

TEAM ID: PNT2022TMID17645

## 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, titled "Dropping Columns", contains Python code to drop 'center\_id' and 'meal\_id' from a DataFrame named 'trainfinal'. The output shows the first five rows of the resulting DataFrame with columns: id, week, checkout\_price, base\_price, emailer\_for\_promotion, homepage\_featured, num\_orders, category, cuisine, city\_code, region\_code, center\_type. The second cell contains code to list the columns and rearrange them. The output shows the rearranged DataFrame with columns: id, week, city\_code, region\_code, center\_type, op\_area, category, cuisine, checkout\_price, base\_price, emailer\_for\_promotion, homepage\_featured, num\_orders. The third cell shows the dtypes of the DataFrame, indicating that 'id' and 'week' are int64 types.

**Dropping Columns**

Let's drop columns "center\_id" and "meal\_id" as they are not required for the further process. Display the changes of trainfinal table using head().

```
In [110]: trainfinal = trainfinal.drop(['center_id', 'meal_id'], axis=1)
trainfinal.head()
```

Out[110]:

	id	week	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type
0	1379560	1	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C
1	1018704	2	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C
2	1196273	3	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C
3	1116527	4	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C
4	1343872	5	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C

Display the list of columns present in trainfinal table and store it in variable "cols"

```
In [111]: cols = trainfinal.columns.tolist()
print(cols)
```

['id', 'week', 'checkout\_price', 'base\_price', 'emailer\_for\_promotion', 'homepage\_featured', 'num\_orders', 'category', 'cuisine', 'city\_code', 'region\_code', 'center\_type', 'op\_area']

Rearrange the columns by slicing the columns of "cols" and print "cols"

```
In [112]: cols = cols[:2] + cols[9:] + cols[7:9] + cols[2:7]
print(cols)
```

['id', 'week', 'city\_code', 'region\_code', 'center\_type', 'op\_area', 'category', 'cuisine', 'checkout\_price', 'base\_price', 'emailer\_for\_promotion', 'homepage\_featured', 'num\_orders']

Store the changes of columns in trainfinal and display the datatypes of trainfinal using trainfinal.dtypes. Here, we can see that, we not only have numerical data but we also have object data.

```
In [113]: trainfinal = trainfinal[cols]
trainfinal.head()
```

Out[113]:

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

```
In [114]: trainfinal.dtypes
```

Out[114]:

	id	week
	int64	int64

The screenshot shows a Jupyter Notebook with the following content:

```

In [113]: trainfinal = trainfinal[trainfinal['center_id'] < 1000]
trainfinal.head()

Out[113]:
```

	id	week	city_code	region_code	center_type	op_area	category	cuisine	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders
0	1379560	1	647	56	TYPE_C	2.0	Beverages	Thai	136.83	152.29	0	0	177
1	1018704	2	647	56	TYPE_C	2.0	Beverages	Thai	135.83	152.29	0	0	323
2	1196273	3	647	56	TYPE_C	2.0	Beverages	Thai	132.92	133.92	0	0	96
3	1116527	4	647	56	TYPE_C	2.0	Beverages	Thai	135.86	134.86	0	0	163
4	1343872	5	647	56	TYPE_C	2.0	Beverages	Thai	146.50	147.50	0	0	215

```

In [114]: trainfinal.dtypes

Out[114]:
```

id	int64
week	int64
city_code	int64
region_code	int64
center_type	object
op_area	float64
category	object
cuisine	object
checkout_price	float64
base_price	float64
emailer_for_promotion	int64
homepage_featured	int64
num_orders	int64
dtype:	object

## Team Member 1

The screenshot shows a Jupyter Notebook with the following content:

### Dropping Columns

Let's drop columns "center\_id" and "meal\_id" as they are not required for the further process. Display the changes of trainfinal table using head().

```

In [110]: trainfinal = trainfinal.drop(['center_id', 'meal_id'], axis=1)
trainfinal.head()

Out[110]:
```

	id	week	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type
0	1379560	1	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C
1	1018704	2	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C
2	1196273	3	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C
3	1116527	4	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C
4	1343872	5	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C

Display the list of columns present in trainfinal table and store it in variable "cols"

```

In [111]: cols = trainfinal.columns.tolist()
print(cols)

['id', 'week', 'checkout_price', 'base_price', 'emailer_for_promotion', 'homepage_featured', 'num_orders', 'category', 'cuisine', 'city_code', 'region_code', 'center_type', 'op_area']

Rearrange the columns by slicing the columns of "cols" and print "cols"

```

IBM Home Page - Select or create a n Code - Jupyter Notebook

localhost:8891/notebooks/Downloads/SBSPS-Challenge-8325-Food-Demand-Forecasting-for-Food-Delivery-Company-using-IBM-Cloud-main/SBSPS-Challenge-...

Gmail YouTube Maps

jupyter Code (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Connected Not Trusted Python 3 (pykernel)

Rearrange the columns by slicing the columns of 'cols' and print 'cols'

```
In [112]: cols = cols[:2] + cols[9:] + cols[7:9] + cols[2:7]
print(cols)

['id', 'week', 'city_code', 'region_code', 'center_type', 'op_area', 'category', 'cuisine', 'checkout_price', 'base_price', 'em
ailer_for_promotion', 'homepage_featured', 'num_orders']
```

Store the changes of columns in trainfinal and display the datatypes of trainfinal using trainfinal.dtypes. Here, we can see that, we not only have numerical data but we also have object data.

```
In [113]: trainfinal = trainfinal[cols]
trainfinal.head()
```

```
Out[113]:
```

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

```
In [114]: trainfinal.dtypes
```

```
Out[114]: id                int64
week                int64
```

IBM Home Page - Select or create a n Code - Jupyter Notebook

localhost:8891/notebooks/Downloads/SBSPS-Challenge-8325-Food-Demand-Forecasting-for-Food-Delivery-Company-using-IBM-Cloud-main/SBSPS-Challenge-...

Gmail YouTube Maps

jupyter Code (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Connected Not Trusted Python 3 (pykernel)

```
In [113]: trainfinal = trainfinal[cols]
trainfinal.head()
```

```
Out[113]:
```

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

```
In [114]: trainfinal.dtypes
```

```
Out[114]: id                int64
week                int64
city_code           int64
region_code         int64
center_type         object
op_area             float64
category            object
cuisine             object
checkout_price       float64
base_price          float64
emailer_for_promotion int64
homepage_featured    int64
num_orders          int64
dtype: object
```

## Team Member 2

IBM

Home Page - Select or create a notebook

Code - Jupyter Notebook

localhost:8891/notebooks/Downloads/SBSPS-Challenge-8325-Food-Demand-Forecasting-for-Food-Delivery-Company-using-IBM-Cloud-main/SBSPS-Challenge-...

Gmail

YouTube

Maps

Jupyter Code (autosaved)

Python 3 (ipykernel)

File

Edit

View

Insert

Cell

Kernel

Widgets

Help

Not Connected

Not Trusted

Run

Code

## Dropping Columns

Let's drop columns "center\_id" and "meal\_id" as they are not required for the further process. Display the changes of trainfinal table using head().

```
In [110]: trainfinal = trainfinal.drop(['center_id', 'meal_id'], axis=1)
trainfinal.head()
```

```
Out[110]:
```

	id	week	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type
0	1379560	1	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C
1	1018704	2	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C
2	1196273	3	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C
3	1116527	4	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C
4	1343872	5	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C

Display the list of columns present in trainfinal table and store it in variable "cols"

```
In [111]: cols = trainfinal.columns.tolist()
print(cols)
```

```
['id', 'week', 'checkout_price', 'base_price', 'emailer_for_promotion', 'homepage_featured', 'num_orders', 'category', 'cuisine', 'city_code', 'region_code', 'center_type', 'op_area']
```

Rearrange the columns by slicing the columns of "cols" and print "cols"

Home Page - create a... Code - Jupyter Notebook +

localhost:8891/notebooks/Downloads/SBSPS-Challenge-8325-Food-Demand-Forecasting-for-Food-Delivery-Company-using-IBM-Cloud-main/SBSPS-Challenge-...

Gmail YouTube Maps

Jupyter Code (autosaved) Python 3 (ipykernel) Logout

File Edit View Insert Cell Kernel Widgets Help

Not Connected Not Trusted Python 3 (ipykernel)

+

Rearrange the columns by slicing the columns of `cols` and print `cols`

```
In [112]: cols = cols[:2] + cols[9:] + cols[7:9] + cols[2:7]
          print(cols)
```

```
['id', 'week', 'city_code', 'region_code', 'center_type', 'op_area', 'category', 'cuisine', 'checkout_price', 'base_price', 'emailer_for_promotion', 'homepage_featured', 'num_orders']
```

Store the changes of columns in trainfinal and display the datatypes of trainfinal using trainfinal.dtypes. Here, we can see that, we not only have numerical data but we also have object data.

```
In [113]: trainfinal = trainfinal[cols]
          trainfinal.head()
```

```
Out[113]:
```

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

```
In [114]: trainfinal.dtypes
```

```
Out[114]: id      int64
          week    int64
```

The screenshot shows a Jupyter Notebook with the following content:

```

In [113]: trainfinal = trainfinal[trainfinal['center_id'] < 1000]
trainfinal.head()

Out[113]:
```

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

```

In [114]: trainfinal.dtypes

Out[114]:
```

id	int64
week	int64
city_code	int64
region_code	int64
center_type	object
op_area	float64
category	object
cuisine	object
checkout_price	float64
base_price	float64
emailer_for_promotion	int64
homepage_featured	int64
num_orders	int64
dtype:	object

## Team Member 3

The screenshot shows a Jupyter Notebook with the following content:

### Dropping Columns

Let's drop columns "center\_id" and "meal\_id" as they are not required for the further process. Display the changes of trainfinal table using head().

```

In [110]: trainfinal = trainfinal.drop(['center_id', 'meal_id'], axis=1)
trainfinal.head()

Out[110]:
```

	id	week	checkout_price	base_price	emailer_for_promotion	homepage_featured	num_orders	category	cuisine	city_code	region_code	center_type
0	1379560	1	136.83	152.29	0	0	177	Beverages	Thai	647	56	TYPE_C
1	1018704	2	135.83	152.29	0	0	323	Beverages	Thai	647	56	TYPE_C
2	1196273	3	132.92	133.92	0	0	96	Beverages	Thai	647	56	TYPE_C
3	1116527	4	135.86	134.86	0	0	163	Beverages	Thai	647	56	TYPE_C
4	1343872	5	146.50	147.50	0	0	215	Beverages	Thai	647	56	TYPE_C

Display the list of columns present in trainfinal table and store it in variable "cols"

```

In [111]: cols = trainfinal.columns.tolist()
print(cols)

['id', 'week', 'checkout_price', 'base_price', 'emailer_for_promotion', 'homepage_featured', 'num_orders', 'category', 'cuisine', 'city_code', 'region_code', 'center_type', 'op_area']

Rearrange the columns by slicing the columns of "cols" and print "cols"
```

IBM Home Page - Select or create a notebook Code - Jupyter Notebook

localhost:8891/notebooks/Downloads/SBSPS-Challenge-8325-Food-Demand-Forecasting-for-Food-Delivery-Company-using-IBM-Cloud-main/SBSPS-Challenge-...

Gmail YouTube Maps

jupyter Code (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Connected Not Trusted Python 3 (ipykernel)

Run Code

Rearrange the columns by slicing the columns of "cols" and print "cols"

```
In [112]: cols = cols[:2] + cols[9:] + cols[7:9] + cols[2:7]
print(cols)

['id', 'week', 'city_code', 'region_code', 'center_type', 'op_area', 'category', 'cuisine', 'checkout_price', 'base_price', 'em
ailer_for_promotion', 'homepage_featured', 'num_orders']
```

Store the changes of columns in trainfinal and display the datatypes of trainfinal using trainfinal.dtypes. Here, we can see that, we not only have numerical data but we also have object data.

```
In [113]: trainfinal = trainfinal[cols]
trainfinal.head()
```

```
Out[113]:
```

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

```
In [114]: trainfinal.dtypes
```

```
Out[114]: id                int64
week                int64
```

IBM Home Page - Select or create a notebook Code - Jupyter Notebook

localhost:8891/notebooks/Downloads/SBSPS-Challenge-8325-Food-Demand-Forecasting-for-Food-Delivery-Company-using-IBM-Cloud-main/SBSPS-Challenge-...

Gmail YouTube Maps

jupyter Code (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Connected Not Trusted Python 3 (ipykernel)

Run Code

```
In [113]: trainfinal = trainfinal[cols]
trainfinal.head()
```

```
Out[113]:
```

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

```
In [114]: trainfinal.dtypes
```

```
Out[114]: id                int64
week                int64
city_code           int64
region_code         int64
center_type         object
op_area             float64
category            object
cuisine             object
checkout_price       float64
base_price           float64
emailer_for_promotion int64
homepage_featured    int64
num_orders           int64
dtype: object
```