In This Part of the project we choose to Buid a recommendatation system for Customers for Olist Ecommerce sales data.

Importing the Dataset

```
# File location and type
file_location = "/FileStore/tables/d17export.csv"
file_type = "csv"

# CSV options
infer_schema = "true"
first_row_is_header = "true"
delimiter = ","

# The applied options are for CSV files. For other file types, these will be ignored.
df = spark.read.format(file_type) \
  .option("inferSchema", infer_schema) \
  .option("header", first_row_is_header) \
  .option("sep", delimiter) \
  .load(file_location)

display(df)
```

product_category_name	product_category_name_english	seller_id	seller_zip_code_prefix	seller_city	seller_state	product
perfumaria	perfumery	325f3178fb58e2a9778334621eecdbf9	6790	taboao da serra	SP	6782d5
esporte_lazer	sports_leisure	a17f621c590ea0fab3d5d883e1630ec6	18055	sorocaba	SP	e95ee6
utilidades_domesticas	housewares	1b4c3a6f53068f0b6944d2d005c9fc89	88730	sao ludgero	SC	e9a693
telefonia	telephony	ea8482cd71df3c1969d7b9473ff13abc	4160	sao paulo	SP	036734

Showing the first 1000 rows.

```
df.printSchema()
```

```
root
 |-- product_category_name: string (nullable = true)
 |-- product_category_name_english: string (nullable = true)
 |-- seller_id: string (nullable = true)
 |-- seller_zip_code_prefix: integer (nullable = true)
 |-- seller_city: string (nullable = true)
 |-- seller_state: string (nullable = true)
 |-- product_id: string (nullable = true)
 |-- product_name_lenght: integer (nullable = true)
 |-- product_description_lenght: integer (nullable = true)
 |-- product_photos_qty: integer (nullable = true)
 |-- product_weight_g: integer (nullable = true)
 |-- product_length_cm: integer (nullable = true)
 |-- product_height_cm: integer (nullable = true)
 |-- product_width_cm: integer (nullable = true)
 |-- order_id: string (nullable = true)
 |-- review_id: string (nullable = true)
 |-- review_score: integer (nullable = true)
 |-- review_comment_title: string (nullable = true)
 |-- review_comment_message: string (nullable = true)
 |-- review_creation_date: string (nullable = true)
```

```
df_rec = df.select('product_category_name_english', 'review_score', 'customer_id', 'order_id')
df_rec = df_rec.dropna(how='any')


# df_rec.show()
```

```
from pyspark.ml.feature import StringIndexer
# Index labels, adding metadata to the label column
labelIndexer = StringIndexer(inputCol='customer_id',
                             outputCol='indexedcustomer_id').fit(df_rec)
df_rec = labelIndexer.transform(df_rec)


from pyspark.ml.feature import StringIndexer
# Index labels, adding metadata to the label column
labelIndexer = StringIndexer(inputCol='product_category_name_english',
                             outputCol='indexedproduct_category_name_english').fit(df_rec)
df_rec = labelIndexer.transform(df_rec)


# df_rec.show()


train, test = df_rec.randomSplit([0.8,0.2])
train.show(5)
test.show(5)
```

```
+-----------------------------+------------+--------------------+--------------------+------------------+--------------------------------
----+
|product_category_name_english|review_score|         customer_id|            order_id|indexedcustomer_id|indexedproduct_category_name_eng
lish|
+-----------------------------+------------+--------------------+--------------------+------------------+--------------------------------
----+
|         agro_industry_and...|           1|572b683cea71aeb65...|5cb6969f0b3cd394d...|           47059.0|
42.0|
|         agro_industry_and...|           1|57a27b2aa5a10e277...|1f68afd1b515cc920...|           33244.0|
42.0|
|         agro_industry_and...|           1|d69adc38c89adba5d...|a7a256d455df2402d...|           48095.0|
42.0|
|         agro_industry_and...|           2|e768022fa47ad4364...|205587440f824c5e5...|           79535.0|
42.0|
|         agro_industry_and...|           4|291fd0287b7cfd757...|04e00ba23c33890ea...|           69613.0|
42.0|
+-----------------------------+------------+--------------------+--------------------+------------------+--------------------------------
```

```
only showing top 5 rows

+-----------------------------+------------+--------------------+--------------------+------------------+-------------------------------
----+
|product_category_name_english|review_score|         customer_id|            order_id|indexedcustomer_id|indexedproduct_category_name_eng
lish|
+-----------------------------+------------+--------------------+--------------------+------------------+-------------------------------
----+
|         agro_industry_and...|           4|dafe240fa4132e5da...|ca00c2ba5781124bd...|            4309.0|
42.0|
|         agro_industry_and...|           5|1eebfdb7083031b40...|21577126c19bf11a0...|              25.0|
42.0|
|         agro_industry_and...|           5|1eebfdb7083031b40...|21577126c19bf11a0...|              25.0|
42.0|
|         agro_industry_and...|           5|1eebfdb7083031b40...|21577126c19bf11a0...|              25.0|
42.0|
|         agro_industry_and...|           5|1eebfdb7083031b40...|21577126c19bf11a0...|              25.0|
42.0|
+-----------------------------+------------+--------------------+--------------------+------------------+-------------------------------
----+
only showing top 5 rows
```

```
import itertools
from math import sqrt
from operator import add
import sys
from pyspark.ml.recommendation import ALS
from pyspark.ml.evaluation import RegressionEvaluator
```

```
als = ALS(userCol="indexedcustomer_id", itemCol="indexedproduct_category_name_english", ratingCol="review_score")
model = als.fit(train)
predictions = model.transform(test)
```

```
# predictions.show(5)
```

```
from pyspark.ml.evaluation import RegressionEvaluator

evaluator = RegressionEvaluator(metricName="rmse", labelCol="review_score", predictionCol="prediction")
print ("The root mean squared error for our model is: " + str(evaluator.evaluate(predictions)))
```

```
The root mean squared error for our model is: nan
```

> The Mean square error by evaluator came to be zero as there are null values in the predictions.So in the next step,an average of review score is calculated and RSME is find out.

```
avgRatings = df_rec.select('review_score').groupBy().avg().first()[0]
print ("The average rating in the dataset is: " + str(avgRatings))
```

```
The average rating in the dataset is: 4.017507140957789
```

RSME for model after filling NA values with average of reviews.

```
evaluator = RegressionEvaluator(metricName="rmse", labelCol="review_score", predictionCol="prediction")
print ("The root mean squared error for our model is: " + str(evaluator.evaluate(predictions.na.fill(avgRatings))))
```

```
The root mean squared error for our model is: 1.1842142770516166
```

Making Predictions by Dropping the NA values

```
als2 = ALS(userCol="indexedcustomer_id", itemCol="indexedproduct_category_name_english", ratingCol="review_score")
model2 = als2.fit(train)
predictions2 = model2.transform(test)
from pyspark.ml.evaluation import RegressionEvaluator

evaluator2 = RegressionEvaluator(metricName="rmse", labelCol="review_score", predictionCol="prediction")
print ("The root mean squared error for our model is: " + str(evaluator2.evaluate(predictions2.na.drop())))
```

```
The root mean squared error for our model is: 0.6237809642666431
```

Product Recommendations For Customers

```
from pyspark.sql.functions import lit

def recommendProducts(model, user, nbRecommendations):
    dataSet = df_rec.select("indexedproduct_category_name_english").distinct().withColumn("indexedcustomer_id", lit(user))
    productsAlreadyRated = df_rec.filter(df_rec.indexedcustomer_id == user).select("indexedproduct_category_name_english",
"indexedcustomer_id")
    predictions = model.transform(dataSet.subtract(productsAlreadyRated)).dropna().orderBy("prediction",
ascending=False).limit(nbRecommendations).select("indexedproduct_category_name_english", "prediction")
    recommendations = predictions.join(df_rec, predictions.indexedproduct_category_name_english ==
df_rec.indexedproduct_category_name_english).select(predictions.indexedproduct_category_name_english,
df_rec.product_category_name_english, predictions.prediction).distinct()

    recommendations.show(truncate=False)


print ("Recommendations for user 3123:")
recommendProducts(model, 3123.0, 10)
```

```
Recommendations for user 3123:
+----------------------------------+----------------------------------------+----------+
|indexedproduct_category_name_english|product_category_name_english          |prediction|
+----------------------------------+----------------------------------------+----------+
|5.0                               |housewares                              |2.323516  |
|35.0                              |kitchen_dining_laundry_garden_furniture|3.612701  |
|23.0                              |musical_instruments                     |2.0694294 |
|70.0                              |security_and_services                   |2.2414503 |
|51.0                              |furniture_bedroom                       |2.483375  |
|61.0                              |music                                   |6.572212  |
|67.0                              |la_cuisine                              |5.7578096 |
|16.0                              |fashion_bags_accessories                |2.225671  |
|68.0                              |cds_dvds_musicals                       |6.375806  |
|46.0                              |signaling_and_security                  |2.4637306 |
```

```
print ("Recommendations for user 4617:")
recommendProducts(model, 4617.0, 10)
```

```
Recommendations for user 4617:
+-----------------------------------+----------------------------+----------+
|indexedproduct_category_name_english|product_category_name_english|prediction|
+-----------------------------------+----------------------------+----------+
|63.0                               |flowers                     |1.2837973 |
|52.0                               |costruction_tools_tools     |1.2110233 |
|29.0                               |home_confort                |0.8164832 |
|39.0                               |fixed_telephony             |1.0650543 |
|33.0                               |construction_tools_lights   |0.81425726|
|45.0                               |art                         |0.9784539 |
|26.0                               |books_general_interest      |0.8667488 |
|27.0                               |furniture_living_room       |1.0595983 |
|41.0                               |home_appliances_2           |1.2334654 |
|51.0                               |furniture_bedroom           |0.85207486|
+-----------------------------------+----------------------------+----------+
```