file or database, maintain program state across sessions, or transport data over the network. Below codes shows how a Python object named "best_model" is dumped or saved as "pro99_model.pkl" (in this case) file:



In order to use keep the model for futre use we saved the model as pickle file. To do this we import pickle and then dump the model as pickle file.

```
In [49]:
# Importing pickle
import pickle
# Saving the model in the form of pickle file for future prediction
pickle.dump(best_model, open("pro99_model.pkl","wb"))
```

13. Conclusion:

Demand prediction plays a crucial role in planning operations for restaurant's management. Dataset loaded from Kaggle is analyzed, understood and fitted into various machine learning algorithms, and discussed advantages and disadvantages for each one of them. It is almost universally agreed in the forecasting literature that no single method is best in every situation; however we extensively fine tuned XGBoost method to yield possible minimum errors to give best predictions. Throughout this whole process we understood that the meal demand highly depends on the discount given on the meal, type of the meal and the location of the store; with promotions on web pages and emails make enhancement on it.

14. References:

- https://www.xpand-it.com/blog/data-science-demand-forecasting-predict-sales/
- Applications of Machine Learning Techniques in Supply Chain Management. ISSN: 2320-2882, Adit Kudtarkar, Danish Shaikh.
- https://www.relexsolutions.com/resources/machine-learning-in-retail-demand-forecasting/
- https://360digitmg.com/blog/applications-of-data-science-in-supply-chain-analytics
- https://dzone.com/articles/xgboost-a-deep-dive-into-boosting
- https://www.genpact.com/insight/the-evolution-of-forecasting-techniques-traditional-versus-machine-learning-methods
- https://xgboost.readthedocs.io/en/stable/python/python_api.html