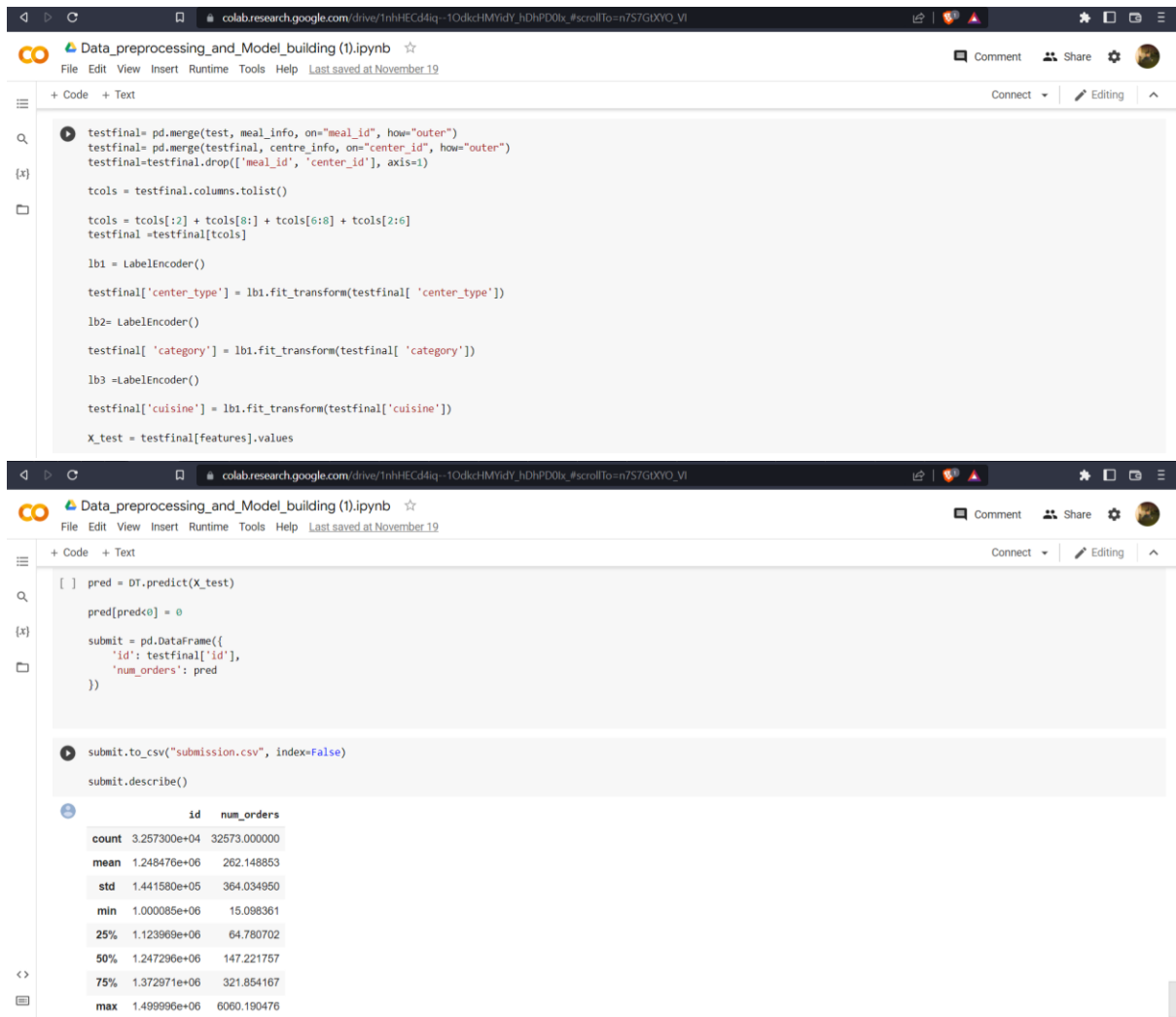


Team ID: PNT2022TMID23576

PROJECT NAME: DemandEst - AI powered Food Demand Forecaster

Team Leader



The image displays two screenshots of a Google Colab notebook titled "Data_preprocessing_and_Model_building (1).ipynb".

The top screenshot shows the following code:

```
testfinal= pd.merge(test, meal_info, on="meal_id", how="outer")
testfinal= pd.merge(testfinal, centre_info, on="center_id", how="outer")
testfinal=testfinal.drop(['meal_id', 'center_id'], axis=1)

tcols = testfinal.columns.tolist()

tcols = tcols[:2] + tcols[8:] + tcols[6:8] + tcols[2:6]
testfinal =testfinal[tcols]

lb1 = LabelEncoder()

testfinal['center_type'] = lb1.fit_transform(testfinal['center_type'])

lb2= LabelEncoder()

testfinal['category'] = lb1.fit_transform(testfinal['category'])

lb3 =LabelEncoder()

testfinal['cuisine'] = lb1.fit_transform(testfinal['cuisine'])

X_test = testfinal[features].values
```

The bottom screenshot shows the following code:

```
[ ] pred = DT.predict(X_test)

pred[pred<0] = 0

submit = pd.DataFrame({
    'id': testfinal['id'],
    'num_orders': pred
})

submit.to_csv("submission.csv", index=False)

submit.describe()
```

Below the code, a summary statistics table is displayed:

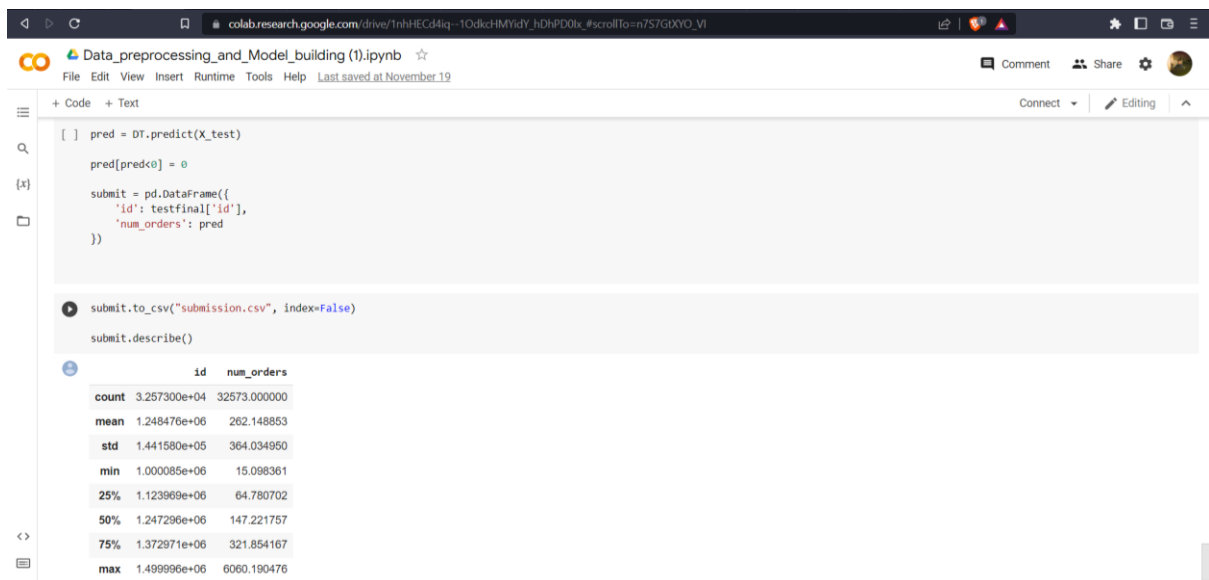
	id	num_orders
count	3.257300e+04	32573.000000
mean	1.248476e+06	262.148853
std	1.441580e+05	364.034950
min	1.000085e+06	15.098361
25%	1.123969e+06	64.780702
50%	1.247296e+06	147.221757
75%	1.372971e+06	321.854167
max	1.499996e+06	6060.190476

Team Member 1



The screenshot shows a Google Colab notebook titled "Data_preprocessing_and_Model_building (1).ipynb". The code in the notebook performs the following steps:

- Merge `test` and `meal_info` on `meal_id`.
- Merge the result with `centre_info` on `center_id`.
- Drop `meal_id` and `center_id` columns.
- Reorder columns: `testfinal = testfinal[tcols]` where `tcols` is `testfinal.columns.tolist()`.
- Initialize `lb1 = LabelEncoder()` and transform `center_type`.
- Initialize `lb2 = LabelEncoder()` and transform `category`.
- Initialize `lb3 = LabelEncoder()` and transform `cuisine`.
- Set `X_test = testfinal[features].values`.



The screenshot shows the continuation of the Google Colab notebook. The code performs the following steps:

- Predict using `DT.predict(X_test)` and store the result in `pred`.
- Create a submission DataFrame with `id` and `num_orders` columns.
- Save the submission to `submission.csv` using `submit.to_csv`.
- Describe the submission data using `submit.describe()`.

The output of `submit.describe()` is shown as a table:

	id	num_orders
count	3.257300e+04	32573.000000
mean	1.248476e+06	262.148853
std	1.441580e+05	364.034950
min	1.000085e+06	15.098361
25%	1.123969e+06	64.780702
50%	1.247296e+06	147.221757
75%	1.372971e+06	321.854167
max	1.499996e+06	6060.190476

Team Member 2



The screenshot shows a Google Colab notebook titled "Data_preprocessing_and_Model_building (1).ipynb". The code in the cell performs the following steps:

- Merge `test`, `meal_info`, and `centre_info` datasets on `meal_id` and `center_id` using `pd.merge` with `how="outer"`.
- Drop the `meal_id` and `center_id` columns from the merged dataset.
- Calculate the total number of orders per meal by summing the `num_orders` column across different dimensions (columns 12, 8, 6:8, and 2:6).
- Use `LabelEncoder` to transform the `center_type`, `category`, and `cuisine` columns into numerical values.
- Extract the feature values from the processed dataset into `X_test`.

```
testfinal= pd.merge(test, meal_info, on="meal_id", how="outer")
testfinal= pd.merge(testfinal, centre_info, on="center_id", how="outer")
testfinal=testfinal.drop(['meal_id', 'center_id'], axis=1)

tcols = testfinal.columns.tolist()

tcols = tcols[:12] + tcols[8:] + tcols[6:8] + tcols[2:6]
testfinal =testfinal[tcols]

lb1 = LabelEncoder()

testfinal['center_type'] = lb1.fit_transform(testfinal['center_type'])

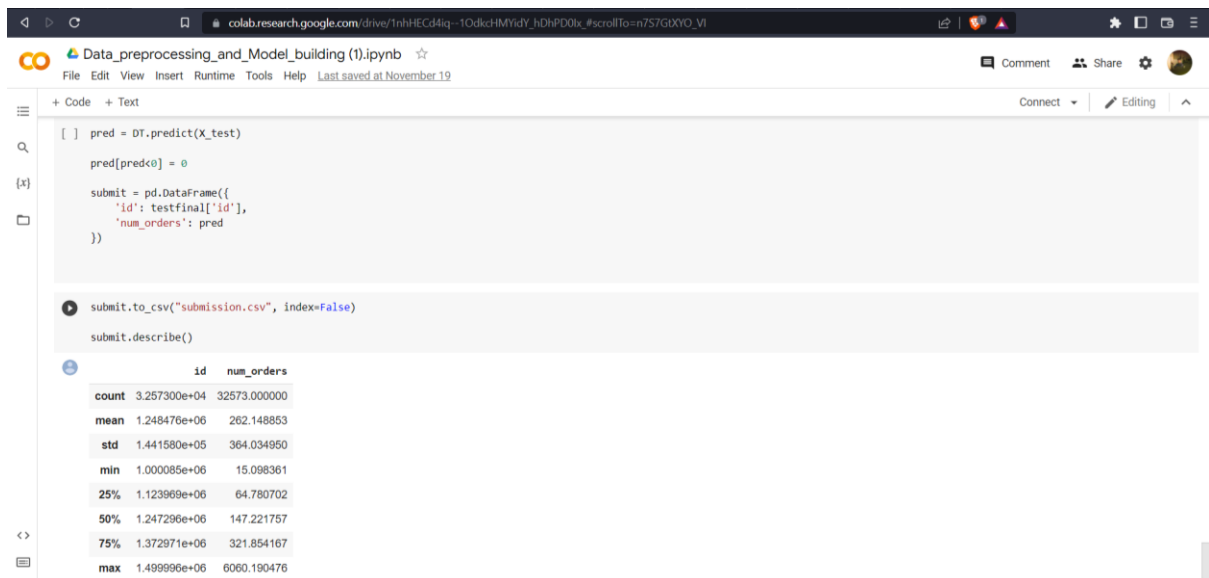
lb2= LabelEncoder()

testfinal['category'] = lb1.fit_transform(testfinal['category'])

lb3 =LabelEncoder()

testfinal['cuisine'] = lb1.fit_transform(testfinal['cuisine'])

X_test = testfinal[features].values
```



The screenshot shows the continuation of the Google Colab notebook. The code in this cell performs the following steps:

- Use the trained model `DT` to predict the number of orders for the test set `X_test`.
- Create a `DataFrame` for submission with columns `id` and `num_orders`.
- Submit the predictions to a CSV file named `submission.csv` using `submit.to_csv`.
- Display the summary statistics of the submission data using `submit.describe()`.

```
[ ] pred = DT.predict(X_test)

pred[pred<0] = 0

submit = pd.DataFrame({
    'id': testfinal['id'],
    'num_orders': pred
})

submit.to_csv("submission.csv", index=False)

submit.describe()
```

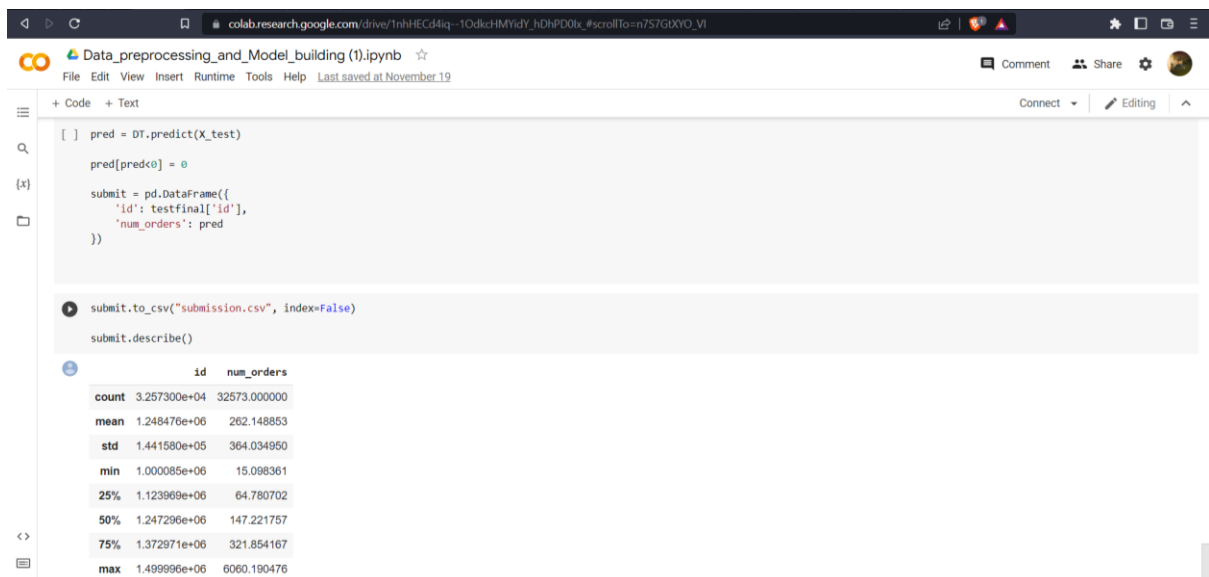
	id	num_orders
count	3.257300e+04	32573.000000
mean	1.248476e+06	262.148853
std	1.441580e+05	364.034950
min	1.000085e+06	15.098361
25%	1.123969e+06	64.780702
50%	1.247296e+06	147.221757
75%	1.372971e+06	321.854167
max	1.499996e+06	6060.190476

Team Member 3



The screenshot shows a Google Colab notebook titled "Data_preprocessing_and_Model_building (1).ipynb". The code in the cell performs the following steps:

- Merge `test` and `meal_info` on `meal_id`.
- Merge the result with `centre_info` on `center_id`.
- Drop `meal_id` and `center_id` columns.
- Reorder columns: `testfinal = testfinal[tcols]` where `tcols` is `testfinal.columns.tolist()` with columns 12, 8, 6, 8, and 2, 6 swapped.
- Encode categorical variables: `lb1 = LabelEncoder()` for `center_type`, `lb2 = LabelEncoder()` for `category`, and `lb3 = LabelEncoder()` for `cuisine`.
- Apply `fit_transform` for each categorical variable.
- Extract features: `X_test = testfinal[features].values`.



The screenshot shows the continuation of the Google Colab notebook. The code in the cell performs the following steps:

- Predict using the model: `pred = DT.predict(X_test)`.
- Initialize a submission DataFrame: `submit = pd.DataFrame({'id': testfinal['id'], 'num_orders': pred})`.
- Save the submission: `submit.to_csv("submission.csv", index=False)`.
- Describe the submission: `submit.describe()`.

The output of `submit.describe()` is displayed as a table:

	id	num_orders
count	3.257300e+04	32573.000000
mean	1.248476e+06	262.148853
std	1.441580e+05	364.034950
min	1.000085e+06	15.098361
25%	1.123969e+06	64.780702
50%	1.247296e+06	147.221757
75%	1.372971e+06	321.854167
max	1.499996e+06	6060.190476