

TEAM ID: PNT2022TMID14136

## PROJECT NAME: DemandEst - AI powered Food Demand Forecaster

### Team Leader

The screenshot displays a Jupyter Notebook environment with two visible code cells. The first cell, labeled 'In [115]:', contains Python code to import LabelEncoder from sklearn.preprocessing and apply it to three categorical columns: 'center\_type', 'category', and 'cuisine'. The second cell, labeled 'In [116]:', shows the output of trainfinal.head(), displaying the first two rows of the dataset. The output table includes columns for id, week, city\_code, region\_code, center\_type, op\_area, category, cuisine, checkout\_price, base\_price, emailer\_for\_promotion, homepage\_featured, and nur.

**Label Encoding**

Typically, any structured dataset includes multiple columns with combination of numerical as well as categorical variables. A machine can only understand the numbers. It cannot understand the text. That's essentially the case with Machine Learning algorithms too.

We need to convert each text category to numbers in order for the machine to process those using mathematical equations. Label Encoding is a popular encoding technique for handling categorical variables implemented using the scikit-learn library in python. In this technique, each label is assigned a unique integer based on alphabetical ordering.

```
In [115]: from sklearn.preprocessing import LabelEncoder

lb1 = LabelEncoder()
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
lb2 = LabelEncoder()
trainfinal['category'] = lb2.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb3.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

```
Out[116]:
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	

The second screenshot shows the continuation of the notebook. It displays the same code as the first cell, followed by the output of trainfinal.head(), which now shows the first five rows of the dataset. Below the table, a text note states: "After performing label encoding, alphabetical classes- 'Center type, Category and City code are converted to numeric values." followed by the instruction "Finally display number of rows and columns of trainfinal using shape()". The final code cell, 'In [117]:', shows the output of trainfinal.shape, which is (456548, 13).

```
In [116]: trainfinal.head()
```

```
Out[116]:
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	
2	1196273	3	647	56	2	2.0	0	3	132.92	133.92	0	0	
3	1116527	4	647	56	2	2.0	0	3	135.86	134.86	0	0	
4	1343872	5	647	56	2	2.0	0	3	146.50	147.50	0	0	

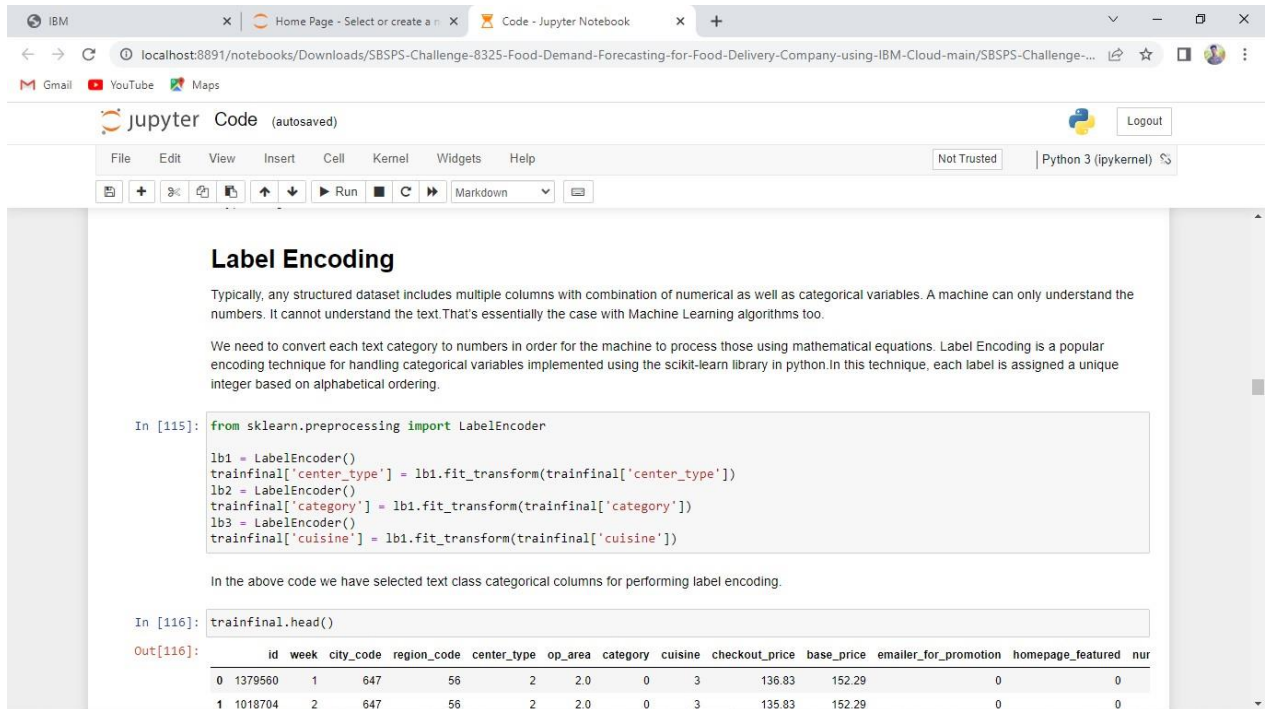
After performing label encoding, alphabetical classes- "Center type, Category and City code are converted to numeric values.

Finally display number of rows and columns of trainfinal using shape()

```
In [117]: trainfinal.shape
```

```
Out[117]: (456548, 13)
```

# Team Member 1



**Label Encoding**

Typically, any structured dataset includes multiple columns with combination of numerical as well as categorical variables. A machine can only understand the numbers. It cannot understand the text. That's essentially the case with Machine Learning algorithms too.

We need to convert each text category to numbers in order for the machine to process those using mathematical equations. Label Encoding is a popular encoding technique for handling categorical variables implemented using the scikit-learn library in python. In this technique, each label is assigned a unique integer based on alphabetical ordering.

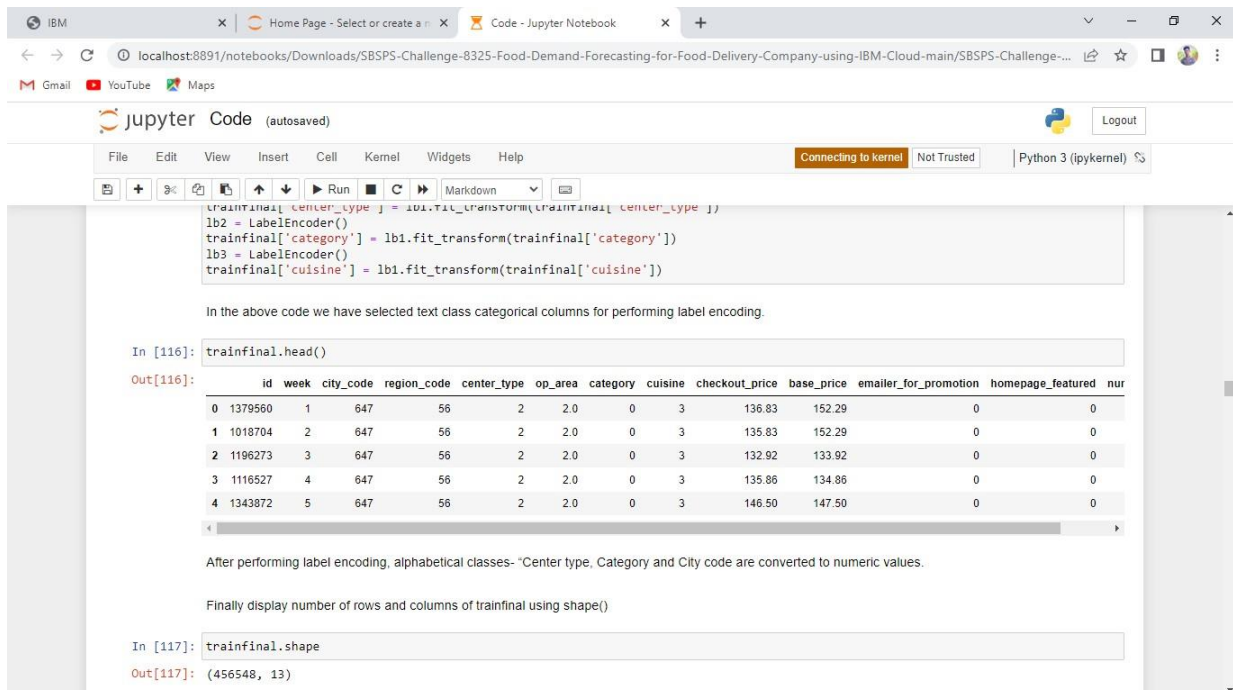
```
In [115]: from sklearn.preprocessing import LabelEncoder

lb1 = LabelEncoder()
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
lb2 = LabelEncoder()
trainfinal['category'] = lb2.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb3.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	



```
(trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type']))
lb2 = LabelEncoder()
trainfinal['category'] = lb2.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb3.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	
2	1196273	3	647	56	2	2.0	0	3	132.92	133.92	0	0	
3	1116527	4	647	56	2	2.0	0	3	135.86	134.86	0	0	
4	1343872	5	647	56	2	2.0	0	3	146.50	147.50	0	0	

After performing label encoding, alphabetical classes- "Center type, Category and City code are converted to numeric values.

Finally display number of rows and columns of trainfinal using shape()

```
In [117]: trainfinal.shape
```

Out[117]: (456548, 13)

## Team Member 2

**Label Encoding**

Typically, any structured dataset includes multiple columns with combination of numerical as well as categorical variables. A machine can only understand the numbers. It cannot understand the text. That's essentially the case with Machine Learning algorithms too.

We need to convert each text category to numbers in order for the machine to process those using mathematical equations. Label Encoding is a popular encoding technique for handling categorical variables implemented using the scikit-learn library in python. In this technique, each label is assigned a unique integer based on alphabetical ordering.

```
In [115]: from sklearn.preprocessing import LabelEncoder

lb1 = LabelEncoder()
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
lb2 = LabelEncoder()
trainfinal['category'] = lb1.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb1.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

```
Out[116]:
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	

```
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
lb2 = LabelEncoder()
trainfinal['category'] = lb1.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb1.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

```
Out[116]:
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	
2	1196273	3	647	56	2	2.0	0	3	132.92	133.92	0	0	
3	1116527	4	647	56	2	2.0	0	3	135.86	134.86	0	0	
4	1343872	5	647	56	2	2.0	0	3	146.50	147.50	0	0	

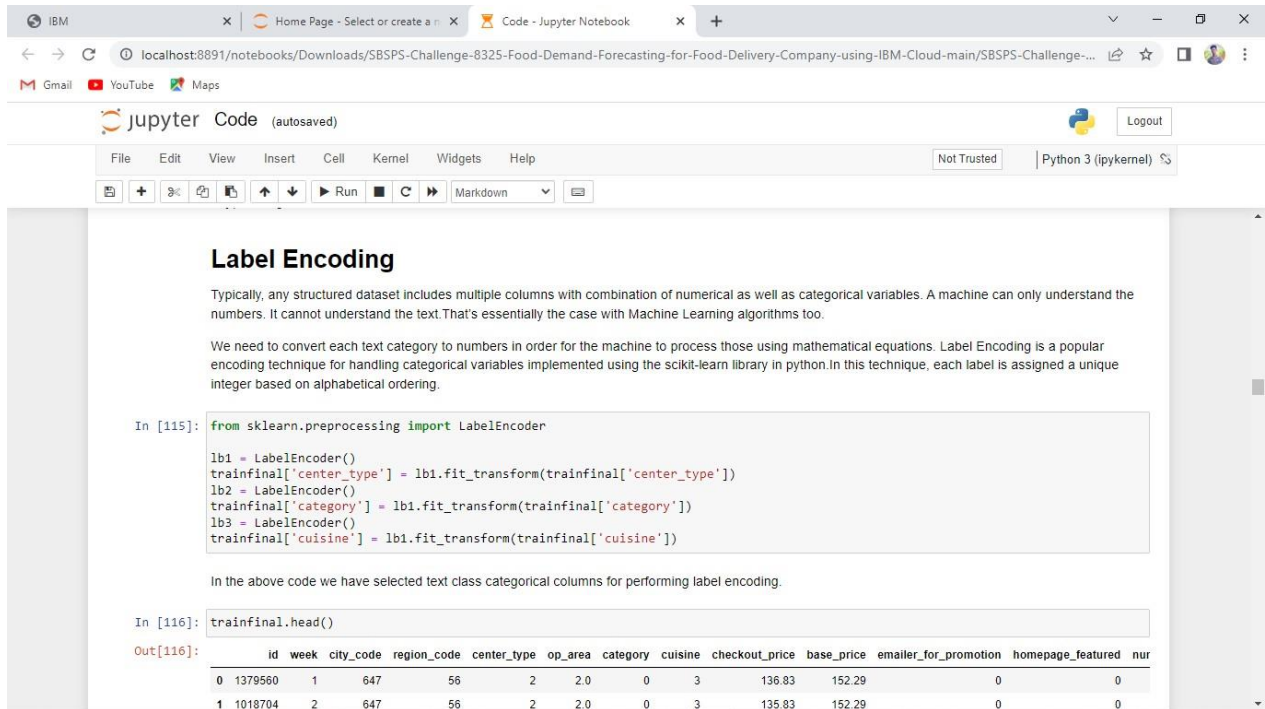
After performing label encoding, alphabetical classes- 'Center type, Category and City code are converted to numeric values.

Finally display number of rows and columns of trainfinal using shape()

```
In [117]: trainfinal.shape
```

```
Out[117]: (456548, 13)
```

## Team Member 3



The screenshot shows a Jupyter Notebook interface with the following content:

### Label Encoding

Typically, any structured dataset includes multiple columns with combination of numerical as well as categorical variables. A machine can only understand the numbers. It cannot understand the text. That's essentially the case with Machine Learning algorithms too.

We need to convert each text category to numbers in order for the machine to process those using mathematical equations. Label Encoding is a popular encoding technique for handling categorical variables implemented using the scikit-learn library in python. In this technique, each label is assigned a unique integer based on alphabetical ordering.

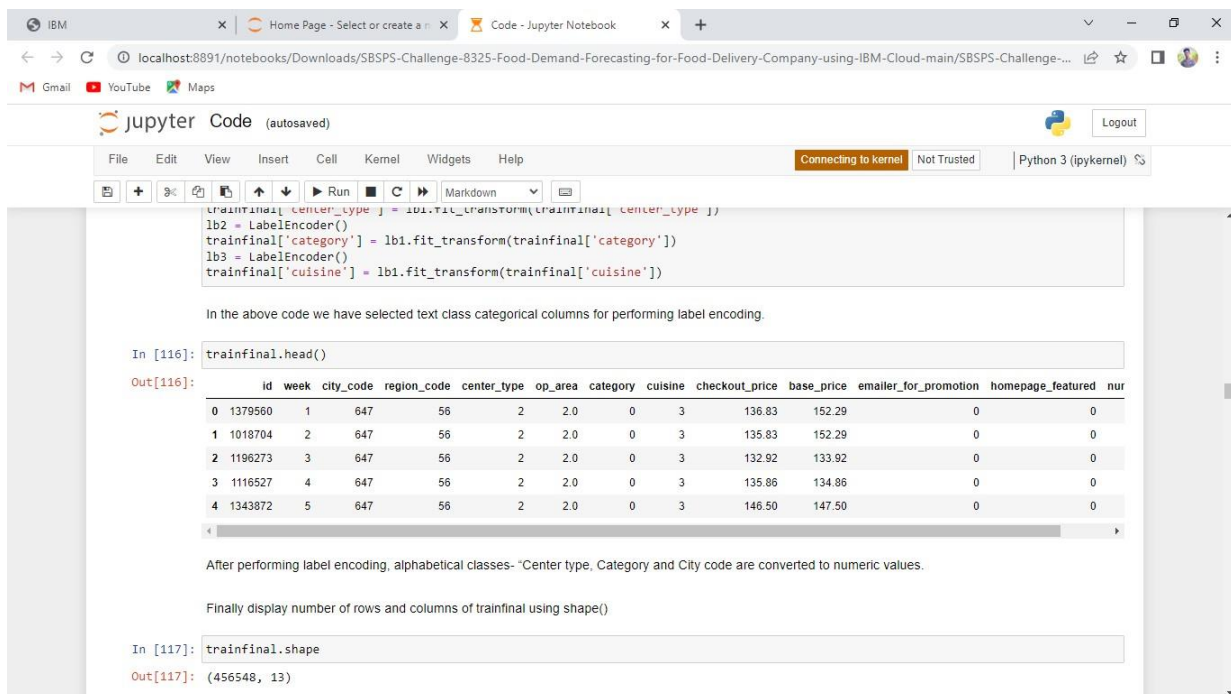
```
In [115]: from sklearn.preprocessing import LabelEncoder

lb1 = LabelEncoder()
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
lb2 = LabelEncoder()
trainfinal['category'] = lb1.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb1.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	



The screenshot shows the continuation of the Jupyter Notebook with the following content:

```
trainfinal['center_type'] = lb1.fit_transform(trainfinal['center_type'])
lb2 = LabelEncoder()
trainfinal['category'] = lb1.fit_transform(trainfinal['category'])
lb3 = LabelEncoder()
trainfinal['cuisine'] = lb1.fit_transform(trainfinal['cuisine'])
```

In the above code we have selected text class categorical columns for performing label encoding.

```
In [116]: trainfinal.head()
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	nur
0	1379560	1	647	56	2	2.0	0	3	136.83	152.29	0	0	
1	1018704	2	647	56	2	2.0	0	3	135.83	152.29	0	0	
2	1196273	3	647	56	2	2.0	0	3	132.92	133.92	0	0	
3	1116527	4	647	56	2	2.0	0	3	135.86	134.86	0	0	
4	1343872	5	647	56	2	2.0	0	3	146.50	147.50	0	0	

After performing label encoding, alphabetical classes- "Center type, Category and City code are converted to numeric values.

Finally display number of rows and columns of trainfinal using shape()

```
In [117]: trainfinal.shape
```

```
Out[117]: (456548, 13)
```