CASCADE CUP

ROUND-3

DATA ANALYSIS REPORT

BY TEAM :
Data_dembele

ALYAL SAMAL
ASHIS KUMAR PARIDA

Introduction and Approach

The Dataset consists of records of **cancelled deliveries** with order and rider details on which we had to create a model that can **predict rider-driven cancellation** in advance (i.e. before getting marked as cancelled or delivered).

For this round, we are provided with two Datasets, i.e. Train Data and Call Data. In the **call data**, there are **4 columns** in which there are **many duplicate rows** and the specific order id also repeats itself many times.

In this call data, only the **reason_text** column contains **6.58% missing** rows.

So, we decided to **merge** the train data and call data together on the **order_id** and then **rider_id** column to gain how the columns were related. After that, we have derived some analyses from the combined Data.

This Dataset is clearly **imbalanced** with respect to the cancelled column which is clearly interpretable from **Fig-1**

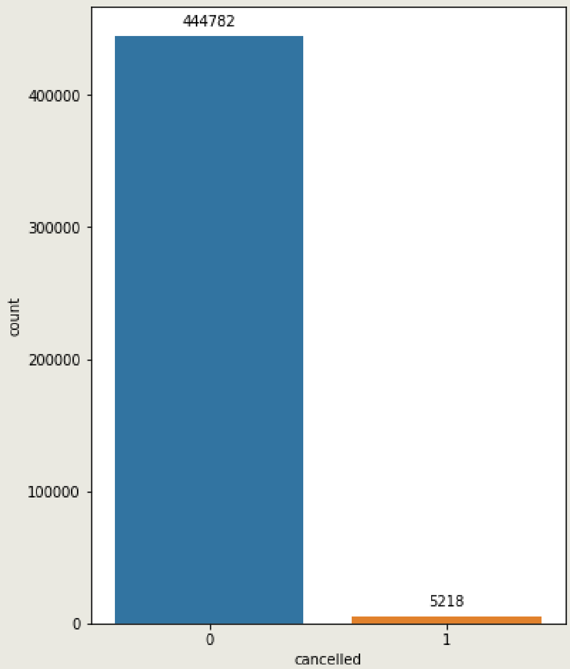


Fig-1

The main task of this competition was to **maximize the ROC-AUC** Score, so our focus was to **improve Recall** by **decreasing False Negative** so that the **area under the ROC curve increases**. Also, while making the predictions, we **predicted the probabilities** and not the class as it is more flexible to predict the probabilities for each class instead. The reason for this is to provide the capability to choose and even calibrate the threshold for how to interpret the predicted probabilities.

We did many univariate and multivariate an each and every column of the data, and then applied various models and finalised on stacking and blending of best performing models..

Features and their infos

Numerical Features

- first_mile_distance
- last_mile_distance
- allotted_orders
- delivered_orders
- undelivered_orders
- lifetime_order_count
- session_time

Datetime Columns

- order_time
- order_date
- allot_time
- accept_time
- pickup_time
- delivered_time
- cancelled_time

Categorical Features

- order_id
- rider_id
- cancelled [Target Column]
- reassignment_method
- reassignment_reason
- reassigned_order
- user_type
- reason_text

Percentage (%) wise missing Data

- accept_time ~ 1 %
- pickup_time ~ 1 %
- delivered_time ~ 1 %
- allotted_orders ~ 4 %
- delivered_orders ~ 4 %
- undelivered_orders ~ 4 %
- lifetime_order_count ~ 1 %
- session_time ~ 1 %
- reassignment_method ~ 97%
- reassignment_reason ~ 97 %
- reason_text ~ 35.5 %
- user_type ~ 35.5 %
- cancelled_time~ 99 %

Data Cleaning

We have observed that this dataset contains **too many nan values** in specific columns. So, we decided to deal with them with different approaches.

On columns like alloted_orders, delivered_orders, undelivered_orders we **replaced the NaN values with 0** because we have **different order_id s for the same rider_id**, it **won,t make sense** if we impute the nan values with the **mean** of the whole column in the data and also there may not be the same groups of rider_id's **frequent occurrence**.

On the session_time column, we check if the rider_id is repeated for that row having no nans, If that is the case, then we fill the nan values with respective means, if not, then we replace them with 0.

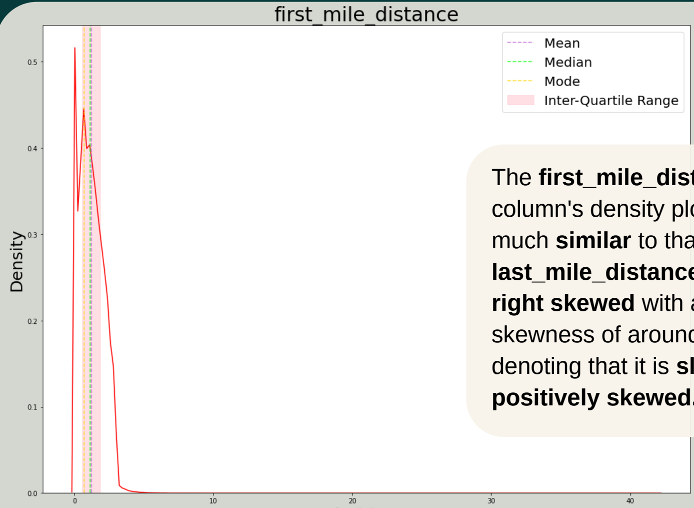
Feature Transformation

We transformed many skewed columns to approximate their distributions to have a **normal distribution**. (Central Limit Theorem). So, we applied **quantile transformations** to those columns, and below are the results of the **KDE plots**.

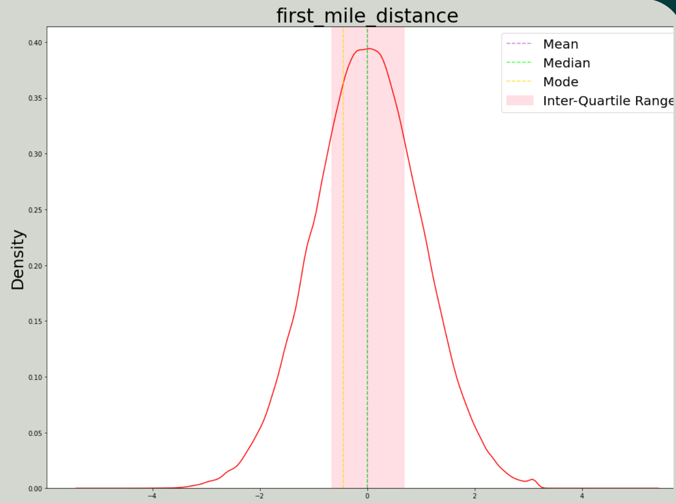
KDE PLOTS

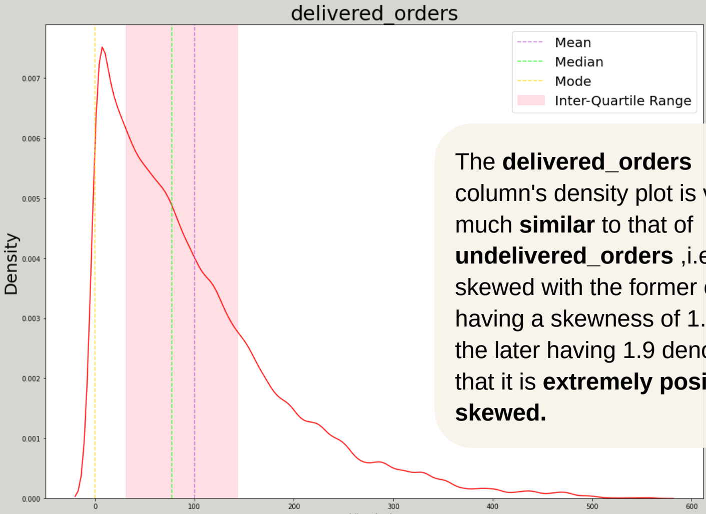
Before Transformation

After Transformation

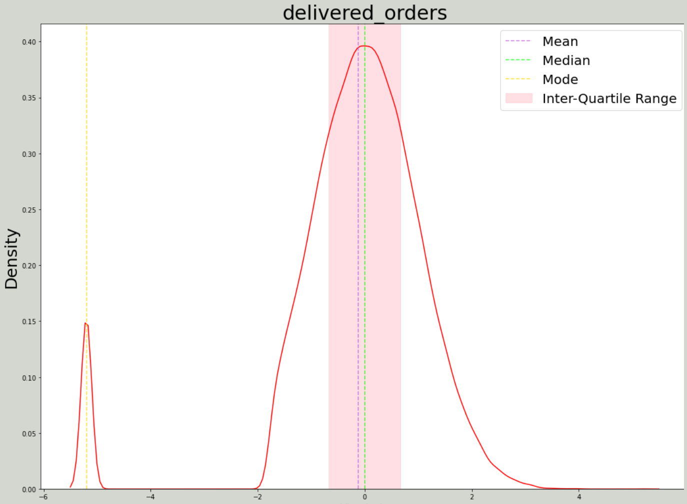


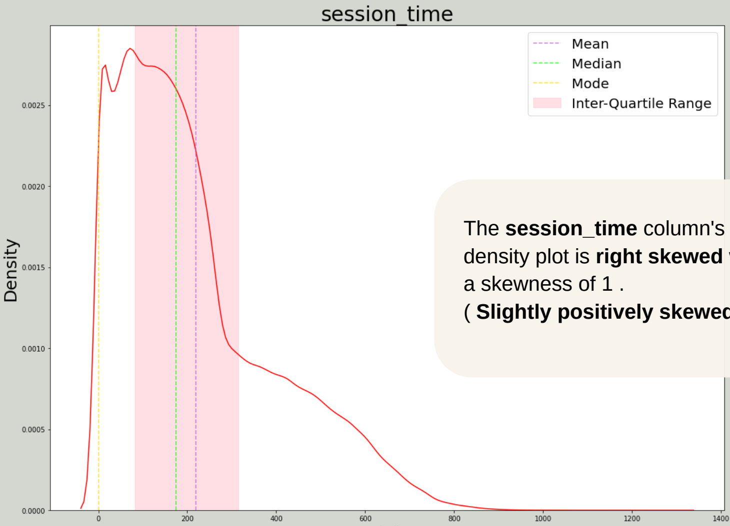
The **first_mile_distance** column's density plot is very much **similar** to that of **last_mile_distance** ,i.e. **right skewed** with a skewness of around 0.80 denoting that it is **slightly positively skewed.**



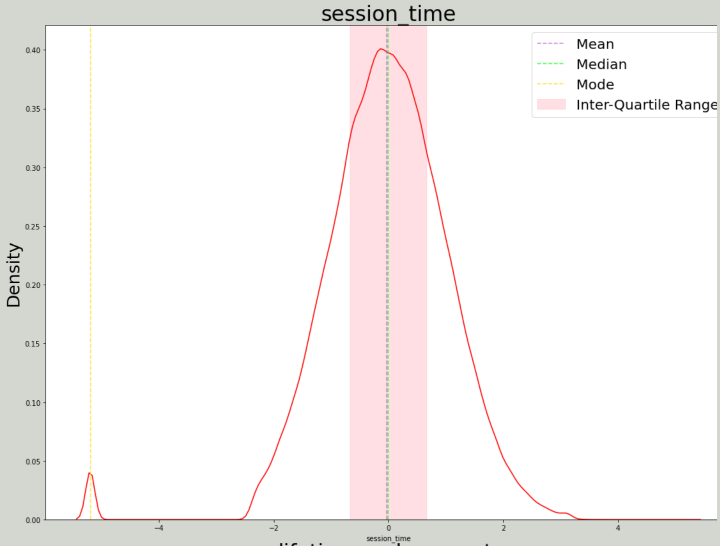


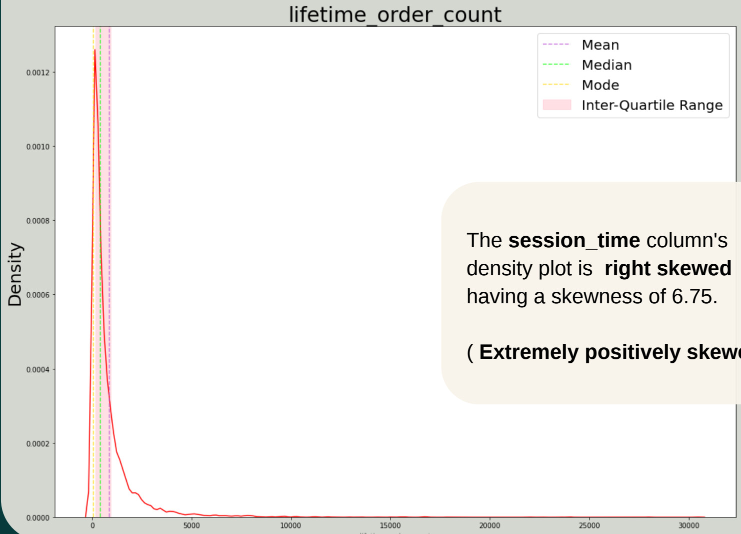
The **delivered_orders** column's density plot is very much **similar** to that of **undelivered_orders** ,i.e. right skewed with the former one having a skewness of 1.4 and the later having 1.9 denoting that it is **extremely positively skewed.**





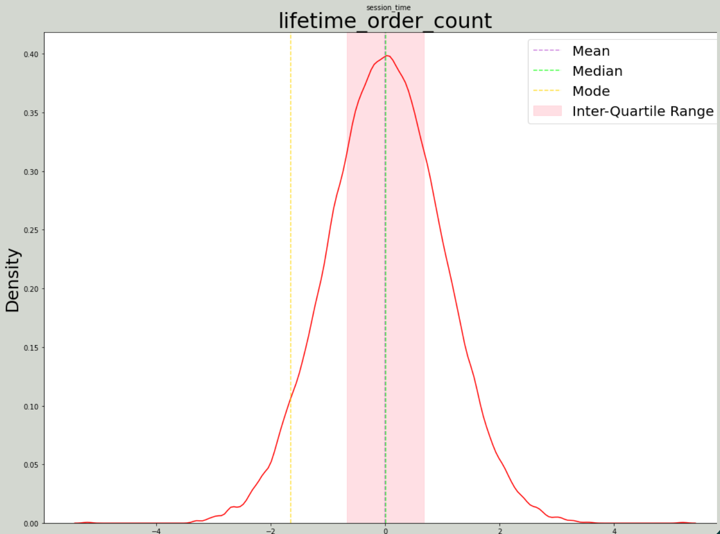
The **session_time** column's density plot is **right skewed** with a skewness of 1 .
(**Slightly positively skewed**)





The **session_time** column's density plot is **right skewed** having a skewness of 6.75.

(**Extremely positively skewed**)



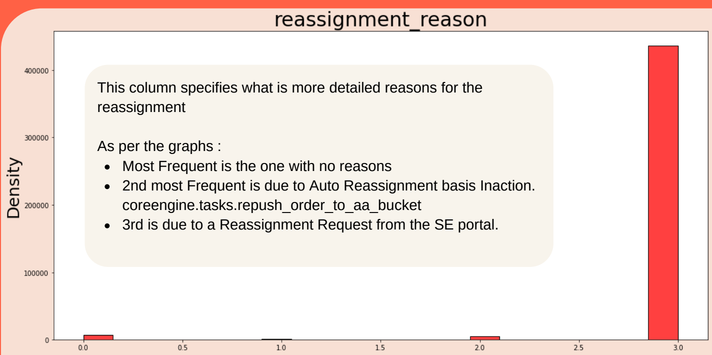
Feature Engineering

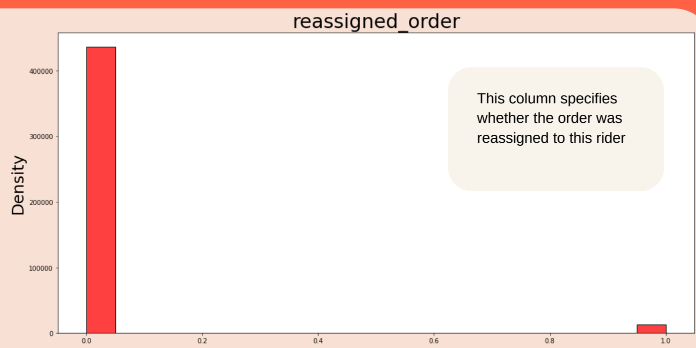
We have engineered some **new features** in accordance with all the observations we have made in the data like **order_hour**, **order_minute**, **order_day_of_week** and **Is_Weekend** from order day itself.

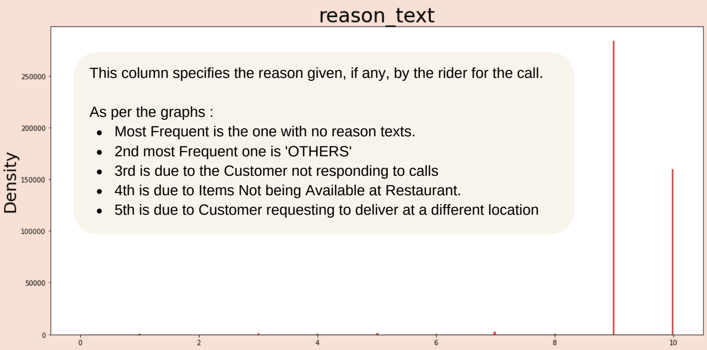
We have also have made features from **difference** of (**order_time** and **allot_time**) and (**allot_time** and **accept_time**) respectively in minutes.

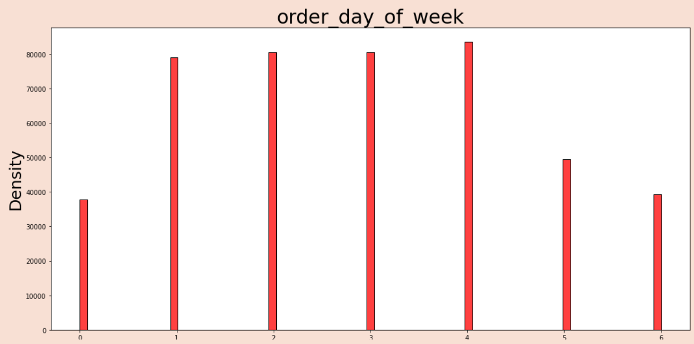
Similarly, we have also engineered an **indicator feature** for **diff_accept_allot_minute** column indicating, if it is a nan or not ?

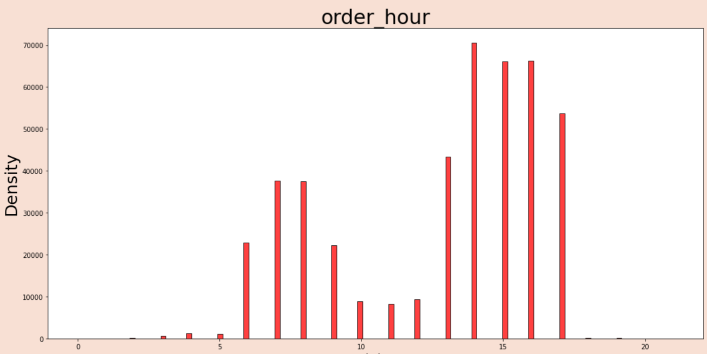
Some Histplots of Categorical Features

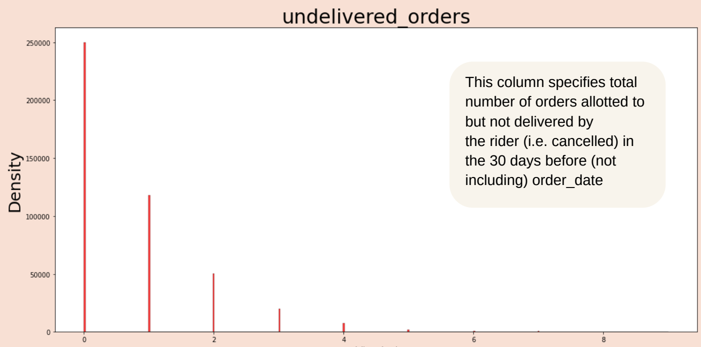


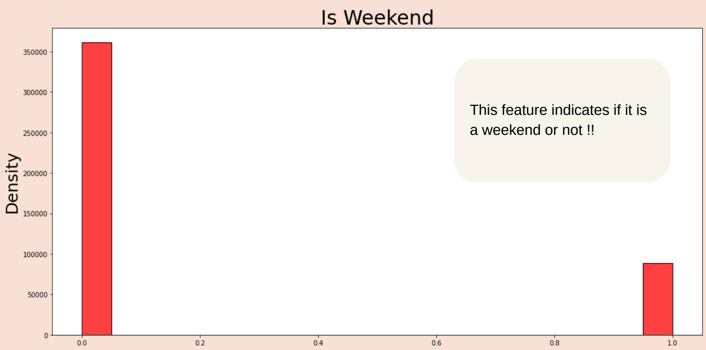


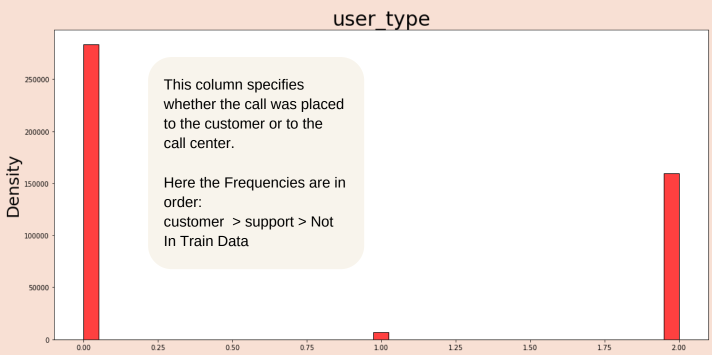












Interesting Graphs Drawn From Data

We wanted to observe if the cancelled orders are mostly by **newly recruited guys** or the ones who are **senior** to them. **But**, there were **no such features** to **indicate whether** they were newly recruited or not. So we made a **new feature solely for analysing this data** which was basically the **difference** between the **lifetime_order_count** and **allotted_orders** column. This column will **never have negative values**. And if the value is very less then we get a **rough estimate** that he/she is a newbie, **ignoring certain cases** like if someone has taken a **long break.**

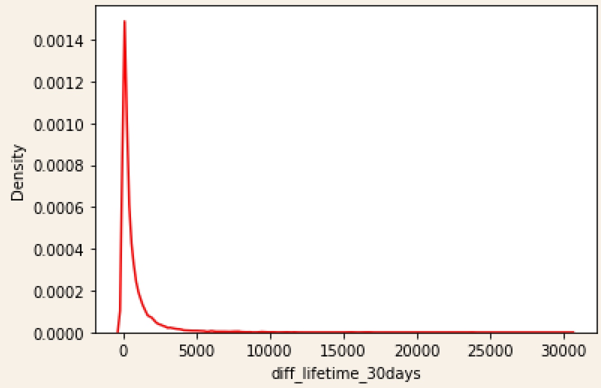


Fig - KDE-Plot of diff_lifetime_30days

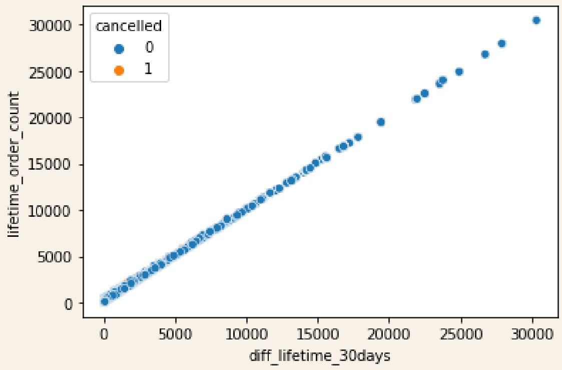


Fig - Scatter-Plot of diff_lifetime_30days and lifetime_order_count

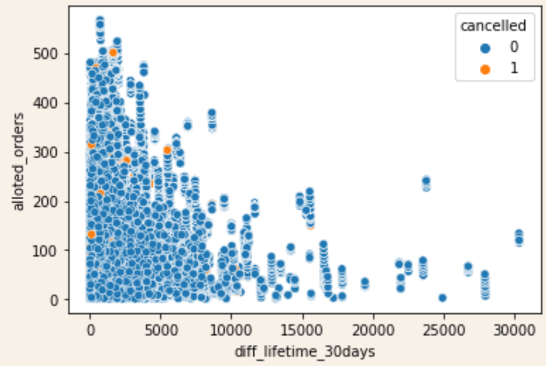


Fig - Scatter-Plot of diff_lifetime_30days and alloted_orders

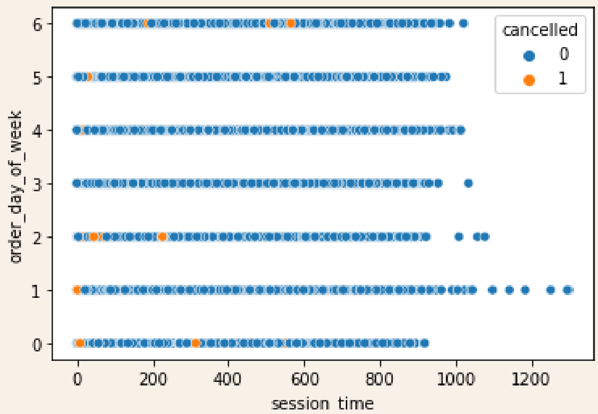


Fig - Scatter-Plot of order_day_of_week and session_time

This is a scatter plot drawn between **order_day_of_week** and **session_time** as we wanted to analyse **how much** do riders **work** on **different days** of the week.

Turns out, **on Tuesday, riders work more** than they do on other days. Also on **Sunday**, **Wednesday** and **Monday**, there are **more no. of cancelled orders.**

*Here 0 means Monday and 6 means Sunday.

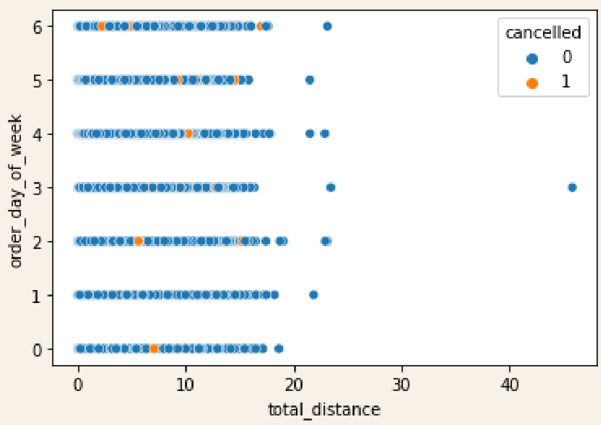


Fig - Scatter-Plot of order_day_of_week and total_distance

First, we created the **total_distance** column by adding **first_mile_distance** and **last_mile_distance**.

Then we created a **scatter plot** drawn between **order_day_of_week** and **total_distance** as we wanted to analyse how much do riders **travel** in total on different days of the week.

Turns out, on **Wednesday** and **Sunday**, riders travel more than they do on other days.

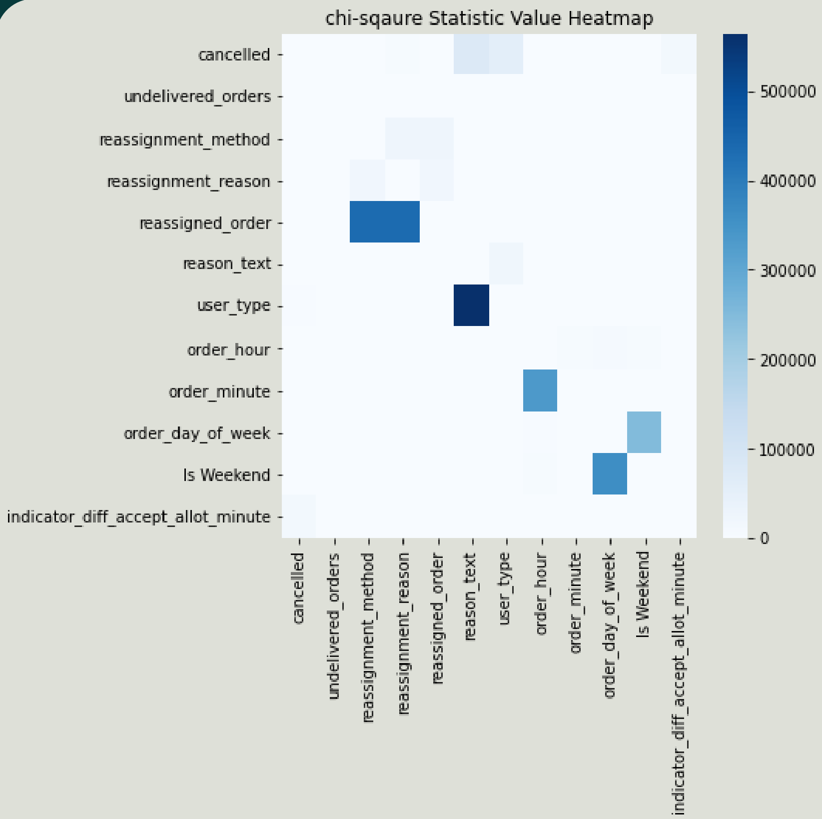
*Here 0 means Monday and 6 means Sunday.

Chi-Square Test Graphs

- The **Chi-Square test** is a **statistical test** that is used to find out the difference between the observed and the expected data we can also use this test to find the **correlation** between two **categorical variables** in our data. The purpose of this test is to determine if the difference between 2 categorical variables is **due to chance**, or if it is **due to a relationship** between them.

. In **our Data**, we have **many categorical features** such as **reassignment_method**, **reassignment_reason**, **user_type**, **reason_text**, **Is_weekend**, **order_hour**, **order_day_of_week** etc on which we want to conduct the chi-square test.
- Here, we have assumed the **alpha** to be **0.05**.

- Our **H0 (Null Hypothesis)** is that the variables to be compared are **independent**. and **H1 (Alternate Hypothesis)** is that variables are **dependent**.

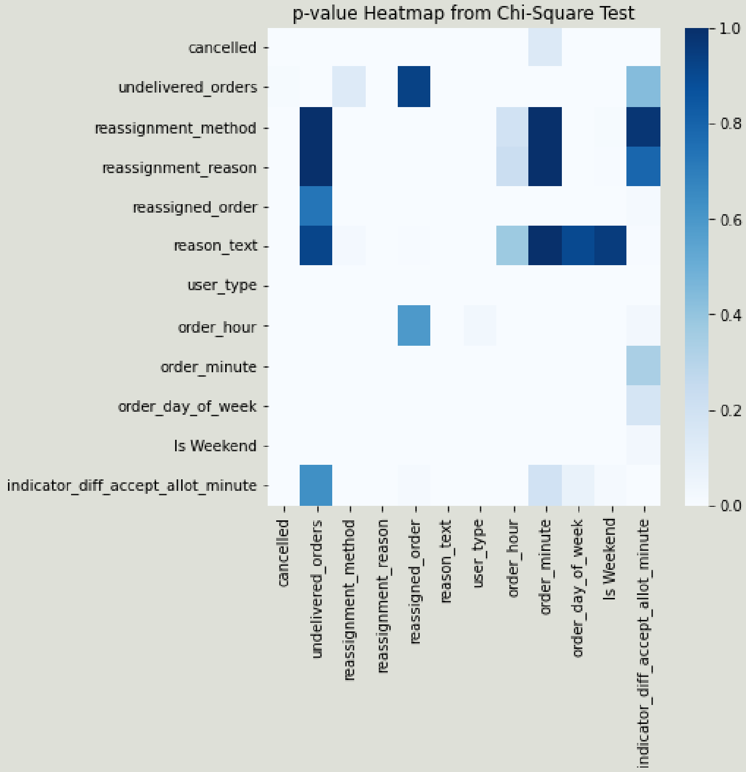


Simply, the **more** these values **diverge** from each other, the **higher** the **chi square score**, the more likely it is to be significant, and the more likely it is we'll **reject** the **null hypothesis** and conclude the variables are associated with each other.

We have done this to get a rough idea of the dependencies.

we have observed very **peculiar results**,i.e. **almost all** of the categorical features are **dependent on each other**.
This is something that **needs treatment** according to the model we decide to use.

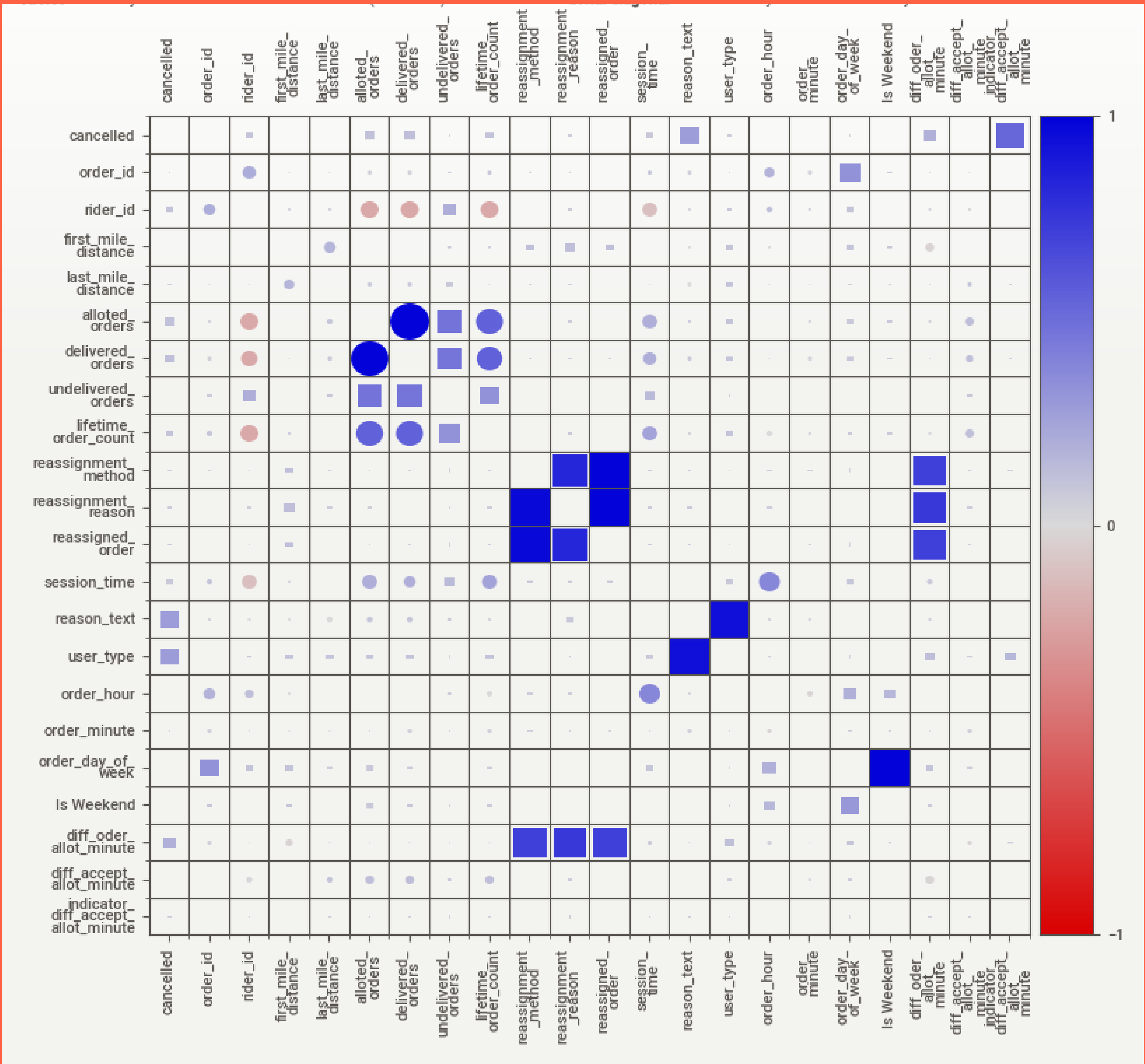
If any of the column's **dependency** with the **target** column is **greater** than its dependency with another column, then we consider the latter one, because we don't want the **interference** of that variable **while determining** the **target** column value



Pearson Correlations

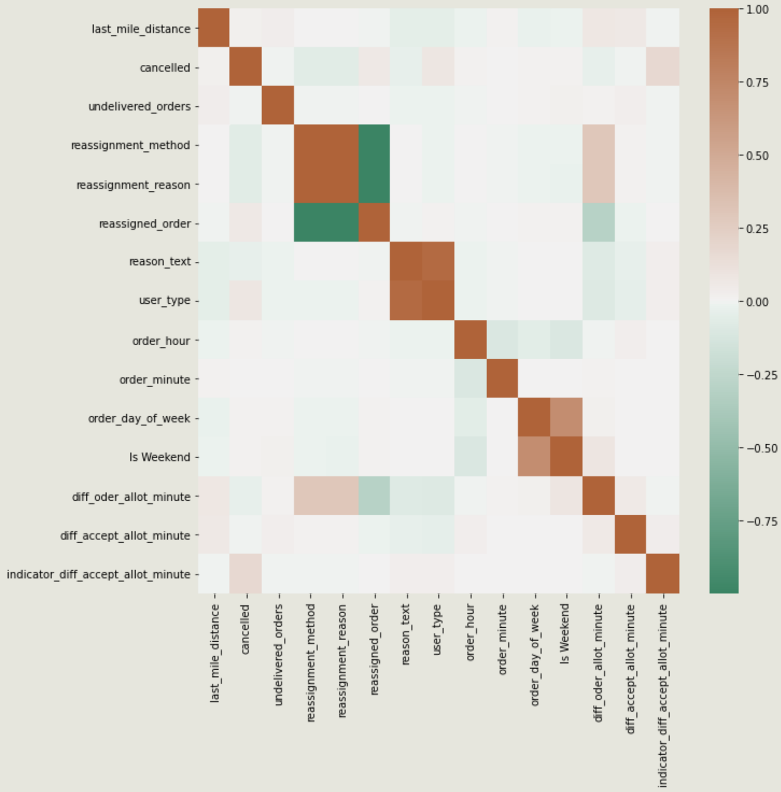
■ Squares are **categorical associations** (uncertainty coefficient & correlation ratio) from 0 to 1. The uncertainty coefficient is asymmetrical, (i.e. ROW LABEL values indicate how much they PROVIDE INFORMATION to each LABEL at the TOP).

• Circles are the **symmetrical numerical correlations** (Pearson's) from -1 to 1. The trivial diagonal is intentionally left blank for clarity.



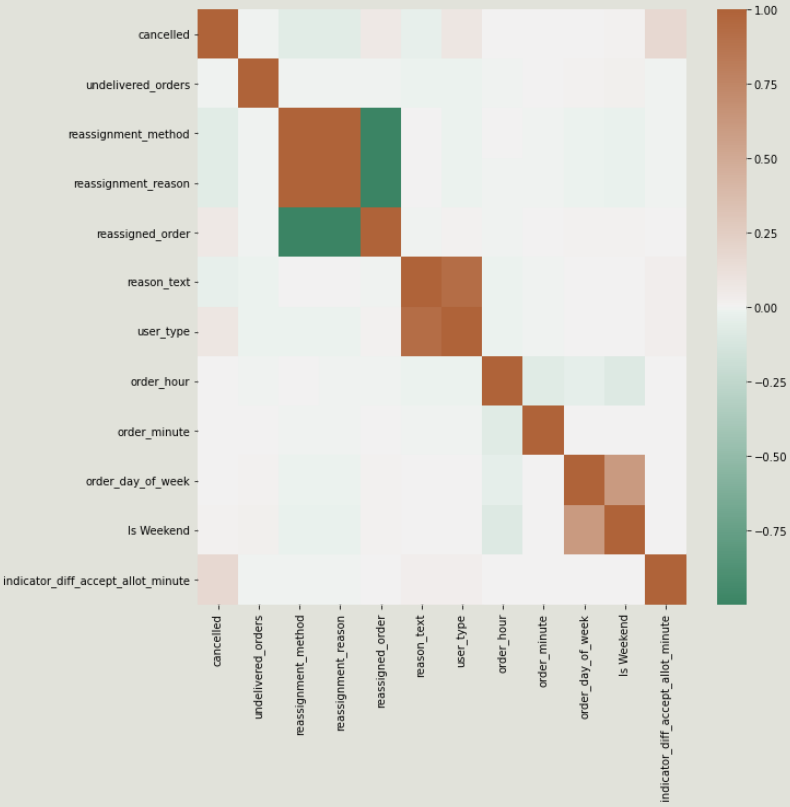
***Pearson's correlation coefficient** is very **sensitive** to **outliers**, which can have a **very large effect** on the **model fitting** and the Pearson correlation coefficient. This means including outliers in our analysis can lead to **misleading results.**

Spearman Correlations



We went with the **Spearman correlation** heatmap because it can **capture nonlinear monotonic relations** between features whereas the **Pearson correlation can capture linear relations only**. Many features have **ordinal values** such as reassignment_method, reassignment_reason, indicator_diff_accept_allot_minute and using **Spearman** correlation **might work better** than **Pearson** correlation.

Kendall's Tau



We have also checked **Kendall's tau** as **Spearman's rho** is **more sensitive to errors** and **discrepancies** in the data. When data is **normal**, Kendall's tau has **smaller gross error sensitivity** and **smaller asymptotic variance.**

Interesting Facts & Findings

1. We have observed that **delivered** and **undelivered** columns have exactly the **same no. of null values, that too on the same rows.** Also, there are **fewer instances of nan** values in **allotted orders**. So, We decided to **replace the extra nans** of delivered and undelivered columns with **Half of the allotted orders' value**, and where there were nan values for all of the aforementioned columns, we decided to impute them with 0.

   The **same** kind of **observation** was also seen in **reassignment_reason** and **reassignment_order**.

```
undelivered_orders        17341
delivered_orders          17341
```

```
alloted_orders            16948
```

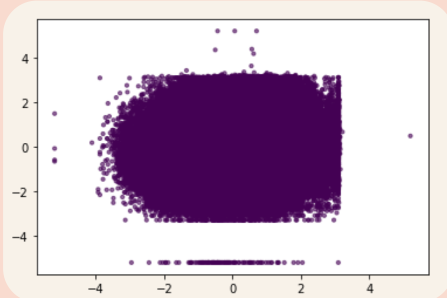
```
reassignment_reason       436247
reassigned_order          436247
```

```
combo[combo['indicator_diff_accept_allot_minute']==1]['cancelled'].value_counts()
```

```
1    157
Name: cancelled, dtype: int64
```

2. In the rows where there is Na**N** in **accept_time**, the order is **100% cancelled**. So, We made an **Indicator column** for that feature to **signify** if the former column **contains nan values** or **not**.

3. Before applying various **clustering techniques** like **K-Means Clustering**, **DBSCAN** and **HDBSCAN**, the optimal number of clusters by **Elbow method**, which came out to be **4**. We were trying to cluster them while **ignoring** the **order_id**, **rider_id** and **cancelled** column. But when we tried to visualize the results with 4 clusters, we got **undifferentiated** data, i.e. the data was **not properly clustered**.



4. There were **LOTS of outliers** in the data, that too in **many columns**. Many were dealt with when applying **Quantile Transformer**, because of the way it **transforms** the data features **using quantiles information.** Also we passed **'normal'** to the **'out' parameter** in the quantile transformer class

5. There is an **observation** we have made which might have occured due to a **bug** while **data entry**, i.e. the **accept_time** is registered for a date that is **earlier** than the **allot_time**, which is never possible. Also, this has happened **182** times.

```
train['check_neg'] = train['accept_time'] > train['allot_time']
train['check_neg'].value_counts()
```

```
True     449818
False       182
Name: check_neg, dtype: int64
```

THANK YOU

We thank **CnA**, **IIT Guwahati** and **Shadowfax** for conducting such an amazing competition where results are not just based on models, but also take into account data analysis, which is crucial for a data scientist. We have thoroughly enjoyed participating in the competition. Please hold more such competitions !! 🙏🙏

Contact Us :

ashiskumarparida73@gmail.com
alyalsonu@gmail.com