DDO Project for “Olist”

Submitted to Prof**.** Brandon Chiazza

Daniel Strauss, Surbhi Nayak, Humayra Kausar and Muhammad Bilal Jamil

daniel.strauss@mail.yu.edu| [SNAYAK@MAIL.YU.EDU|](mailto:snayak@mail.yu.edu|)

kausar@mail.yu.edu|mjamil@mail.yu.edu

Table of Contents

[**1. Strategy**  2](#_Toc451376266)

**1.1** **Introduction:** 2

[**1.2** **Competitive Landscape Analysis:** 2](#_Toc451376268)

[**1.3** **Balanced Score-Card:** 5](#_Toc451376268)

[**1.4** **Business Model Canvas:** 6](#_Toc451376268)

[**1.5** **OKRs:** 6](#_Toc451376268)

[**1.6** **Team Chart:** 7](#_Toc451376268)

[**2. Design**  8](#_Toc451376266)

[**2.1** **Personas:** 8](#_Toc451376268)

[**2.2** **Journey Maps:** 9](#_Toc451376268)

[**3. Dimensional Model** 10](#_Toc451376266)

[**3.1** **OLTP Database and ERD:** 10](#_Toc451376268)

[**3.2** **OLAP and ERD:** 14](#_Toc451376268)

[**4. Governance Model** 20](#_Toc451376271)

[**5. AI Use Case** 39](#_Toc451376277)

[**6. References** 41](#_Toc451376277)

1. **Strategy:**

**Introduction:**

Olist, a Brazilian e-commerce platform founded in 2015, connects small businesses to larger product marketplaces to help entrepreneurs sell their products to a larger customer base. Olist connects entrepreneurs with major online retailers and allows shopkeepers to advertise and sell in the marketplaces without complication, enabling retail companies to reach out to the international marketplaces, improve the shopping experience and modify their purchasing behavior. Through a single, seamless integration, Olist provides full stack operational support to merchants by managing product catalogues, inventory, pricing, fulfillment, customer service, and payments in a single place. It has attracted more than 200,000 users in 180 countries, according to the company.

**Problem Statement:**

How Olist can better serve their customers and help businesses to increase their e-commerce platform presence to reach out to the larger customer base as well as heighten their market share through data driven approach

**Solution**

* Collect customer data from various sources to understand customers purchasing behavior.
* Build AI model to predict customers purchasing pattern and to understand their needs and desires  which will leads to serve them better, maintain current customer base, upsell to current customer base, increase market penetration, and create brand addiction.
* Based on customer purchasing pattern and behavior we will provide tailor solutions to the businesses to heighten their online marketplace presence to reach out to the larger customer base and drive business growth.

**Competitive Landscape Analysis**

**Olist’s Major Competitors:**

**Ikman:** Online platform based in Sri-Lanka founded in 2012, specialized in buy and sell secondhand consumer goods locally. Ikman has the widest selection of popular secondhand items all over Sri Lanka. And they are in mobile app. Revenue $13.6M. Consumers have problems with their mobile app, price, and overall services.

**Top Ikman Integrations and Technologies :** Google Analytics, Google Global Site Tag, Google Universal Analytics, and Snowplow.

**Udaan** : Udaan is an Indian business-to-business e-commerce company based in Bangalore, India. It was registered as Hiveloop Technology pvt. ltd. in 2016. The company picks up products from manufacturers and sellers in 80-100 cities and delivers across 800-900 cities and towns. It has operations across categories including lifestyle, electronics, home & kitchen, staples, fruits, and vegetables, FMCG, pharma, toys and general merchandise. The company operates regionally and helps users to solve trade issues between small, medium, and large businesses across India looking to source merchandise from manufacturers, brands, white labels, and importers. They are in mobile app with 5 hundred thousand of categories of products. Udaan is solving core trade problems faced by small and medium businesses, that are unique to India, through its unique India-fit low-cost business model by leveraging technology and bringing the benefits of eCommerce to them.

* Udaan offers credit lines to buyers and sellers on their platform financed by Udaan and non-banking partners. Non-Banking Financial Companies working with Udaan are Hiveloop Capital Private Limited and Northern Arc Capital Limited.
* The platform has enabled logistics focused on b2b trade built on strong technology and operations for fulfilment and delivery service through udaanExpress. udaanCapital, focused on SME trade financing provides financial products and services for sellers and buyers to expand their business.
* The platform’s SaaS offerings such as analysis of real time marketing feedback through app data analytics enables brands and manufacturers to make well-informed decisions about product launches and testing of new products in different markets.

**Competitive Advantages :**

* Provide trade financing to buyers and sellers.
* Operates regionally

**Fyndiq :** Fyndiq is a Swedish online bargain shopping portal containing hundreds of thousands of bargains from hundreds of stores. Funded in 2009, the company popularly known as Sweden’s largest bargain house. Fashion and accessories, health and beauty goods, kid toys, home & living, and gifts are among the most common categories. They claim to have 1.6 million subscribers and 2 million website visitors each month.

**Competitive Advantages:**

* Provide Free Consultation to new business starters from an eCommerce Expert
* Perceived as Sweden’s largest bargain house
* Strong loyal customer base, 68% of sales are generated from returning customers
* 1700 active merchants
* The future generation, 99% of young adults know Fyndiq
* Sweden has the largest eCommerce market in the Nordic Region
* In Sweden, brand awareness is strong, particularly among young adults between the ages of 18 and 25
* Swedes are excellent online shoppers for 95 percent internet penetration and 67 percent do so on a monthly basis and spend an average of £ 1,668 per annum.

**Competitive Stance of Olist**

Porter’s Five Forces analysis to understand the industry in which Olist is operating

**Threat of new entrants – Low to Moderate**

**High Cost of Brand Development Online**

**Low Switching Cost**

**Strong Players in the Market**

**Rivalry among existing competitors - High**

**Numerous and Strong Competitors in the Industry**

**Similar Product and Services Offering**

**Difficult to Differentiate**

**Bargaining power of buyers – High**

**Very Low Switching Cost**

**Well-informed Customer**

**Low Price Sensitivity**

**Many Players in the marker**

**Bargaining power of suppliers – Moderate**

**Code of Conduct Encompassing Quality, Labor, Wages, as well as Sustainability**

**Threat of substitute products or services – High**

**Physical Retailers**

**Low Product Cost**

**Low Switching Cost**

Porter’s Five Competitive Forces model. The above figure exhibits how the five competitive forces apply to Olist and the E-Commerce industry.

From the model above, it can be professed that bargaining power of buyers, rivalry among competitors, and threat of substitute products are high. Due to the intense competition in the industry and high-level threats of substitute products, Olist must find solid ground to ensure long term competence and success. Therefore, creating solid and strong connections with customers and sellers can ensure that.

**Balanced Scorecard:**

Diagram

Description automatically generated

**Business Model Canvas:**

**A picture containing graphical user interface

Description automatically generated**

**Recommended OKRs:**

**Objective:** On-time product delivery.

**Key Result:** Find out the issues associated with on-time delivery where the estimated deliveries are late and resolve it.

**Objective:** Attract businesses to sell their product in Olist’s platform.

**Key Result**: Reach out to the business personally and explain how Olist can add value to their business

**Objective:** Helping customers to choose the right product

**Key Result** : Encourage customer to write reviews, encourage businesses to provide testimonials, and provide product details as much as possible such as details pictures

**Objective** : Improving Customer Satisfaction

**Key Result** : Ensure to collect 100% customer feedback

**Key Result** : Increase positive feedback and decrease negative feedback by 10% by end of next quarter.

**Objective:** Ensure installment paid in timely fashion

**Key Result** : Find out the issues associated with the installment payment, reach out to customers, and solve it.

**Team Chart:**

The roles and responsibility we need to achieve our objectives are following :

CTO Data engineer, Data analyst, Data Scientist, Data Stewards, Operation Manager,

Marketing Manager, Digital Marketing, Analyst VP sales.

1. **CTO (Chief Technology Officer) :**

* CTO will be responsible for overseeing the development and dissemination of technology for external customers, vendors, and other clients to achieve intended objectives as the company operates in digital platform.

1. **Data Analyst** :

* Acquiring sellers and consumers’ data from primary and secondary sources.
* Analyzing data using statistical techniques and providing reports.
* Identifying, analyzing, and interpreting trends or patterns of sellers’ objectives and buyers needs and desires.

1. **Data Engineer** :

* Data Engineer will be responsible for the following functions.
* Developing, constructing, testing & maintaining a complete architecture of data for Olist.
* Preparing data for prescriptive and predictive modeling to predict sellers needs, and consumers behaviors and purchasing pattern to achieve intended goals which are serving customers better, help sellers for their business growth.

1. **Data Scientist:**

* Identify valuable data sources for Olist and automate the collection processes
* Build predictive models and machine-learning algorithms to achieve intended objectives
* Propose solutions and strategies to business challenges Olist might encounter while attaining those goals
* Collaborate with data engineering and data analyst teams to achieve shared objectives

1. **VP sales**

* Ensure the sales team are meeting sales targets, close more deals and help generate revenue.

1. **Marketing Manager:**

* Will oversee the marketing campaign to bring in new customers.
* Promote businesses, services, products, and create brand awareness

1. **Operations Manager:**

* Making sure all the operations are running smoothly without any bottlenecks.

1. **Digital marketing analyst:**

* Monitor online marketing trends, statistics and develop strategies for marketing campaigns with marketing Manager.

1. **Data Steward:**

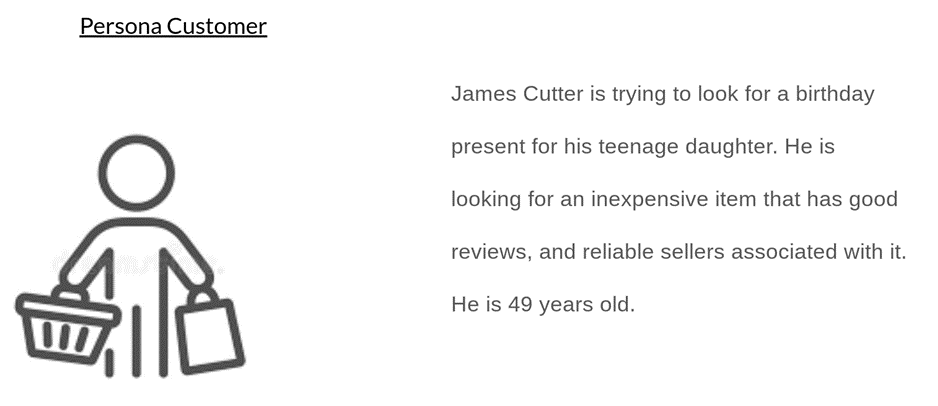
* Perform oversight tasks on the data. Perform data governance to ensure the overall quality of the data and make sure the data is fit for the organization’s purposes.
* Help Olist to adhering to changing data laws locally, regionally, and nationally to avoid penalties or fines
* Educating employees and keeping a pulse on new best practices and technology

1. **Design:**

**Personas:**

Graphical user interface

Description automatically generated with low confidence



**Journey Maps & Process Flow:**

Diagram

Description automatically generated

Diagram

Description automatically generated

1. **Use Cases:**

**Dimensional Model Use Case:**

For our dimensional model we used a PostgreSQL database, we created it by first developing an OLTP database and then we created an OLAP database.

**DDL for OLTP:**

CREATE TABLE olist\_customers (

customer\_id VARCHAR(250) PRIMARY KEY,

customer\_unique\_id VARCHAR(250),

customer\_zip\_code\_prefix int,

customer\_city VARCHAR(250),

customer\_state VARCHAR(250)

);

CREATE TABLE olist\_geolocation (

geolocation\_zip\_code\_prefix int,

geolocation\_lat float,

geolocation\_lng float,

geolocation\_city VARCHAR(250),

geolocation\_state VARCHAR(250)

);

CREATE TABLE olist\_order\_items(

order\_id VARCHAR(250),

order\_item\_id int,

product\_id VARCHAR(250),

seller\_id VARCHAR(250),

shipping\_limit\_date VARCHAR(250),

price float,

freight\_value float

);

CREATE TABLE olist\_order\_payments(

order\_id VARCHAR(250),

payment\_sequential int,

payment\_type VARCHAR(250),

payment\_installments int,

payment\_value float

);

CREATE TABLE olist\_order\_review(

review\_id VARCHAR(250),

order\_id VARCHAR(250),

review\_score int,

review\_comment\_title VARCHAR(250),

review\_comment\_message VARCHAR(250),

review\_creation\_date VARCHAR(250),

review\_answer\_timestamp VARCHAR(250)

);

CREATE TABLE olist\_orders\_data(

order\_id VARCHAR(250),

customer\_id VARCHAR(250),

order\_status VARCHAR(250),

order\_purchase\_timestamp VARCHAR(250),

order\_approved\_at VARCHAR(250),

order\_delivered\_carrier\_date VARCHAR(250),

order\_delivered\_customer\_date VARCHAR(250),

order\_estimated\_delivery\_date VARCHAR(250)

);

CREATE TABLE olist\_products(

product\_id VARCHAR(250),

product\_category\_name VARCHAR(250),

product\_name\_lenght float,

product\_description\_lenght float,

product\_photos\_qty float,

product\_weight\_g float,

product\_length\_cm float,

product\_height\_cm float,

product\_width\_cm float

);

CREATE TABLE olist\_seller(

seller\_id VARCHAR(250),

seller\_zip\_code\_prefix int,

seller\_city VARCHAR(250),

seller\_state VARCHAR(250)

);

CREATE TABLE product\_category\_name\_translation(

product\_category\_name VARCHAR(250),

product\_category\_name\_english VARCHAR(250)

);

**ERD for OLTP:**

Graphical user interface, table

Description automatically generated with medium confidence

**DDL for OLAP:**

--Dimension table 1 for customers:

Create table olist\_dw.customer\_dim (

customer\_id serial primary key,

customer\_city varchar(250),

customer\_state varchar (250)

);

/\*Dimension 1 ETL\*/

INSERT INTO olist\_dw.customer\_dim(customer\_city,customer\_state)

(SELECT DISTINCT customer\_city,customer\_state FROM public.olist\_customers);

--Dimension table 2 for orders\_items:

Create table olist\_dw.order\_items\_dim (

order\_id serial primary key,

product\_id serial,

seller\_id serial,

order\_item\_id int,

shipping\_limit\_date varchar(250)

);

/\*Dimension 2 ETL\*/

INSERT INTO olist\_dw.order\_items\_dim(order\_item\_id,shipping\_limit\_date)

(SELECT DISTINCT order\_item\_id,shipping\_limit\_date FROM public.olist\_order\_items);

--price and freight value

--will come as a measure in main fact table so will all the dates

--Dimension table 3 for payments:

Create table olist\_dw.payments\_dim (

order\_id serial primary key,

Payment\_installments int,

Payment\_sequential int,

Payment\_type varchar(250)

);

--payment\_value will be in fact table

/\*Dimension 3 ETL\*/

INSERT INTO olist\_dw.payments\_dim(Payment\_installments,Payment\_sequential,Payment\_type)

(SELECT DISTINCT Payment\_installments,Payment\_sequential,Payment\_type FROM public.olist\_order\_payments);

--Dimension table 4 for reviews:

Create table olist\_dw.review\_dim (

review\_id serial primary key,

order\_id serial,

review\_score int,

review\_comment\_title varchar(250),

review\_creation\_date varchar(250),

review\_answer\_timestamp varchar(250)

);

/\*Dimension 4 ETL\*/

INSERT INTO olist\_dw.review\_dim(review\_score,review\_comment\_title,review\_creation\_date,review\_answer\_timestamp)

(SELECT DISTINCT review\_score,review\_comment\_title,review\_creation\_date,review\_answer\_timestamp FROM public.olist\_order\_review);

--Dimension table 5 for order\_data:

Create table olist\_dw.order\_data\_dim (

order\_id serial primary key,

Customer\_id serial,

Order\_status varchar(250),

order\_purchase\_timestamp VARCHAR(250),

order\_approved\_at VARCHAR(250),

order\_delivered\_carrier\_date VARCHAR(250),

order\_delivered\_customer\_date VARCHAR(250),

order\_estimated\_delivery\_date VARCHAR(250)

);

/\*Dimension 5 ETL\*/

INSERT INTO olist\_dw.order\_data\_dim(Order\_status,order\_purchase\_timestamp,order\_approved\_at,order\_delivered\_carrier\_date,

order\_delivered\_customer\_date,order\_estimated\_delivery\_date)

(SELECT DISTINCT Order\_status,order\_purchase\_timestamp,order\_approved\_at,order\_delivered\_carrier\_date,

order\_delivered\_customer\_date,order\_estimated\_delivery\_date FROM public.olist\_orders\_data);

--Dimension table 6 for products:

Create table olist\_dw.products\_dim (

product\_id serial primary key,

product\_category\_name varchar(250)

);

/\*Dimension 6 ETL\*/

INSERT INTO olist\_dw.products\_dim(product\_category\_name)

(SELECT DISTINCT product\_category\_name FROM public.olist\_products);

/\* --measures:

Product\_name\_length float

Product\_description\_lenght float

Product\_photos\_qty float

Product\_weight\_g float

Product\_length\_cm float

Product\_height\_cm float

Product\_width\_cm float \*/

--Dimension table 7 for seller:

Create table Olist\_dw.seller\_dim (

seller\_id serial primary key,

Seller\_zip\_code\_prefix int,

Seller\_city varchar(250),

Seller\_state varchar (250)

);

/\*Dimension 7 ETL\*/

INSERT INTO olist\_dw.seller\_dim (Seller\_zip\_code\_prefix,seller\_city, seller\_state)

(SELECT DISTINCT Seller\_zip\_code\_prefix,seller\_city, seller\_state FROM public.olist\_seller);

--Dimension table 8 for transalation:

Create table Olist\_dw.transalation\_dim (

transalation\_id serial primary key,

Product\_category\_name varchar(250),

Product\_category\_name\_english varchar (250)

);

/\*Dimension 8 ETL\*/

INSERT INTO olist\_dw.transalation\_dim (Product\_category\_name, Product\_category\_name\_english)

(SELECT DISTINCT Product\_category\_name, Product\_category\_name\_english FROM public.product\_category\_name\_translation);

-- Dimension 9 Fact Table:

CREATE TABLE olist\_dw.fact\_dim (

customer\_id int references olist\_dw.customer\_dim(customer\_id),

order\_id int references olist\_dw.order\_items\_dim (order\_id),

order\_payment\_id int references olist\_dw.payments\_dim (order\_id),

review\_id int references olist\_dw.review\_dim (review\_id),

order\_data\_id int references olist\_dw.order\_data\_dim (order\_id),

product\_id int references olist\_dw.products\_dim (product\_id),

seller\_id int references olist\_dw.seller\_dim (seller\_id),

transalation\_id int references olist\_dw.transalation\_dim (transalation\_id),

price float,

freight\_value float,

payment\_value float,

Product\_name\_length float,

Product\_description\_lenght float,

Product\_photos\_qty float,

Product\_weight\_g float,

Product\_length\_cm float,

Product\_height\_cm float,

Product\_width\_cm float,

primary key (customer\_id, order\_id, order\_payment\_id,review\_id,order\_data\_id,product\_id,seller\_id,transalation\_id));

**ERD for OLAP:**

**Diagram

Description automatically generated**

**Governance Model:**

For governance model we used Jupyter Notebooks, following are the details:

#importing required libraries

import pandas as pd

from termcolor import colored

# path of the datasets

customers = 'olist\_customers\_dataset.csv'

geolocation = 'olist\_geolocation\_dataset.csv'

order\_item = 'olist\_order\_items\_dataset.csv'

order\_payment = 'olist\_order\_payments\_dataset.csv'

order\_review = 'olist\_order\_reviews\_dataset.csv'

orders = 'olist\_orders\_dataset.csv'

products = 'olist\_products\_dataset.csv'

sellers = 'olist\_sellers\_dataset.csv'

product\_category\_name\_translation = 'product\_category\_name\_translation.csv'

lst\_dataset = [customers,geolocation,order\_item,order\_payment,order\_review,

orders,products,sellers,product\_category\_name\_translation]

df\_olist = {}

for i in lst\_dataset:

df = pd.read\_csv(i)

i = i.replace('.csv','')

i = i.replace('\_dataset','')

i = i.replace('olist\_','')

df\_olist[i] = df

customers\_df = df\_olist['customers']

customers\_df.head()

| **customer\_id** | **customer\_unique\_id** | **customer\_zip\_code\_prefix** | **customer\_city** | **customer\_state** |
| --- | --- | --- | --- | --- |
| **0** | 06b8999e2fba1a1fbc88172c00ba8bc7 | 861eff4711a542e4b93843c6dd7febb0 | 14409 | franca | SP |
| **1** | 18955e83d337fd6b2def6b18a428ac77 | 290c77bc529b7ac935b93aa66c333dc3 | 9790 | sao bernardo do campo | SP |
| **2** | 4e7b3e00288586ebd08712fdd0374a03 | 060e732b5b29e8181a18229c7b0b2b5e | 1151 | sao paulo | SP |
| **3** | b2b6027bc5c5109e529d4dc6358b12c3 | 259dac757896d24d7702b9acbbff3f3c | 8775 | mogi das cruzes | SP |
| **4** | 4f2d8ab171c80ec8364f7c12e35b23ad | 345ecd01c38d18a9036ed96c73b8d066 | 13056 | campinas | SP |

geolocation\_df = df\_olist['geolocation']

geolocation\_df.head()

|  | **geolocation\_zip\_code\_prefix** | **geolocation\_lat** | **geolocation\_lng** | **geolocation\_city** | **geolocation\_state** |
| --- | --- | --- | --- | --- | --- |
| **0** | 1037 | -23.545621 | -46.639292 | sao paulo | SP |
| **1** | 1046 | -23.546081 | -46.644820 | sao paulo | SP |
| **2** | 1046 | -23.546129 | -46.642951 | sao paulo | SP |
| **3** | 1041 | -23.544392 | -46.639499 | sao paulo | SP |
| **4** | 1035 | -23.541578 | -46.641607 | sao paulo | SP |

order\_items\_df = df\_olist['order\_items']

order\_items\_df.head()

|  | **order\_id** | **order\_item\_id** | **product\_id** | **seller\_id** | **shipping\_limit\_date** | **price** | **freight\_value** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 00010242fe8c5a6d1ba2dd792cb16214 | 1 | 4244733e06e7ecb4970a6e2683c13e61 | 48436dade18ac8b2bce089ec2a041202 | 2017-09-19 09:45:35 | 58.90 | 13.29 |
| **1** | 00018f77f2f0320c557190d7a144bdd3 | 1 | e5f2d52b802189ee658865ca93d83a8f | dd7ddc04e1b6c2c614352b383efe2d36 | 2017-05-03 11:05:13 | 239.90 | 19.93 |
| **2** | 000229ec398224ef6ca0657da4fc703e | 1 | c777355d18b72b67abbeef9df44fd0fd | 5b51032eddd242adc84c38acab88f23d | 2018-01-18 14:48:30 | 199.00 | 17.87 |
| **3** | 00024acbcdf0a6daa1e931b038114c75 | 1 | 7634da152a4610f1595efa32f14722fc | 9d7a1d34a5052409006425275ba1c2b4 | 2018-08-15 10:10:18 | 12.99 | 12.79 |
| **4** | 00042b26cf59d7ce69dfabb4e55b4fd9 | 1 | ac6c3623068f30de03045865e4e10089 | df560393f3a51e74553ab94004ba5c87 | 2017-02-13 13:57:51 | 199.90 |  |

order\_payment\_df = df\_olist['order\_payments']

order\_payment\_df.head()

|  | **order\_id** | **payment\_sequential** | **payment\_type** | **payment\_installments** | **payment\_value** |
| --- | --- | --- | --- | --- | --- |
| **0** | b81ef226f3fe1789b1e8b2acac839d17 | 1 | credit\_card | 8 | 99.33 |
| **1** | a9810da82917af2d9aefd1278f1dcfa0 | 1 | credit\_card | 1 | 24.39 |
| **2** | 25e8ea4e93396b6fa0d3dd708e76c1bd | 1 | credit\_card | 1 | 65.71 |
| **3** | ba78997921bbcdc1373bb41e913ab953 | 1 | credit\_card | 8 | 107.78 |
| **4** | 42fdf880ba16b47b59251dd489d4441a | 1 | credit\_card | 2 | 128.45 |

order\_review\_df = df\_olist['order\_reviews']

order\_review\_df.head()

| **w\_id** | **order\_id** | **review\_score** | **review\_comment\_title** | **review\_comment\_message** | **review\_creation\_date** | **review\_answer\_timestamp** |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | 7bc2406110b926393aa56f80a40eba40 | 73fc7af87114b39712e6da79b0a377eb | 4 | NaN | NaN | 2018-01-18 00:00:00 | 2018-01-18 21:46:59 |
| **1** | 80e641a11e56f04c1ad469d5645fdfde | a548910a1c6147796b98fdf73dbeba33 | 5 | NaN | NaN | 2018-03-10 00:00:00 | 2018-03-11 03:05:13 |
| **2** | 228ce5500dc1d8e020d8d1322874b6f0 | f9e4b658b201a9f2ecdecbb34bed034b | 5 | NaN | NaN | 2018-02-17 00:00:00 | 2018-02-18 14:36:24 |
| **3** | e64fb393e7b32834bb789ff8bb30750e | 658677c97b385a9be170737859d3511b | 5 | NaN | Recebi bem antes do prazo estipulado. | 2017-04-21 00:00:00 | 2017-04-21 22:02:06 |
| **4** | f7c4243c7fe1938f181bec41a392bdeb | 8e6bfb81e283fa7e4f11123a3fb894f1 | 5 | NaN | Parabéns lojas lannister adorei comprar pela I... | 2018-03-01 00:00:00 | 2018-03-02 10:26:53 |

orders\_df = df\_olist['orders']

orders\_df.head()

| **order\_id** | **customer\_id** | **order\_status** | **order\_purchase\_timestamp** | **order\_approved\_at** | **order\_delivered\_carrier\_date** | **order\_delivered\_customer\_date** | **order\_estimated\_delivery\_date** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | e481f51cbdc54678b7cc49136f2d6af7 | 9ef432eb6251297304e76186b10a928d | delivered | 2017-10-02 10:56:33 | 2017-10-02 11:07:15 | 2017-10-04 19:55:00 | 2017-10-10 21:25:13 | 2017-10-18 00:00:00 |
| **1** | 53cdb2fc8bc7dce0b6741e2150273451 | b0830fb4747a6c6d20dea0b8c802d7ef | delivered | 2018-07-24 20:41:37 | 2018-07-26 03:24:27 | 2018-07-26 14:31:00 | 2018-08-07 15:27:45 | 2018-08-13 00:00:00 |
| **2** | 47770eb9100c2d0c44946d9cf07ec65d | 41ce2a54c0b03bf3443c3d931a367089 | delivered | 2018-08-08 08:38:49 | 2018-08-08 08:55:23 | 2018-08-08 13:50:00 | 2018-08-17 18:06:29 | 2018-09-04 00:00:00 |
| **3** | 949d5b44dbf5de918fe9c16f97b45f8a | f88197465ea7920adcdbec7375364d82 | delivered | 2017-11-18 19:28:06 | 2017-11-18 19:45:59 | 2017-11-22 13:39:59 | 2017-12-02 00:28:42 | 2017-12-15 00:00:00 |
| **4** | ad21c59c0840e6cb83a9ceb5573f8159 | 8ab97904e6daea8866dbdbc4fb7aad2c | delivered | 2018-02-13 21:18:39 | 2018-02-13 22:20:29 | 2018-02-14 19:46:34 |  |  |

products\_df = df\_olist['products']

products\_df.head()

| **product\_id** | **product\_category\_name** | **product\_name\_lenght** | **product\_description\_lenght** | **product\_photos\_qty** | **product\_weight\_g** | **product\_length\_cm** | **product\_height\_cm** | **product\_width\_cm** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 1e9e8ef04dbcff4541ed26657ea517e5 | perfumaria | 40.0 | 287.0 | 1.0 | 225.0 | 16.0 | 10.0 | 14.0 |
| **1** | 3aa071139cb16b67ca9e5dea641aaa2f | artes | 44.0 | 276.0 | 1.0 | 1000.0 | 30.0 | 18.0 | 20.0 |
| **2** | 96bd76ec8810374ed1b65e291975717f | esporte\_lazer | 46.0 | 250.0 | 1.0 | 154.0 | 18.0 | 9.0 | 15.0 |
| **3** | cef67bcfe19066a932b7673e239eb23d | bebes | 27.0 | 261.0 | 1.0 | 371.0 | 26.0 | 4.0 | 26.0 |
| **4** | 9dc1a7de274444849c219cff195d0b71 | utilidades\_domesticas | 37.0 | 402.0 | 4.0 | 625.0 | 20.0 |  |  |

sellers\_df = df\_olist['sellers']

sellers\_df.head()

| **seller\_id** | **seller\_zip\_code\_prefix** | **seller\_city** | **seller\_state** |
| --- | --- | --- | --- |
| **0** | 3442f8959a84dea7ee197c632cb2df15 | 13023 | campinas | SP |
| **1** | d1b65fc7debc3361ea86b5f14c68d2e2 | 13844 | mogi guacu | SP |
| **2** | ce3ad9de960102d0677a81f5d0bb7b2d | 20031 | rio de janeiro | RJ |
| **3** | c0f3eea2e14555b6faeea3dd58c1b1c3 | 4195 | sao paulo | SP |
| **4** | 51a04a8a6bdcb23deccc82b0b80742cf | 12914 | braganca paulista |  |

product\_category\_name\_translation\_df = df\_olist['product\_category\_name\_translation']

product\_category\_name\_translation\_df.head()

|  | **product\_category\_name** | **product\_category\_name\_english** |
| --- | --- | --- |
| **0** | beleza\_saude | health\_beauty |
| **1** | informatica\_acessorios | computers\_accessories |
| **2** | automotivo | auto |
| **3** | cama\_mesa\_banho | bed\_bath\_table |
| **4** | moveis\_decoracao | furniture\_decor |

for i in df\_olist:

print(colored(i[0].upper()+i[1:]+' DataFrame', 'red'),'\n')

print('---------------------------------------------------------------------','\n')

print(colored('No. of rows and columns :', 'blue'),'\n')

print(df\_olist[i].shape,'\n')

print(colored('Percentage of null values in dataframe for each column :','blue'),'\n')

print(df\_olist[i].isnull().sum()/df\_olist[i].shape[0],'\n')

print(colored('Information about dataframe :','blue'),'\n')

print(df\_olist[i].info(),'\n')

print(colored('Statstical summary of dataframe :','blue'),'\n')

print(df\_olist[i].describe(),'\n')

print(colored('Number of unique values:','blue'),'\n')

print(df\_olist[i].nunique(),'\n')

print('---------------------------------------------------------------------','\n')

Customers DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(99441, 5)

Percentage of null values in dataframe for each column :

customer\_id 0.0

customer\_unique\_id 0.0

customer\_zip\_code\_prefix 0.0

customer\_city 0.0

customer\_state 0.0

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 99441 entries, 0 to 99440

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 customer\_id 99441 non-null object

1 customer\_unique\_id 99441 non-null object

2 customer\_zip\_code\_prefix 99441 non-null int64

3 customer\_city 99441 non-null object

4 customer\_state 99441 non-null object

dtypes: int64(1), object(4)

memory usage: 3.8+ MB

None

Statstical summary of dataframe :

customer\_zip\_code\_prefix

count 99441.000000

mean 35137.474583

std 29797.938996

min 1003.000000

25% 11347.000000

50% 24416.000000

75% 58900.000000

max 99990.000000

Number of unique values:

customer\_id 99441

customer\_unique\_id 96096

customer\_zip\_code\_prefix 14994

customer\_city 4119

customer\_state 27

dtype: int64

---------------------------------------------------------------------

Geolocation DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(1000163, 5)

Percentage of null values in dataframe for each column :

geolocation\_zip\_code\_prefix 0.0

geolocation\_lat 0.0

geolocation\_lng 0.0

geolocation\_city 0.0

geolocation\_state 0.0

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 1000163 entries, 0 to 1000162

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 geolocation\_zip\_code\_prefix 1000163 non-null int64

1 geolocation\_lat 1000163 non-null float64

2 geolocation\_lng 1000163 non-null float64

3 geolocation\_city 1000163 non-null object

4 geolocation\_state 1000163 non-null object

dtypes: float64(2), int64(1), object(2)

memory usage: 38.2+ MB

None

Statstical summary of dataframe :

geolocation\_zip\_code\_prefix geolocation\_lat geolocation\_lng

count 1.000163e+06 1.000163e+06 1.000163e+06

mean 3.657417e+04 -2.117615e+01 -4.639054e+01

std 3.054934e+04 5.715866e+00 4.269748e+00

min 1.001000e+03 -3.660537e+01 -1.014668e+02

25% 1.107500e+04 -2.360355e+01 -4.857317e+01

50% 2.653000e+04 -2.291938e+01 -4.663788e+01

75% 6.350400e+04 -1.997962e+01 -4.376771e+01

max 9.999000e+04 4.506593e+01 1.211054e+02

Number of unique values:

geolocation\_zip\_code\_prefix 19015

geolocation\_lat 717358

geolocation\_lng 717613

geolocation\_city 8011

geolocation\_state 27

dtype: int64

---------------------------------------------------------------------

Order\_items DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(112650, 7)

Percentage of null values in dataframe for each column :

order\_id 0.0

order\_item\_id 0.0

product\_id 0.0

seller\_id 0.0

shipping\_limit\_date 0.0

price 0.0

freight\_value 0.0

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 112650 entries, 0 to 112649

Data columns (total 7 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 order\_id 112650 non-null object

1 order\_item\_id 112650 non-null int64

2 product\_id 112650 non-null object

3 seller\_id 112650 non-null object

4 shipping\_limit\_date 112650 non-null object

5 price 112650 non-null float64

6 freight\_value 112650 non-null float64

dtypes: float64(2), int64(1), object(4)

memory usage: 6.0+ MB

None

Statstical summary of dataframe :

order\_item\_id price freight\_value

count 112650.000000 112650.000000 112650.000000

mean 1.197834 120.653739 19.990320

std 0.705124 183.633928 15.806405

min 1.000000 0.850000 0.000000

25% 1.000000 39.900000 13.080000

50% 1.000000 74.990000 16.260000

75% 1.000000 134.900000 21.150000

max 21.000000 6735.000000 409.680000

Number of unique values:

order\_id 98666

order\_item\_id 21

product\_id 32951

seller\_id 3095

shipping\_limit\_date 93318

price 5968

freight\_value 6999

dtype: int64

---------------------------------------------------------------------

Order\_payments DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(103886, 5)

Percentage of null values in dataframe for each column :

order\_id 0.0

payment\_sequential 0.0

payment\_type 0.0

payment\_installments 0.0

payment\_value 0.0

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 103886 entries, 0 to 103885

Data columns (total 5 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 order\_id 103886 non-null object

1 payment\_sequential 103886 non-null int64

2 payment\_type 103886 non-null object

3 payment\_installments 103886 non-null int64

4 payment\_value 103886 non-null float64

dtypes: float64(1), int64(2), object(2)

memory usage: 4.0+ MB

None

Statstical summary of dataframe :

payment\_sequential payment\_installments payment\_value

count 103886.000000 103886.000000 103886.000000

mean 1.092679 2.853349 154.100380

std 0.706584 2.687051 217.494064

min 1.000000 0.000000 0.000000

25% 1.000000 1.000000 56.790000

50% 1.000000 1.000000 100.000000

75% 1.000000 4.000000 171.837500

max 29.000000 24.000000 13664.080000

Number of unique values:

order\_id 99440

payment\_sequential 29

payment\_type 5

payment\_installments 24

payment\_value 29077

dtype: int64

---------------------------------------------------------------------

Order\_reviews DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(99224, 7)

Percentage of null values in dataframe for each column :

review\_id 0.000000

order\_id 0.000000

review\_score 0.000000

review\_comment\_title 0.883415

review\_comment\_message 0.587025

review\_creation\_date 0.000000

review\_answer\_timestamp 0.000000

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 99224 entries, 0 to 99223

Data columns (total 7 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 review\_id 99224 non-null object

1 order\_id 99224 non-null object

2 review\_score 99224 non-null int64

3 review\_comment\_title 11568 non-null object

4 review\_comment\_message 40977 non-null object

5 review\_creation\_date 99224 non-null object

6 review\_answer\_timestamp 99224 non-null object

dtypes: int64(1), object(6)

memory usage: 5.3+ MB

None

Statstical summary of dataframe :

review\_score

count 99224.000000

mean 4.086421

std 1.347579

min 1.000000

25% 4.000000

50% 5.000000

75% 5.000000

max 5.000000

Number of unique values:

review\_id 98410

order\_id 98673

review\_score 5

review\_comment\_title 4527

review\_comment\_message 36159

review\_creation\_date 636

review\_answer\_timestamp 98248

dtype: int64

---------------------------------------------------------------------

Orders DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(99441, 8)

Percentage of null values in dataframe for each column :

order\_id 0.000000

customer\_id 0.000000

order\_status 0.000000

order\_purchase\_timestamp 0.000000

order\_approved\_at 0.001609

order\_delivered\_carrier\_date 0.017930

order\_delivered\_customer\_date 0.029817

order\_estimated\_delivery\_date 0.000000

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 99441 entries, 0 to 99440

Data columns (total 8 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 order\_id 99441 non-null object

1 customer\_id 99441 non-null object

2 order\_status 99441 non-null object

3 order\_purchase\_timestamp 99441 non-null object

4 order\_approved\_at 99281 non-null object

5 order\_delivered\_carrier\_date 97658 non-null object

6 order\_delivered\_customer\_date 96476 non-null object

7 order\_estimated\_delivery\_date 99441 non-null object

dtypes: object(8)

memory usage: 6.1+ MB

None

Statstical summary of dataframe :

order\_id customer\_id \

count 99441 99441

unique 99441 99441

top b1adda8b2734d4a0bae4885a90722859 97324b134a7e75d33decb2a65d637343

freq 1 1

order\_status order\_purchase\_timestamp order\_approved\_at \

count 99441 99441 99281

unique 8 98875 90733

top delivered 2018-07-28 13:11:22 2018-02-27 04:31:10

freq 96478 3 9

order\_delivered\_carrier\_date order\_delivered\_customer\_date \

count 97658 96476

unique 81018 95664

top 2018-05-09 15:48:00 2018-02-14 21:09:19

freq 47 3

order\_estimated\_delivery\_date

count 99441

unique 459

top 2017-12-20 00:00:00

freq 522

Number of unique values:

order\_id 99441

customer\_id 99441

order\_status 8

order\_purchase\_timestamp 98875

order\_approved\_at 90733

order\_delivered\_carrier\_date 81018

order\_delivered\_customer\_date 95664

order\_estimated\_delivery\_date 459

dtype: int64

---------------------------------------------------------------------

Products DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(32951, 9)

Percentage of null values in dataframe for each column :

product\_id 0.000000

product\_category\_name 0.018512

product\_name\_lenght 0.018512

product\_description\_lenght 0.018512

product\_photos\_qty 0.018512

product\_weight\_g 0.000061

product\_length\_cm 0.000061

product\_height\_cm 0.000061

product\_width\_cm 0.000061

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 32951 entries, 0 to 32950

Data columns (total 9 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 product\_id 32951 non-null object

1 product\_category\_name 32341 non-null object

2 product\_name\_lenght 32341 non-null float64

3 product\_description\_lenght 32341 non-null float64

4 product\_photos\_qty 32341 non-null float64

5 product\_weight\_g 32949 non-null float64

6 product\_length\_cm 32949 non-null float64

7 product\_height\_cm 32949 non-null float64

8 product\_width\_cm 32949 non-null float64

dtypes: float64(7), object(2)

memory usage: 2.3+ MB

None

Statstical summary of dataframe :

product\_name\_lenght product\_description\_lenght product\_photos\_qty \

count 32341.000000 32341.000000 32341.000000

mean 48.476949 771.495285 2.188986

std 10.245741 635.115225 1.736766

min 5.000000 4.000000 1.000000

25% 42.000000 339.000000 1.000000

50% 51.000000 595.000000 1.000000

75% 57.000000 972.000000 3.000000

max 76.000000 3992.000000 20.000000

product\_weight\_g product\_length\_cm product\_height\_cm \

count 32949.000000 32949.000000 32949.000000

mean 2276.472488 30.815078 16.937661

std 4282.038731 16.914458 13.637554

min 0.000000 7.000000 2.000000

25% 300.000000 18.000000 8.000000

50% 700.000000 25.000000 13.000000

75% 1900.000000 38.000000 21.000000

max 40425.000000 105.000000 105.000000

product\_width\_cm

count 32949.000000

mean 23.196728

std 12.079047

min 6.000000

25% 15.000000

50% 20.000000

75% 30.000000

max 118.000000

Number of unique values:

product\_id 32951

product\_category\_name 73

product\_name\_lenght 66

product\_description\_lenght 2960

product\_photos\_qty 19

product\_weight\_g 2204

product\_length\_cm 99

product\_height\_cm 102

product\_width\_cm 95

dtype: int64

---------------------------------------------------------------------

Sellers DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(3095, 4)

Percentage of null values in dataframe for each column :

seller\_id 0.0

seller\_zip\_code\_prefix 0.0

seller\_city 0.0

seller\_state 0.0

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 3095 entries, 0 to 3094

Data columns (total 4 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 seller\_id 3095 non-null object

1 seller\_zip\_code\_prefix 3095 non-null int64

2 seller\_city 3095 non-null object

3 seller\_state 3095 non-null object

dtypes: int64(1), object(3)

memory usage: 96.8+ KB

None

Statstical summary of dataframe :

seller\_zip\_code\_prefix

count 3095.000000

mean 32291.059451

std 32713.453830

min 1001.000000

25% 7093.500000

50% 14940.000000

75% 64552.500000

max 99730.000000

Number of unique values:

seller\_id 3095

seller\_zip\_code\_prefix 2246

seller\_city 611

seller\_state 23

dtype: int64

---------------------------------------------------------------------

Product\_category\_name\_translation DataFrame

---------------------------------------------------------------------

No. of rows and columns :

(71, 2)

Percentage of null values in dataframe for each column :

product\_category\_name 0.0

product\_category\_name\_english 0.0

dtype: float64

Information about dataframe :

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 71 entries, 0 to 70

Data columns (total 2 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 product\_category\_name 71 non-null object

1 product\_category\_name\_english 71 non-null object

dtypes: object(2)

memory usage: 1.2+ KB

None

Statstical summary of dataframe :

product\_category\_name product\_category\_name\_english

count 71 71

unique 71 71

top fashion\_roupa\_masculina office\_furniture

freq 1 1

Number of unique values:

product\_category\_name 71

product\_category\_name\_english 71

dtype: int64

---------------------------------------------------------------------

**AI Use Case**

* Describe predictive analytics use cases involving your data set. Describe in plain English what your model would predict and explain the benefit this would provide, be it a business benefit, a social benefit, or something else.

We have two personas to work with this dataset. One Customer and the other a Seller. This means whatever predictive analytics we do we need for it to effectively be informative for all three of our personas. The best predictive analytics that could be employed may differ for each of these two personas.

For our Seller, we may wish to do some predictions on customer purchases. In this prediction we would be looking at the products sold, the number of products sold, and the overall amount earned by each product. This could potentially be used to inform predictions for future earnings for each product sold by sellers.

For our customers, we want to predict which Sellers offer the best deals. We want low prices. So, we could potentially use this data to inform predictions on which sellers will have the best prices in the future. Likewise, we can inform predictions on which seller locations have the best pricing.

These models would allow both our personas to have better access to the information they need to inform their decisions. Customers can find cheaper products, Sellers can predict future purchases, and Investors can determine which sellers will be profitable investments.

* Identify the subset of your data that could be used to train the ML model and specify your feature columns and label column. **You do not have to create the model**-- that goes beyond the scope of this class.

The data we will be working with for each model type is different. Suffice to say we will need payment\_value, Geolocation\_state, Customer\_unique\_id, Seller\_id, Order\_purcase\_timestamp, Product\_id, order\_id, freight\_value. We may need several other columns, but these ones would be the main ones used to inform the predictions. The idea would be to use the order\_purchase\_timestamp to inform sales data, purchases, and Seller profit over time. This way we could use time series analysis to predict future earnings, sales, and purchases.

* Identify five potential ethical issues that could crop up in your use case (every use case has them -- think and you will come up with them) and how you would mitigate or prevent them from causing harm.

The first major issue is that sellers trying to predict what purchases users may make in the future could potentially violate customers' privacy. For example, the famous scenario of sending a teenage girl's home and parents’ information for pregnancy-related products without the parents or even their daughter knowing that she is pregnant.

The second major issue with predicting purchases of customers is that there may be selective stocks purchased or produced by the seller for resale. Non-Vital products that are popular to most customers may be prioritized over products that are essential but are unpopular. There may be ethical issues of sellers trying to keep less of such essential products in stock. For example, an uncommonly used medical product.

Another issue with trying to predict customers' purchases is related to trust. People may be buying things that they don't want to buy. That they don't want to be encouraged to buy. If you make those purchases more accessible it would be counterproductive for them. This may cause a lack of trust. For example, a person who is trying to lose weight may want to cut out sugary sweets. If your model predicts an increase in the number of sweets bought you might target sales of those items at that customer. This would violate the customer’s trust.

Another ethical issue is for predicting where to find the lowest cost for customers. Sellers may take advantage of this in an unethical fashion if they have a higher capital reserve than their competitors. They may artificially lower prices and take the financial loss too slowly, driving competitors bankrupt. Then raises prices after their competitors are out of business.

Finally, predicting which sellers have the lowest prices for a product may cause issues with driving newer, and less well-established companies out of business. Newer companies will not have the luxury of charging as little as well-established companies. It may also lead to the bankruptcy of sellers in high rent areas. As they would have to charge more for a product to make up for the higher rents.

**References:**

<https://pitchbook.com/profiles/company/102473-65#overview>

<https://valorcapitalgroup.com/case-studies/olist-redesigned-the-marketplace-business-model-to-fit-the-realities-of-ecommerce-in-brazil/>

<https://www.datanyze.com/companies/ikman/455449763>

<https://play.google.com/store/apps/details?id=lk.ikman&hl=en_US&gl=US&showAllReviews=true>

<https://golden.com/wiki/Udaan_(company)-DB5E534>

<https://udaan.com/about-us>

<https://cedcommerce.com/blog/selling-on-fyndiq-marketplace-bargain-superstore/>

<https://www.investopedia.com/terms/b/balancedscorecard.asp>

<https://www.dnb.com/perspectives/master-data/6-key-responsibilities-of-data-stewards.html>