Final_Project (4)

June 13, 2020

0.1 Ecommerce Analysis Final Project

Group Menbers: Shuyun Tang and Yuehan Guo

1 Part I Abstract(100+)

One paragraph that briefly summarizes the problem you addressed, questions of interest, the data, the techniques used, findings and key results, and conclusions.

In our final project, we analyzed a real 100k+ real-time local online orders dataset in Brazil in 2019, and explored potential factors involved in e-shopping process that could influence customers' satisfaction levels. In doing this, we used Numpy, Pandas, and datetime for processing data; Altair, Seaborn, Matplot for visualization; and logistic regression, random forest, and NLP for fitting our classification dataframe (influential factors) and predicting our target dataframe (review score). We found out that delivery time and product details have a huge impact on customers' satisfaction level. Shorter delivery period and more detailed product description could lead to better comments and review scores. By comparing different models, we've also found that if e-shops want to improve their service, they should pay attention to review comments prior to all other factors mentioned above.

2 Part II Introduction

Part 2 Introduction (200 words or more):**

In this section you should describe 1. the primary goal(s) or question(s) that your project addresses, 1. the motivation for your project, i.e., why your readers should be interested, 1. the relevant background of your topic, including a brief literature review (a paragraph with 1-3 references) describing any prior related work. 1. the dataset you are using to answer your question. You should address why this data is appropriate for answering your question.

Nowadays, online shopping is becoming more and more popular. People could shop whatever they want at home with a few finger clicks, and need not to consider all those bothersome transportations to get to the shopping mall and the energy and time spent with the in-store assistants. People could even spend more time on comparing different products and deciding what to buy. Driven by this trend, it is very important for eshop owners to explore what is really valued by customers in order to make more profit. Besides those eshop owners, other readers who are online shopper can gain insightful, experienced reports from most of the other online shoppers. So they can aviod some fraud info in the eshops.

The primary goal of our project is to explore differenct influencing factors (namely product price, delivery length, payment method, quantity of product photos, and the geographic position of customers) on customer's e-purshace bahaviors and satisfaction, so that we could predict customers' purchase preference and the trend of ecommerce.

Shuyun is an experienced accessories e-shop owner, and he would like to get some inspiration on how to improve his e-shop to make more benefit by analyzing the customer data; and Yuehan is an experienced online shopper who is very familiar with different factors involved in online shopping that could potentially affect customers' behavior and preference. Hence, this data exploration is especially meaningful and interesting for us.

This data was from https://www.kaggle.com/jainaashish/orders-merged, and it doesn't have a specific license. The publisher does not provide detailed information either. However, after our analysis, we found out that most of the buyers were located in Brazil and that this was a real 100k+ realtime local online orders dataset in Brazil in 2019. It provides lots of clean, insightful data about those influencing factors listed above and thus, it is appropriate for us to use.

2.1 Questions of interest

We want to explore the review score and other influential factors' relationships and patterns. These influential factors include delivery time, delivery time difference (actual-estimated), products' photos, description length in words, review comments. By estimating the review scores based on these factors, we want to get an idea of what factor could influence customers' satisfaction level in which way so that we could find potential ways to improve our e-shop and provide customers better experience.

[1]:
```python
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
## Seaborn is a Python data visualization library based on matplotlib.
## It provides a high-level interface for drawing attractive and informative␣
↪statistical graphics.

import altair as alt
%matplotlib inline
pd.set_option('display.max_columns', None)
```

[2]:
```python
from datetime import datetime
```

[3]:
```python
# from google.colab import drive
# drive.mount('/content/drive')
```

3 Ecommerce Data

[4]:
```python
customers = pd.read_csv("Orders_merged.csv")
```

[5]:
```python
customers.head()
```

[5]:
```
                         product_id                         seller_id  \
0  00066f42aeeb9f3007548bb9d3f33c38  5670f4db5b62c43d542e1b2d56b0cf7c
1  00088930e925c41fd95ebfe695fd2655  7142540dd4c91e2237acb7e911c4eba2
2  0009406fd7479715e4bef61dd91f2462  4a3ca9315b744ce9f8e9374361493884
3  000b8f95fcb9e0096488278317764d19  40ec8ab6cdafbcc4f544da38c67da39a
4  000b8f95fcb9e0096488278317764d19  40ec8ab6cdafbcc4f544da38c67da39a

                           order_id                       customer_id  \
0  f30149f4a8882a08895b6a242aa0d612  86c180c33f454b35e1596a99da3dddc4
1  f5eda0ded77c1293b04c953138c8331d  68f2b37558e27791155db34bcded5ac0
2  0bf736fd0fd5169d60de3699fcbcf986  6cd217b674e22cf568f6a2cf6060fd07
3  6f0dfb5b5398b271cc6bbd9ee263530e  8517e7c86998bf39a540087da6f115d9
4  3aba44d8e554ab4bb8c09f6f78032ca8  82b838f513e00463174cc7cae7e76c1f

  order_status order_purchase_timestamp    order_approved_at  \
0    delivered      2018-05-20 18:45:00  2018-05-20 18:58:59
1    delivered      2017-12-12 19:20:00  2017-12-12 19:32:19
2    delivered      2017-12-21 16:21:00  2017-12-22 17:31:27
3    delivered      2018-08-01 22:00:00  2018-08-01 22:15:19
4    delivered      2018-08-10 13:24:00  2018-08-10 13:35:21

  order_delivered_carrier_date order_delivered_customer_date  \
0          2018-05-21 16:09:00           2018-06-06 22:11:00
1          2017-12-20 20:12:42           2017-12-23 17:11:00
2          2018-01-02 22:27:47           2018-01-06 15:03:00
3          2018-08-02 14:20:00           2018-08-07 17:38:00
4          2018-08-13 14:43:00           2018-08-17 21:33:00

  order_estimated_delivery_date                customer_unique_id  \
0           2018-06-20 00:00:00  cd929c5ecff5fc60e9d808d33702e434
1           2018-01-05 00:00:00  cbbeff6b693e69511cf9d059f4b71036
2           2018-01-16 00:00:00  f51fb63558e88eb3373773d106fa6880
3           2018-08-24 00:00:00  7f2dfd48dba158dbf61ba2ea631d93df
4           2018-08-27 00:00:00  4e32da06df703a2561f63e75b13f6260

   customer_zip_code_prefix  customer_city customer_state  \
0                     95890       teutonia             RS
1                     14403         franca             SP
2                      2883      sao paulo             SP
3                     93530  novo hamburgo             RS
4                     95174    farroupilha             RS
```

```
                          review_id  review_score review_comment_title  \
0  91845d1f2ee1fdb677c769fad86f2109             5                  NaN
1  e5636189f943b2589b37f715a3bcae96             4                  NaN
2  32247878e34bd6e8d7dbf7b31a4ae0b0             1                  NaN
3  14303ce09673466b69c4354628aa5a84             5          Produto bom
4  40f2e7bbfda859ba75411743546849b0             5                  NaN

                              review_comment_message review_creation_date  \
0                                                NaN  2018-06-07 00:00:00
1                                                NaN  2017-12-24 00:00:00
2        Meu produto não foi entregue até o momento!  2018-01-07 00:00:00
3  Produto bom, mas o pegador da tampa é de plást…  2018-08-08 00:00:00
4  Produto igual ao anunciado, de excelente quali…  2018-08-18 00:00:00

  review_answer_timestamp  payment_sequential payment_type  \
0     2018-06-08 10:59:20                 1.0  credit_card
1     2017-12-27 13:23:27                 1.0  credit_card
2     2018-01-11 11:03:53                 1.0  credit_card
3     2018-08-08 23:48:48                 1.0  credit_card
4     2018-08-22 12:40:29                 1.0  credit_card

   payment_installments  payment_value  order_item_id   price  freight_value  \
0                   3.0         120.24              1  101.65          18.59
1                   1.0         143.83              1  129.90          13.93
2                  10.0         242.10              1  229.00          13.10
3                   1.0          78.50              1   58.90          19.60
4                   4.0          78.50              1   58.90          19.60

   seller_zip_code_prefix        seller_city seller_state  \
0                    3694          sao paulo           SP
1                   16301          penapolis           SP
2                   14940           ibitinga           SP
3                   85603  francisco beltrao           PR
4                   85603  francisco beltrao           PR

   product_category_name  product_name_lenght  product_description_lenght  \
0             perfumaria                 53.0                       596.0
1             automotivo                 56.0                       752.0
2         cama_mesa_banho                 50.0                       266.0
3  utilidades_domesticas                 25.0                       364.0
4  utilidades_domesticas                 25.0                       364.0

   product_photos_qty  product_weight_g  product_length_cm  product_height_cm  \
0                 6.0             300.0               20.0               16.0
1                 4.0            1225.0               55.0               10.0
2                 2.0             300.0               45.0               15.0
```

```
3              3.0          550.0           19.0           24.0
4              3.0          550.0           19.0           24.0

   product_width_cm
0              16.0
1              26.0
2              35.0
3              12.0
4              12.0
```

4 Data Cleaning

Since there are too many columns, we should create a new dataframe with only the columns that we are interested in. Let's create a dataframe **"df"**, which is the target column **'review__score'**(customers' satisfaction) with the "influencing factors" colums **'order__purchase__timestamp'**(when did the customers place the orders), **'order__estimated__delivery__date'**(when did the customers estimated to receive), **'product__description__lenght'**(the description length listed in the e-shops) **'product__photos__qty'**(how many photos did the shopper provide), **'price'** (purchase price), **'freight value'** (the shipping fee). Those columns are enough for our questions of interest.

[47]:
```
# make a new df contained the above columns
df=customers.drop(columns=["order_status","order_delivered_carrier_date",
          "order_approved_at","customer_zip_code_prefix","customer_unique_id",
         ␣
 ↪"customer_city","review_id","review_comment_title","review_comment_message",
         ␣
 ↪"review_creation_date","review_answer_timestamp","order_item_id","seller_zip_code_prefix",
          "seller_city","product_category_name","product_id","seller_id",
          "order_id","customer_id",
         ␣
 ↪"customer_state","seller_state","payment_sequential","payment_value","payment_type","produc
         ␣
 ↪"product_height_cm","product_length_cm","product_width_cm","payment_installments","product_


df.head()
```

[47]:
```
  order_purchase_timestamp order_delivered_customer_date  \
0      2018-05-20 18:45:00           2018-06-06 22:11:00
1      2017-12-12 19:20:00           2017-12-23 17:11:00
2      2017-12-21 16:21:00           2018-01-06 15:03:00
3      2018-08-01 22:00:00           2018-08-07 17:38:00
4      2018-08-10 13:24:00           2018-08-17 21:33:00
```

```
  order_estimated_delivery_date  review_score   price  freight_value  \
0           2018-06-20 00:00:00             5  101.65          18.59
1           2018-01-05 00:00:00             4  129.90          13.93
2           2018-01-16 00:00:00             1  229.00          13.10
3           2018-08-24 00:00:00             5   58.90          19.60
4           2018-08-27 00:00:00             5   58.90          19.60

   product_description_lenght  product_photos_qty
0                       596.0                 6.0
1                       752.0                 4.0
2                       266.0                 2.0
3                       364.0                 3.0
4                       364.0                 3.0
```

4.1 Features Engineering

We will then add some new features based on the current columns: **del_time** and **est_del_time** are the datetime conversion of the 'order_delivered_customer_date' and 'order_estimated_delivery_date' columns. **timediff** is the delivery time minus the estimated delivery time and converts to the days difference to the integers. **deliver_time** is the delivery time minus the purchase time and converts to the integers. **purchase_weekday** is the weekday of the purchase time, in which 0 is Monday and 6 is Sunday. **Late** is the categorical variable that indicates 0 if the timediff is less than 0, which means delivery time is earlier than estimated delivery time, or the delivery is not late than the estimated time and vice versa.

We will add those columns to our dataframe and explore their relationships to the review score.

[48]:
```
df['del_time']= pd.to_datetime(df['order_delivered_customer_date']).dt.
 ↪floor('d')
df['est_del_time']= pd.to_datetime(df['order_estimated_delivery_date'])

df['order_purchase_timestamp'] = pd.to_datetime(df['order_purchase_timestamp']).
 ↪dt.floor('d')
df['timediff']=df['del_time']-df['est_del_time']
df['timediff'] = pd.to_numeric(df['timediff'].dt.days, downcast='integer')

df['deliver_time']=df['del_time']-df['order_purchase_timestamp']
df['deliver_time'] = pd.to_numeric(df['deliver_time'].dt.days,␣
 ↪downcast='integer')

df['purchase_weekday']=df['order_purchase_timestamp'].dt.weekday ##Monday=0,␣
 ↪Sunday=6.

df['Late']=np.where(df['timediff']<0,0,1) ##0 means timediff<0, actual del time␣
 ↪is earlier than estimated del
```

```
df=df.
↪drop(columns=['order_delivered_customer_date','order_estimated_delivery_date','del_time','o
```

4.2 Review Score Category redesign

As the eshop owner, Shuyun will usually classify the 1, 2 review scores as the "bad reviews".

In order to make our model more accurate, we will sort out the review scores that were lower or equal than 2 into the negative reviews and denote it by "0"; and the review scores higher than 2 into the positive reviews, denoted by "1".

[49]:
```
df['review_score']=(df['review_score']>2)+0
```

[50]:
```
df.head()
```

[50]:
```
   review_score   price  freight_value  product_description_lenght  \
0             1  101.65          18.59                       596.0
1             1  129.90          13.93                       752.0
2             0  229.00          13.10                       266.0
3             1   58.90          19.60                       364.0
4             1   58.90          19.60                       364.0

   product_photos_qty  timediff  deliver_time  purchase_weekday  Late
0                 6.0     -14.0          17.0                 6     0
1                 4.0     -13.0          11.0                 1     0
2                 2.0     -10.0          16.0                 3     0
3                 3.0     -17.0           6.0                 2     0
4                 3.0     -10.0           7.0                 4     0
```

5 Pre-analysis Data Exploring

Before we begin analyzing our dataset, we need to do a little bit of data exploration.

- Are there any missing values? If so, how many?

[10]:
```
customers.isnull().sum().sum()
#total count is 3522735, rows are 96479
```

[10]: 147613

[51]:
```
df.isnull().sum().sum()
# df after adjusting columns total Nah count is 2728
```

[51]: 2728

[52]:
```python
# we will drop the null values here
df=df.dropna()
```

- Are there any unique values? Any values that seems strange?

All the values are in our expectation and no unique values. Some large payment values are probably luxury goods.

[12]:
```python
df.info
```

[12]:
```
<bound method DataFrame.info of        review_score   price  freight_value
product_description_lenght  \
0                 1  101.65          18.59                        596.0
1                 1  129.90          13.93                        752.0
2                 0  229.00          13.10                        266.0
3                 1   58.90          19.60                        364.0
4                 1   58.90          19.60                        364.0
…               …     …            …                          …
96473             1   34.99           7.39                        501.0
96474             1   29.99          18.23                        501.0
96475             1   34.99           7.51                        501.0
96476             1   34.99          18.23                        501.0
96477             1  249.99          53.88                       1536.0

       product_photos_qty  timediff  deliver_time  purchase_weekday  Late
0                     6.0       -14            17                 6     0
1                     4.0       -13            11                 1     0
2                     2.0       -10            16                 3     0
3                     3.0       -17             6                 2     0
4                     3.0       -10             7                 4     0
…                   …       …          …               …   …
96473                 5.0       -12             2                 1     0
96474                 5.0         0            19                 6     1
96475                 5.0        -1             4                 4     0
96476                 5.0       -20             6                 3     0
96477                 3.0         2            23                 2     1

[95114 rows x 9 columns]>
```

[13]:
```python
df.describe() #all values are making sense
```

[13]:
```
       review_score         price  freight_value  product_description_lenght  \
count  95114.000000  95114.000000   95114.000000                95114.000000
mean       0.869073    125.308130      20.181286                  793.259205
std        0.337322    189.635046      15.817141                  653.533103
min        0.000000      0.850000       0.000000                    4.000000
25%        1.000000     41.792500      13.310000                  349.000000
50%        1.000000     79.000000      16.390000                  607.000000
```

```
75%        1.000000    139.900000      21.250000                    995.000000
max        1.000000   6735.000000     409.680000                   3992.000000

       product_photos_qty      timediff  deliver_time  purchase_weekday  \
count        95114.000000  95114.000000  95114.000000      95114.000000
mean             2.250584    -11.883834     12.493566          2.755861
std              1.745368     10.183467      9.551784          1.966987
min              1.000000   -147.000000      0.000000          0.000000
25%              1.000000    -17.000000      7.000000          1.000000
50%              2.000000    -12.000000     10.000000          3.000000
75%              3.000000     -7.000000     16.000000          4.000000
max             20.000000    188.000000    210.000000          6.000000

               Late
count  95114.000000
mean       0.080987
std        0.272817
min        0.000000
25%        0.000000
50%        0.000000
75%        0.000000
max        1.000000
```

5.1 Check out the details of each factors based on groupby the review score

[14]:
```python
df.groupby('review_score').describe()
```

[14]:
```
                price                                                     \
                count        mean         std   min   25%   50%    75%
review_score
0             12453.0  132.769346  213.404498  3.54  44.0  79.9  141.8
1             82661.0  124.184087  185.766219  0.85  40.9  79.0  139.4

                   freight_value                                          \
                 max         count       mean        std  min    25%    50%
review_score
0             6729.0       12453.0  21.590711  17.791020  0.0  14.10  17.06
1             6735.0       82661.0  19.968955  15.487005  0.0  13.14  16.29

                          product_description_lenght              \
                75%    max                      count        mean
review_score
0             22.80  321.88                   12453.0  771.465510
1             21.13  409.68                   82661.0  796.542457

                                                               \
```

```
                    std   min    25%    50%    75%     max
review_score
0             643.626381  20.0  340.0  590.0  982.0  3950.0
1             654.953569   4.0  352.0  610.0  999.0  3992.0

             product_photos_qty                                            \
                          count      mean       std  min  25%  50%  75%   max
review_score
0                       12453.0  2.166787  1.723093  1.0  1.0  1.0  3.0  19.0
1                       82661.0  2.263208  1.748361  1.0  1.0  2.0  3.0  20.0

             timediff                                                   \
                count       mean        std    min   25%   50%  75%    max
review_score
0             12453.0  -5.096282  15.572905  -69.0 -15.0  -8.0  5.0  188.0
1             82661.0 -12.906389   8.649106 -147.0 -17.0 -13.0 -8.0  162.0

             deliver_time                                                \
                    count      mean        std  min  25%   50%   75%    max
review_score
0                 12453.0  20.19947  15.592059  1.0  9.0  16.0  29.0  210.0
1                 82661.0  11.33266   7.620058  0.0  6.0  10.0  15.0  195.0

             purchase_weekday                                            \
                        count      mean       std  min  25%  50%  75%  max
review_score
0                     12453.0  2.769614  1.963611  0.0  1.0  3.0  4.0  6.0
1                     82661.0  2.753790  1.967499  0.0  1.0  3.0  4.0  6.0

                 Late
                count      mean       std  min  25%  50%  75%  max
review_score
0             12453.0  0.337188  0.472769  0.0  0.0  0.0  1.0  1.0
1             82661.0  0.042390  0.201478  0.0  0.0  0.0  0.0  1.0
```

•

5.2 Which parts of the data were entered by a human? Are there any other potential sources of error?

All the data were supposed to be generated by the e-shop platform, and there should be no data entered by a human after our obeservation to the data source. There might be some potential sources of error on the other columns such as the length/ weight of the products, but our dataframe will not address those potential columns here.

We could also see some noticable large or small values for delivery time. The reasons are probably custom regulation or the pre-order sales.

-

5.3 What are the ethical considerations regarding this dataset?

The primary ethical considerations are the privacy of each customers, especially their payment infomation. Thus, when generating the processed dataset, we will exclude those sensitive columns to protect their infomation.

Another consideration is the customers' locations. Since the original dataset contains their exact address and zip code, it is not safe to use them in our final project without the instructions of the original pubulisher. So we exclude those columns in our processed dataset.

There is no specific groups in our processed dataset over-represented or underrepresented, but the locations are in Brazil, which might possibly make our exploration lack of diversity due to cultural differences.

-

5.4 What "principles of measurement" might be particularly relevant to our questions (distortion, relevance, precision, cost)?

Since all the records were given precise, the precision of our analysis should be guaranteed. Also, we likely have no measurement distortion since the original dataset did not distort the ecommerce system and structure under study. The customers' purchase behaviors and reviews won't be affected by the observation.

The most questionable part might be the relevance, since our records come from different individuals with different backgrounds. Also, the whole dataset is collected from Brazil, which might be different to the dominant e-commerce system in Asia. Sample size and selection will result in lots of time cost since we need to consider and extract the useful variables from more than 30 columns.

5.5 EDA

Our ultimate goal is to explore how to make potential customers more satisfied. In order to do this, we need to analyze the potential factors which could possibly bring us a better understanding in marketing. To be specific, we will explore the following: 1. Delivery length& delivery late vs review score 2. Freight value & purchase behavior& payment amount vs review score 3. product photo quantity&description length vs review score and price 4. Beyond the processed features, we will also analyze some interesting patterns from the original dataset–**bonus parts**: Which weekday usually has the most buyers? Which region usually has the most buyers and why? Which payment methods is the most popular? What other insightful thoughts will be drilled from those patterns?

[15]:
```
#disable the Altair maximum rows limit
alt.data_transformers.disable_max_rows()
```

[15]: DataTransformerRegistry.enable('default')

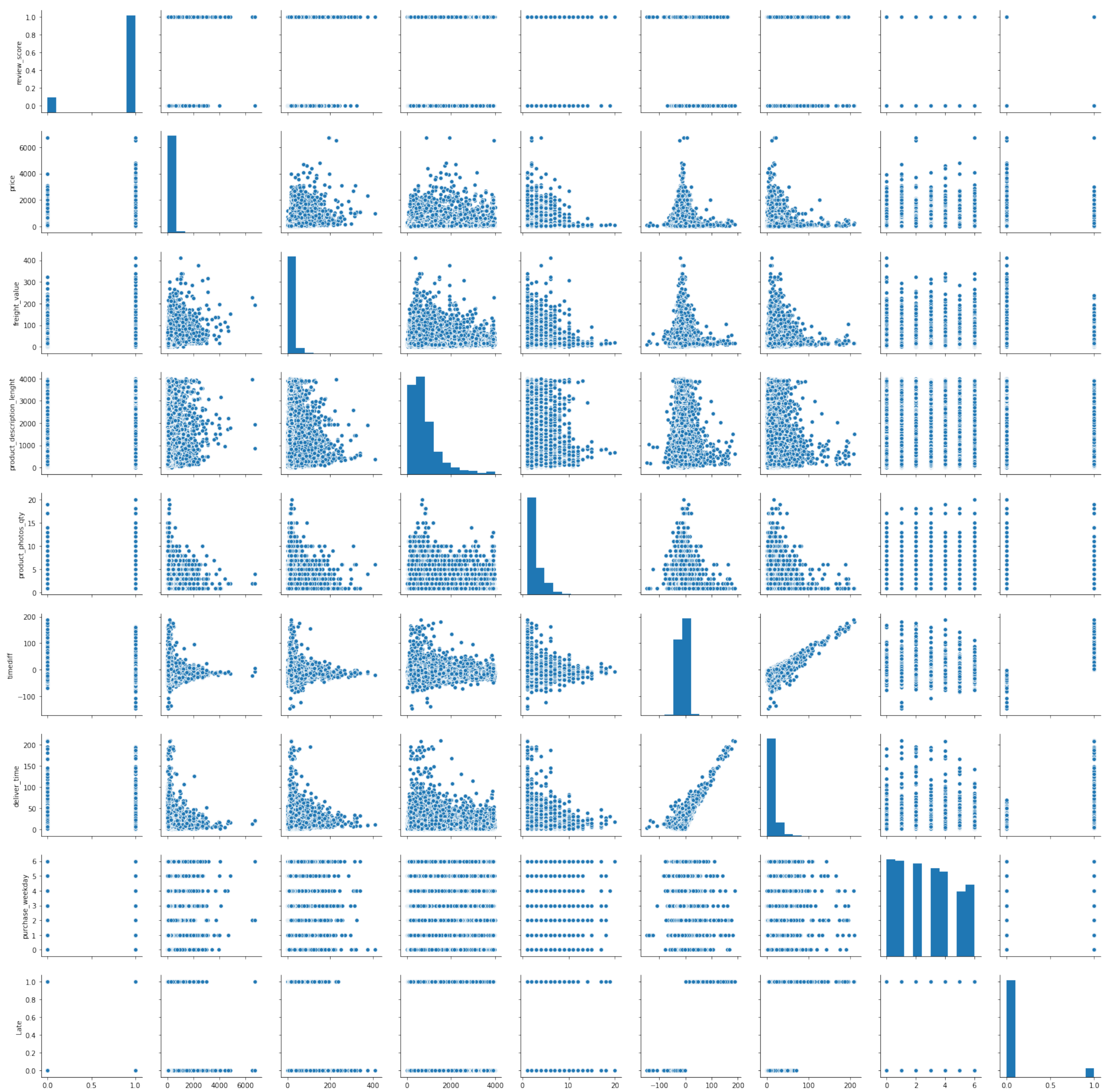
5.6 Pair plot

Let's create a pair-plot first to have a brief glance of some of the potentially interesting relationships.

There are some noticable patterns between the **review score and delivery time**, **product description length, product photo quantity and price**, **delivery time and freight value**, etc

```
[16]: sns.pairplot(df)
```

[16]: <seaborn.axisgrid.PairGrid at 0x7f0f78ccf490>



5.7 Correlation Matrix

We will then conduct the correlation matrix to further explore those relationships that we found interesting.

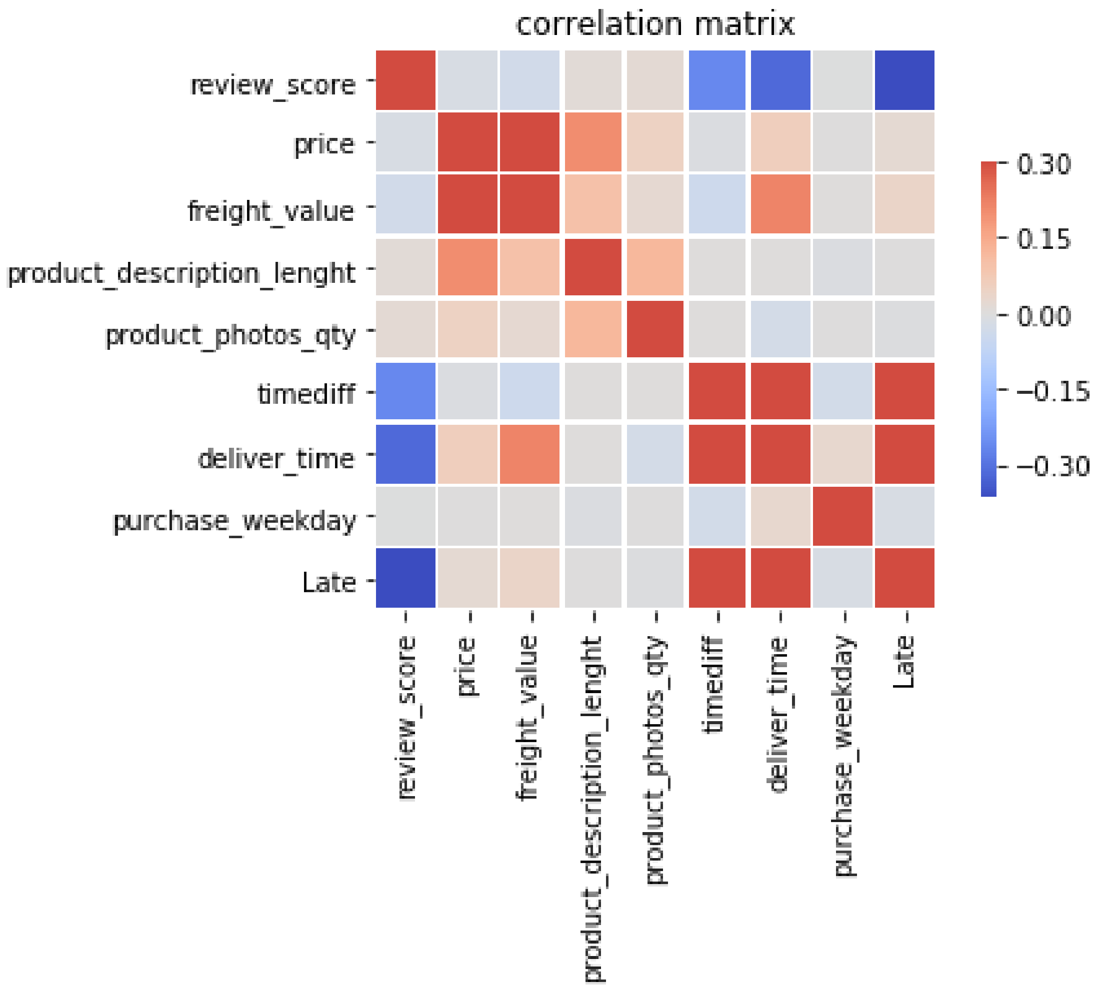
The relationships are shown as: negative relationship between the timediff, delivery time, and late delivery vs the review score. It is obvious that most of the buyers don't wanna wait too long. Also, the freight value, price, product description length and product photo quantities all have the positive relationship. We can assumn the goods that are more complex and valuable usually have more detailed information listed in the eshops.

[17]:
```
# Compute the correlation matrix
corr = df.corr()



# Draw the heatmap with the mask and correct aspect ratio
sns.heatmap(corr, cmap='coolwarm', vmax=.3, center=0,
            square=True, linewidths=.6, cbar_kws={"shrink": .6}).
 ↪set_title('correlation matrix')
```

[17]:
```
Text(0.5, 1, 'correlation matrix')
```



The delivery time clearly has a negative relationship to the review score. The review score 0 (negative review score) has a greater density in the longer delivery time

[18]:
```
alt.Chart(df).transform_density(
    'deliver_time',
    as_=['deliver_time', 'density'],
    groupby=['review_score']
).mark_area(orient='horizontal').encode(
    y=alt.Y('deliver_time', axis = alt.Axis(title = "delivery time (days)")),
    color='review_score:N',
    x=alt.X(
        'density:Q',
        stack='center',
        impute=None,
        title=None,
        axis=alt.Axis(labels=False, values=[0],grid=False, ticks=True),
```

```
    ),
    column=alt.Column(
        'review_score:N',
        header=alt.Header(
            titleOrient='bottom',
            labelOrient='bottom',
            labelPadding=0,
        ),
    )
).properties(
    title='Review Score vs Delivery Time'
).properties(
    width=100
).configure_facet(
    spacing=0
).configure_view(
    stroke=None
)
```

[18]: alt.Chart(…)

[19]:
```
alt.Chart(df).transform_density(
    'timediff',
    as_=['timediff', 'density'],
    groupby=['review_score']
).mark_area(orient='horizontal').encode(
    y=alt.Y('timediff', axis = alt.Axis(title = "delivery time difference␣
↪(days)")),
    color='review_score:N',
    x=alt.X(
        'density:Q',
        stack='center',
        impute=None,
        title=None,
        axis=alt.Axis(labels=False, values=[0],grid=False, ticks=True),
    ),
    column=alt.Column(
        'review_score:N',
        header=alt.Header(
            titleOrient='bottom',
            labelOrient='bottom',
            labelPadding=0,
        ),
    )
).properties(
    title='Review Score vs Delivery Time Difference (Actual-Estimated Delivery␣
↪Time)'
```

```
).properties(
    width=100
).configure_facet(
    spacing=0
).configure_view(
    stroke=None
)
```

[19]: alt.Chart(…)

5.8 Photo Quantities vs Review Scores

From the charts and details below we can discover that the review score 0 usually has less product photos and shorter description, which could probably because the customers are misled by the fraud or imcomplete product photos and information.

[20]:
```
df.groupby('review_score').agg('product_photos_qty').describe()
```

[20]:

review_score	count	mean	std	min	25%	50%	75%	max
0	12453.0	2.166787	1.723093	1.0	1.0	1.0	3.0	19.0
1	82661.0	2.263208	1.748361	1.0	1.0	2.0	3.0	20.0

[21]:
```
df.groupby('review_score').agg('product_description_lenght').describe()
```

[21]:

review_score	count	mean	std	min	25%	50%	75%	max
0	12453.0	771.465510	643.626381	20.0	340.0	590.0	982.0	3950.0
1	82661.0	796.542457	654.953569	4.0	352.0	610.0	999.0	3992.0

[22]:
```
chart1=alt.Chart(df).mark_line().encode(
    x = alt.X('product_photos_qty:Q',axis = alt.Axis(title = "product photos␣
↪quantity")),
    y=alt.Y('count()',axis = alt.Axis(title = "count")),
    color='review_score:N'
).properties(
    title='Review Score vs Photo Quantities'
).properties(
    width=120
)
chart2=alt.Chart(df).mark_line().encode(
```

```
    x=alt.X('product_description_lenght:Q',axis = alt.Axis(title = "product␣
↪description length(word)")),
    y='count()',
    color='review_score:N'
).properties(
    title='Review Score vs Description Length'
).properties(
    width=120
)
chart1|chart2
```

[22]: alt.HConcatChart(…)

5.9 Payment values

Now, let's see if there is some relationship between Payment Values(Product Price) and Review Scores.

[23]:
```
alt.Chart(df).transform_density(
    'price',
    as_=['price', 'density'],
    groupby=['review_score']
).mark_area(orient='horizontal').encode(
    y=alt.Y('price',axis = alt.Axis(title = "price(Brazilian Real)")),
    color='review_score:N',
    x=alt.X(
        'density:Q',
        stack='center',
        impute=None,
        title=None,
        axis=alt.Axis(labels=False, values=[0],grid=False, ticks=True),
    ),
    column=alt.Column(
        'review_score:N',
        header=alt.Header(
            titleOrient='bottom',
            labelOrient='bottom',
            labelPadding=0,
        ),
    )
).properties(
    title='Review Score vs Price (Brazilian Real)'
).properties(
    width=100
).configure_facet(
    spacing=0
```

```
).configure_view(
    stroke=None
)
```

[23]: alt.Chart(…)

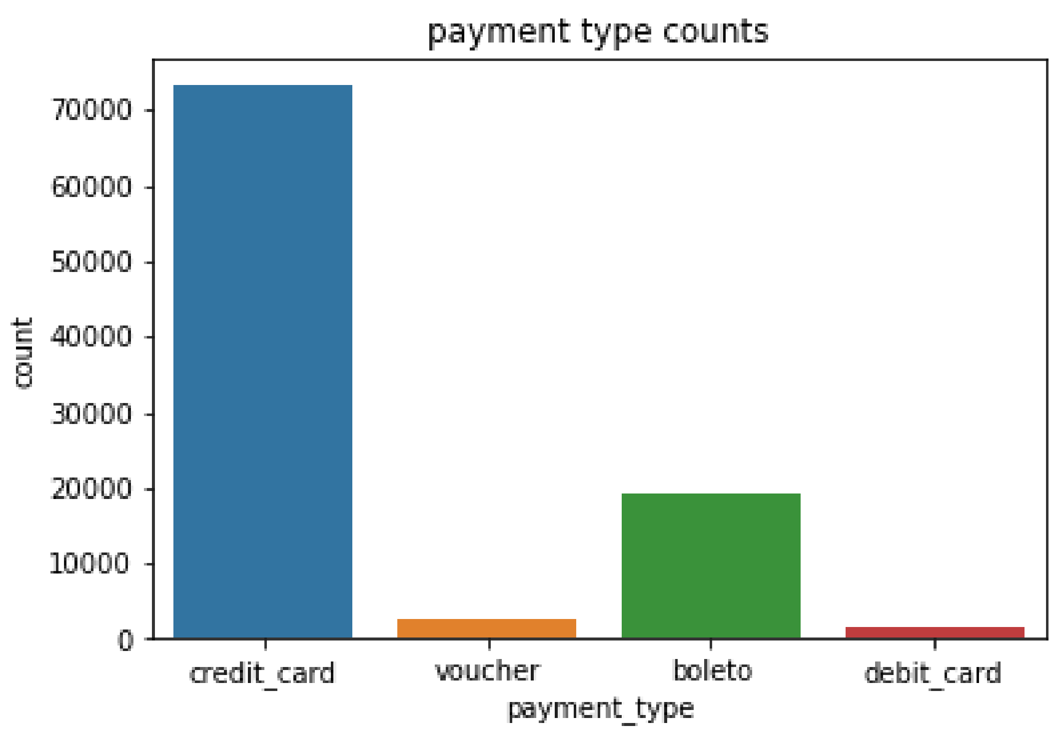
From the chart above, we could see that when the price is low, there are more reviews, resulting in more good reviews and bad reviews. While with higher prices, the amount of both kinds of reviews are much fewer. However, when the price is very low, there is much more bad reviews than good reviews. This is possibly because cheaper goods might have poorer quality which could lead to lower review scores.

5.10 Payment type (Bonus)

from this plot we can find out the most frequent payment type is credit card. The second one is boleto, which is a growing payment method popular in Latin America. Whereas debit cards and voucher are rarely used.

[24]:
```
sns.countplot(x='payment_type',data=customers).set_title('payment type counts')
```

[24]: Text(0.5, 1.0, 'payment type counts')

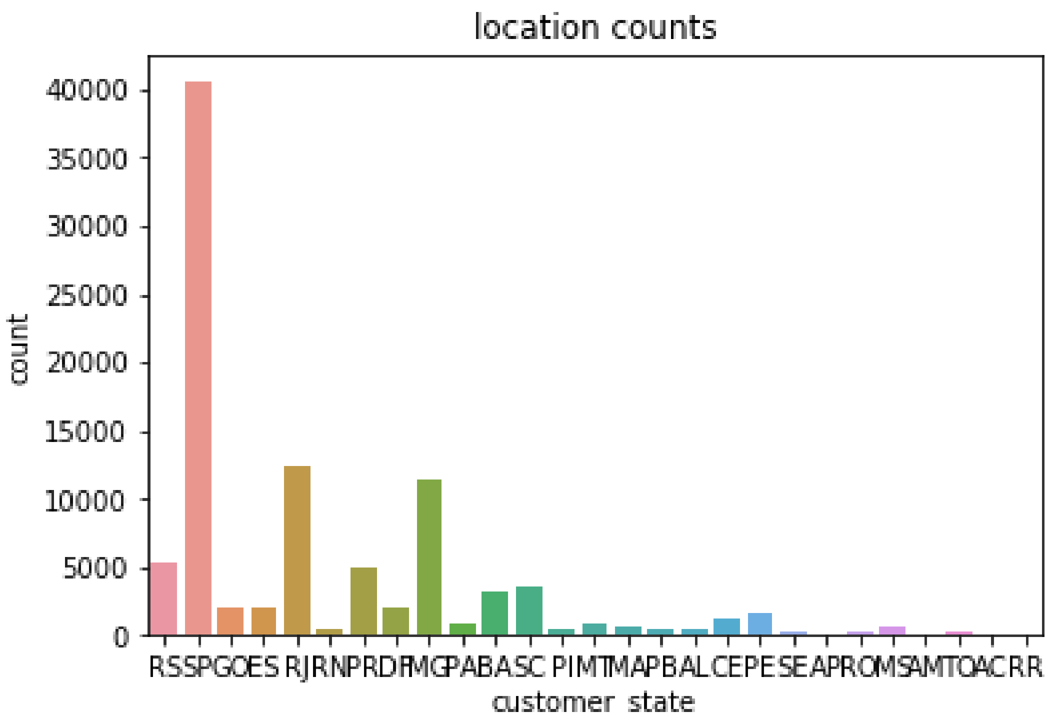


5.11 Location (Bonus)

From the plot below we can discover that the SP (São Paulo) has the most customers. Because São Paulo is the municipality with large population (world's 12th largest city proper by population). Other two state RJ (Rio, lots of carnivals), and MG (a large inland colonial state before, prosperity phase of economy)

[26]:
```
sns.countplot(x='customer_state',data=customers).set_title('location counts')
```

[26]:
```
Text(0.5, 1.0, 'location counts')
```

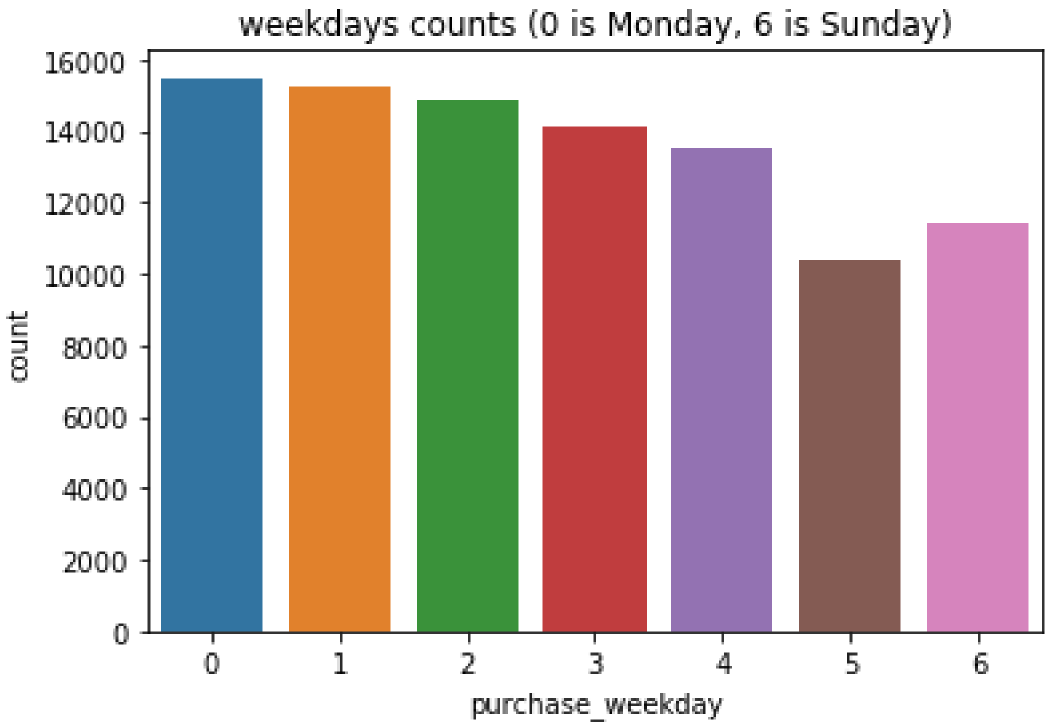


5.12 Weekdays (Bonus)

The Monday has the most customers. Based on our experience, it is not hard to deduce that people like to shopping at the beginning of the week and expect to recieve them in the weekends.

[27]:
```
sns.countplot(x='purchase_weekday',data=df).set_title('weekdays counts (0 is␣
↪Monday, 6 is Sunday)')
```

[27]:
```
Text(0.5, 1.0, 'weekdays counts (0 is Monday, 6 is Sunday)')
```



6 Sampling, partitioning and balancing the data

Due to the extremely large sample, which is more than 90000 rows, it will take lots of time and memory to analyze all of them. Also, the positive reviews are significantly more than the negative reviews (people are nice). We will sample and partition the dataset into the classfication columns and target columns before we put it into our model.

6.1 sampling the unbalanced (original ratio) dataset

The reason we do this is to compare these two datasets in the machine learning part. Both will have different outcomes and we will analyze the advantages of choosing the balanced dataset or not.

[28]:
```
import random
n_sample = 4000
df_unbalanced = df.sample(n_sample)
df_target_unbalanced=df_unbalanced['review_score']
df_classfication_unbalanced=df_unbalanced.drop(columns=['review_score'])
df_classfication_unbalanced
```

[28]:
```
        price  freight_value  product_description_lenght  product_photos_qty  \
26723  127.00           9.08                      1120.0                 1.0
53299   26.90          15.11                       801.0                 3.0
49241   99.99          25.54                       513.0                 1.0
51388   49.90          15.97                       184.0                 1.0
27764   65.00          16.89                       722.0                 4.0
…        …              …                          …                    …
42475  109.90          14.52                       826.0                 1.0
55840  149.90          18.93                       408.0                 1.0
32927  143.90          31.76                       503.0                 1.0
50196   18.90          13.47                       343.0                 1.0
88544   21.99          12.79                       662.0                 1.0

       timediff  deliver_time  purchase_weekday  Late
26723       -12             2                 2     0
53299       -12            10                 0     0
49241        -4            19                 6     0
51388        -9            16                 6     0
27764       -24             9                 2     0
…          …            …               …   …
42475       -20             4                 6     0
55840       -16            11                 5     0
32927        11            31                 2     1
50196       -14            17                 0     0
88544       -12             7                 4     0

[4000 rows x 8 columns]
```

6.2 sampling the balanced dataset

we will take 2000 random samples. 1000 from those review score higher than 2 (positive reviews) and 1000 from those lower or equal than 2 (negative reviews) So that it will make the sample set more balanced and easy to analyze.

[29]:
```
n_sample = 2000
df = df.dropna()
df1 = df[df['review_score']==1]
df2 = df[df['review_score']==0]
df1 = df1.sample(n_sample)
df2 = df2.sample(n_sample)
df_balanced = pd.concat([df1,df2],axis=0)
df_target_balanced=df_balanced['review_score']
df_classfication_balanced=df_balanced.drop(columns=['review_score'])
##df_classfication
df_classfication_balanced
```

[29]:

```
         price  freight_value  product_description_lenght  product_photos_qty  \
27631    49.90           8.27                       605.0                 2.0
73242    49.90          19.32                        77.0                 2.0
29837   110.00          19.01                       235.0                 2.0
33384    84.99          16.79                      2164.0                 5.0
61977    41.90          35.48                       555.0                 1.0
…          …              …                          …                   …
33138    98.44          13.81                       385.0                 1.0
57036   129.00          12.40                       471.0                 5.0
61899    69.90          10.75                      1176.0                 1.0
6757     29.90          13.37                        68.0                 2.0
60765   119.99          14.93                       421.0                 1.0

       timediff  deliver_time  purchase_weekday  Late
27631       -19             2                 1     0
73242        -6            30                 3     0
29837       -17             8                 0     0
33384       -10             6                 1     0
61977        -7            14                 4     0
…          …            …                …    …
33138       -20             5                 3     0
57036       -19             4                 0     0
61899       -14             9                 5     0
6757          3            23                 2     1
60765        -1            13                 4     0

[4000 rows x 8 columns]
```

6.3 Standardization

Standardize the classfication dataframe so that all the columns will have the same weight during the machine learning process.

[30]:

```python
#function that standardize the input dataframe
def standardization(data):
    mu = np.mean(data, axis=0)
    sigma = np.std(data, axis=0)
    return (data - mu) / sigma
```

[31]:

```python
df_classfication_balanced = standardization(df_classfication_balanced)
df_classfication_unbalanced = standardization(df_classfication_unbalanced)
```

7 Examine our questions of interest with the help of Machine Learning (Sklearn)

The reason why we use logistic regression is that the review score is categorical variables rather than continues numerical variables. Thus, we need to apply the logistic regression to examine how well those classification columns (df_classification) could predict the target column (review score). The packages are used from sklearn.

7.1 logistic regression about the unbalanced sample

The accuracy for the unbalanced data is 0.88 (weighted avg is 0.88). It is significantly higher than the balanced sample(see later). Why would this happen?

- The original ratio (a higher proportion of those give positive review scores) is a better fit to the model.
- The balanced data amplifies the incertainty of those give bad review scores.
- We assumn the model fits the positive review scores better. Because the pattens and relations in the good review scores are easy to find.

[32]:
```python
from sklearn.linear_model import LogisticRegression
logmodel = LogisticRegression()
logmodel_balanced = LogisticRegression()
```

[33]:
```python
from sklearn.model_selection import train_test_split
X = df_classfication_unbalanced
y = df_target_unbalanced
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,␣
↪random_state=42)
X_train.head(4)
```

[33]:
```
          price  freight_value  product_description_lenght  \
18945 -0.214205       0.476815                   -0.524089
8720  -0.367484      -0.488710                    0.127860
12491  1.116071      -0.271973                   -0.543754
74435 -0.474045      -0.839929                   -0.173156

       product_photos_qty  timediff  deliver_time  purchase_weekday      Late
18945           -0.146740 -1.523437     -0.482181          0.099363 -0.306256
8720            -0.723887  2.723783      2.690061         -1.417630  3.265242
12491           -0.146740 -0.107697     -0.482181         -0.911966 -0.306256
74435           -0.146740  0.701297     -0.372793         -0.406301 -0.306256
```

[34]:
```python
logmodel.fit(X_train,y_train)
```

[34]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
```

```
                    random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                    warm_start=False)
```

[35]:
```
pred = logmodel.predict(X_test)
```

[36]:
```
from sklearn.metrics import classification_report
print(classification_report(y_test,pred))
```

```
              precision    recall  f1-score   support

           0       0.77      0.20      0.31       168
           1       0.89      0.99      0.94      1152

    accuracy                           0.89      1320
   macro avg       0.83      0.59      0.63      1320
weighted avg       0.88      0.89      0.86      1320
```

7.2 logistic regression about the balanced sample

From the results above, we discover that the unbalanced data is a nice fit overall. However, we notice that the 0 review score has a very low accuracy (which is 0.45). After the weighted, its accuracy is underestimated. This is the main issue we discover. Thus, we will use the balanced sample below to re-check the model's real accuracy.

The reason why we use logistic regression is that the review score is a categorical variable rather than continues numerical variables. Thus, we need to apply the logistic regression to examine how well those classification columns (df_classification) could predict the target column (review score). The packages are used from Sklearn.

After our first attempt to conduct logistic regression using the unbalanced sample, we were actually pretty happy with the result at first, because the accuracy for the unbalanced data was 0.89 (weighted avg is 0.87) (See Table 1). It is actually significantly higher than the balanced sample. This might because of the facts that the original ratio (a higher proportion of the give positive review scores) is a better fit to the model and that the balanced data amplifies the uncertainty of those give bad review scores.

[37]:
```
#do the train test split to our standerdized, balanced classification dataset
X_balanced = df_classfication_balanced
y_balanced = df_target_balanced

X_train_balanced, X_test_balanced, y_train_balanced, y_test_balanced =␣
 ↪train_test_split(X_balanced, y_balanced, test_size=0.33, random_state=42)
X_train_balanced.head(4)
```

[37]:
```
          price  freight_value  product_description_lenght  \
62312 -0.551194      -0.428982                    0.146991
13927  4.928064       0.148014                    1.018577
```

```
94524 -0.584099      -0.682886                          -0.819900
55915 -0.111339      -0.613701                          -0.606231

       product_photos_qty  timediff  deliver_time  purchase_weekday      Late
62312           -0.112172  0.290405     -0.067396         -0.386718 -0.486681
13927            1.680191 -0.519393     -0.665640         -0.386718 -0.486681
94524           -0.112172 -0.961101     -1.039542         -0.895892 -0.486681
55915            2.875099  1.100202     -0.142176         -0.895892  2.054734
```

[38]:

```python
logmodel.fit(X_train_balanced,y_train_balanced)
pred_balanced = logmodel.predict(X_test_balanced)
```

[39]:

```python
print(classification_report(y_test_balanced,pred_balanced))
```

```
              precision    recall  f1-score   support

           0       0.77      0.42      0.54       632
           1       0.62      0.89      0.73       688

    accuracy                           0.66      1320
   macro avg       0.70      0.65      0.64      1320
weighted avg       0.70      0.66      0.64      1320
```

From the results above, in fact, the accuracy of our logistic model is very low. We observed the possible reasons:

- There are lots of factors such as the weekdays that are completely uncorrelated to the review score.
- The target column itself is hard to predict. Since the review scores are too subjective. And there are lots of people never leave their review scores (the systems will automatically assign the score 5).

In all ,those variables may not fit the models well since each individuals have different situation. Thus, applying those objective variables to predict the subjective variable may not work. We will then try to use the review comments (subjective variable) to predict the review score

7.3 NLP For Review Comments on the balanced sample

The review comments might be a good variable to predict the review score

Build the NLP_df first, applying all the process steps above and only take the review_comment_message as the classification column.

Why this model is good for us to predict the review score?

We will use the Natural Language Processing to analyze the correlation of the strings in review comment and the review scores. The reason we use NLP is it did a good job in translating the human readable language into computer readable language, which is from the strings in review comments to TF-IDF scores. We will then apply the Naive Bayes to classify in the training model.

Naive Bayes applies similar method to predict the probability of different class based on various attributes. Its algorithm is mostly used in text classification. Then we will have our trained model that predicts the review scores.

[40]:
```
NLP_df=customers[['review_score','review_comment_message']]
NLP_df.dropna()
NLP_df['review_score']=(NLP_df['review_score']>2)+0
NLP_df.head()
```

```
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:3:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so we can avoid doing imports
until
```

[40]:

	review_score	review_comment_message
0	1	NaN
1	1	NaN
2	0	Meu produto não foi entregue até o momento!
3	1	Produto bom, mas o pegador da tampa é de plást…
4	1	Produto igual ao anunciado, de excelente quali…

[41]:
```
import random
n_sample = 2000
df1 = NLP_df[NLP_df['review_score']==1]
df2 = NLP_df[NLP_df['review_score']==0]
df1 = df1.sample(n_sample)
df2 = df2.sample(n_sample)
df_balanced = pd.concat([df1,df2],axis=0)
y=df_balanced['review_score']
X=df_balanced['review_comment_message']
```

Import CountVectorizer, TfidfTransformer, MultinomialNB to create a pipeline that can process those comments

[42]:
```
from sklearn.feature_extraction.text import  TfidfTransformer
from sklearn.pipeline import Pipeline
from sklearn.naive_bayes import MultinomialNB
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split

pipeline = Pipeline([
    ('bow', CountVectorizer()),  # strings to token integer counts
    ('tfidf', TfidfTransformer()),  # integer counts to weighted TF-IDF scores
```

```
    ('classifier', MultinomialNB()),  # train on TF-IDF vectors w/ Naive Bayes␣
 ↪classifier
])
```

[43]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y,test_size=0.
 ↪3,random_state=101)
```

Fit the models and make the predictions soley based on the review comment

[44]:
```
pipeline.fit(X_train.values.astype('U'),y_train.values.astype('U'))
```

[44]:
```
Pipeline(memory=None,
         steps=[('bow',
                 CountVectorizer(analyzer='word', binary=False,
                                 decode_error='strict',
                                 dtype=<class 'numpy.int64'>, encoding='utf-8',
                                 input='content', lowercase=True, max_df=1.0,
                                 max_features=None, min_df=1,
                                 ngram_range=(1, 1), preprocessor=None,
                                 stop_words=None, strip_accents=None,
                                 token_pattern='(?u)\\b\\w\\w+\\b',
                                 tokenizer=None, vocabulary=None)),
                ('tfidf',
                 TfidfTransformer(norm='l2', smooth_idf=True,
                                  sublinear_tf=False, use_idf=True)),
                ('classifier',
                 MultinomialNB(alpha=1.0, class_prior=None, fit_prior=True))],
         verbose=False)
```

[45]:
```
predictions = pipeline.predict(X_test.astype('U'))
```

[46]:
```
from sklearn.metrics import confusion_matrix,classification_report

print(confusion_matrix(y_test.astype('U'),predictions))
print(classification_report(y_test.astype('U'),predictions))
```

```
[[436 167]
 [ 61 536]]
              precision    recall  f1-score   support

           0       0.88      0.72      0.79       603
           1       0.76      0.90      0.82       597

    accuracy                           0.81      1200
   macro avg       0.82      0.81      0.81      1200
weighted avg       0.82      0.81      0.81      1200
```

The results are surprising! We get relatively better accuracy in both bad review scores and good review scores. The reason is probably that the review comments have a closer relationship to the review scores. It is obvious that people would leave their unsatisfied comment if they receive a bad shopping experience. On the other side, if people forgot to leave the positive comment when they receive a good shopping experience, the system will automatically assign the default good review comment to the eshop. That's why on both sides the NLP works well.

8 Discussion and Future Works (about 200 words)

In this section you should summarize your findings based on your final model in clearly understandable, non-statistical terms. What is the main message produced by your analysis? There may also be additional questions that arise, problems you encounter, or possible extensions of your analysis that could be addressed here.

Include any final comments and thoughts about your project. For example, do you trust your results? How general are your results, to what situations do they apply? Add any other comments that are relevant.

- Did you achieve your goal? If not, why? What were some challenges and lessons you learned from them?
- What were your primary conclusions and how do your results support these conclusions?
- What extensions or future work would you recommend?
- Key Results

and Conclusion We partially achieved our goal to analyze the customers' shopping behaviors (feedbacks) based on those influential factors. After cleaning, processing, and re-designing our features and target data frame, we addressed our questions of interest. The delivery length and late delivery have a clear negative correlation to the review scores. It informs the e-sellers that they should improve their shipping service and make sure to deliver the products as soon as possible. Another feature that exerts a huge impact on customers' satisfaction level is photo quantity and description. After analyzing the dataset, we found that products with more photos and descriptions tend to have better reviews. This indicates that sellers should provide more product photos and product descriptions in order to fully display the product and avoid misleading potential customers. The payment amount also affects the overall reviews because cheaper goods usually have worse quality controls, leading to lower review scores. However, we should see this problem critically. For example, for shops whose target customers have high consumption, they could consider only Table 3: Random Forrest Results Table 4: NLP Results producing good quality stuffs and reduce the production and selling of low-price-poor-quality product in order to maintain the shop's reputation. Yet for shops whose target customer have low consumption level, they usually have larger audience, so they could still make profits even if there might be a risk of receiving more bad reviews.

We also find that if e-shops want to improve they service, they should pay attention to review comments prior to all other factors discussed above. There is a tight connection between review comment and review score. According to our model fit results, review comment could be an important predictor to review score, because it has a high accuracy, especially to bad reviews. Eshops should carefully read through the comments and improve what is mentioned in the bad comments.

- Discussion

At the beginning, we spent a lot of time designing and cleaning our ideal dataset due to the original dataset's complexity. Throughout our approaching, we made some struggles choosing the right models to fit our datasets. We started with logistic regression. Due to the fact that linear regression only handles continuous numerical variables and that multi-categorical logistic regression will amplify the weight of the review score 5 which is a large proportion of the whole dataset, sorting out the review scores into two categories (positive and negative) provides a better fit to the logistic regression. Since our results were not ideal, we tried the Random Forest model, which ensures the outliers and large variance samples to have smaller effect on the classification process by making collective decisions by each decision tree. After getting another frustrating result, we realized that the predictors that we used might be not so proper. Review score is very subjective, yet the factors that we previously used were all pretty objective, hence there might be a significant inaccuracy in the prediction. Therefore, we introduced another variable: review comments, which is also a subjective factor. After applying Natural Language Processing, we finally got a satisfying model to predict the review score.

- Future Work

In the future, we will collect more datasets from different regions since Asian E-commerce usually has different structures from Brazil's. We will also try to analyze their review comments with the help of Natural Language Processing. Also, when preparing the random sampling, we will adjust different categories' ratio since balanced and unbalanced dataset often has different outcomes in the models.