RETAIL RECOMMENDER

Tredence ML Hackathon

Team Boilermakers:

Nikhil Katiki
Pavan Ghantasala
Srinikhil Bolneyti
Pratyusha Gajavalli

PURDUE UNIVERSITY | Krannert School of Management

Agenda

- ***Problem Description***
- ***Data Understanding***
- ***Exploratory Data Analysis***
- ***Hypothesis***
- ***Feature Engineering***
- ***Model Development***
- ***Model Selection***
- ***Results and Future Scope***

Problem Description

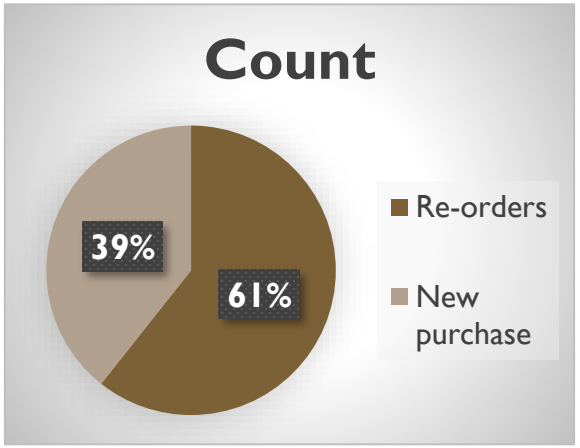
Re-order Prediction

- Instacart is an e-commerce grocery shopping site where a customer can purchase a product online from nearby grocery stores
- To predict which of the previously purchased products will be in a customer's next order, based on the purchase history of each customer
- Identify customer's ordering pattern and target customers with surprise product recommendations using tremendous customer order data
- To suggest new customers with potential product recommendations.

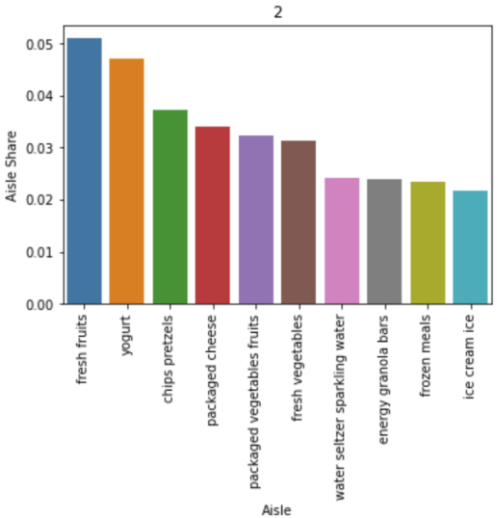
Data Understanding

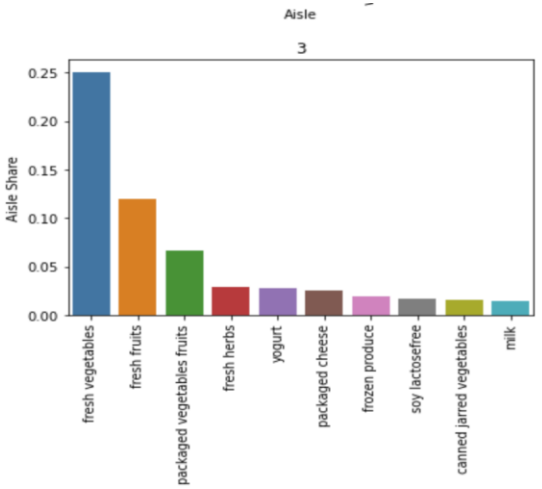
- **99,462** distinct users and 1 order details of each user is available in the training dataset
- **34,238** unique products are available in the training dataset.
- Add to cart order is the order in which each product is added to the cart during purchase
- Re-order is a binary variable indicating whether that product is previously purchased or not by that customer. Of all these purchases **477,908** are re-orders and **310,003** are new purchases.

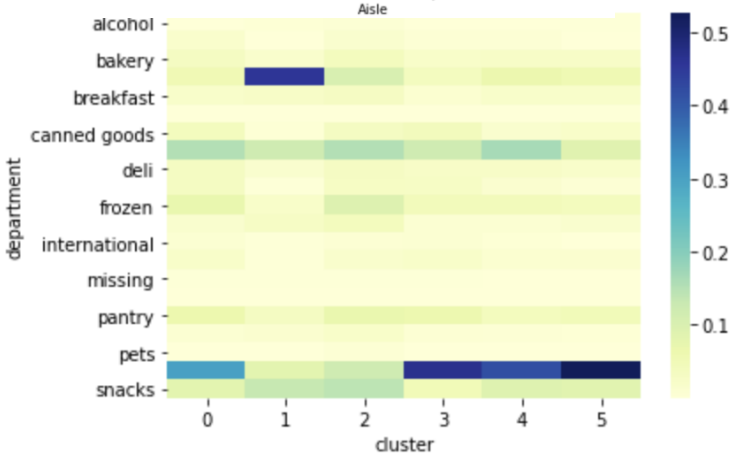
	order_id	product_id	add_to_cart_order	reordered	user_id	ID
0	1187899	27845	9	0	1	14
1	1187899	38928	3	1	1	2
2	1187899	39657	5	1	1	21
3	1187899	26405	4	1	1	119001
4	1187899	196	1	1	1	10

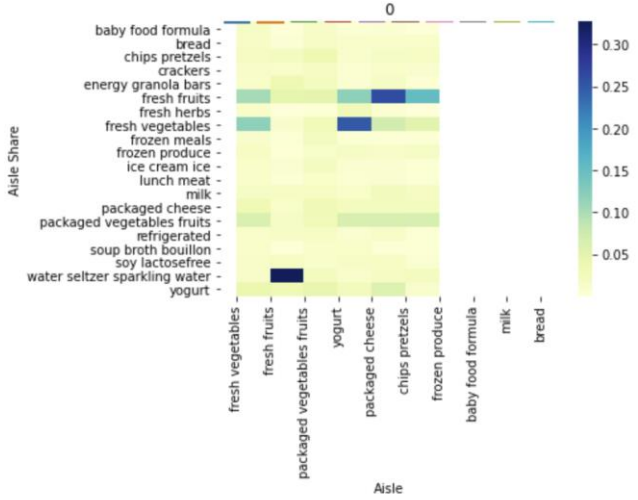


Exploratory Data Analysis









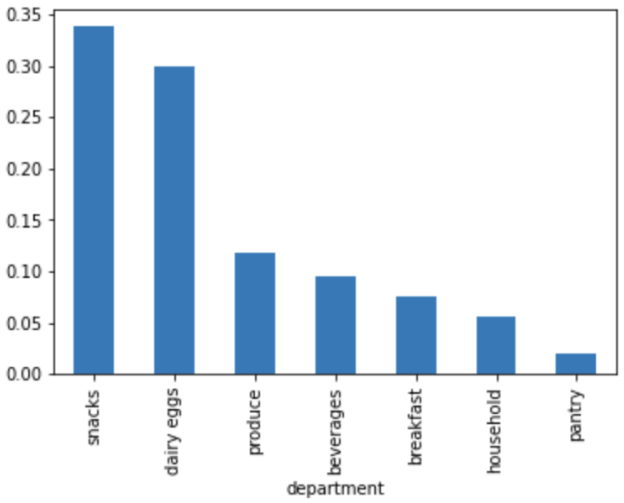
Hypothesis

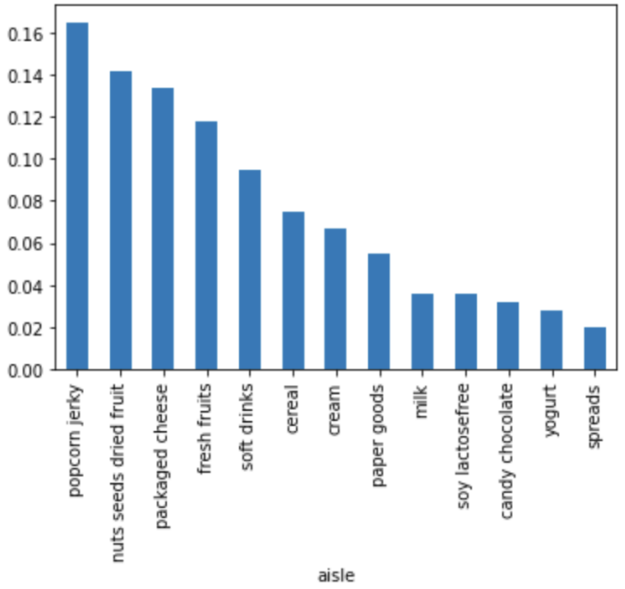
- Sum of add-to-cart order grouped by department indicates the customer's interest towards that department in an inverse fashion. If the sum is high, customer interest is less and vice-versa. Similarly, aisle interest is also captured.
- This customer interest is derived from the interaction between add-to-cart order and department or aisle.
- **Customer's interest towards a department or aisle is different across departments or aisles.**
- If these two derived variables are significantly explaining the re-order instance, we can use them as predictor variables for future orders.

Feature Engineering

- order_products_prior, departments, aisles, products datasets are merged with train dataset to get more purchase history of a customer and department, aisle details of the products
- Department share is the ratio between sum of add-to-cart order of a particular department and total sum of add-to-cart order of all departments
- Aisle share is the ratio between sum of add-to-cart order of a particular aisle and total sum of add-to-cart order of all aisles.

user_id = 1



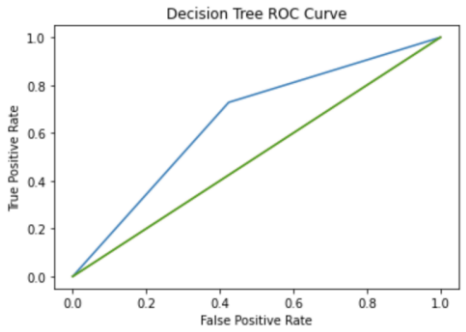


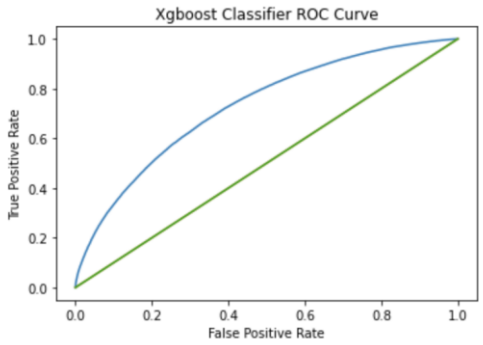
Model Development

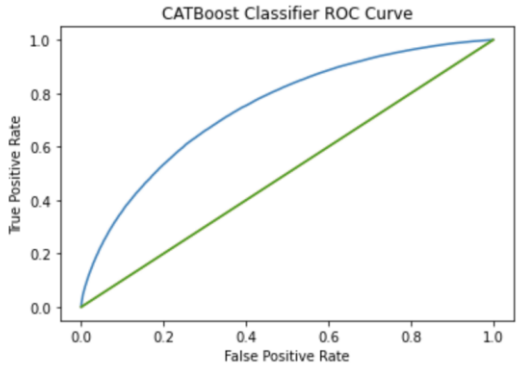
- Different models such as Logistic Regression, CatBoost, XGBoost, LightGBM, Random Forest and Decision Tree have been applied on the train data with all given and derived features
- Since add-to-cart order is a categorical variable while department_share and aisle_share are continuous variables, algorithm which works for both kinds should be chosen.
- As expected, CatBoost and Random Forest performed well.

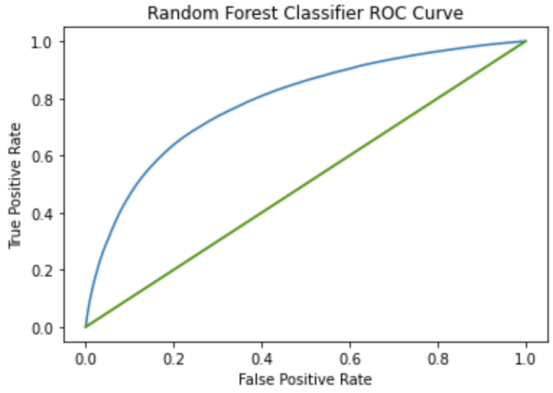
Model	Accuracy
Logistic	0.61
CatBoost	0.7
XGBoost	0.69
Random Forest	0.73
Light GBM	0.67
Decision Tree	0.67

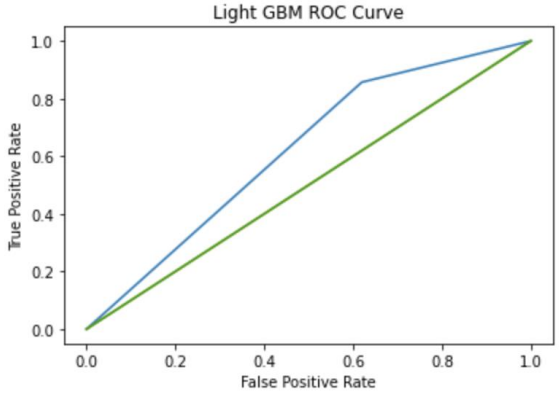
Model Selection

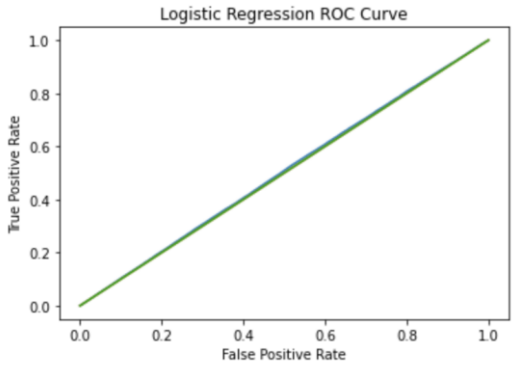












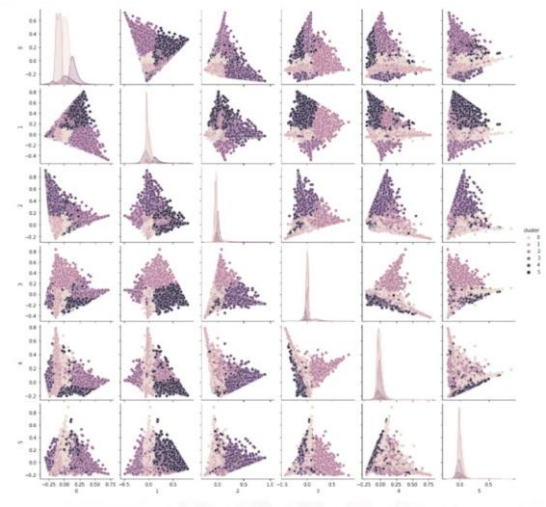
Model Selection

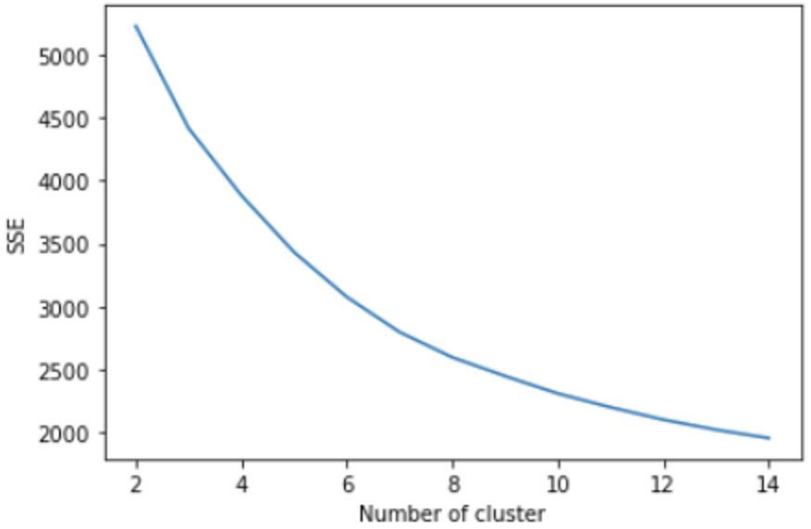
- From the ROC AUC curves, it is observed that Random Forest and Cat Boost stood out to be the best performing models.
- Furthermore, Grid Search CV is employed to tune hyper parameters to improve test accuracy.

```
import numpy as np
from sklearn.model_selection import RandomizedSearchCV
from pprint import pprint
# Number of trees in random forest
n_estimators = [int(x) for x in np.linspace(start = 200, stop = 2000, num = 10)]
# Number of features to consider at every split
max_features = ['auto', 'sqrt']
# Maximum number of levels in tree
max_depth = [int(x) for x in np.linspace(10, 110, num = 11)]
max_depth.append(None)
# Minimum number of samples required to split a node
min_samples_split = [2, 5, 10]
# Minimum number of samples required at each leaf node
min_samples_leaf = [1, 2, 4]
# Method of selecting samples for training each tree
bootstrap = [True, False]
# Create the random grid
random_grid = {'n_estimators': n_estimators,
               'max_features': max_features,
               'max_depth': max_depth,
               'min_samples_split': min_samples_split,
               'min_samples_leaf': min_samples_leaf,
               'bootstrap': bootstrap}
pprint(random_grid)
{'bootstrap': [True, False],
 'max_depth': [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None],
 'max_features': ['auto', 'sqrt'],
 'min_samples_leaf': [1, 2, 4],
 'min_samples_split': [2, 5, 10],
 'n_estimators': [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000]}
```

Results and Future Scope

- The reorder predictions can be used in inventory planning, customer targeting and product recommendation systems.
- Customers are clustered into natural groupings based on their department and aisle preferences.
- This can be used in a collaborative filtering scenario with association rule mining to recommend new products.





THANK YOU

Boilermakers

MS Business Analytics and Information Management