Order Delivery Time Prediction

Objectives

The objective of this assignment is to build a regression model that predicts the delivery time for orders placed through Porter. The model will use various features such as the items ordered, the restaurant location, the order protocol, and the availability of delivery partners.

The key goals are:
- Predict the delivery time for an order based on multiple input features
- Improve delivery time predictions to optimize operational efficiency
- Understand the key factors influencing delivery time to enhance the model's accuracy

Solution

1. First load the data , as dataset only has 175777 rows whole file can be loaded in single dataframe
2. Data processing
    a. Convert created_at' and 'actual_delivery_time' columns to datetime format
    b. We have below
    category_columns = ['market_id', 'store_primary_category', 'order_protocol']
    c. Convert them into integer type .('store_primary_category' is already integer type)
3. Feature engineering
    a. Calculate time_taken in minutes with 'actual_delivery_time' and 'created_at'
    porter_df['time_taken'] = (porter_df['actual_delivery_time'] - porter_df['created_at']).dt.total_seconds() / 60

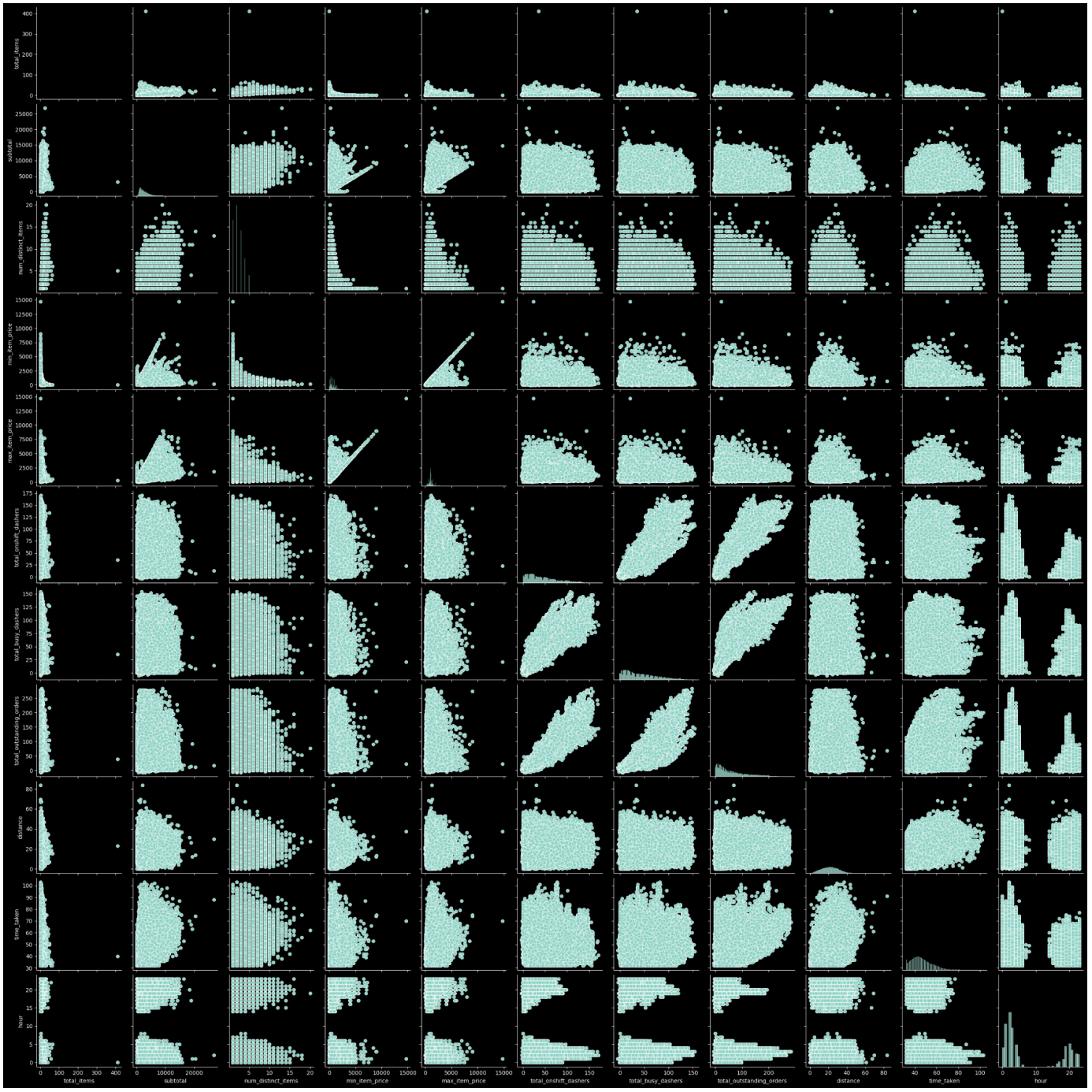
    b. Create hour and day_of_week features
    c. Add isWeekend Categorical feature which has binary values
    d. As we have create desired features from 'actual_delivery_time' and 'created_at' we can drop these columns from dataset

4. Creating training and validations sets
    a. Split dataset in 80-20 % ratio
5. Feature distribution
    a. Below are the numerical columns in the dataset

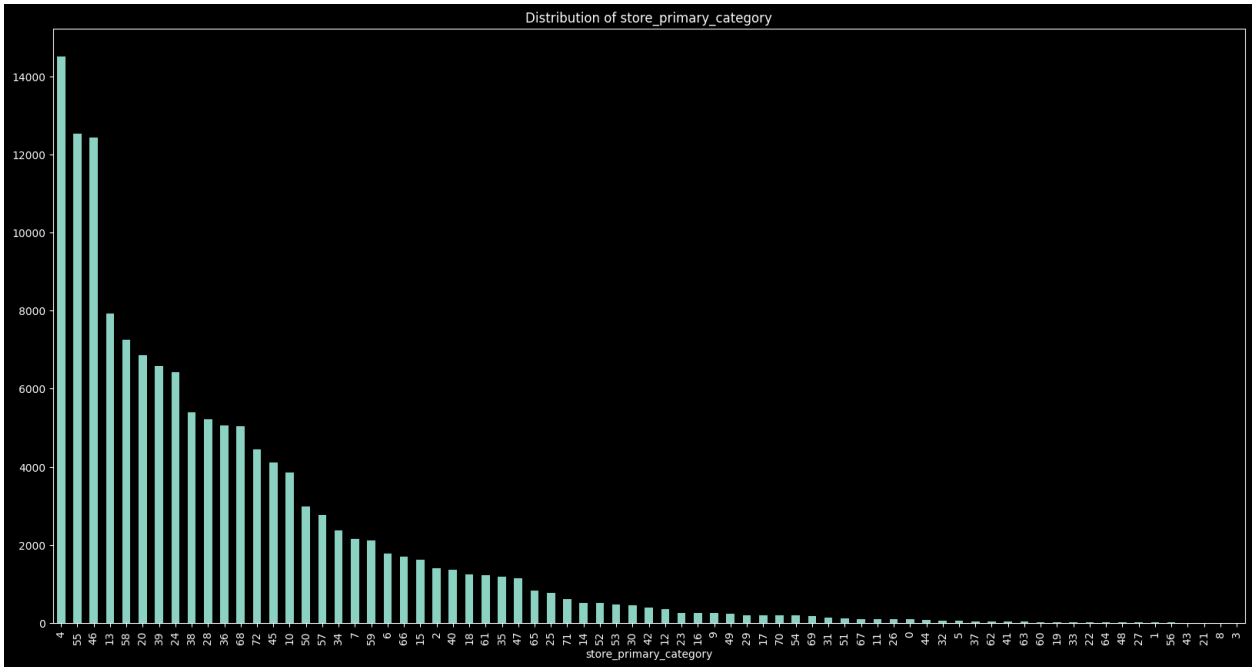
num_columns = ['total_items', 'subtotal', 'num_distinct_items', 'min_item_price', 'max_item_price', 'total_onshift_dashers', 'total_busy_dashers', 'total_outstanding_orders', 'distance', 'time_taken', 'hour']

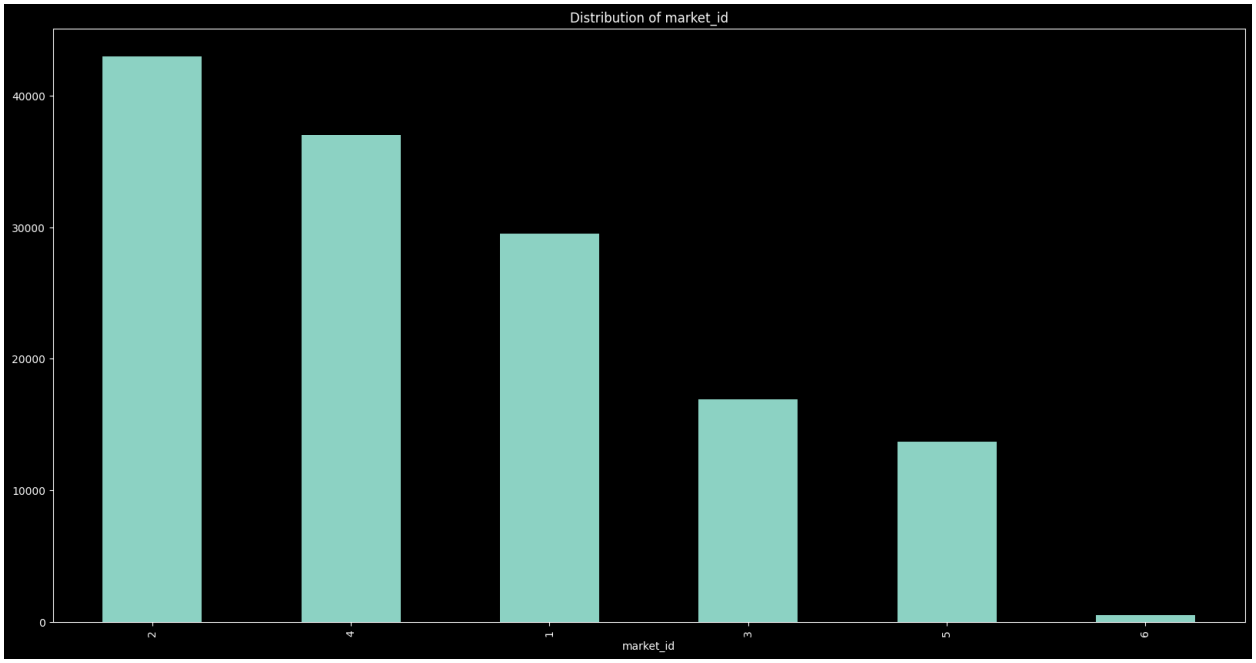
b. Below are the categorical columns in the dataset
cat_columns = ['market_id', 'store_primary_category', 'order_protocol', 'day_of_week', 'isWeekend']

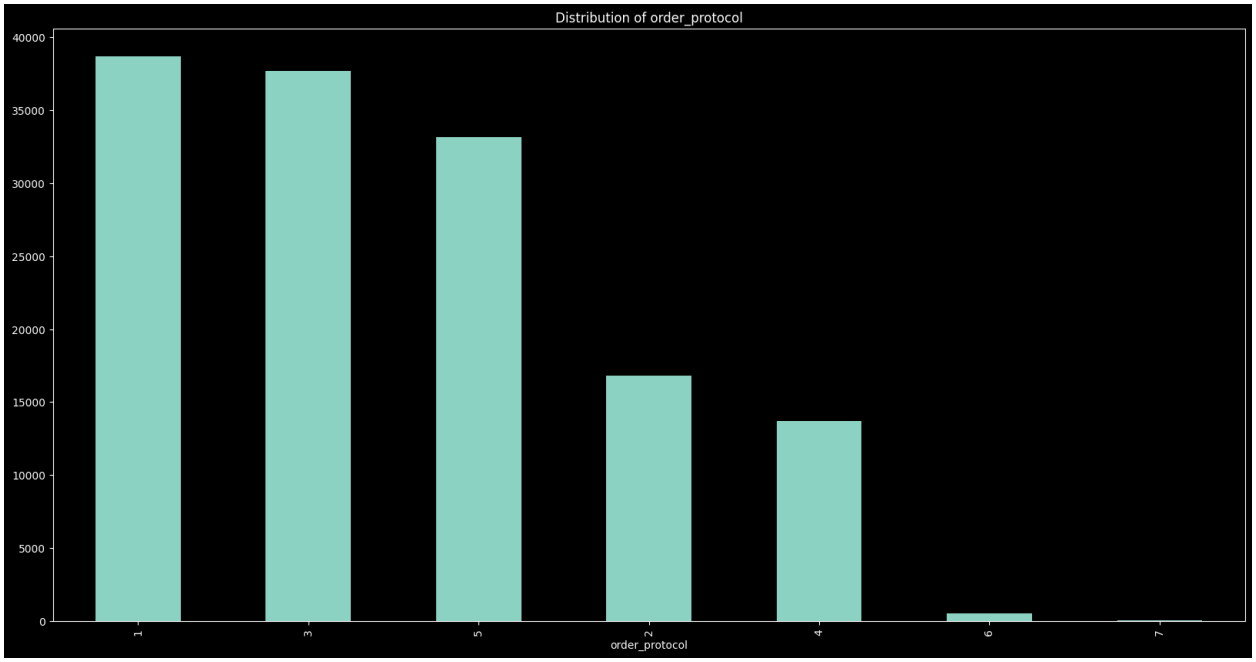
c. As Hour can also be treated as categorical data but treating it as categorical will create 24 features which will overfit the model .So assumption here is to use it as numerical data.
d. Distributions for all numerical columns (Univariate analysis)

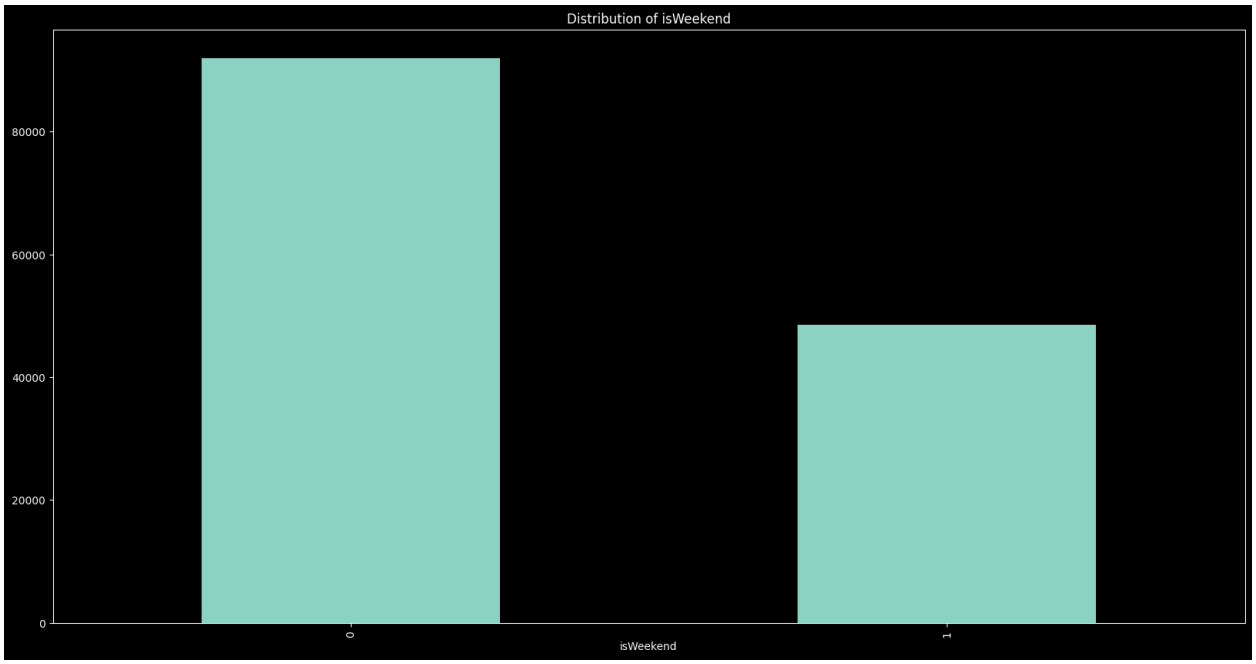


e. Distribution of categorical columns

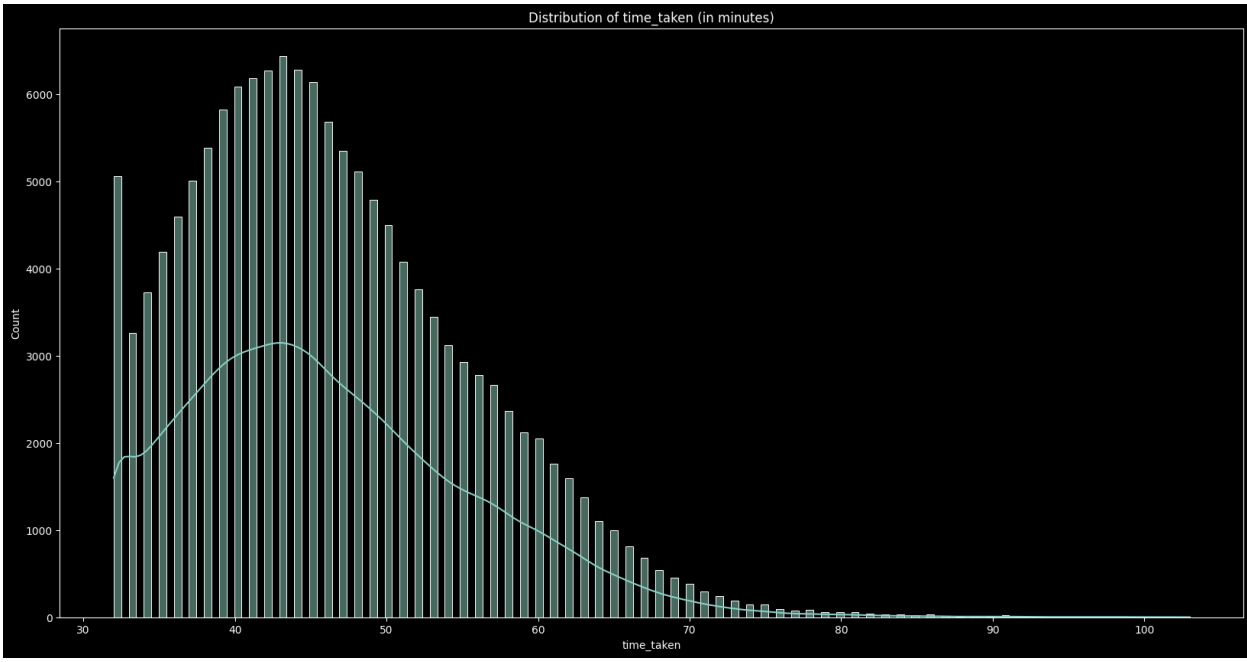

Distribution of store_primary_category
14000
12000
10000
8000
6000
4000
2000
0
4 55 46 13 58 20 39 24 38 28 36 68 72 45 10 50 57 34 7 59 6 66 15 2 40 18 61 35 47 65 25 71 14 52 53 30 42 12 23 16 9 49 29 17 70 54 69 31 51 67 11 26 0 44 32 5 37 62 41 63 60 19 33 22 64 48 27 1 56 43 21 8 3
store_primary_category


Distribution of market_id
40000
30000
20000
10000
0
2 4 1 3 5 6
market_id

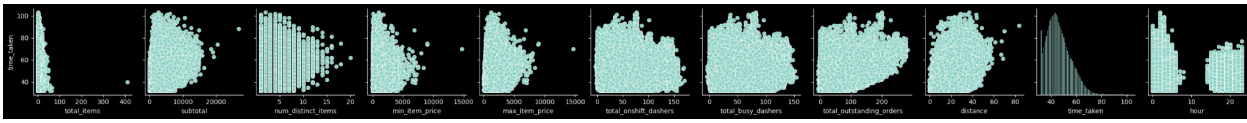




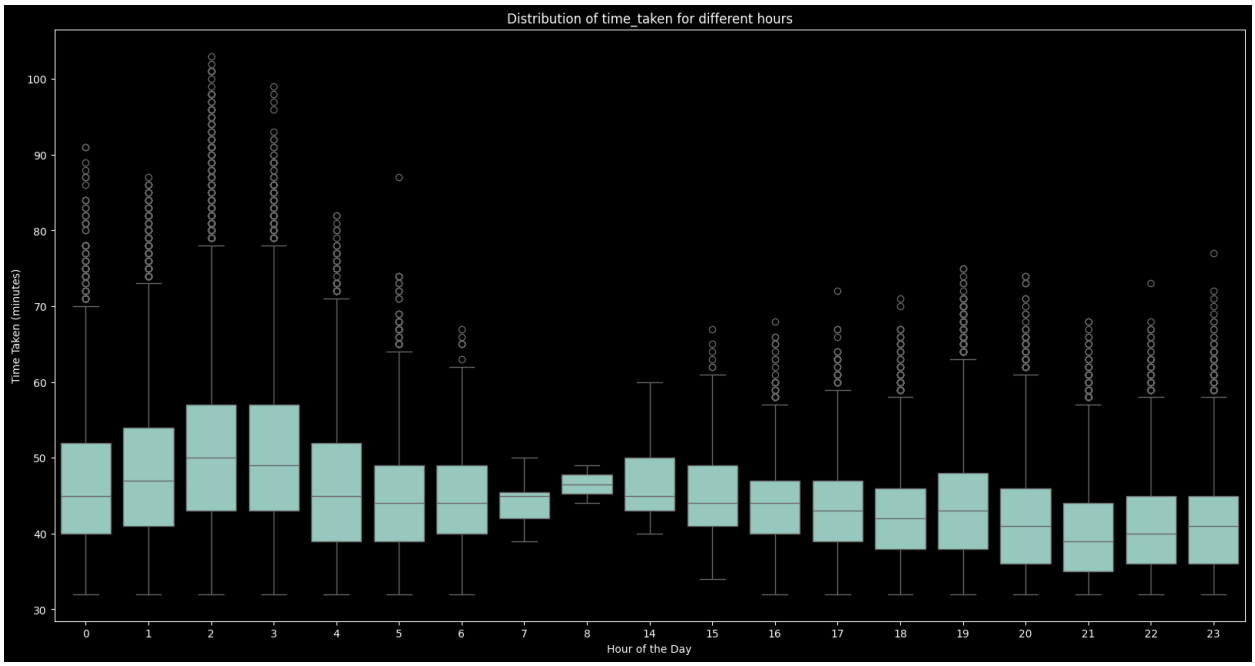
f. Visualize the distribution of time_taken



6. Relationships Between Features(Bivariate analysis)
    a. Scatter plot between time_taken numerical columns

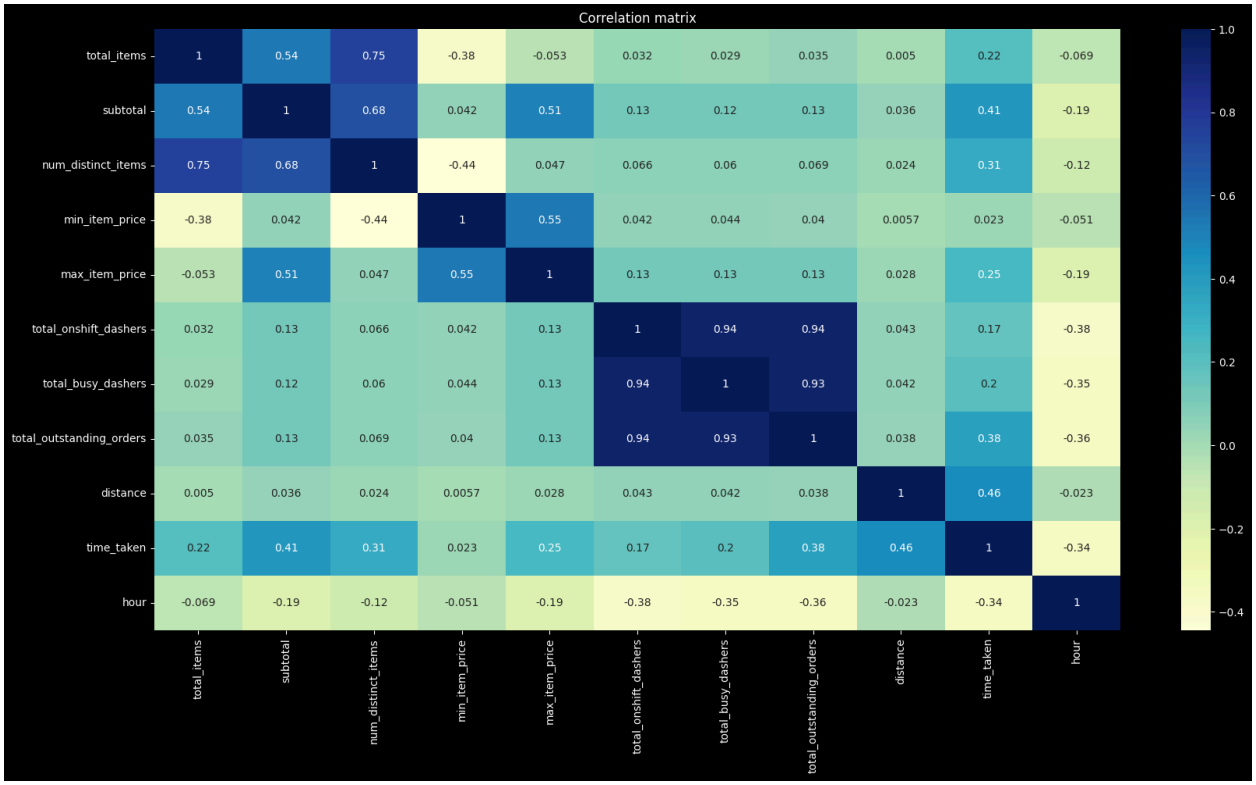


    b. distribution of time_taken for different hours



The median time taken for delivery increases from 0 to 16 hours, then decreases from 16 to 23 hours. The variability in the time taken for delivery is highest for the first few hours of the day and the last few hours of the day.

C. Heatmap of the correlation matrix

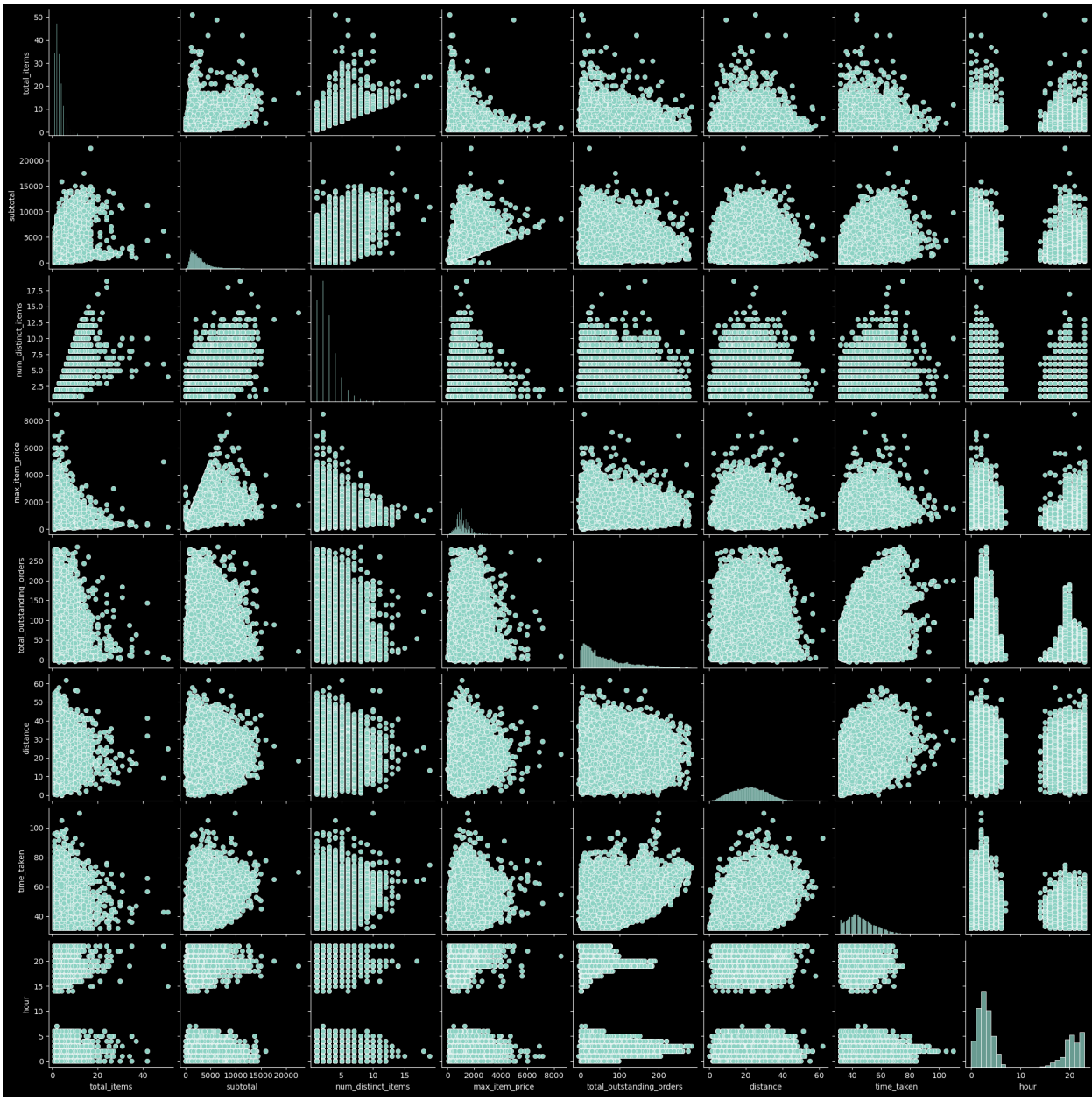


The factors that most strongly influence delivery time (time_taken) in this dataset appear to be the distance of the delivery and the total number of outstanding orders. The number of dashers on shift and busy also have a moderate positive influence, possibly indicating that high dasher activity corresponds to high demand. There's a slight tendency for deliveries to be faster later in the day.

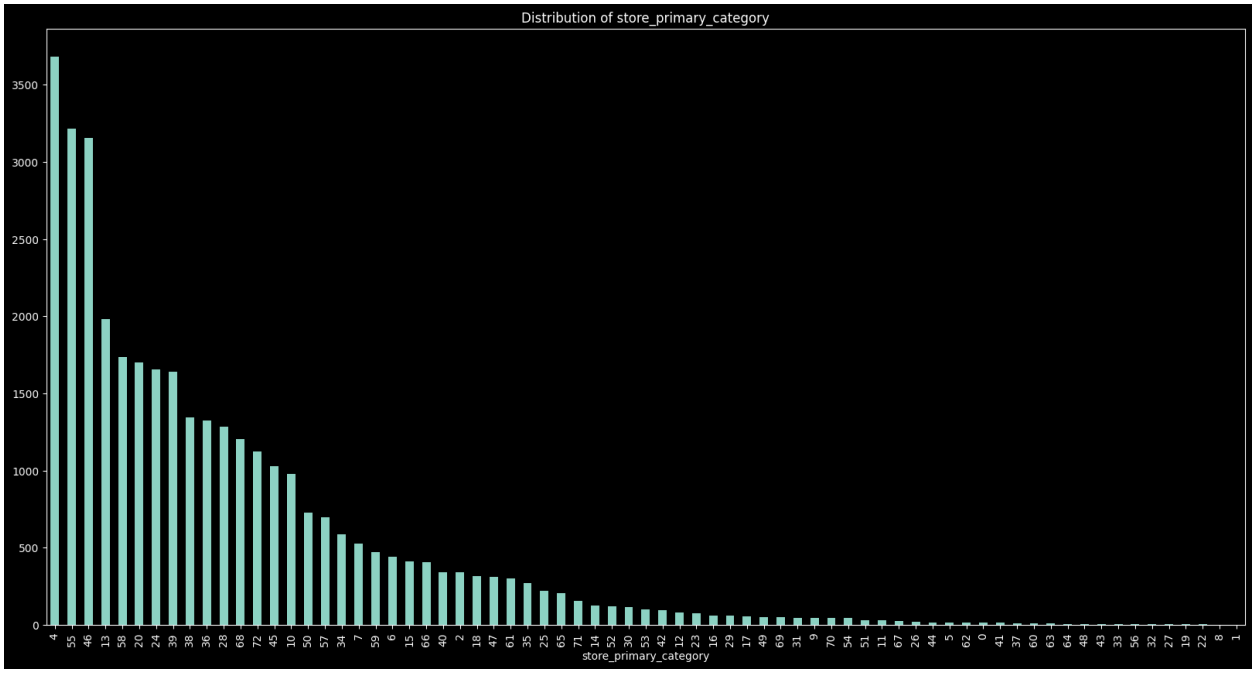
7. Remove weakly correlated feature
   weakly_correlated_columns = ['min_item_price',
   'total_onshift_dashers','total_busy_dashers']

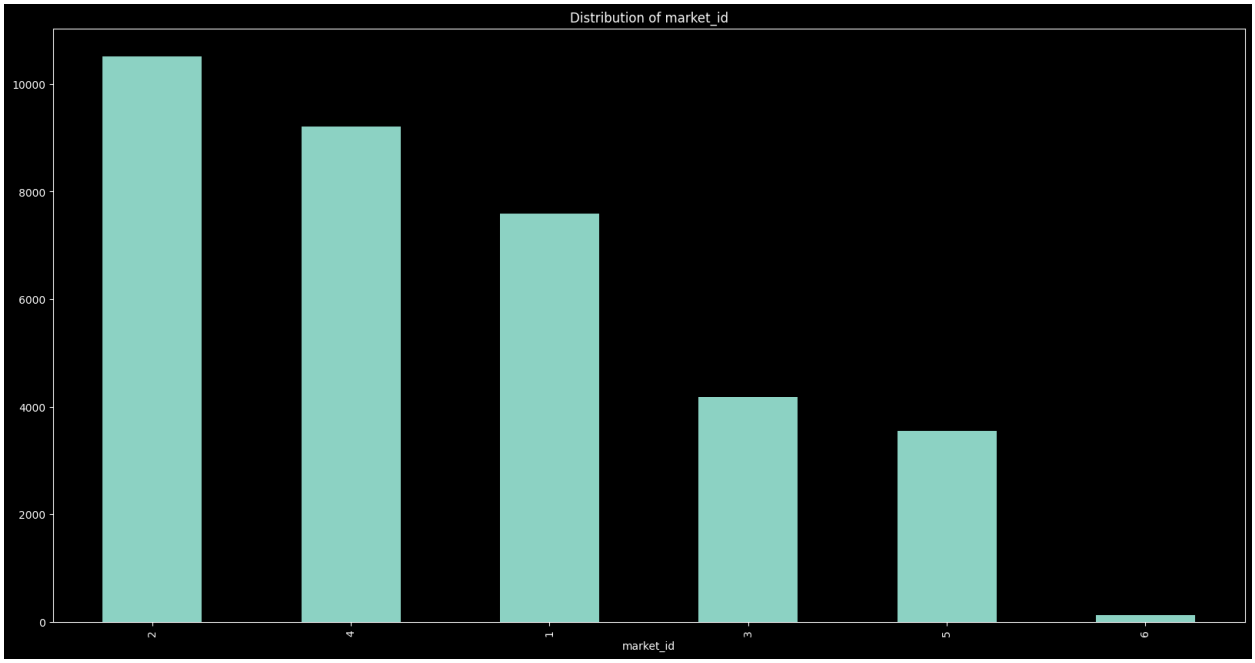
   input_df_train.drop(weakly_correlated_columns, axis=1, inplace=True)

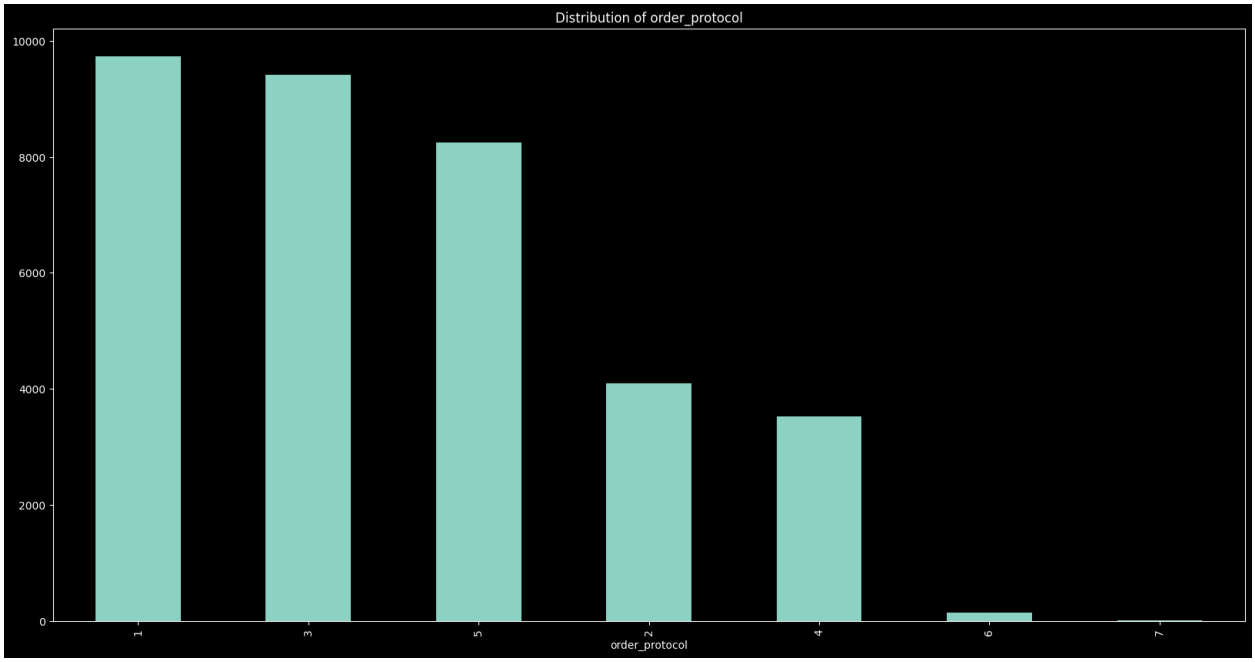
8. Remove outlier using IQR method
9. EDA on validation dataset
   a. Distribution for all numeric columns

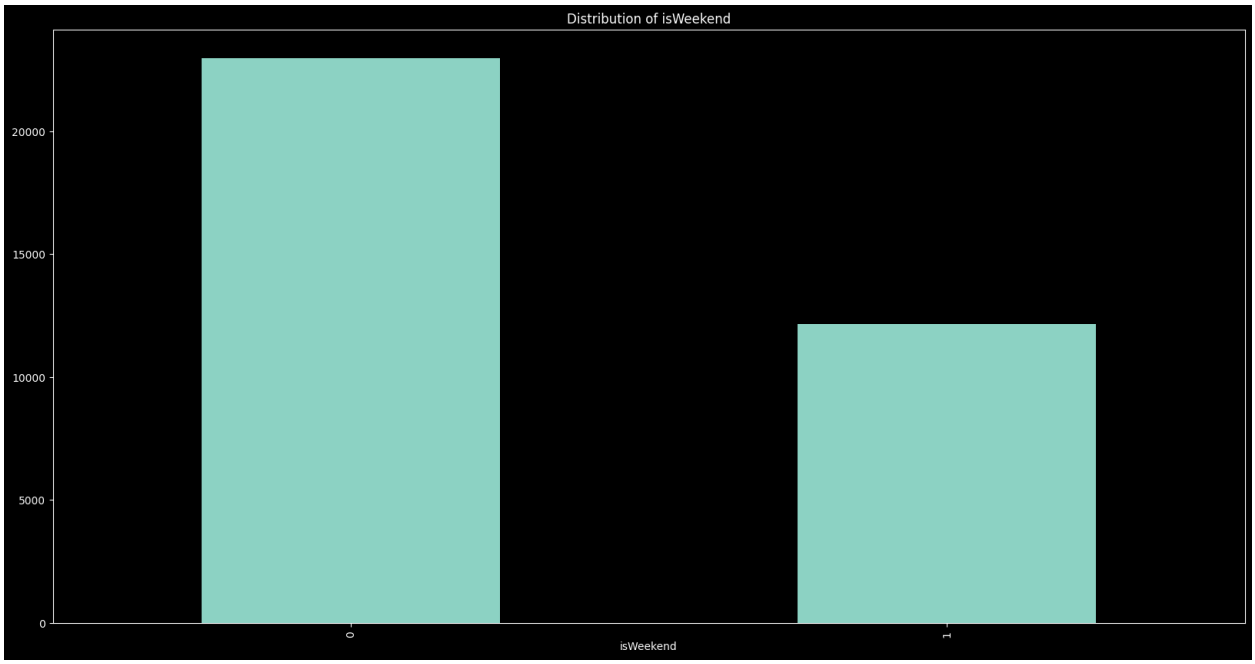


b. Distribution of categorical columns

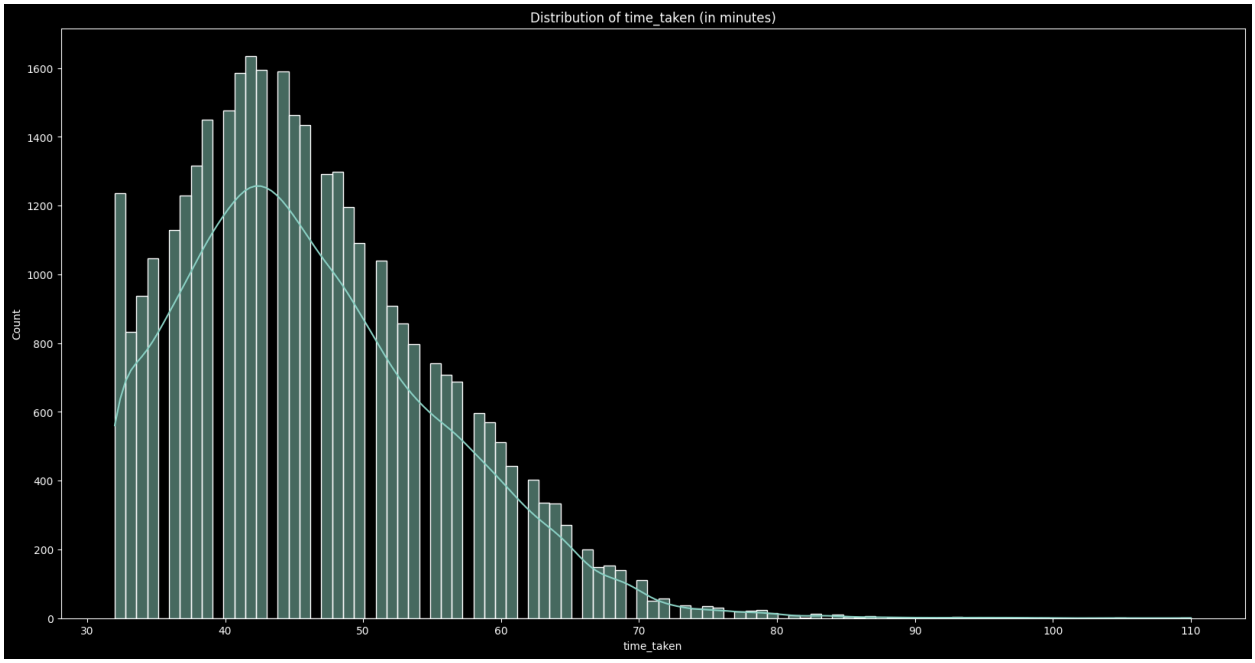

Distribution of store_primary_category
3500
3000
2500
2000
1500
1000
500
0
4 55 46 13 58 20 24 39 38 36 28 68 72 45 10 50 57 34 7 59 6 15 66 40 2 18 47 61 35 25 65 71 14 52 30 53 42 12 23 16 29 17 49 69 31 9 70 54 51 11 67 26 44 5 62 0 41 37 60 63 64 48 43 33 56 32 27 19 22 8 1
store_primary_category


Distribution of market_id
10000
8000
6000
4000
2000
0
2 4 1 3 5 6
market_id

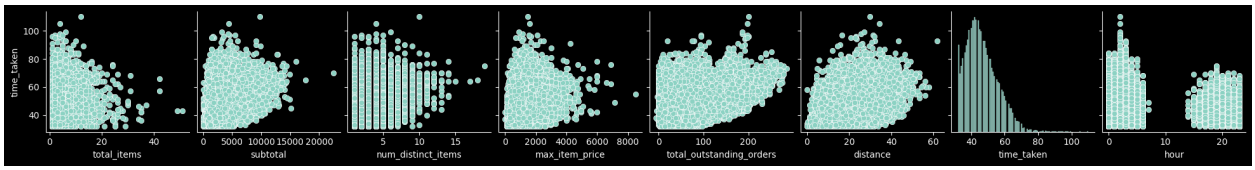




c. Distribution of time_taken



d. Scatter plot between time_taken numerical columns



10. **Assumptions :** We need to convert categorical columns into dummies. Converting all categorical columns into dummies will create lots of features and will result in overfitting . I am considering only two categorical columns **'market_id','day_of_week'.** Also dropping Store_primary_category','order_protocol' from test and training datasets as it wont impact our model .
11. Apply scaling to the numerical columns

12. Model summary

OLS Regression Results

Dep. Variable:	time_taken	R-squared:	0.569
Model:	OLS	Adj. R-squared:	0.569
Method:	Least Squares	F-statistic:	8721.
Date:	Wed, 26 Mar 2025	Prob (F-statistic):	0.00
Time:	18:40:54	Log-Likelihood:	-1.1889e+05
No. Observations:	119075	AIC:	2.378e+05
Df Residuals:	119056	BIC:	2.380e+05
Df Model:	18		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	4.186e-16	0.002	2.2e-13	1.000	-0.004	0.004
total_items	-0.0148	0.005	-3.144	0.002	-0.024	-0.006
subtotal	0.1749	0.004	46.687	0.000	0.168	0.182
num_distinct_items	0.0929	0.004	22.295	0.000	0.085	0.101
max_item_price	0.0273	0.003	9.396	0.000	0.022	0.033
total_outstanding_orders	0.3374	0.002	135.389	0.000	0.333	0.342
distance	0.5073	0.002	265.370	0.000	0.504	0.511
hour	-0.1406	0.002	-65.758	0.000	-0.145	-0.136
isWeekend	-0.0096	0.002	-5.932	0.000	-0.013	-0.006
market_id_2	-0.4660	0.003	-171.527	0.000	-0.471	-0.461
market_id_3	-0.1687	0.002	-74.708	0.000	-0.173	-0.164
market_id_4	-0.3418	0.003	-127.466	0.000	-0.347	-0.337
market_id_5	-0.1347	0.002	-61.016	0.000	-0.139	-0.130
market_id_6	-0.0309	0.002	-16.093	0.000	-0.035	-0.027
day_of_week_1	-0.0748	0.003	-29.887	0.000	-0.080	-0.070
day_of_week_2	-0.1117	0.003	-44.532	0.000	-0.117	-0.107
day_of_week_3	-0.0840	0.003	-33.328	0.000	-0.089	-0.079
day_of_week_4	-0.1109	0.003	-43.230	0.000	-0.116	-0.106
day_of_week_5	-0.0114	0.002	-7.078	0.000	-0.015	-0.008
day_of_week_6	-0.0009	0.002	-0.573	0.567	-0.004	0.002

Omnibus:	281.777	Durbin-Watson:	2.013
Prob(Omnibus):	0.000	Jarque-Bera (JB):	350.816
Skew:	0.034	Prob(JB):	6.63e-77
Kurtosis:	3.257	Cond. No.	1.06e+15

13. Check VIF to measure of multicollinearity among the independent variables within a multiple regression model

	Features	VIF
18	day_of_week_6	inf
7	isWeekend	inf
17	day_of_week_5	inf
1	subtotal	3.43
2	num_distinct_items	3.23
0	total_items	2.41
8	market_id_2	2.07
10	market_id_4	1.98
3	max_item_price	1.89
16	day_of_week_4	1.75
4	total_outstanding_orders	1.74
15	day_of_week_3	1.69
14	day_of_week_2	1.68
13	day_of_week_1	1.67
9	market_id_3	1.39
11	market_id_5	1.33
6	hour	1.26
12	market_id_6	1.02
5	distance	1.01

14. Generally VIF that is less than 5 is recommended. So there are clearly some variables we need to drop.
15. Drop day_of_week_6 and check VIF

	Features	VIF
1	subtotal	3.43
2	num_distinct_items	3.23
7	isWeekend	2.99
0	total_items	2.41
8	market_id_2	2.07
10	market_id_4	1.98
3	max_item_price	1.89
16	day_of_week_4	1.75
4	total_outstanding_orders	1.74
15	day_of_week_3	1.69
17	day_of_week_5	1.69
14	day_of_week_2	1.68
13	day_of_week_1	1.67
9	market_id_3	1.39
11	market_id_5	1.33
6	hour	1.26
12	market_id_6	1.02
5	distance	1.01

- After Getting dummies for categorical feature and Elementing features with RFE we will have below feature in the model

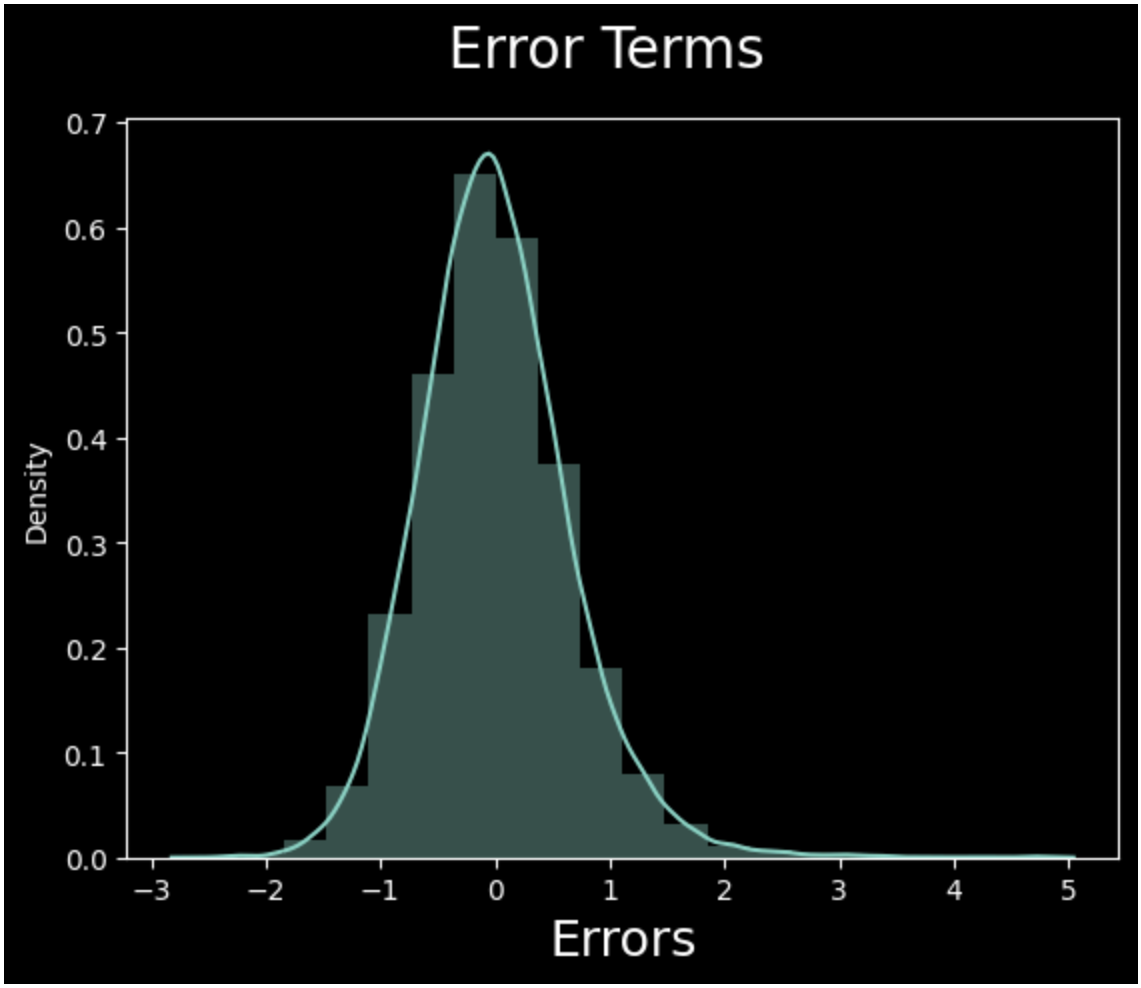
OLS Regression Results

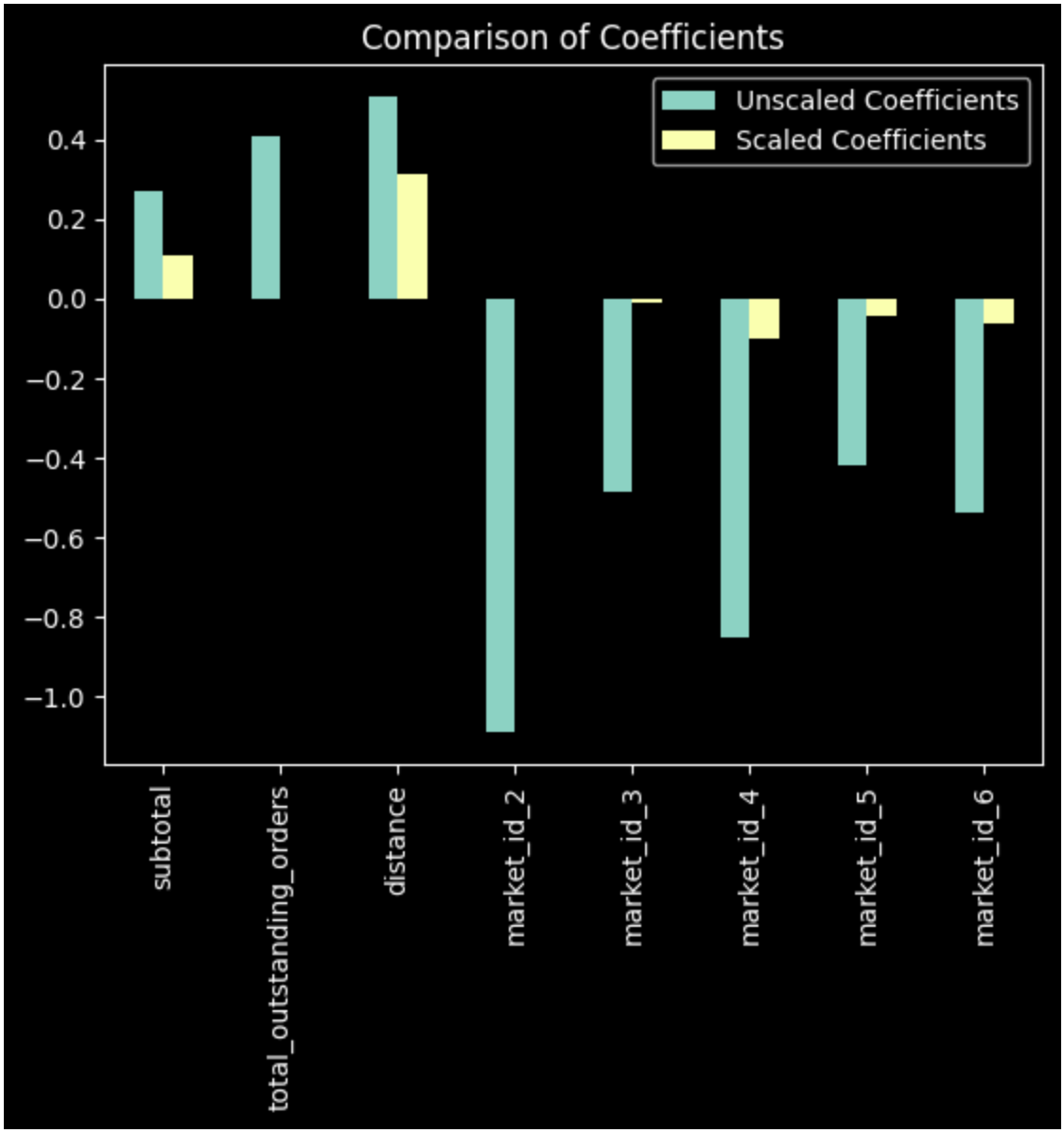
Dep. Variable:	time_taken	R-squared:	0.598
Model:	OLS	Adj. R-squared:	0.598
Method:	Least Squares	F-statistic:	2.616e+04
Date:	Wed, 26 Mar 2025	Prob (F-statistic):	0.00
Time:	18:24:45	Log-Likelihood:	-1.3542e+05
No. Observations:	140614	AIC:	2.709e+05
Df Residuals:	140605	BIC:	2.709e+05
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-2.002e-16	0.002	-1.18e-13	1.000	-0.003	0.003
subtotal	0.3196	0.002	184.677	0.000	0.316	0.323
total_outstanding_orders	0.4465	0.002	204.125	0.000	0.442	0.451
distance	0.4540	0.002	267.672	0.000	0.451	0.457
hour	-0.1146	0.002	-60.969	0.000	-0.118	-0.111
market_id_2	-0.4432	0.002	-183.828	0.000	-0.448	-0.438
market_id_3	-0.1445	0.002	-72.694	0.000	-0.148	-0.141
market_id_4	-0.3430	0.002	-145.205	0.000	-0.348	-0.338
market_id_5	-0.1176	0.002	-60.571	0.000	-0.121	-0.114

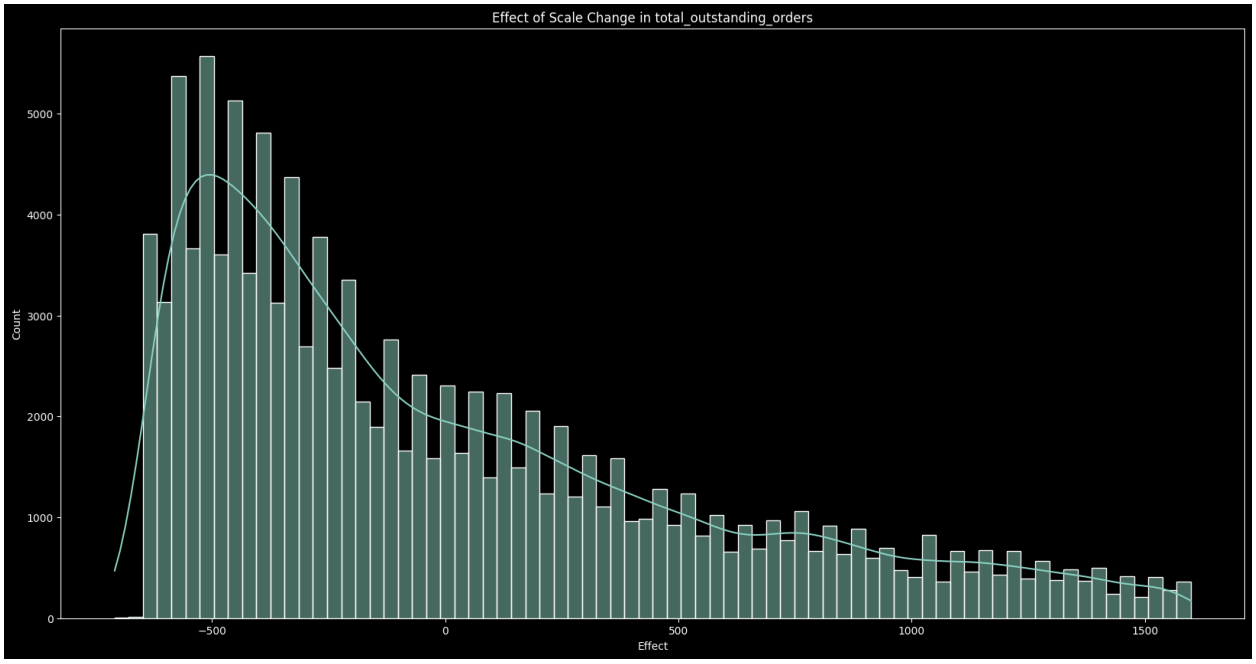
Omnibus:	10412.254	Durbin-Watson:	2.002
Prob(Omnibus):	0.000	Jarque-Bera (JB):	25833.861
Skew:	0.447	Prob(JB):	0.00
Kurtosis:	4.900	Cond. No.	2.78

- residual analysis


Error Terms
Density
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
−3
−2
−1
0
1
2
3
4
5
Errors


Comparison of Coefficients
Unscaled Coefficients
Scaled Coefficients
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
subtotal
total_outstanding_orders
distance
market_id_2
market_id_3
market_id_4
market_id_5
market_id_6

Get the scaled coefficient for 'total_outstanding_orders' as total_itme is removed from the model



Final Metrics

Mean Squared Error: 0.43308272301325407
R-squared: 0.566917276986746

```
<class 'pandas.core.frame.DataFrame'>
Index: 35156 entries, 139667 to 98870
Data columns (total 8 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   subtotal                  35156 non-null  float64
 1   total_outstanding_orders  35156 non-null  float64
 2   distance                  35156 non-null  float64
 3   market_id_2               35156 non-null  int64
 4   market_id_3               35156 non-null  int64
 5   market_id_4               35156 non-null  int64
 6   market_id_5               35156 non-null  int64
 7   market_id_6               35156 non-null  int64
dtypes: float64(3), int64(5)
memory usage: 2.4 MB
```

Model 1 seems to be the best model with 8 features.