In [1]:
```
import os
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn import metrics
from sklearn.model_selection import train_test_split
```

In [2]:
```
df= pd.read_parquet(r"C:\Users\Pulkit\Downloads\yellow_tripdata_2023-11.pa
```

In [3]:
```
df.head(2)
```

Out[3]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance | Ratec |
|---|---|---|---|---|---|---|
| **0** | 1 | 2023-11-01 00:03:03 | 2023-11-01 01:04:08 | 2.0 | 13.6 | |
| **1** | 1 | 2023-11-01 00:03:28 | 2023-11-01 00:23:59 | 0.0 | 3.5 | |

In [4]:
```
df.shape
```

Out[4]:
```
(3339715, 19)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3339715 entries, 0 to 3339714
Data columns (total 19 columns):
 #   Column                 Dtype
---  ------                 -----
 0   VendorID               int32
 1   tpep_pickup_datetime   datetime64[ns]
 2   tpep_dropoff_datetime  datetime64[ns]
 3   passenger_count        float64
 4   trip_distance          float64
 5   RatecodeID             float64
 6   store_and_fwd_flag     object
 7   PULocationID           int32
 8   DOLocationID           int32
 9   payment_type           int64
 10  fare_amount            float64
 11  extra                  float64
 12  mta_tax                float64
 13  tip_amount             float64
 14  tolls_amount           float64
 15  improvement_surcharge  float64
 16  total_amount           float64
 17  congestion_surcharge   float64
 18  Airport_fee            float64
dtypes: datetime64[ns](2), float64(12), int32(3), int64(1), object(1)
memory usage: 445.9+ MB
```

# Univariate Analysis

## Categorical

In [6]:
```
1 df["store_and_fwd_flag"].nunique()
```

Out[6]:
```
2
```

In [7]:
```
1 df["store_and_fwd_flag"].unique()
```

Out[7]:
```
array(['N', 'Y', None], dtype=object)
```

In [8]:
```
1 df["store_and_fwd_flag"].value_counts(dropna=False)
```

Out[8]:
```
N       3192960
None     132675
Y         14080
Name: store_and_fwd_flag, dtype: int64
```

In [9]:
```
1 (132675/3339715)*100
2 # We can see that there are 132675 Null values in the data which is only 
3 # less than 5% so we will drop these values
```

Out[9]:
```
3.972644372349138
```

In [10]:
```
1 df1= df.dropna()
```

In [11]:
```
1 df1.shape
```

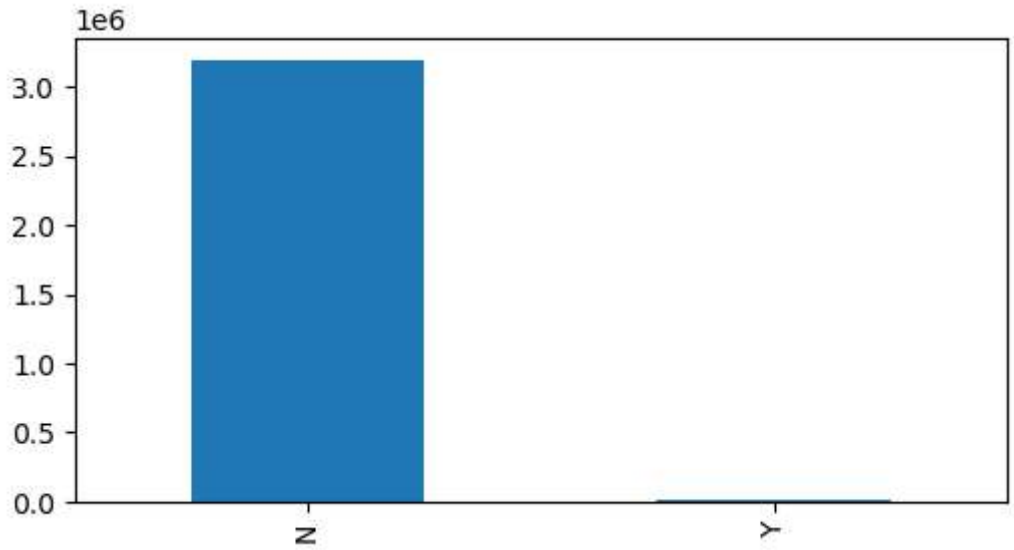
Out[11]:
```
(3207040, 19)
```

In [12]:
```
1 df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3339715 entries, 0 to 3339714
Data columns (total 19 columns):
 #   Column                 Dtype
---  ------                 -----
 0   VendorID               int32
 1   tpep_pickup_datetime   datetime64[ns]
 2   tpep_dropoff_datetime  datetime64[ns]
 3   passenger_count        float64
 4   trip_distance          float64
 5   RatecodeID             float64
 6   store_and_fwd_flag     object
 7   PULocationID           int32
 8   DOLocationID           int32
 9   payment_type           int64
 10  fare_amount            float64
 11  extra                  float64
 12  mta_tax                float64
 13  tip_amount             float64
 14  tolls_amount           float64
 15  improvement_surcharge  float64
 16  total_amount           float64
 17  congestion_surcharge   float64
 18  Airport_fee            float64
dtypes: datetime64[ns](2), float64(12), int32(3), int64(1), object(1)
memory usage: 445.9+ MB
```

In [13]:
```
1 df1["store_and_fwd_flag"].value_counts().plot(kind="bar",figsize=(6,3))
```

Out[13]: <Axes: >




In [14]:
```
1 df1["VendorID"].unique()
```

Out[14]: array([1, 2])

In [15]:
```
1 df1["VendorID"].value_counts()
```

Out[15]:
```
2    2437585
1     769455
Name: VendorID, dtype: int64
```

In [16]:
```
1 df1["passenger_count"].nunique()
```

Out[16]:
```
10
```

In [17]:
```
1 df1["passenger_count"].unique()
```

Out[17]:
```
array([2., 0., 4., 1., 3., 5., 6., 8., 7., 9.])
```

In [18]:
```
1 df1["passenger_count"].value_counts(dropna=False)   # Dropna=False will sh
```

Out[18]:
```
1.0    2438039
2.0     484579
3.0     113651
4.0      64725
0.0      41443
5.0      39579
6.0      24969
8.0         42
7.0          9
9.0          4
Name: passenger_count, dtype: int64
```

In [19]:
```
1 (132675/3339715)*100   # 3% of the data in the passenger count is null
```

Out[19]:
```
3.972644372349138
```

In [20]:
```
1 df1["RatecodeID"].nunique()
```

Out[20]:
```
7
```

In [21]:
```
1 df1["RatecodeID"].unique()
```

Out[21]:
```
array([ 1.,  2., 99.,  5.,  4.,  3.,  6.])
```

In [22]:
```
1 df1["RatecodeID"].value_counts(dropna=False)
```

Out[22]:
```
1.0     3025815
2.0      121449
99.0      21342
5.0       20966
3.0       10695
4.0        6767
6.0           6
Name: RatecodeID, dtype: int64
```

In [23]:
```
(21342/3207040)*100
# We can see that there is a term with 99 value is present in the data but
# dictionary that it is present in the value. and we have computed that th
# we will delete this ratecode id
```

Out[23]: 0.6654734583915386

In [24]:
```
df1=df1.drop(df1[df1["RatecodeID"]==99].index)
```

In [25]:
```
df1.shape
```

Out[25]: (3185698, 19)

In [26]:
```
df1["store_and_fwd_flag"].nunique()
```

Out[26]: 2

In [27]:
```
df1["store_and_fwd_flag"].unique()
```

Out[27]: array(['N', 'Y'], dtype=object)

In [28]:
```
df1["store_and_fwd_flag"].value_counts(dropna=False)
```

Out[28]:
```
N    3171683
Y      14015
Name: store_and_fwd_flag, dtype: int64
```

In [29]:
```
df1["PULocationID"].nunique()
```

Out[29]: 256

In [30]:
```
1 df1["PULocationID"].unique()
```

Out[30]:
```
array([132, 140, 236, 141, 114,  48, 170, 144,  90, 249, 230,  79, 234,
       107, 142,  68, 137, 158, 237,  50,  43, 148, 239, 238, 231, 162,
       229, 211, 246, 261, 163, 100, 161, 164, 224, 186,   6, 233,  66,
        87, 263,  80, 113, 262,   4,  75,  88, 125, 265, 143, 152,  83,
       232, 264, 179, 226,  24, 168,  45, 151, 209, 255,  37,  63, 138,
       146, 129,   7, 157,  17, 215,  74,  12,  28,  97, 173,  41, 256,
       166,  69, 145, 116,  56,  42,  25, 130,  91, 244, 198, 243, 165,
       188, 136, 223, 235,  20,  13,  82, 248, 193, 216,  65, 181, 217,
        49,  70,  89, 131, 189,  26, 258,  61,  14, 112, 174,  93,   1,
       203,  35, 139,  51,  40, 191,  62,  33,  95,  78, 121, 202,  52,
       134,  18, 178, 196, 247, 197, 207, 218, 205, 242,   3, 260, 159,
        10, 212, 102,  32, 126,  76,  92, 119,  81, 220, 185,  72, 222,
        39, 149,  22,  71, 213, 135,   9, 160,  47, 177, 241,  23, 219,
       194, 127, 195, 133,  64, 225, 253,   8, 106,  36,  98, 208, 155,
       190, 147,  38, 154,  29, 122, 221,  85, 257, 252, 228, 182, 169,
       101,  86, 124,  31, 227,  55,  67,  11, 167, 240,  94,  54, 150,
       180,  53,  19, 210, 251, 184,  77, 200,  34, 250,  16, 192, 204,
       120, 115,  96,  21, 259,  57, 175, 111, 254, 171,  60, 199,  15,
       153, 128, 183, 117, 123, 109, 108, 245,  73, 201,  58,  84, 206,
        59,  46, 105, 187, 156,   5,   2, 118, 172])
```

In [31]:
```
1 df1["PULocationID"].value_counts(dropna=False)
```

Out[31]:
```
237    164310
161    160799
132    157985
236    143964
162    119132
        ...
201         1
204         1
245         1
109         1
172         1
Name: PULocationID, Length: 256, dtype: int64
```

In [32]:
```
1 df1.drop(columns="PULocationID",inplace=True)   # because it is a spars va
```

In [33]:
```
1 # df1["PULocationID"].isnull().sum()  # there are no null values
```

In [34]:
```
1 df1["DOLocationID"].nunique()
```

Out[34]:
```
261
```

In [35]:
```
df1["DOLocationID"].unique()
```

Out[35]:
```
array([ 26,   7, 230, 236, 141, 263,  62, 246,  42, 148, 164, 231,  25,
       224, 255,  48, 143, 229, 163,  75, 140,  76,  13, 186,  79, 100,
       234, 244, 137, 174, 162, 177, 257, 142, 205, 113, 237, 116, 158,
       238, 165, 216, 239,  90,  74, 155,  41, 114,  49, 181, 136, 261,
       112,  80,  87, 166, 107, 146, 180, 249, 169,  68,  50, 190, 161,
        93,  97, 188,   4, 258, 262,   6, 170, 241,  33, 243,  65, 233,
       125, 144, 202, 151,  34, 219,  18, 218,  40, 223,  14, 226, 130,
        36, 256, 211, 178,  43,  69, 232, 265,  95,  39, 264, 196,  45,
        91, 168, 129,  92,  37, 242,  17, 179, 209, 145, 189, 157,  67,
       220,  71, 152,  88,  83, 203, 260,  10, 192,  52, 132,  11, 208,
        24, 225,  54, 221, 195, 119, 198, 182, 123,  98,  35,  72, 217,
        63,  89,  61, 171, 210, 228, 248,  64, 106,  22,  66, 215, 213,
       138,  82, 135, 235, 127,  28, 247, 134, 147, 131, 197, 133, 121,
       101,  85, 159, 102,  56, 149, 173,   1,  73, 167, 212, 126,  16,
       153, 194,  78, 250,  20,   8, 183, 227, 154,  53,  60,  70, 193,
       200,  38,  15,  77,  51, 160, 254,  12,  21, 252, 120, 128, 222,
       108, 191, 139, 207, 118, 184,  81, 175, 124,  58,  19, 185,  94,
        55,  86, 214,   3, 150,  96,  29,   9, 117,  23, 259,  30,  47,
        57, 201, 122, 111, 245, 240,  27,  32, 253, 251, 176, 187,  46,
       206,  31, 115, 156,  84, 109, 172, 204,  44,   5,   2,  99,  59,
       105])
```

In [36]:
```
df1["DOLocationID"].value_counts(dropna=False)
```

Out[36]:
```
236    151731
237    147745
161    131035
230     97844
162     94333
        ...
59         11
44          6
2           4
99          1
105         1
Name: DOLocationID, Length: 261, dtype: int64
```

In [37]:
```
df1["DOLocationID"].isnull().sum() # there are no null values in the DO lo
```

Out[37]:
```
0
```

In [38]:
```
df1.drop(columns="DOLocationID",inplace=True)
```

**Payment type**

In [39]:
```
df1["payment_type"].nunique()
```

Out[39]:
```
4
```

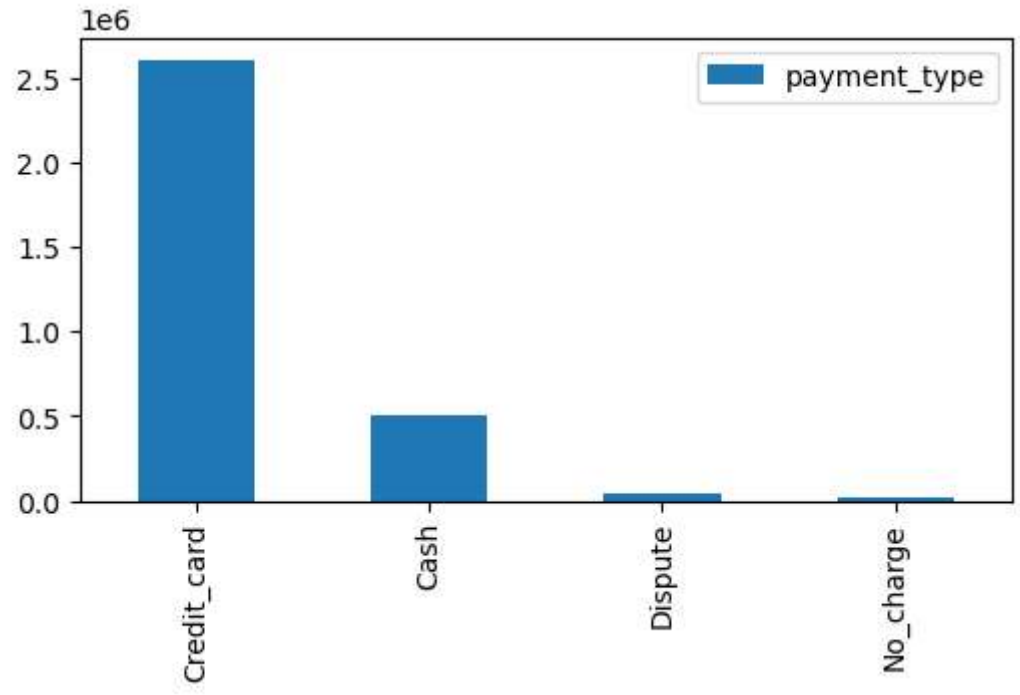
In [40]:
```
1 df1["payment_type"].unique()
```

Out[40]:
```
array([2, 1, 3, 4], dtype=int64)
```

In [41]:
```
1 df1["payment_type"].value_counts(dropna=False)
```

Out[41]:
```
1    2608850
2     507414
4      48425
3      21009
Name: payment_type, dtype: int64
```

In [42]:
```
1 df1["payment_type"].value_counts().plot(kind="bar", figsize=(6,3), legend=
2 plt.xticks([0,1,3,2],["Credit_card","Cash","No_charge","Dispute"])
3 plt.show()
```




We can understand from the obove graph that most of the American people are Paying their taxi expenses using credit card. As Over 81.9% people are paying using Credit card. And only 15.92% people are paying using Cash.

**MTA TAx**

In [43]:
```
1 df1["mta_tax"].nunique()
```

Out[43]:
```
6
```

In [44]:
```
1 df1["mta_tax"].value_counts()
```

Out[44]:
```
 0.50    3117571
-0.50      35600
 0.00      32523
 3.50          2
 4.00          1
 0.05          1
Name: mta_tax, dtype: int64
```

**Improvement surcharge**

In [45]:
```
1 df1["improvement_surcharge"].nunique()
```

Out[45]:
```
4
```

In [46]:
```
1 df["improvement_surcharge"].unique()
```

Out[46]:
```
array([ 1. , -1. ,  0. ,  0.3])
```

In [47]:
```
1 df["improvement_surcharge"].value_counts()
```

Out[47]:
```
 1.0    3300737
-1.0      36725
 0.0       1501
 0.3        752
Name: improvement_surcharge, dtype: int64
```

## Numerical Variable

In [48]:
```
1 df1.describe(percentiles=[.01,.02,.03,.04,.05,.1,.25,.5,.75,.9,.95,.96,.97
```

Out[48]:

| | count | mean | std | min | 1% | 2% | 3% | 4% | 5% | |
|---|---|---|---|---|---|---|---|---|---|---|
| **VendorID** | 3185698.0 | 1.765164 | 0.423896 | 1.00 | 1.0 | 1.0 | 1.00 | 1.0 | 1.00 | |
| **passenger_count** | 3185698.0 | 1.360424 | 0.869648 | 0.00 | 0.0 | 1.0 | 1.00 | 1.0 | 1.00 | |
| **trip_distance** | 3185698.0 | 3.336554 | 4.823657 | 0.00 | 0.0 | 0.2 | 0.35 | 0.4 | 0.47 | |
| **RatecodeID** | 3185698.0 | 1.077545 | 0.412316 | 1.00 | 1.0 | 1.0 | 1.00 | 1.0 | 1.00 | |
| **payment_type** | 3185698.0 | 1.218071 | 0.524319 | 1.00 | 1.0 | 1.0 | 1.00 | 1.0 | 1.00 | |
| **fare_amount** | 3185698.0 | 19.475394 | 18.891461 | -872.99 | -3.7 | 4.4 | 5.10 | 5.1 | 5.80 | |
| **extra** | 3185698.0 | 1.558659 | 1.855843 | -7.50 | 0.0 | 0.0 | 0.00 | 0.0 | 0.00 | |
| **mta_tax** | 3185698.0 | 0.483723 | 0.116068 | -0.50 | -0.5 | 0.0 | 0.50 | 0.5 | 0.50 | |
| **tip_amount** | 3185698.0 | 3.701480 | 4.123650 | -100.00 | 0.0 | 0.0 | 0.00 | 0.0 | 0.00 | |
| **tolls_amount** | 3185698.0 | 0.602621 | 2.221272 | -77.75 | 0.0 | 0.0 | 0.00 | 0.0 | 0.00 | |
| **improvement_surcharge** | 3185698.0 | 0.976653 | 0.214100 | -1.00 | -1.0 | 1.0 | 1.00 | 1.0 | 1.00 | |
| **total_amount** | 3185698.0 | 28.660435 | 23.923081 | -801.00 | -8.7 | 8.7 | 9.85 | 10.5 | 11.20 | 1 |
| **congestion_surcharge** | 3185698.0 | 2.288166 | 0.776688 | -2.50 | 0.0 | 0.0 | 0.00 | 0.0 | 0.00 | |
| **Airport_fee** | 3185698.0 | 0.150005 | 0.499241 | -1.75 | 0.0 | 0.0 | 0.00 | 0.0 | 0.00 | |

In [49]:
```
# We have seen from the above numerical univariate analysis that we have c
df1[df1["trip_distance"]>20.06]
```

Out[49]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **36** | 2 | 2023-11-01 00:21:03 | 2023-11-01 01:00:35 | 1.0 | 22.01 |
| **79** | 1 | 2023-11-01 00:34:02 | 2023-11-01 01:16:06 | 2.0 | 22.20 |
| **113** | 1 | 2023-11-01 00:33:38 | 2023-11-01 01:10:01 | 1.0 | 20.10 |
| **180** | 2 | 2023-11-01 00:14:23 | 2023-11-01 01:15:09 | 2.0 | 22.01 |
| **409** | 2 | 2023-11-01 00:48:06 | 2023-11-01 01:48:18 | 1.0 | 25.46 |
| **...** | ... | ... | ... | ... | ... |
| **3206671** | 2 | 2023-11-30 23:56:37 | 2023-12-01 00:24:07 | 2.0 | 25.76 |
| **3206841** | 2 | 2023-11-30 22:58:52 | 2023-11-30 23:34:35 | 1.0 | 20.41 |
| **3206859** | 2 | 2023-11-30 23:55:33 | 2023-12-01 00:35:34 | 2.0 | 32.81 |
| **3206890** | 2 | 2023-11-30 23:53:29 | 2023-12-01 00:29:44 | 4.0 | 20.92 |
| **3207033** | 2 | 2023-11-30 23:36:01 | 2023-12-01 00:27:27 | 1.0 | 27.89 |

31759 rows × 17 columns

We can clearly see that there are only 31759 who have had a trip distance of more than 20.6 which is less than 1% so we will drop those rows for higher accuracy.

In [50]:
```
df1=df1.drop(df1[df1["trip_distance"]>20.06].index)
```

In [51]:
```
df1.shape
```

Out[51]:
```
(3153939, 17)
```

**Fare amount**

In [52]:
```
1 df1[df1["fare_amount"]<-3.7]
```

Out[52]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **51** | 2 | 2023-11-01 00:23:27 | 2023-11-01 00:27:43 | 1.0 | 1.45 |
| **54** | 2 | 2023-11-01 00:23:03 | 2023-11-01 00:56:14 | 2.0 | 9.95 |
| **107** | 2 | 2023-11-01 00:13:19 | 2023-11-01 01:03:26 | 1.0 | 14.10 |
| **205** | 2 | 2023-11-01 00:53:33 | 2023-11-01 00:56:27 | 2.0 | 0.67 |
| **275** | 2 | 2023-11-01 00:27:57 | 2023-11-01 00:47:39 | 1.0 | 3.94 |
| **...** | ... | ... | ... | ... | ... |
| **3206749** | 2 | 2023-11-30 23:44:38 | 2023-12-01 00:13:54 | 1.0 | 8.50 |
| **3206857** | 2 | 2023-11-30 23:40:48 | 2023-11-30 23:50:00 | 2.0 | 0.68 |
| **3206908** | 2 | 2023-11-30 23:28:32 | 2023-11-30 23:40:13 | 1.0 | 1.01 |
| **3206910** | 2 | 2023-11-30 23:42:31 | 2023-12-01 00:03:06 | 1.0 | 4.07 |
| **3206916** | 2 | 2023-11-30 23:56:05 | 2023-12-01 00:00:27 | 1.0 | 0.61 |

31007 rows × 17 columns

In [53]:
```
1 (31007/3153939)*100
2 # Only 31007 has the fare less than -3.7, which is even less than 1% so we
```

Out[53]:
```
0.9831198383989038
```

In [54]:
```
1 df1=df1.drop(df1[df1["fare_amount"]<-3.7].index)
```

In [55]:
```
1 df1.shape
```

Out[55]:
```
(3122932, 17)
```

In [56]:
```
1 df1[df1["fare_amount"]>79]
```

Out[56]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **1229** | 2 | 2023-11-01 00:08:05 | 2023-11-01 00:08:16 | 1.0 | 0.00 |
| **1627** | 1 | 2023-11-01 00:03:14 | 2023-11-01 01:52:55 | 1.0 | 18.90 |
| **1677** | 2 | 2023-11-01 00:33:38 | 2023-11-01 00:33:43 | 1.0 | 0.00 |
| **1691** | 2 | 2023-11-01 00:18:57 | 2023-11-01 00:58:01 | 1.0 | 5.84 |
| **2014** | 2 | 2023-11-01 00:16:27 | 2023-11-01 00:16:31 | 1.0 | 0.02 |
| **...** | ... | ... | ... | ... | ... |
| **3206816** | 2 | 2023-11-30 22:44:42 | 2023-11-30 22:45:07 | 1.0 | 0.00 |
| **3206817** | 1 | 2023-11-30 23:05:43 | 2023-11-30 23:35:50 | 1.0 | 13.80 |
| **3206837** | 2 | 2023-11-30 23:35:11 | 2023-11-30 23:35:18 | 2.0 | 0.02 |
| **3206858** | 2 | 2023-11-30 23:40:48 | 2023-11-30 23:50:00 | 2.0 | 0.68 |
| **3206923** | 1 | 2023-11-30 23:27:44 | 2023-12-01 00:10:17 | 1.0 | 18.40 |

19024 rows × 17 columns

In [57]:
```
1 (19024/3122932)*100
2 # Only 19024 has the fare greater than 79, which is even less than 1% so w
```

Out[57]:
```
0.6091711250837354
```

In [58]:
```
1 df1=df1.drop(df1[df1["fare_amount"]>79].index)
```

In [59]:
```
1 df1.shape
```

Out[59]:
```
(3103908, 17)
```

**Extra**

In [60]:
```
1 df1[df1["extra"]<0]
```

Out[60]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **461** | 2 | 2023-11-01 00:03:17 | 2023-11-01 00:04:02 | 2.0 | 0.25 |
| **757** | 2 | 2023-11-01 00:27:41 | 2023-11-01 00:28:55 | 1.0 | 0.17 |
| **1227** | 2 | 2023-11-01 00:04:46 | 2023-11-01 00:04:52 | 1.0 | 0.01 |
| **1831** | 2 | 2023-11-01 00:07:05 | 2023-11-01 00:07:11 | 1.0 | 0.06 |
| **2184** | 2 | 2023-11-01 00:36:38 | 2023-11-01 00:38:06 | 1.0 | 0.07 |
| **...** | ... | ... | ... | ... | ... |
| **3203017** | 2 | 2023-11-30 23:31:03 | 2023-11-30 23:32:38 | 1.0 | 0.27 |
| **3204698** | 2 | 2023-11-30 23:22:05 | 2023-11-30 23:23:02 | 1.0 | 0.00 |
| **3204974** | 2 | 2023-11-30 23:39:37 | 2023-11-30 23:39:44 | 1.0 | 0.00 |
| **3206029** | 2 | 2023-11-30 23:23:41 | 2023-11-30 23:25:14 | 2.0 | 0.12 |
| **3206172** | 2 | 2023-11-30 23:37:59 | 2023-11-30 23:39:50 | 1.0 | 0.14 |

2471 rows × 17 columns

In [61]:
```
1 (2471/3103908)*100
2 # We can see that the extra amount paid to driver with less than 0 is ever
```

Out[61]:
```
0.07960931831742436
```

In [62]:
```
1 df1=df1.drop(df1[df1["extra"]<0].index)
```

In [63]:
```
1 df1.shape
```

Out[63]:
```
(3101437, 17)
```

In [64]:
```
1 df1[df1["extra"]>7.5]
```

Out[64]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **3819** | 1 | 2023-11-01 02:52:36 | 2023-11-01 03:20:21 | 1.0 | 11.5 |
| **4738** | 1 | 2023-11-01 02:45:18 | 2023-11-01 03:05:30 | 1.0 | 9.6 |
| **4919** | 1 | 2023-11-01 03:46:47 | 2023-11-01 04:07:01 | 1.0 | 9.1 |
| **4979** | 1 | 2023-11-01 03:54:33 | 2023-11-01 04:16:21 | 1.0 | 10.6 |
| **5315** | 1 | 2023-11-01 03:42:33 | 2023-11-01 04:00:40 | 1.0 | 5.1 |
| **...** | ... | ... | ... | ... | ... |
| **3206883** | 1 | 2023-11-30 23:13:27 | 2023-11-30 23:28:31 | 1.0 | 7.7 |
| **3206992** | 1 | 2023-11-30 23:16:56 | 2023-11-30 23:37:03 | 1.0 | 9.0 |
| **3206994** | 1 | 2023-11-30 23:42:51 | 2023-12-01 00:07:22 | 1.0 | 10.7 |
| **3207017** | 1 | 2023-11-30 23:36:34 | 2023-12-01 00:09:18 | 4.0 | 12.7 |
| **3207021** | 1 | 2023-11-30 23:23:41 | 2023-11-30 23:49:20 | 1.0 | 9.8 |

24350 rows × 17 columns

In [65]:
```
1 (24350/3101437)*100
2 # we will delete these rows as they are outliers also the extras are less
```

Out[65]:
```
0.7851199298905636
```

In [66]:
```
1 df1=df1.drop(df1[df1["extra"]>7.5].index)
```

In [67]:
```
1 df1.shape
```

Out[67]:
```
(3077087, 17)
```

**MTA tax**

In [68]:
```
1 df1[df1["mta_tax"]>.5]
```

Out[68]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **24099** | 1 | 2023-11-01 10:08:53 | 2023-11-01 10:26:44 | 1.0 | 6.5 |
| **1542180** | 1 | 2023-11-14 17:47:19 | 2023-11-14 18:04:03 | 1.0 | 3.0 |
| **2483903** | 1 | 2023-11-23 12:28:51 | 2023-11-23 12:49:41 | 3.0 | 1.1 |

In [69]:
```
# Removing these rows
df1=df1.drop(df1[df1["mta_tax"]>.5].index)
```

In [70]:
```
df1.shape
```

Out[70]:
```
(3077084, 17)
```

**Tip amount**

In [71]:
```
1 df1[df1["tip_amount"]<0]
```

Out[71]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **54795** | 2 | 2023-11-01 15:11:32 | 2023-11-01 15:11:39 | 1.0 | 0.0 |
| **137971** | 2 | 2023-11-02 09:17:11 | 2023-11-02 09:17:29 | 1.0 | 0.0 |
| **138690** | 2 | 2023-11-02 09:27:41 | 2023-11-02 09:27:57 | 1.0 | 0.0 |
| **165831** | 2 | 2023-11-02 13:19:53 | 2023-11-02 13:20:04 | 1.0 | 0.0 |
| **165833** | 2 | 2023-11-02 13:32:33 | 2023-11-02 13:32:44 | 1.0 | 0.0 |
| **173303** | 2 | 2023-11-02 15:20:37 | 2023-11-02 15:20:47 | 1.0 | 0.0 |
| **179307** | 2 | 2023-11-02 15:25:48 | 2023-11-02 15:26:09 | 1.0 | 0.0 |
| **295388** | 2 | 2023-11-03 15:58:30 | 2023-11-03 15:58:42 | 1.0 | 0.0 |
| **296812** | 2 | 2023-11-03 15:25:43 | 2023-11-03 15:25:50 | 1.0 | 0.0 |
| **408677** | 2 | 2023-11-04 13:05:08 | 2023-11-04 13:05:22 | 1.0 | 0.0 |
| **416401** | 2 | 2023-11-04 14:27:51 | 2023-11-04 14:27:51 | 1.0 | 0.0 |
| **597045** | 2 | 2023-11-06 09:24:46 | 2023-11-06 09:25:03 | 1.0 | 0.0 |
| **730374** | 2 | 2023-11-07 15:16:34 | 2023-11-07 15:16:40 | 1.0 | 0.0 |
| **820759** | 2 | 2023-11-08 11:36:42 | 2023-11-08 11:37:13 | 1.0 | 0.0 |
| **926581** | 2 | 2023-11-09 08:44:26 | 2023-11-09 08:44:41 | 1.0 | 0.0 |
| **963387** | 2 | 2023-11-09 14:28:14 | 2023-11-09 14:28:25 | 1.0 | 0.0 |
| **1322445** | 2 | 2023-11-12 13:45:40 | 2023-11-12 13:45:55 | 1.0 | 0.0 |
| **1361789** | 2 | 2023-11-12 19:07:07 | 2023-11-12 19:07:15 | 1.0 | 0.0 |
| **1414495** | 2 | 2023-11-13 13:30:32 | 2023-11-13 13:31:44 | 1.0 | 0.0 |
| **1512437** | 2 | 2023-11-14 12:25:11 | 2023-11-14 12:26:08 | 1.0 | 0.0 |
| **1641598** | 2 | 2023-11-15 14:42:35 | 2023-11-15 14:43:06 | 1.0 | 0.0 |
| **1851493** | 2 | 2023-11-17 08:49:03 | 2023-11-17 08:49:30 | 1.0 | 0.0 |
| **1980063** | 2 | 2023-11-18 09:57:25 | 2023-11-18 09:57:32 | 1.0 | 0.0 |
| **2010129** | 2 | 2023-11-18 14:46:25 | 2023-11-18 14:46:59 | 1.0 | 0.0 |
| **2010177** | 2 | 2023-11-18 14:18:52 | 2023-11-18 14:19:18 | 1.0 | 0.0 |
| **2026387** | 2 | 2023-11-18 16:24:46 | 2023-11-18 16:25:21 | 1.0 | 0.0 |
| **2113172** | 2 | 2023-11-19 12:51:13 | 2023-11-19 12:51:19 | 1.0 | 0.0 |
| **2121281** | 2 | 2023-11-19 13:57:58 | 2023-11-19 13:58:07 | 1.0 | 0.0 |
| **2181500** | 2 | 2023-11-20 07:54:12 | 2023-11-20 07:54:18 | 1.0 | 0.0 |
| **2613720** | 2 | 2023-11-25 10:03:56 | 2023-11-25 10:04:20 | 1.0 | 0.0 |
| **2698206** | 2 | 2023-11-26 09:26:58 | 2023-11-26 09:27:28 | 1.0 | 0.0 |
| **2811275** | 2 | 2023-11-27 15:27:57 | 2023-11-27 15:28:10 | 1.0 | 0.0 |
| **2821562** | 2 | 2023-11-27 16:00:21 | 2023-11-27 16:00:33 | 1.0 | 0.0 |
| **2891251** | 2 | 2023-11-28 11:43:53 | 2023-11-28 11:44:50 | 1.0 | 0.0 |
| **2894177** | 2 | 2023-11-28 12:29:49 | 2023-11-28 12:31:14 | 1.0 | 0.0 |

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **2910770** | 2 | 2023-11-28 15:53:01 | 2023-11-28 15:53:13 | 1.0 | 0.0 |

In [72]:
```
# There are very less low with less than 0 tip so we will delete it.
df1=df1.drop(df1[df1["tip_amount"]<0].index)
```

In [73]:
```
df1[df1["tip_amount"]>18.28]
```

Out[73]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **85** | 2 | 2023-11-01 00:58:18 | 2023-11-01 01:22:26 | 1.0 | 4.92 |
| **144** | 2 | 2023-11-01 01:03:35 | 2023-11-01 01:14:51 | 1.0 | 1.54 |
| **200** | 2 | 2023-11-01 00:19:37 | 2023-11-01 01:00:27 | 1.0 | 17.93 |
| **613** | 1 | 2023-11-01 00:14:34 | 2023-11-01 01:05:24 | 2.0 | 17.70 |
| **621** | 2 | 2023-11-01 00:43:15 | 2023-11-01 01:24:01 | 2.0 | 18.31 |
| **...** | ... | ... | ... | ... | ... |
| **3206231** | 2 | 2023-11-30 23:44:22 | 2023-12-01 00:15:21 | 1.0 | 19.72 |
| **3206639** | 2 | 2023-11-30 23:14:47 | 2023-11-30 23:45:59 | 1.0 | 18.78 |
| **3206711** | 2 | 2023-11-30 23:56:14 | 2023-12-01 00:08:25 | 1.0 | 2.99 |
| **3206771** | 2 | 2023-11-30 23:51:53 | 2023-12-01 00:17:07 | 1.0 | 9.45 |
| **3206900** | 2 | 2023-11-30 23:13:14 | 2023-11-30 23:42:50 | 1.0 | 15.88 |

16560 rows × 17 columns

In [74]:
```
(16560/3077084)*100
#since it is less than 1% we will delete it.
```

Out[74]: 0.5381718536120561

In [75]:
```
df1=df1.drop(df1[df1["tip_amount"]>18.28].index)
```

In [76]:
```
df1.shape
```

Out[76]: (3060488, 17)

### Tolls Amount

In [77]:
```
1 df1[df1["tolls_amount"]<0]
```

Out[77]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **546366** | 2 | 2023-11-05 16:28:52 | 2023-11-05 16:29:04 | 1.0 | 0.00 |
| **552281** | 2 | 2023-11-05 17:46:27 | 2023-11-05 17:47:28 | 1.0 | 0.00 |
| **1296010** | 2 | 2023-11-12 07:57:03 | 2023-11-12 07:58:54 | 1.0 | 0.00 |
| **1417866** | 2 | 2023-11-13 13:59:14 | 2023-11-13 14:00:17 | 1.0 | 0.00 |
| **1534455** | 2 | 2023-11-14 15:19:54 | 2023-11-14 16:07:03 | 1.0 | 17.46 |
| **1814813** | 2 | 2023-11-16 20:00:49 | 2023-11-16 20:01:10 | 2.0 | 0.00 |
| **1982160** | 2 | 2023-11-18 10:26:51 | 2023-11-18 10:27:43 | 2.0 | 0.07 |
| **2181498** | 2 | 2023-11-20 07:52:00 | 2023-11-20 07:52:42 | 1.0 | 0.00 |
| **2793664** | 2 | 2023-11-27 11:32:32 | 2023-11-27 11:32:56 | 1.0 | 0.00 |

In [78]:
```
1 df1=df1.drop(df1[df1["tolls_amount"]<0].index)
```

In [79]:
```
1 df1[df1["tolls_amount"]>6.94]
```

Out[79]:

| | VendorID | tpep_pickup_datetime | tpep_dropoff_datetime | passenger_count | trip_distance |
|---|---|---|---|---|---|
| **151** | 2 | 2023-11-01 00:06:17 | 2023-11-01 00:06:41 | 2.0 | 0.28 |
| **480** | 2 | 2023-11-01 00:26:59 | 2023-11-01 00:45:35 | 1.0 | 9.47 |
| **819** | 1 | 2023-11-01 00:47:04 | 2023-11-01 01:33:10 | 2.0 | 3.00 |
| **2161** | 2 | 2023-11-01 00:29:57 | 2023-11-01 00:59:52 | 5.0 | 16.90 |
| **2472** | 2 | 2023-11-01 01:48:04 | 2023-11-01 02:16:55 | 1.0 | 15.36 |
| **...** | ... | ... | ... | ... | ... |
| **3203080** | 2 | 2023-11-30 23:06:41 | 2023-11-30 23:25:40 | 1.0 | 8.82 |
| **3204166** | 2 | 2023-11-30 23:04:46 | 2023-11-30 23:42:25 | 1.0 | 18.63 |
| **3206229** | 2 | 2023-11-30 23:28:55 | 2023-12-01 00:06:24 | 1.0 | 14.59 |
| **3206825** | 1 | 2023-11-30 23:45:01 | 2023-12-01 00:14:56 | 1.0 | 5.10 |
| **3206927** | 1 | 2023-11-30 23:20:13 | 2023-12-01 00:14:23 | 2.0 | 0.00 |

4516 rows × 17 columns

In [80]:
```
1 (4516/3060488)*100
2 # We will delete these rows
```

Out[80]:
```
0.14755816719425136
```

In [81]:
```
df1=df1.drop(df1[df1["tolls_amount"]>6.94].index)
```

In [82]:
```
df1.shape
```

Out[82]: (3055963, 17)

In [83]:
```
df1.describe(percentiles=[.01,.02,.03,.04,.05,.1,.25,.5,.75,.9,.95,.96,.97
```

Out[83]:

| | count | mean | std | min | 1% | 2% | 3% | 4% | 5% |
|---|---|---|---|---|---|---|---|---|---|
| **VendorID** | 3055963.0 | 1.767355 | 0.422518 | 1.00 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 |
| **passenger_count** | 3055963.0 | 1.357573 | 0.869769 | 0.00 | 0.0 | 1.00 | 1.00 | 1.00 | 1.00 |
| **trip_distance** | 3055963.0 | 2.948851 | 3.634133 | 0.00 | 0.0 | 0.28 | 0.38 | 0.42 | 0.49 |
| **RatecodeID** | 3055963.0 | 1.043770 | 0.288337 | 1.00 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 |
| **payment_type** | 3055963.0 | 1.195899 | 0.469545 | 1.00 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 |
| **fare_amount** | 3055963.0 | 18.240963 | 14.398333 | -3.70 | 4.4 | 5.10 | 5.10 | 5.80 | 5.80 |
| **extra** | 3055963.0 | 1.526355 | 1.691570 | 0.00 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 |
| **mta_tax** | 3055963.0 | 0.497400 | 0.041547 | -0.50 | 0.5 | 0.50 | 0.50 | 0.50 | 0.50 |
| **tip_amount** | 3055963.0 | 3.429283 | 3.294571 | 0.00 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 |
| **tolls_amount** | 3055963.0 | 0.431328 | 1.672591 | 0.00 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 |
| **improvement_surcharge** | 3055963.0 | 0.997991 | 0.061181 | -1.00 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 |
| **total_amount** | 3055963.0 | 27.046327 | 18.475121 | -10.20 | 8.7 | 9.80 | 10.50 | 11.10 | 11.50 |
| **congestion_surcharge** | 3055963.0 | 2.361353 | 0.578065 | -2.50 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 |
| **Airport_fee** | 3055963.0 | 0.124255 | 0.450196 | -1.75 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 |

## Bivariate

In [84]:
```
# CAT-CAT
```

### Vendor ID-PU LOcation ID

In [85]:
```
from scipy.stats import chi2_contingency
```

In [86]:
```
df_chi1 = pd.crosstab(df1["VendorID"],df1["RatecodeID"])
df_chi1
```

Out[86]:

| RatecodeID | 1.0 | 2.0 | 3.0 | 4.0 | 5.0 | 6.0 |
|---|---|---|---|---|---|---|
| VendorID | | | | | | |
| 1 | 693652 | 14839 | 230 | 449 | 1779 | 4 |
| 2 | 2261582 | 73494 | 943 | 1369 | 7620 | 2 |

```
In Chi Square Test:
Null hypothesis: There is no relationship between Vendor ID and PULocationID
Alternate hypothesis: There is relationship between Vendor ID and PULocationID

    Test: Chi square test
Confidence_interval: 95%
alpha / threshold: 5% = 0.05

    Decision rule:
if p value > alpha, then accept null hypothesis
if p value <= alpha, then reject null hypothesis and accept alternate hypothesis

Conclusion: As p value <= alpha, there is no statistical significant evidence in support of Null Hypothesis, So we fail to
accept null hypothesis.
We accept alternative hypothesis which means is relationship between Manufacturer and Vehicle_Type.

stats.chi2_contingency(vt_vs_manufctrr)
```

In [87]:
```
chi2_contingency(df_chi1)
```

Out[87]:
```
Chi2ContingencyResult(statistic=2255.9595087734547, pvalue=0.0, dof=5, expected_freq=array([[6.87518952e+05, 2.05501871e+04, 2.72892005e+02, 4.22947710e+02,
        2.18662570e+03, 1.39586703e+00],
       [2.26771505e+06, 6.77828129e+04, 9.00107995e+02, 1.39505229e+03,
        7.21237430e+03, 4.60413297e+00]]))
```

## Cat Num

In [88]:
```
1 df1.groupby(["store_and_fwd_flag"]).agg({"total_amount":["min","mean","ma
```

Out[88]:

| | total_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| store_and_fwd_flag | | | | |
| N | -10.2 | 27.054928 | 114.34 | 3042529 |
| Y | -4.5 | 25.098341 | 105.84 | 13434 |

### *Vendor ID*

In [89]:
```
1 df1.groupby(["VendorID"]).agg({"passenger_count":["min","mean","max","cou
```

Out[89]:

| | passenger_count | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 1.218245 | 7.0 | 710953 |
| 2 | 0.0 | 1.399814 | 8.0 | 2345010 |

In [90]:
```
1 df1.groupby(["VendorID"]).agg({"trip_distance":["min","mean","max","count"
```

Out[90]:

| | trip_distance | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 2.525533 | 20.00 | 710953 |
| 2 | 0.0 | 3.077191 | 20.06 | 2345010 |

In [91]:
```
1 df1.groupby(["VendorID"]).agg({"fare_amount":["min","mean","max","count"]}
```

Out[91]:

| | fare_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 16.410027 | 79.0 | 710953 |
| 2 | -3.7 | 18.796061 | 79.0 | 2345010 |

In [92]:
```
1 df1.groupby(["VendorID"]).agg({"extra":["min","mean","max","count"]})
```

Out[92]:

| | extra | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 3.225501 | 7.5 | 710953 |
| 2 | 0.0 | 1.011213 | 7.5 | 2345010 |

In [93]:
```
1 df1.groupby(["VendorID"]).agg({"mta_tax":["min","mean","max","count"]})
```

Out[93]:

| | mta_tax | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 0.497861 | 0.5 | 710953 |
| 2 | -0.5 | 0.497260 | 0.5 | 2345010 |

In [94]:
```
1 df1.groupby(["VendorID"]).agg({"tip_amount":["min","mean","max","count"]})
```

Out[94]:

| | tip_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 3.058068 | 18.25 | 710953 |
| 2 | 0.0 | 3.541827 | 18.28 | 2345010 |

In [95]:
```
1 df1.groupby(["VendorID"]).agg({"tolls_amount":["min","mean","max","count"]
```

Out[95]:

| | tolls_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 0.230304 | 6.94 | 710953 |
| 2 | 0.0 | 0.492273 | 6.94 | 2345010 |

In [96]:
```
1 df1.groupby(["VendorID"]).agg({"improvement_surcharge":["min","mean","max"
```

Out[96]:

| | improvement_surcharge | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 0.999423 | 1.0 | 710953 |
| 2 | -1.0 | 0.997557 | 1.0 | 2345010 |

In [97]:
```
1 df1.groupby(["VendorID"]).agg({"total_amount":["min","mean","max","count"]
```

Out[97]:

| | total_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 24.421184 | 109.54 | 710953 |
| 2 | -10.2 | 27.842210 | 114.34 | 2345010 |

In [98]:
```
1 df1.groupby(["VendorID"]).agg({"congestion_surcharge":["min","mean","max",
```

Out[98]:

| | congestion_surcharge | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.0 | 2.378670 | 2.5 | 710953 |
| 2 | -2.5 | 2.356103 | 2.5 | 2345010 |

In [99]:
```
1 df1.groupby(["VendorID"]).agg({"Airport_fee":["min","mean","max","count"]]
```

Out[99]:

| | Airport_fee | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| VendorID | | | | |
| 1 | 0.00 | 0.055927 | 1.75 | 710953 |
| 2 | -1.75 | 0.144970 | 1.75 | 2345010 |

### RAtecode ID

In [100]:
```
1 df1.groupby(["RatecodeID"]).agg({"passenger_count":["min","mean","max","co
```

Out[100]:

| | passenger_count | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | 0.0 | 1.350790 | 7.0 | 2955234 |
| 2.0 | 0.0 | 1.562904 | 6.0 | 88333 |
| 3.0 | 0.0 | 1.695652 | 6.0 | 1173 |
| 4.0 | 0.0 | 1.364136 | 6.0 | 1818 |
| 5.0 | 0.0 | 1.517183 | 8.0 | 9399 |
| 6.0 | 1.0 | 1.333333 | 3.0 | 6 |

In [101]:
```
1 df1.groupby(["RatecodeID"]).agg({"trip_distance":["min","mean","max","cour
```

Out[101]:

| | trip_distance | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | 0.0 | 2.541288 | 20.06 | 2955234 |
| 2.0 | 0.0 | 16.573583 | 20.06 | 88333 |
| 3.0 | 0.0 | 2.475584 | 18.10 | 1173 |
| 4.0 | 0.0 | 10.163806 | 19.60 | 1818 |
| 5.0 | 0.0 | 1.713183 | 20.05 | 9399 |
| 6.0 | 0.0 | 0.115000 | 0.30 | 6 |

In [102]:
```
1 df1.groupby(["RatecodeID"]).agg({"fare_amount":["min","mean","max","count"
```

Out[102]:

| | fare_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | -3.7 | 16.614035 | 78.60 | 2955234 |
| 2.0 | 0.0 | 69.919722 | 75.75 | 88333 |
| 3.0 | 0.0 | 32.788065 | 79.00 | 1173 |
| 4.0 | -3.0 | 52.047690 | 78.60 | 1818 |
| 5.0 | -3.5 | 35.750243 | 79.00 | 9399 |
| 6.0 | 2.5 | 4.745000 | 8.97 | 6 |

In [103]:
```
1 df1.groupby(["RatecodeID"]).agg({"extra":["min","mean","max","count"]})
```

Out[103]:

| | extra | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | 0.0 | 1.532472 | 7.50 | 2955234 |
| 2.0 | 0.0 | 1.419503 | 7.50 | 88333 |
| 3.0 | 0.0 | 1.502344 | 7.50 | 1173 |
| 4.0 | 0.0 | 1.704070 | 7.50 | 1818 |
| 5.0 | 0.0 | 0.576870 | 6.75 | 9399 |
| 6.0 | 0.0 | 0.000000 | 0.00 | 6 |

In [104]:
```
1 df1.groupby(["RatecodeID"]).agg({"mta_tax":["min","mean","max","count"]})
```

Out[104]:

| | mta_tax | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | -0.5 | 0.498914 | 0.5 | 2955234 |
| 2.0 | -0.5 | 0.499287 | 0.5 | 88333 |
| 3.0 | 0.0 | 0.000000 | 0.0 | 1173 |
| 4.0 | -0.5 | 0.495875 | 0.5 | 1818 |
| 5.0 | -0.5 | 0.066124 | 0.5 | 9399 |
| 6.0 | 0.5 | 0.500000 | 0.5 | 6 |

In [105]:
```
1 df1.groupby(["RatecodeID"]).agg({"tip_amount":["min","mean","max","count"]
```

Out[105]:

| | tip_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | 0.0 | 3.226836 | 18.28 | 2955234 |
| 2.0 | 0.0 | 10.068666 | 18.19 | 88333 |
| 3.0 | 0.0 | 2.379318 | 18.26 | 1173 |
| 4.0 | 0.0 | 5.203245 | 18.20 | 1818 |
| 5.0 | 0.0 | 4.475128 | 18.20 | 9399 |
| 6.0 | 0.0 | 0.000000 | 0.00 | 6 |

In [106]:
```
1 df1.groupby(["RatecodeID"]).agg({"tolls_amount":["min","mean","max","count
```

Out[106]:

| | tolls_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | 0.0 | 0.278418 | 6.94 | 2955234 |
| 2.0 | 0.0 | 5.530835 | 6.94 | 88333 |
| 3.0 | 0.0 | 0.696488 | 6.94 | 1173 |
| 4.0 | 0.0 | 0.490523 | 6.94 | 1818 |
| 5.0 | 0.0 | 0.538978 | 6.94 | 9399 |
| 6.0 | 0.0 | 0.000000 | 0.00 | 6 |

In [107]:
```
1 df1.groupby(["RatecodeID"]).agg({"improvement_surcharge":["min","mean","ma
```

Out[107]:

| | improvement_surcharge | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | -1.0 | 0.998014 | 1.0 | 2955234 |
| 2.0 | -1.0 | 0.998917 | 1.0 | 88333 |
| 3.0 | -1.0 | 0.988917 | 1.0 | 1173 |
| 4.0 | -1.0 | 0.997800 | 1.0 | 1818 |
| 5.0 | -1.0 | 0.983147 | 1.0 | 9399 |
| 6.0 | 1.0 | 1.000000 | 1.0 | 6 |

In [108]:
```
1 df1.groupby(["RatecodeID"]).agg({"total_amount":["min","mean","max","count
```

Out[108]:

| | total_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | -10.20 | 25.044966 | 114.34 | 2955234 |
| 2.0 | -5.75 | 91.360232 | 108.23 | 88333 |
| 3.0 | -1.00 | 38.781816 | 111.24 | 1173 |
| 4.0 | -4.00 | 62.255484 | 109.68 | 1818 |
| 5.0 | -7.00 | 43.622269 | 108.53 | 9399 |
| 6.0 | 4.00 | 6.370000 | 10.47 | 6 |

In [109]:
```
1 df1.groupby(["RatecodeID"]).agg({"congestion_surcharge":["min","mean","ma
```

Out[109]:

| | congestion_surcharge | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | -2.5 | 2.369632 | 2.50 | 2955234 |
| 2.0 | -2.5 | 2.294329 | 2.50 | 88333 |
| 3.0 | 0.0 | 0.014919 | 2.50 | 1173 |
| 4.0 | 0.0 | 0.192519 | 2.50 | 1818 |
| 5.0 | -2.5 | 1.101979 | 2.50 | 9399 |
| 6.0 | 0.0 | 0.125000 | 0.75 | 6 |

In [110]:
```
1 df1.groupby(["RatecodeID"]).agg({"Airport_fee":["min","mean","max","count"
```

Out[110]:

| | Airport_fee | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| RatecodeID | | | | |
| 1.0 | -1.75 | 0.091770 | 1.75 | 2955234 |
| 2.0 | -1.75 | 1.174241 | 1.75 | 88333 |
| 3.0 | 0.00 | 0.474425 | 1.75 | 1173 |
| 4.0 | -1.75 | 1.505501 | 1.75 | 1818 |
| 5.0 | -1.75 | 0.159565 | 1.75 | 9399 |
| 6.0 | 0.00 | 0.000000 | 0.00 | 6 |

### Payment_type

In [111]:
```
1 df1.groupby(["payment_type"]).agg({"passenger_count":["min","mean","max","
```

Out[111]:

| | passenger_count | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | 0.0 | 1.346396 | 8.0 | 2526679 |
| 2 | 0.0 | 1.423460 | 7.0 | 486716 |
| 3 | 0.0 | 1.211435 | 8.0 | 15759 |
| 4 | 0.0 | 1.300757 | 6.0 | 26809 |

In [112]:
```
1 df1.groupby(["payment_type"]).agg({"fare_amount":["min","mean","max","cour
```

Out[112]:

| | fare_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | -3.0 | 18.256882 | 79.0 | 2526679 |
| 2 | -3.7 | 18.395389 | 79.0 | 486716 |
| 3 | -3.7 | 13.383537 | 79.0 | 15759 |
| 4 | -3.7 | 16.792416 | 78.6 | 26809 |

In [113]:
```
1 df1.groupby(["payment_type"]).agg({"extra":["min","mean","max","count"]})
```

Out[113]:

| | extra | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | 0.0 | 1.552957 | 7.5 | 2526679 |
| 2 | 0.0 | 1.388304 | 7.5 | 486716 |
| 3 | 0.0 | 2.004201 | 7.5 | 15759 |
| 4 | 0.0 | 1.244582 | 7.5 | 26809 |

In [114]:
```
1 df1.groupby(["payment_type"]).agg({"mta_tax":["min","mean","max","count"]]
```

Out[114]:

| | mta_tax | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | -0.5 | 0.498457 | 0.5 | 2526679 |
| 2 | -0.5 | 0.497164 | 0.5 | 486716 |
| 3 | -0.5 | 0.423441 | 0.5 | 15759 |
| 4 | -0.5 | 0.445578 | 0.5 | 26809 |

In [115]:
```
1 df1.groupby(["payment_type"]).agg({"fare_amount":["min","mean","max","cour
```

Out[115]:

| | fare_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | -3.0 | 18.256882 | 79.0 | 2526679 |
| 2 | -3.7 | 18.395389 | 79.0 | 486716 |
| 3 | -3.7 | 13.383537 | 79.0 | 15759 |
| 4 | -3.7 | 16.792416 | 78.6 | 26809 |

In [116]:
```
1 df1.groupby(["payment_type"]).agg({"tip_amount":["min","mean","max","count
```

Out[116]:

| | tip_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | 0.0 | 4.147532 | 18.28 | 2526679 |
| 2 | 0.0 | 0.000288 | 16.19 | 486716 |
| 3 | 0.0 | 0.003112 | 10.08 | 15759 |
| 4 | 0.0 | 0.003454 | 12.40 | 26809 |

In [117]:
```
1 df1.groupby(["payment_type"]).agg({"tolls_amount":["min","mean","max","cou
```

Out[117]:

| | tolls_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | 0.0 | 0.442765 | 6.94 | 2526679 |
| 2 | 0.0 | 0.387713 | 6.94 | 486716 |
| 3 | 0.0 | 0.177086 | 6.94 | 15759 |
| 4 | 0.0 | 0.294666 | 6.94 | 26809 |

In [118]:
```
1 df1.groupby(["payment_type"]).agg({"improvement_surcharge":["min","mean","
```

Out[118]:

| | improvement_surcharge | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | -1.0 | 0.999861 | 1.0 | 2526679 |
| 2 | -1.0 | 0.997196 | 1.0 | 486716 |
| 3 | -1.0 | 0.872581 | 1.0 | 15759 |
| 4 | -1.0 | 0.909918 | 1.0 | 26809 |

In [119]:
```
1 df1.groupby(["payment_type"]).agg({"total_amount":["min","mean","max","cou
```

Out[119]:

| | total_amount | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | -7.0 | 27.849127 | 114.34 | 2526679 |
| 2 | -10.2 | 23.494842 | 98.79 | 486716 |
| 3 | -10.2 | 17.481652 | 89.44 | 15759 |
| 4 | -10.2 | 21.483882 | 94.09 | 26809 |

In [120]:
```
1 df1.groupby(["payment_type"]).agg({"congestion_surcharge":["min","mean","r
```

Out[120]:

| | congestion_surcharge | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | -2.5 | 2.395628 | 2.5 | 2526679 |
| 2 | -2.5 | 2.226022 | 2.5 | 486716 |
| 3 | -2.5 | 1.722508 | 2.5 | 15759 |
| 4 | -2.5 | 1.963520 | 2.5 | 26809 |

In [121]:
```
1 df1.groupby(["payment_type"]).agg({"Airport_fee":["min","mean","max","cour
```

Out[121]:

| | Airport_fee | | | |
|---|---|---|---|---|
| | min | mean | max | count |
| payment_type | | | | |
| 1 | 0.00 | 0.116712 | 1.75 | 2526679 |
| 2 | -1.75 | 0.160695 | 1.75 | 486716 |
| 3 | -1.75 | 0.139809 | 1.75 | 15759 |
| 4 | -1.75 | 0.164497 | 1.75 | 26809 |

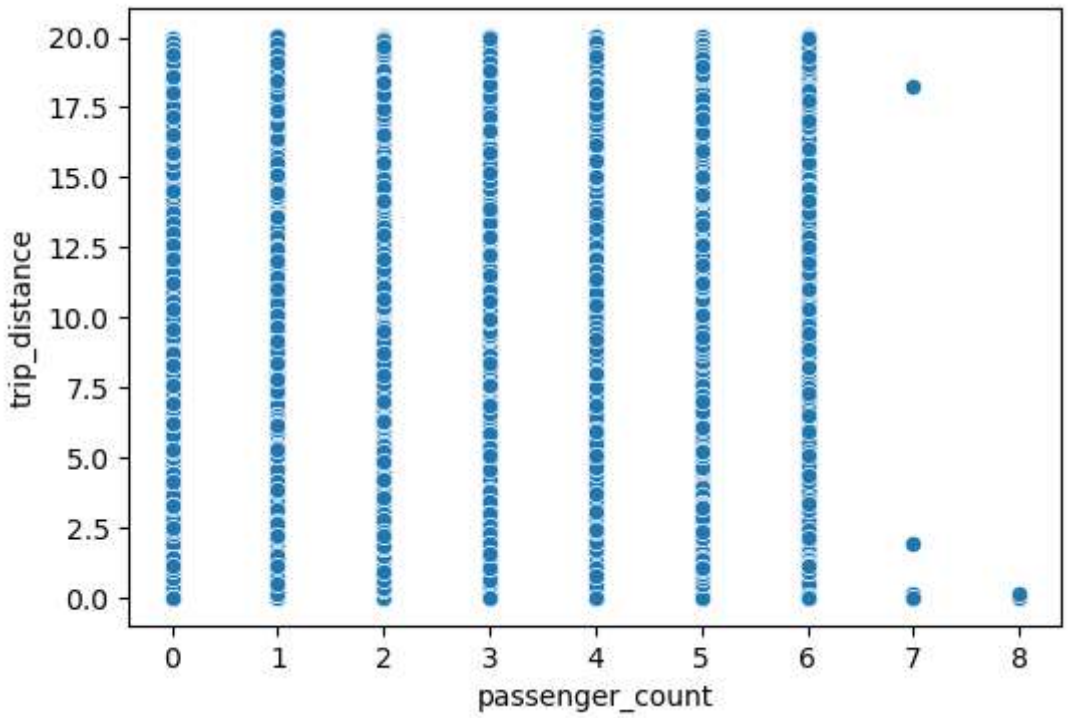
# Num-Num

In [122]:
```
1 df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3055963 entries, 0 to 3207039
Data columns (total 17 columns):
 #   Column                 Dtype
---  ------                 -----
 0   VendorID               int32
 1   tpep_pickup_datetime   datetime64[ns]
 2   tpep_dropoff_datetime  datetime64[ns]
 3   passenger_count        float64
 4   trip_distance          float64
 5   RatecodeID             float64
 6   store_and_fwd_flag     object
 7   payment_type           int64
 8   fare_amount            float64
 9   extra                  float64
 10  mta_tax                float64
 11  tip_amount             float64
 12  tolls_amount           float64
 13  improvement_surcharge  float64
 14  total_amount           float64
 15  congestion_surcharge   float64
 16  Airport_fee            float64
dtypes: datetime64[ns](2), float64(12), int32(1), int64(1), object(1)
memory usage: 408.0+ MB
```

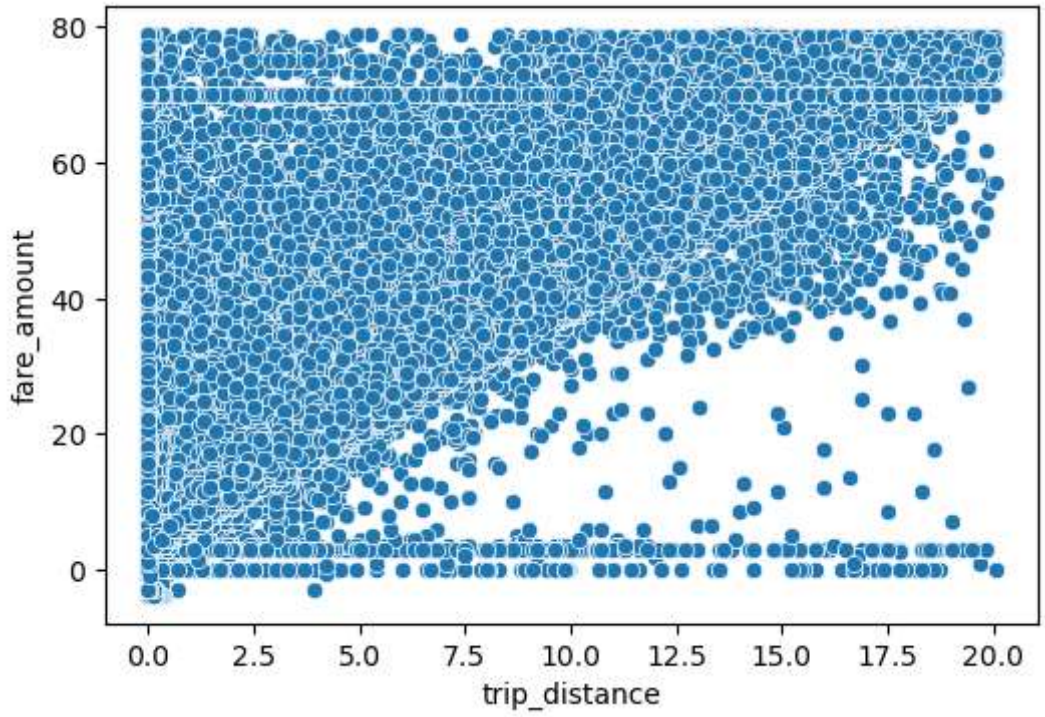
In [123]:
```
plt.figure(figsize=(6,4))
sns.scatterplot(data=df1, x=df1["passenger_count"], y=df1["trip_distance"]
# There is no correlation between passenger count and trip distance which
# should not be correlated.
```

Out[123]: <Axes: xlabel='passenger_count', ylabel='trip_distance'>



In [124]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["fare_amount"])
```

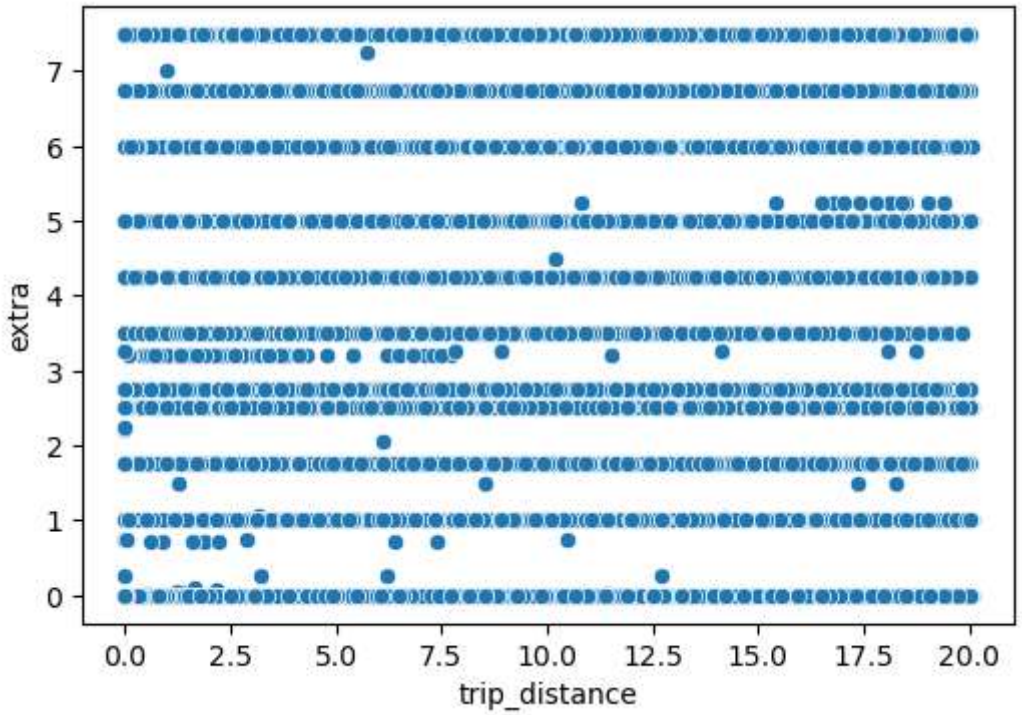
Out[124]: <Axes: xlabel='trip_distance', ylabel='fare_amount'>




**There is no correlation between trip distance and fare amount which is a good thing.**

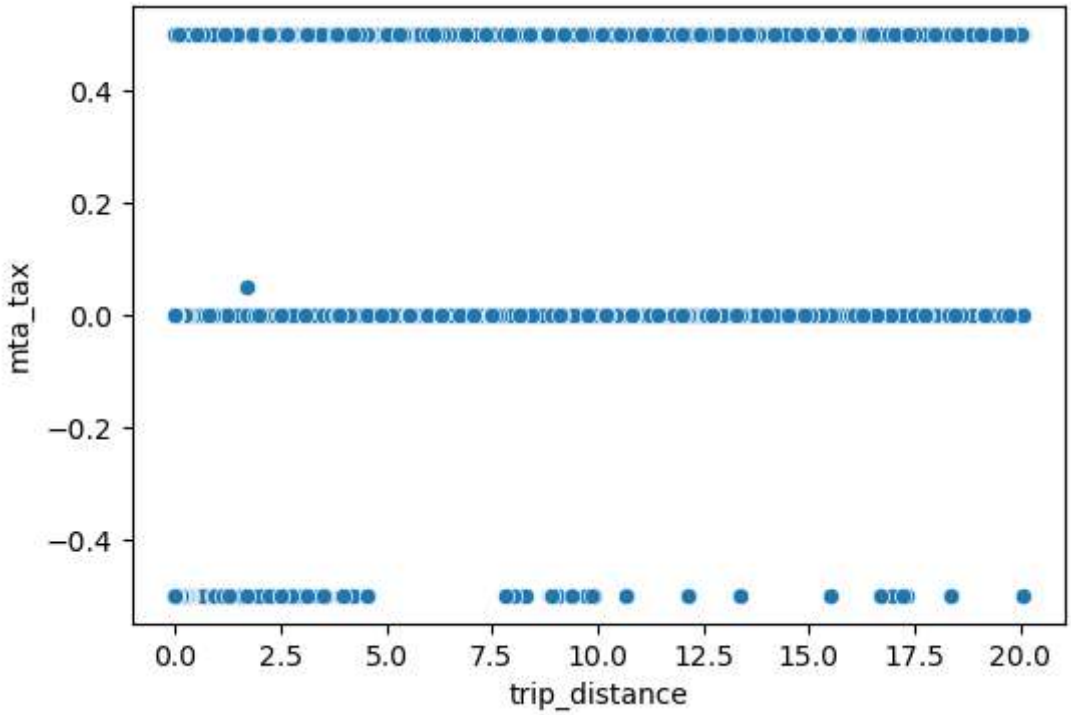
In [125]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["extra"])
```

Out[125]: <Axes: xlabel='trip_distance', ylabel='extra'>



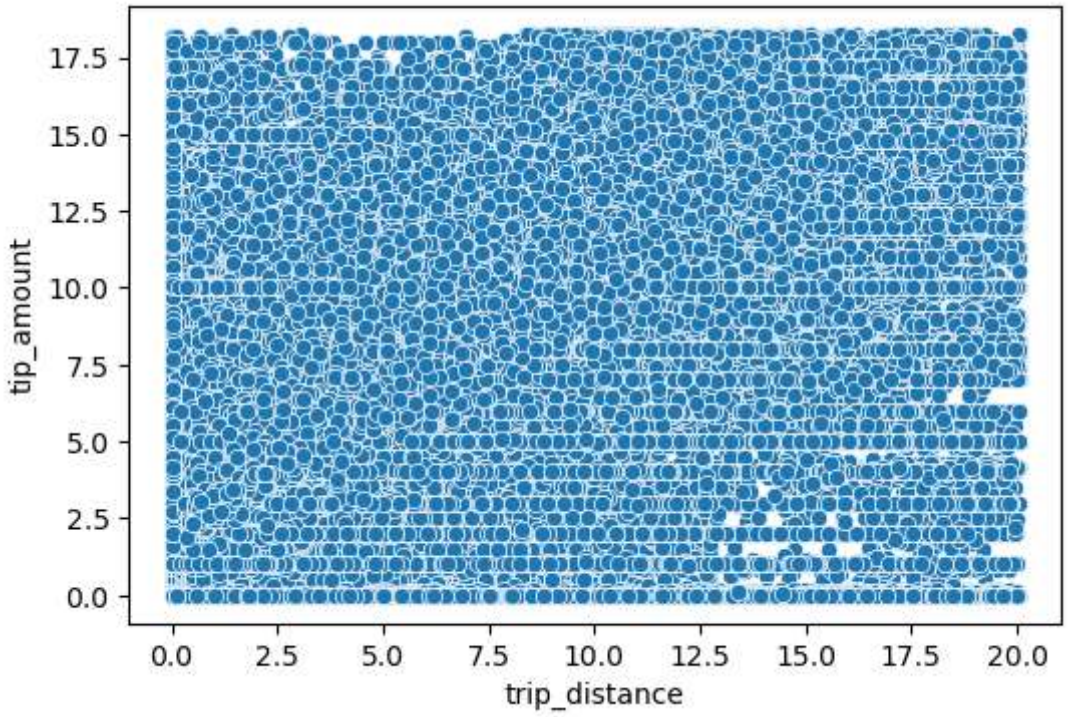

In [126]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["mta_tax"])
```

Out[126]: <Axes: xlabel='trip_distance', ylabel='mta_tax'>



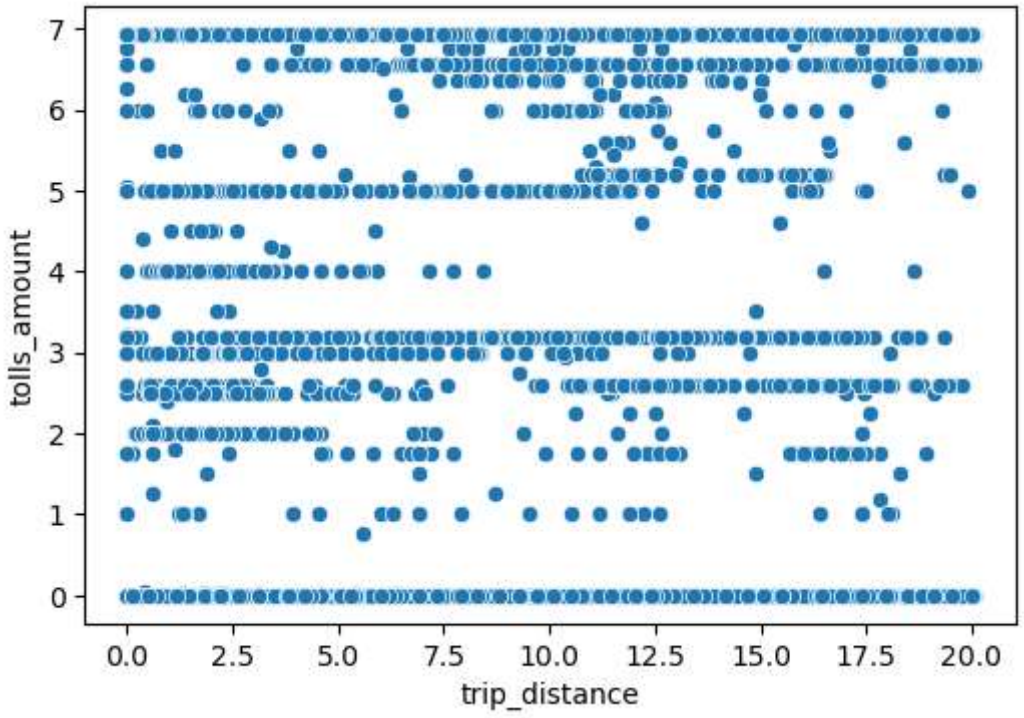
In [127]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["tip_amount"])
```

Out[127]: <Axes: xlabel='trip_distance', ylabel='tip_amount'>



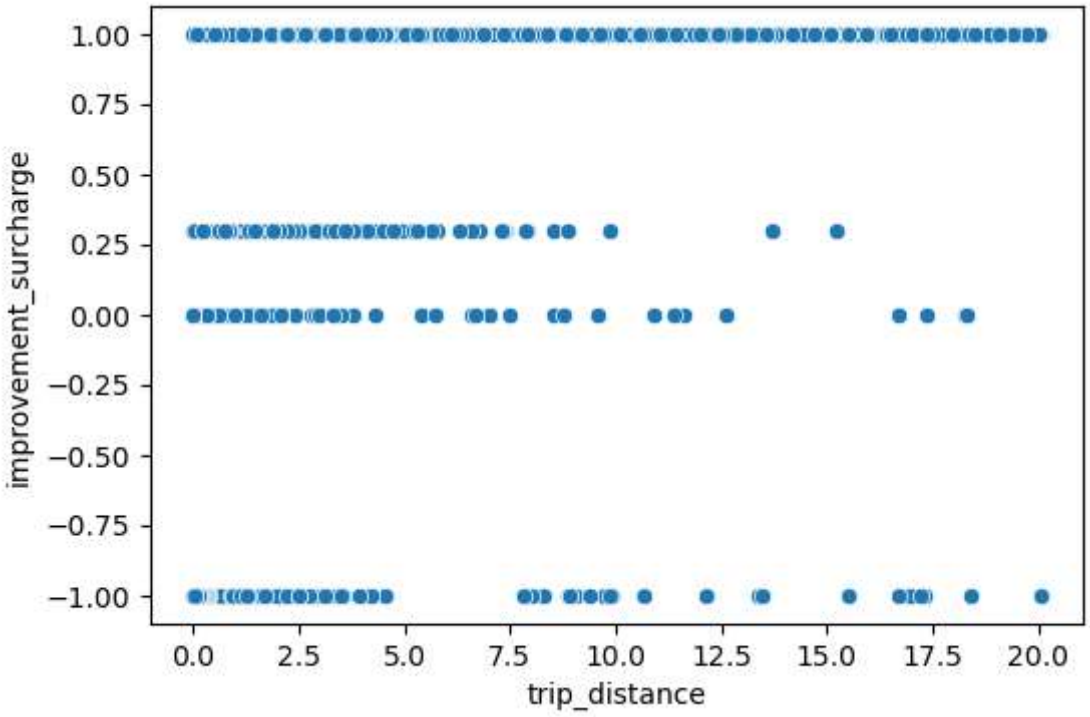

In [128]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["tolls_amount"])
```

Out[128]: <Axes: xlabel='trip_distance', ylabel='tolls_amount'>



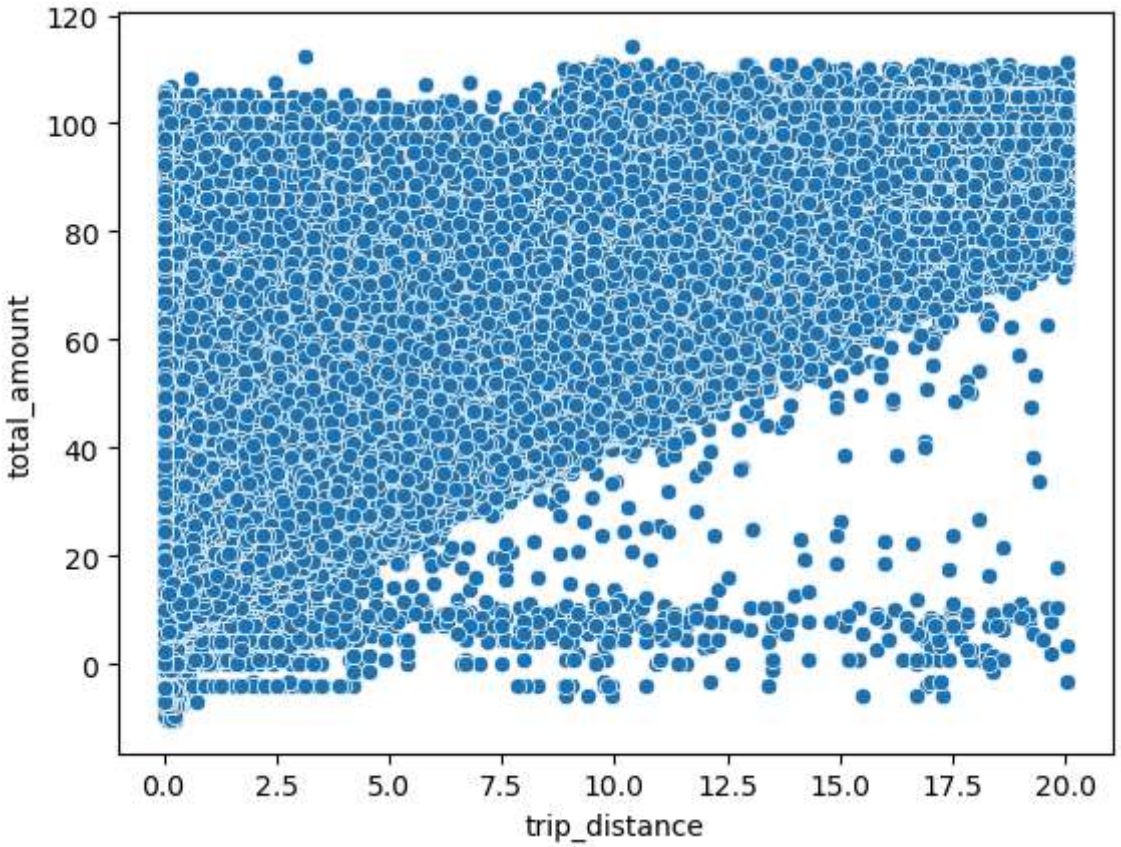
In [129]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["improvement_surch
```

Out[129]: <Axes: xlabel='trip_distance', ylabel='improvement_surcharge'>



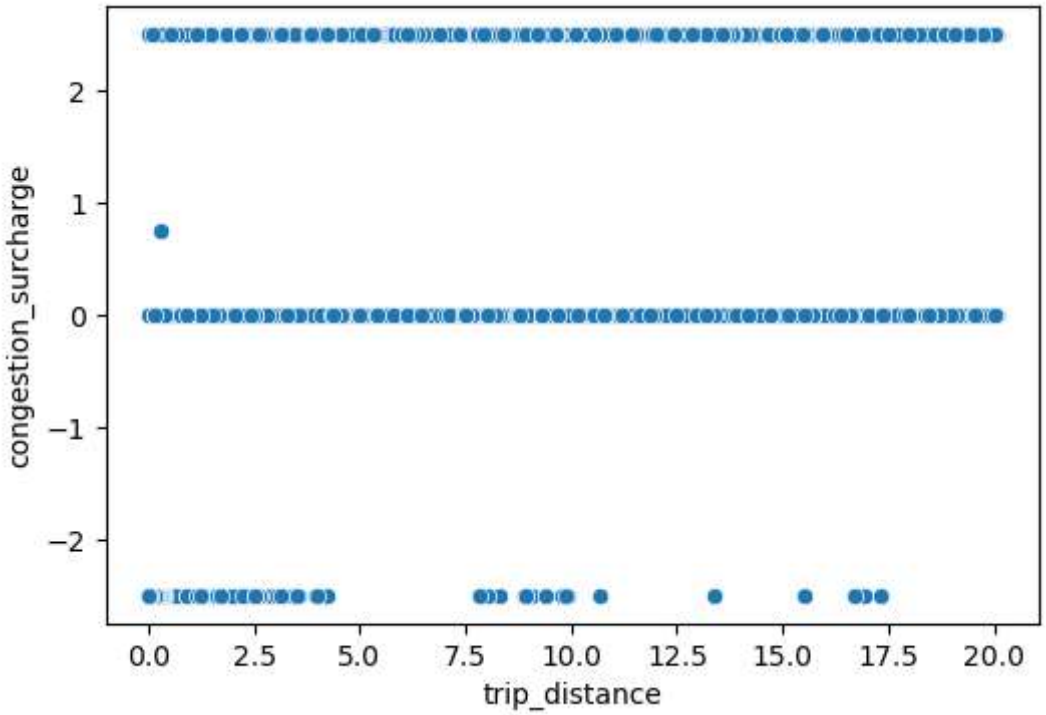
In [130]:
```
# plt.figure(figsize=(6,6))
sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["total_amount"])
```

Out[130]: <Axes: xlabel='trip_distance', ylabel='total_amount'>



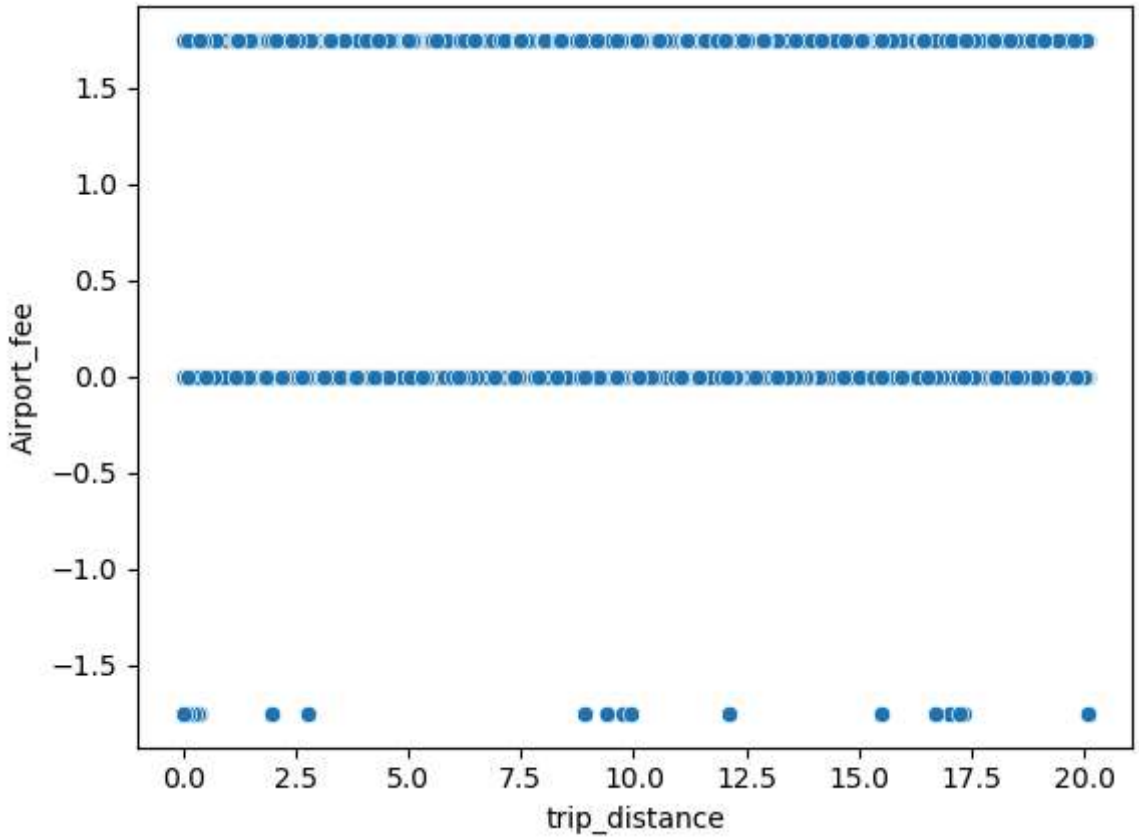
In [131]:
```
1 plt.figure(figsize=(6,4))
2 sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["congestion_surcha
```

Out[131]: <Axes: xlabel='trip_distance', ylabel='congestion_surcharge'>



In [132]:
```
# plt.figure(figsize=(6,6))
sns.scatterplot(data=df1, x=df1["trip_distance"], y=df1["Airport_fee"])
```

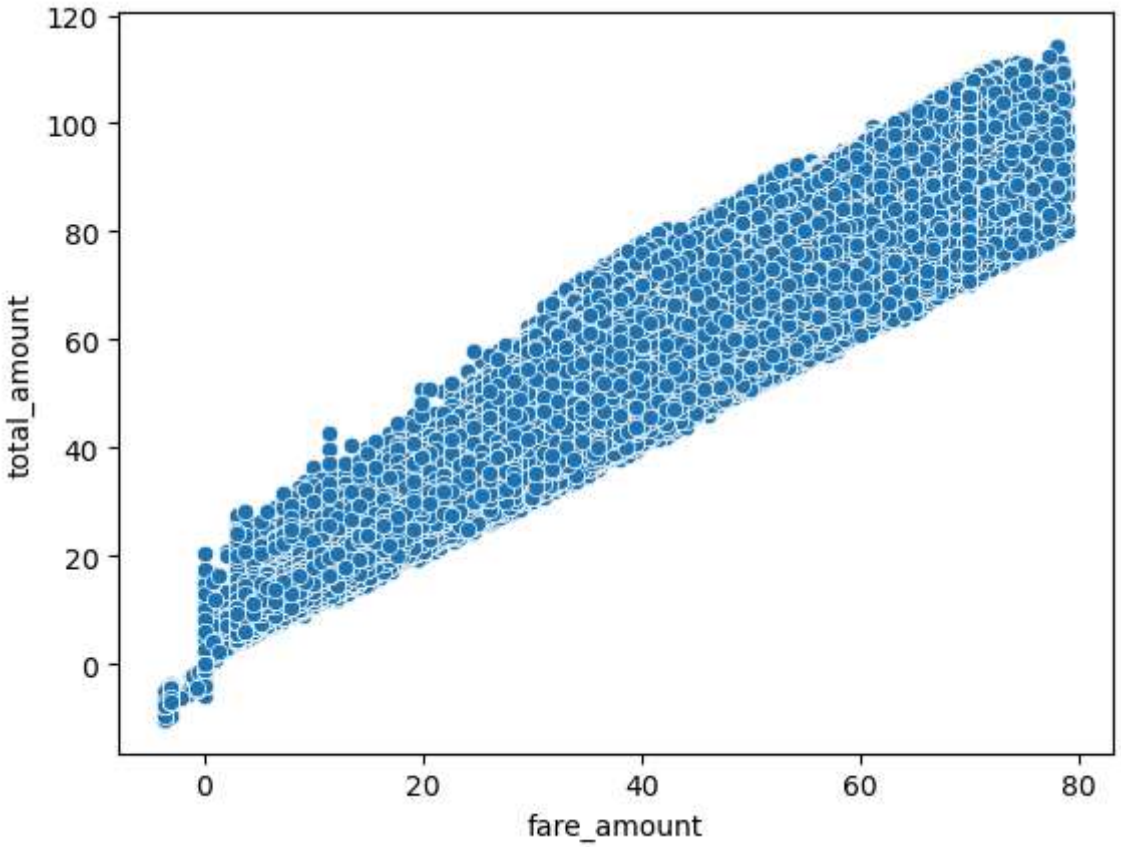
Out[132]: <Axes: xlabel='trip_distance', ylabel='Airport_fee'>




**Fare amount**

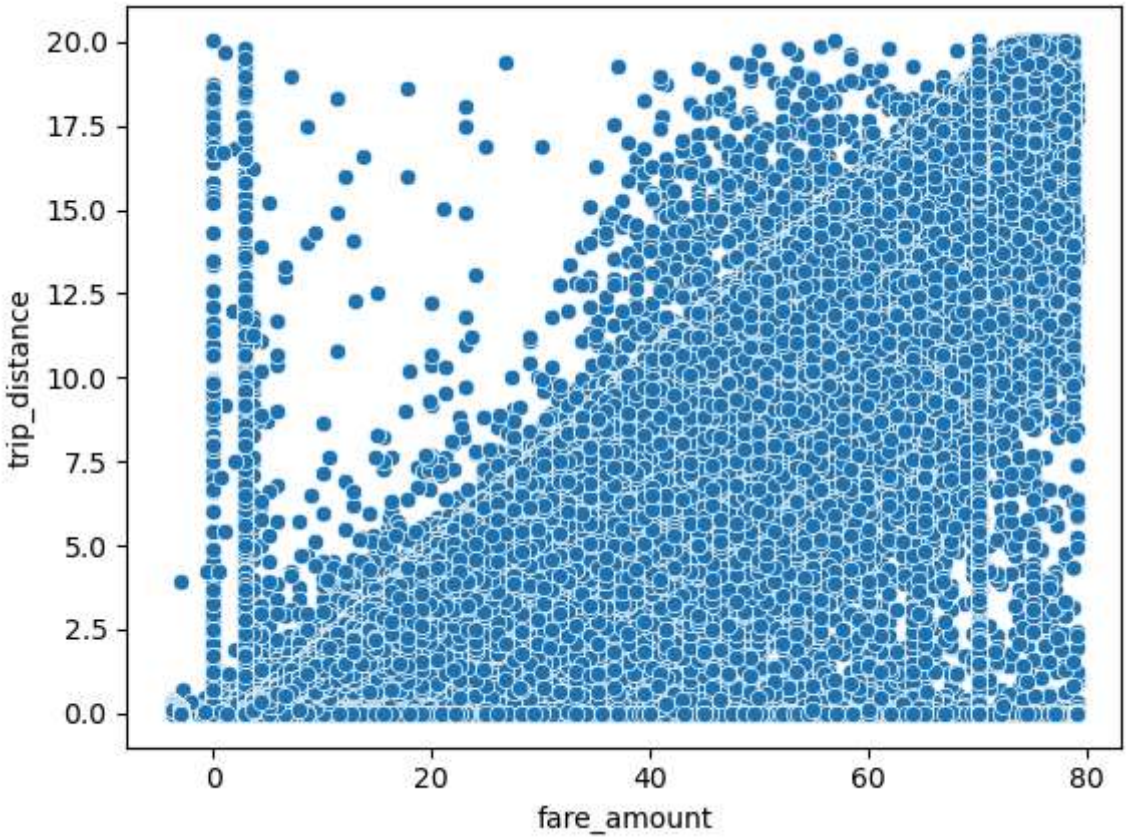
In [133]:
```
# plt.figure(figsize=(6,6))
sns.scatterplot(data=df1, x=df1["fare_amount"], y=df1["total_amount"])
```

Out[133]: <Axes: xlabel='fare_amount', ylabel='total_amount'>



In [134]:
```
# plt.figure(figsize=(6,6))
sns.scatterplot(data=df1, x=df1["fare_amount"], y=df1["trip_distance"])
```

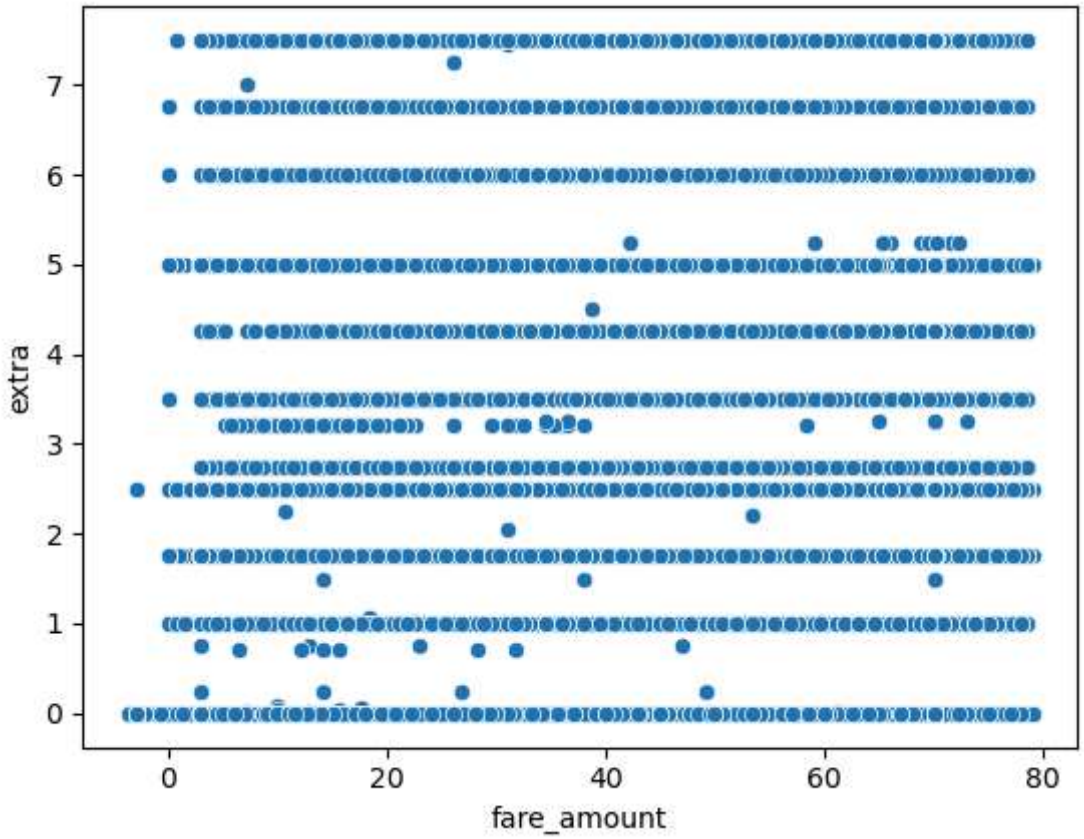
Out[134]: <Axes: xlabel='fare_amount', ylabel='trip_distance'>




In [135]:
```
# there is no correlation between fare amount and trip distance.
```

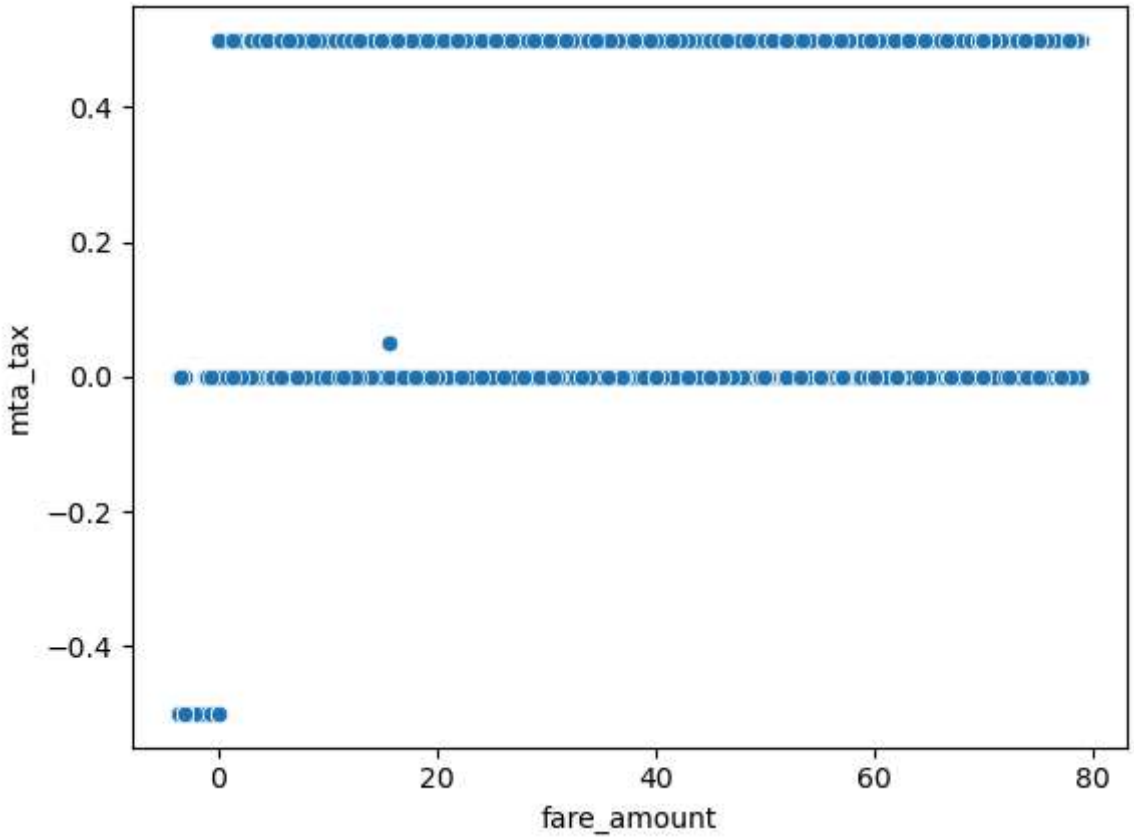
In [136]:
```
1 sns.scatterplot(data=df1, x=df1["fare_amount"], y=df1["extra"])
```

Out[136]: <Axes: xlabel='fare_amount', ylabel='extra'>



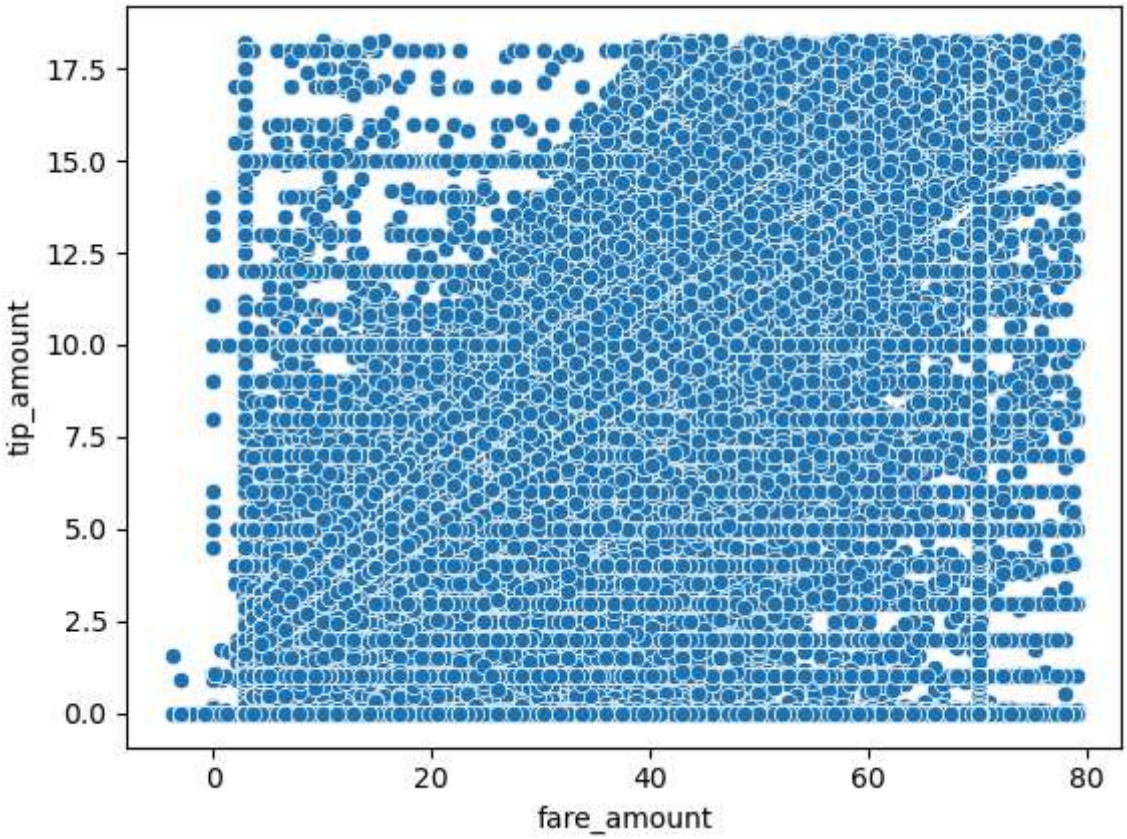
In [137]:
```
1 sns.scatterplot(data=df1, x=df1["fare_amount"], y=df1["mta_tax"])
```

Out[137]: <Axes: xlabel='fare_amount', ylabel='mta_tax'>



In [138]:
```
1 sns.scatterplot(data=df1, x=df1["fare_amount"], y=df1["tip_amount"])
```

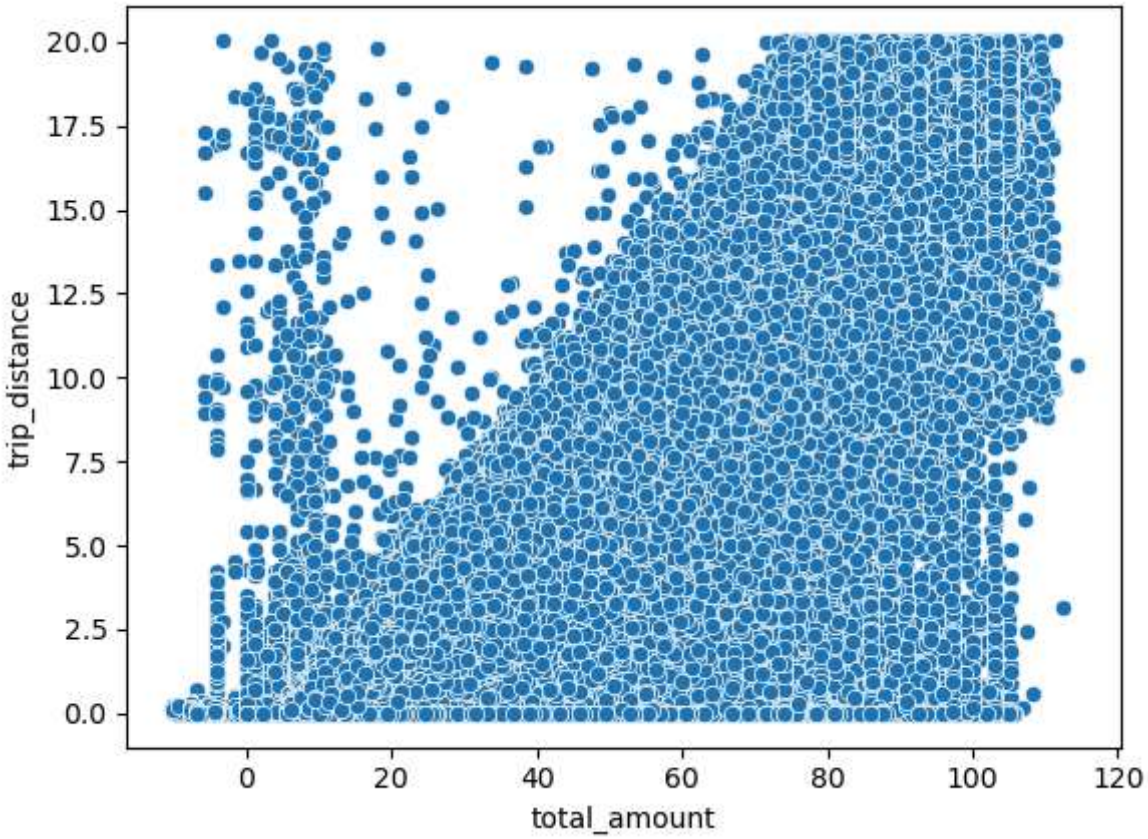
Out[138]: <Axes: xlabel='fare_amount', ylabel='tip_amount'>




**Total Amount**

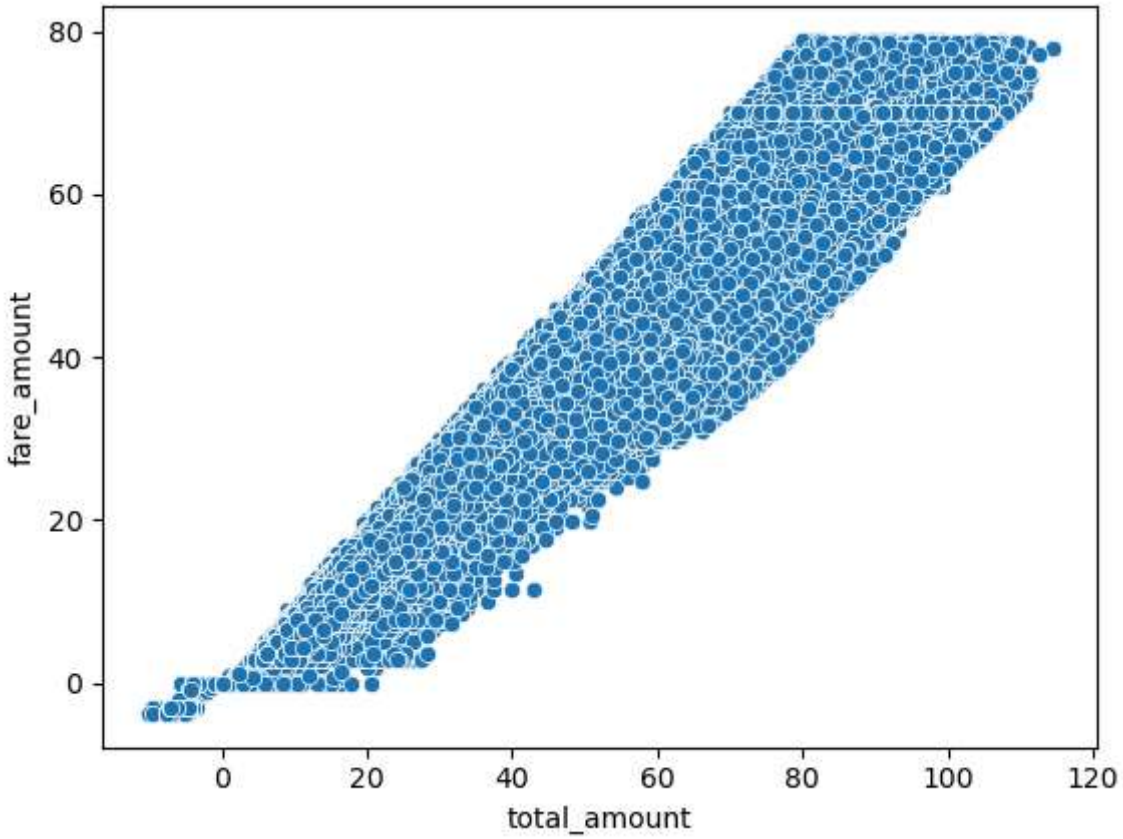
In [139]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["trip_distance"])
```

Out[139]: <Axes: xlabel='total_amount', ylabel='trip_distance'>



In [140]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["fare_amount"])
```

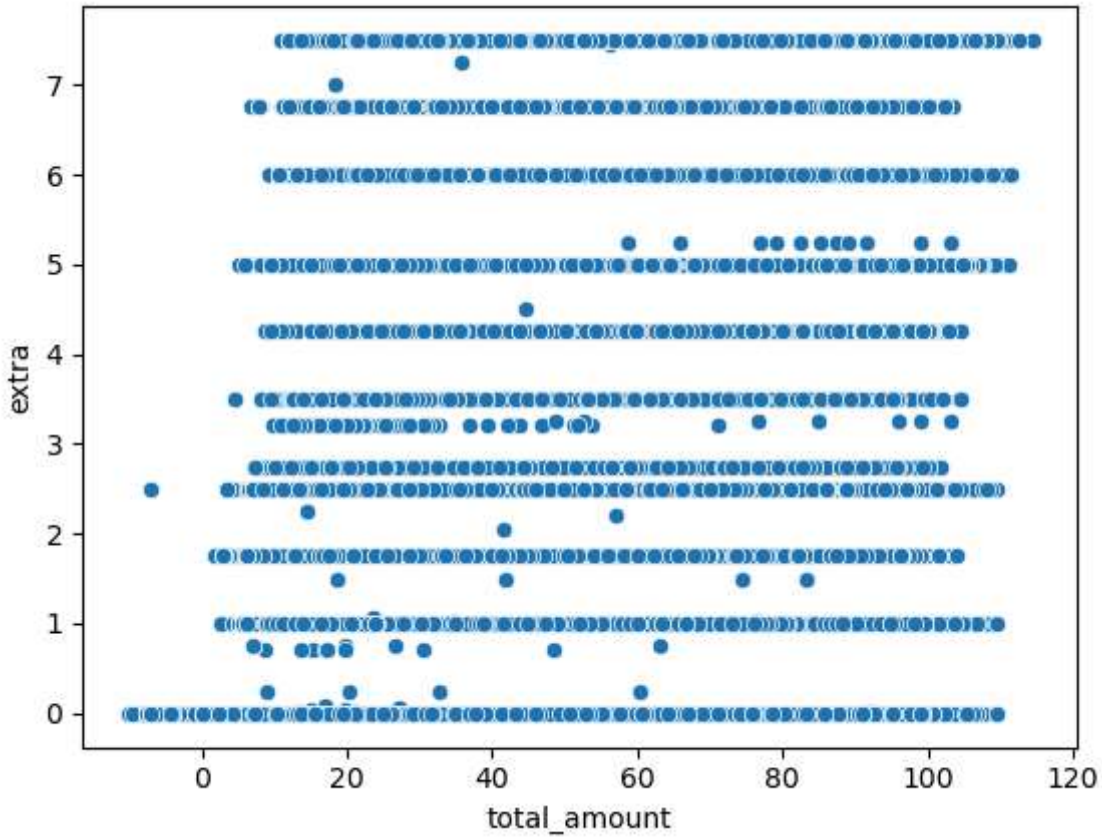
Out[140]: <Axes: xlabel='total_amount', ylabel='fare_amount'>




It can be seen that they are highly correlated.

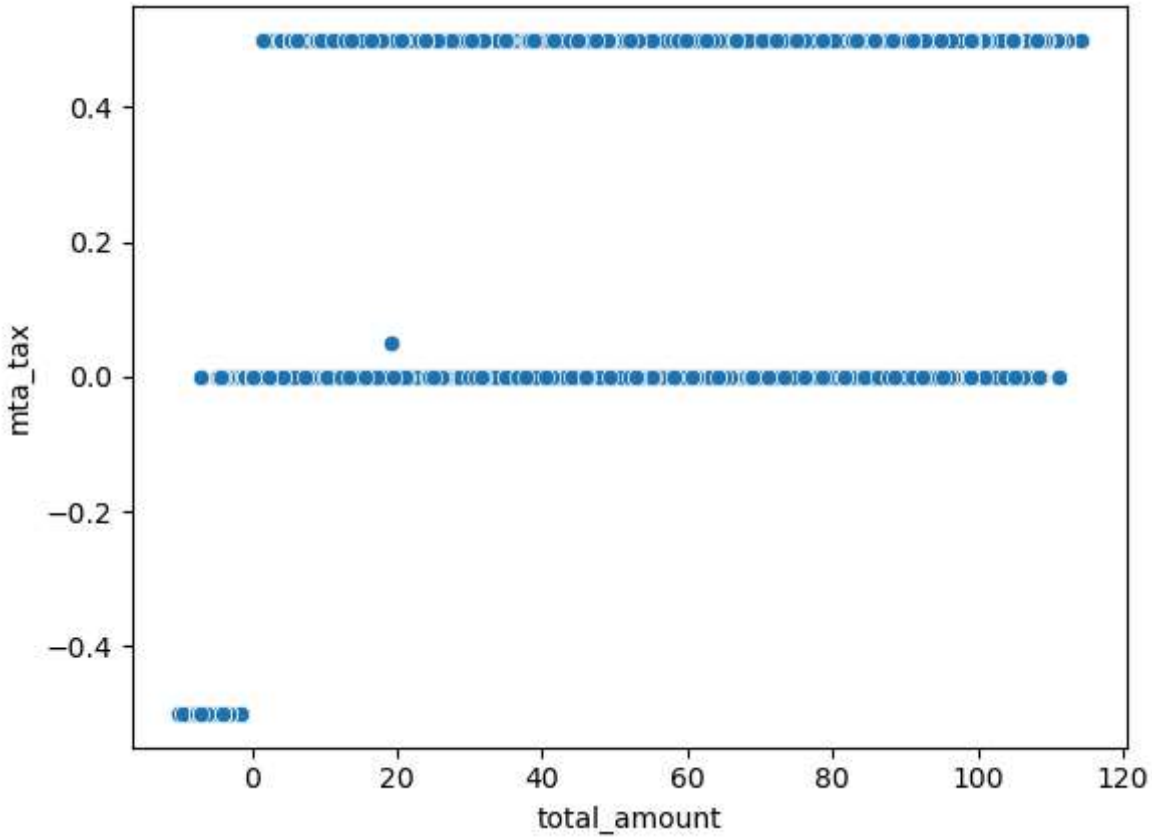
In [141]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["extra"])
```

Out[141]: <Axes: xlabel='total_amount', ylabel='extra'>



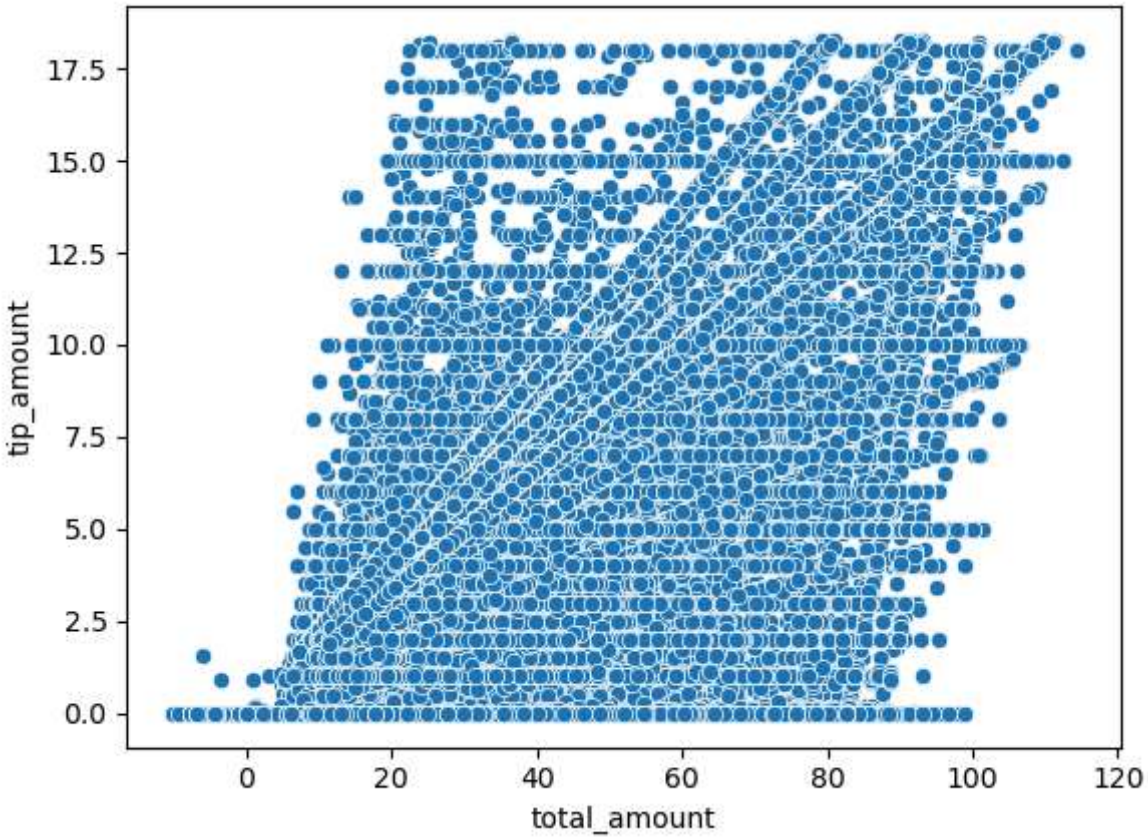
In [142]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["mta_tax"])
```

Out[142]: <Axes: xlabel='total_amount', ylabel='mta_tax'>



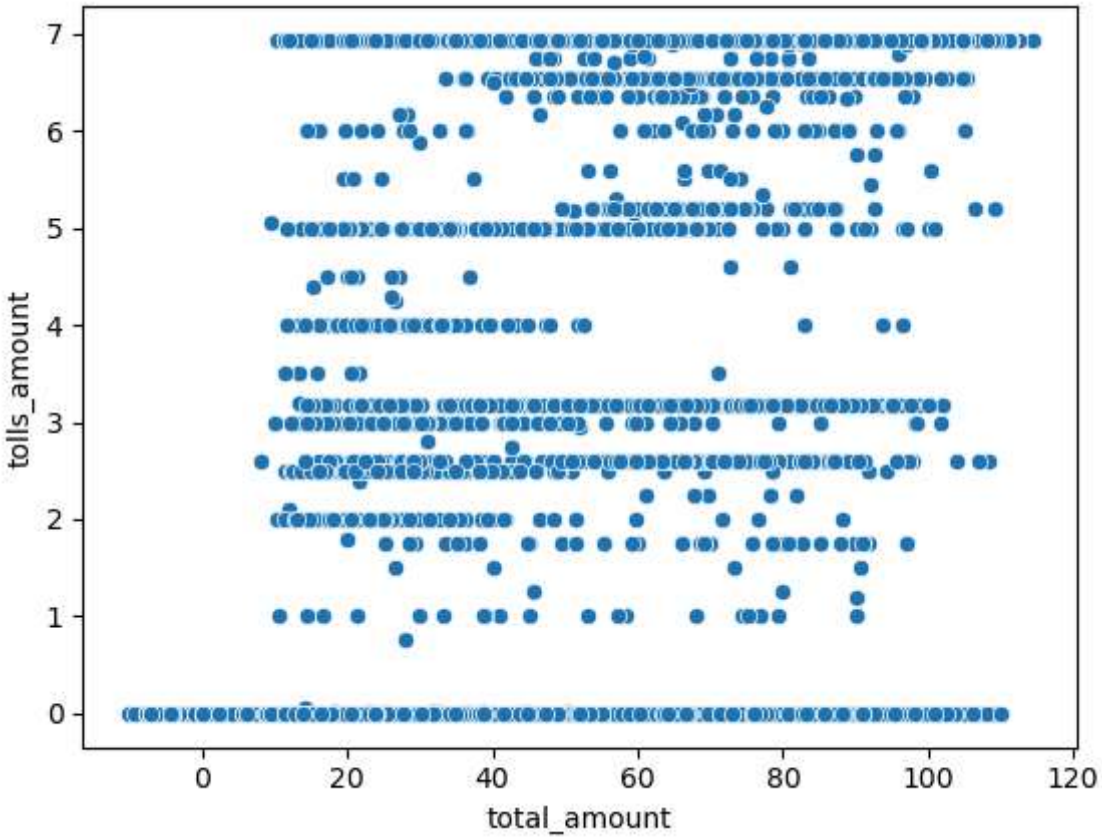
In [143]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["tip_amount"])
```

Out[143]: <Axes: xlabel='total_amount', ylabel='tip_amount'>



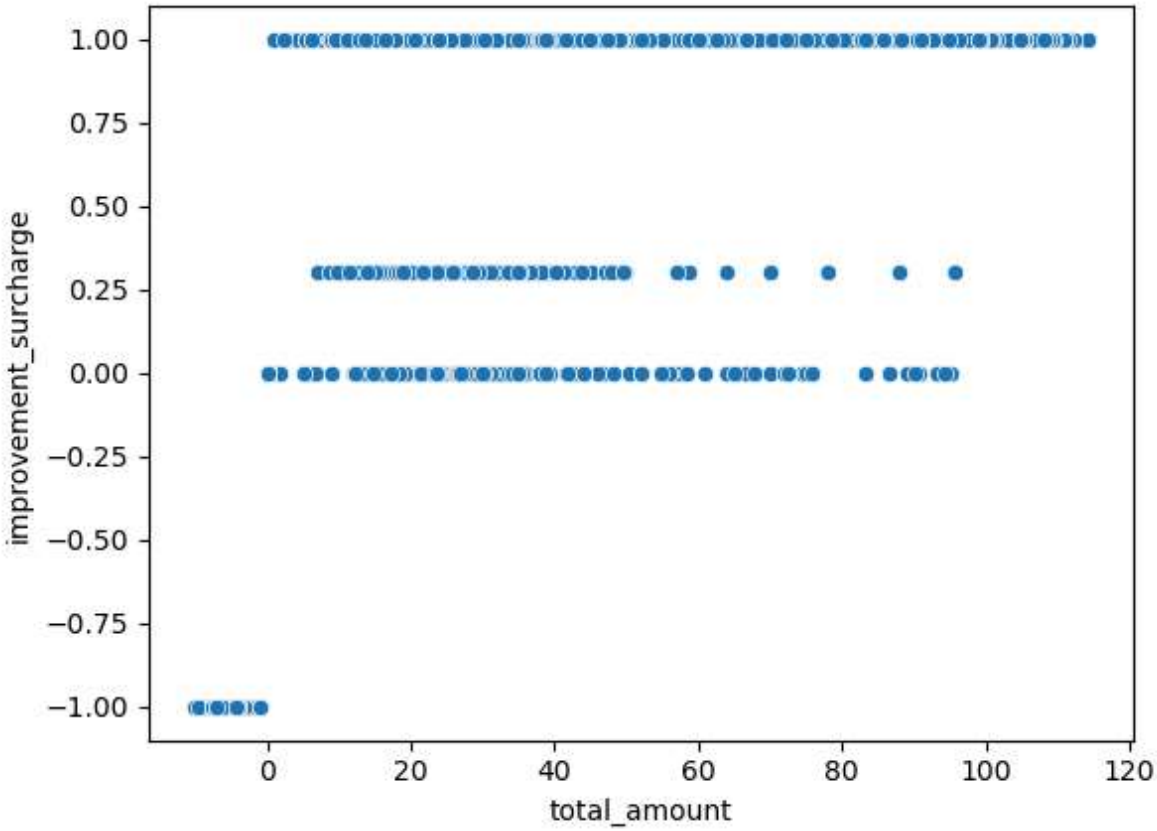
In [144]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["tolls_amount"])
```

Out[144]: <Axes: xlabel='total_amount', ylabel='tolls_amount'>



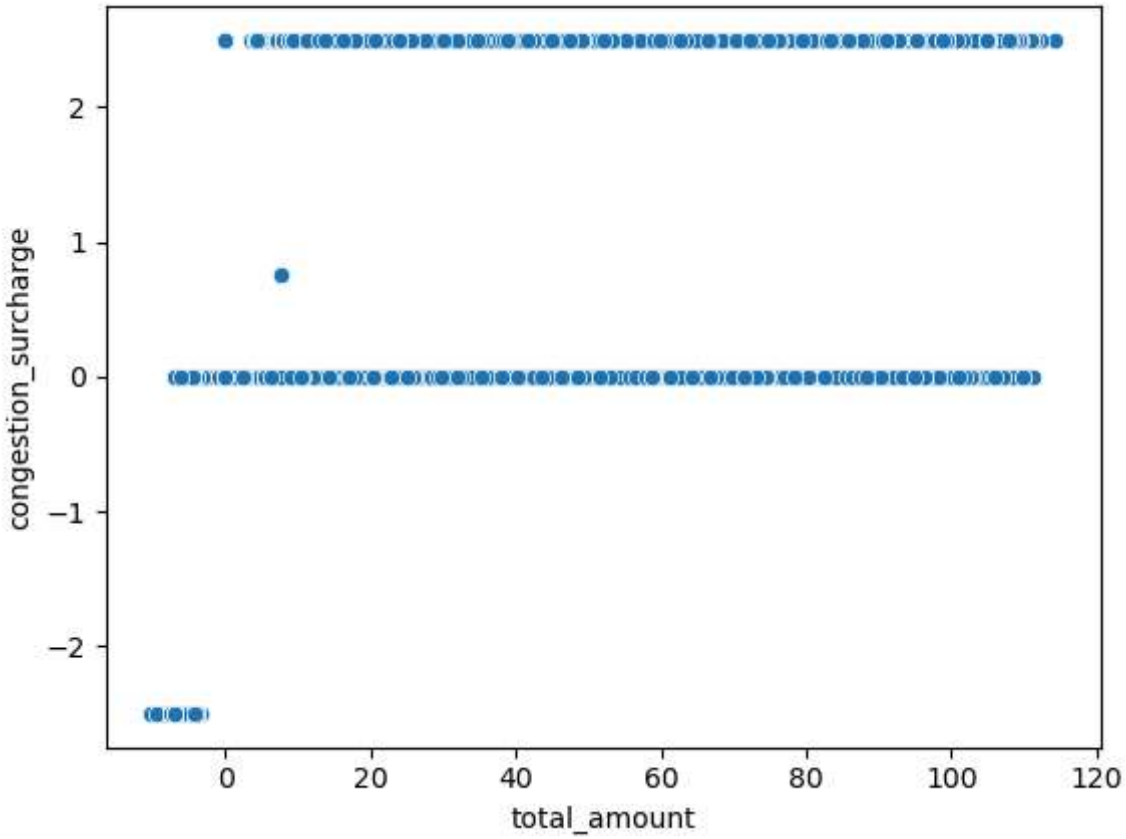
In [145]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["improvement_surcha
```

Out[145]: <Axes: xlabel='total_amount', ylabel='improvement_surcharge'>



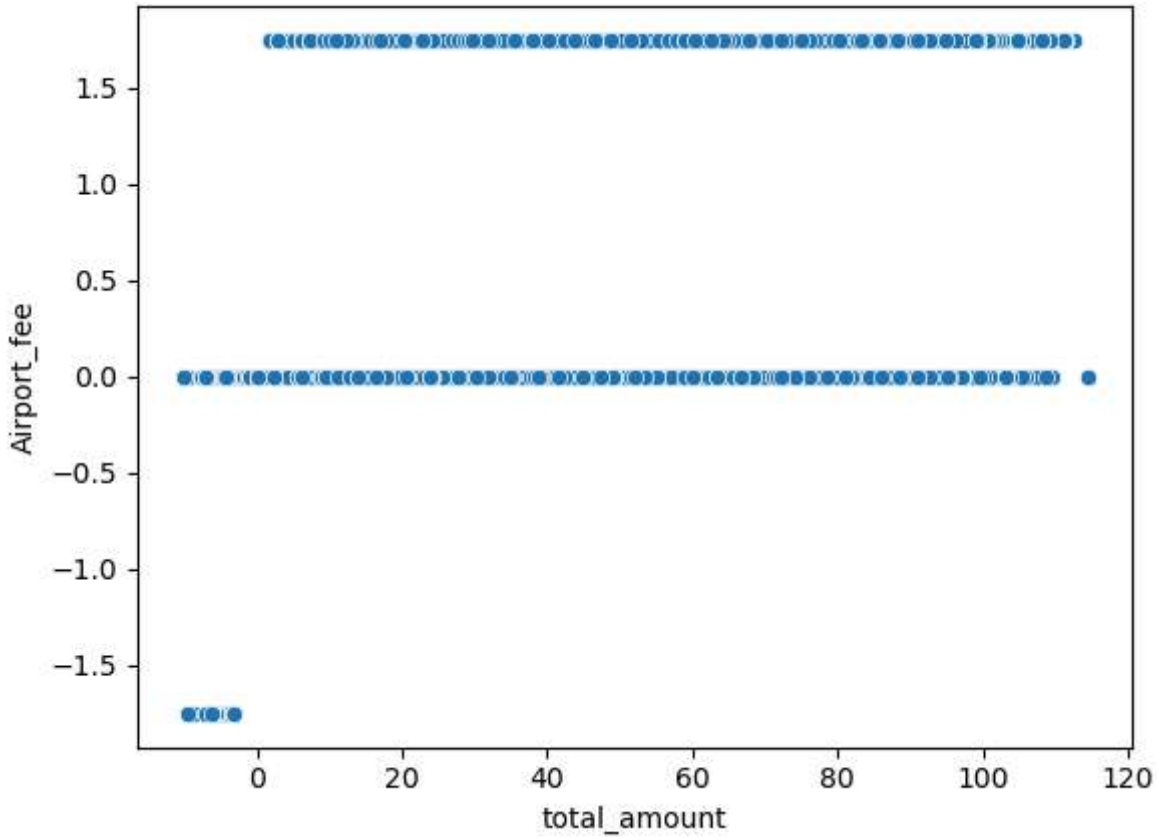
In [146]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["congestion_surchar
```

Out[146]: <Axes: xlabel='total_amount', ylabel='congestion_surcharge'>



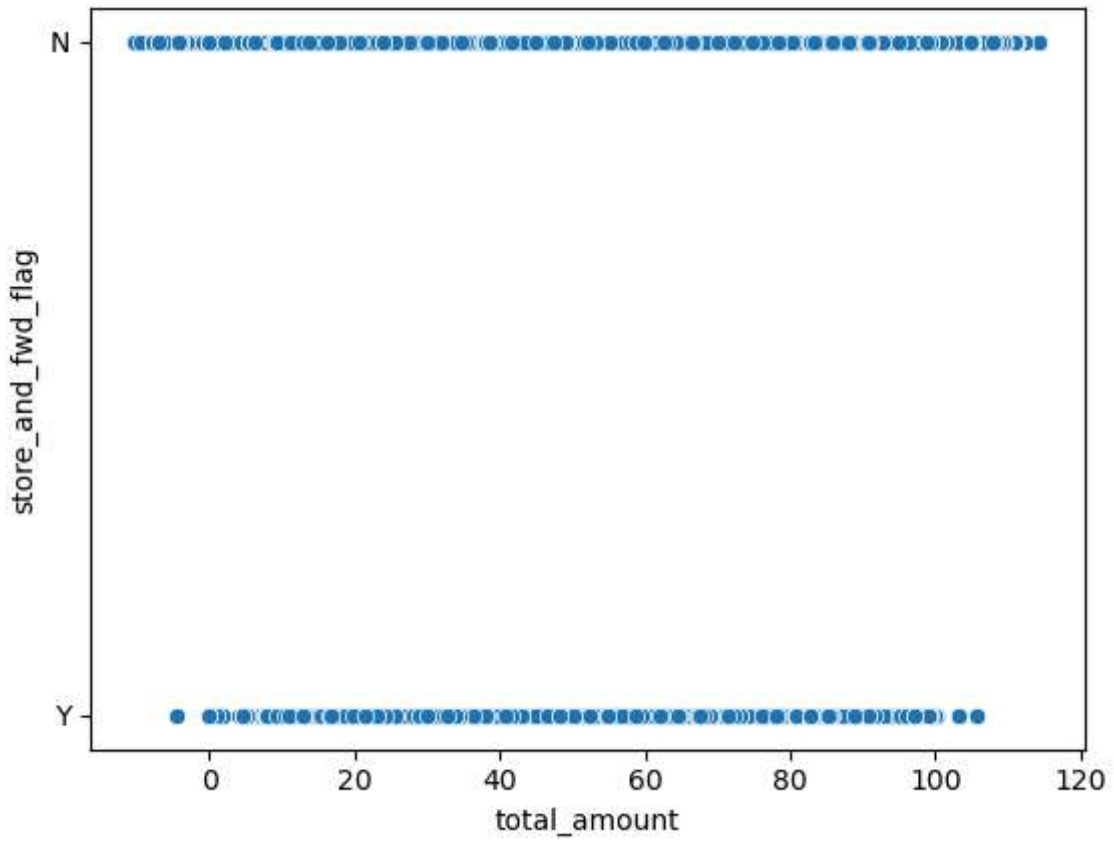
In [147]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["Airport_fee"])
```

Out[147]: <Axes: xlabel='total_amount', ylabel='Airport_fee'>



In [148]:
```
1 sns.scatterplot(data=df1, x=df1["total_amount"], y=df1["store_and_fwd_flag
```

Out[148]: <Axes: xlabel='total_amount', ylabel='store_and_fwd_flag'>




## Feature Engineering

In [149]:
```
1 # df1.info()
```

In [150]:
```
1 #dropping time variables
2 df1.drop(columns=["tpep_pickup_datetime","tpep_dropoff_datetime"], inplace
```

In [151]:
```
1 df1.head()
```

Out[151]:

| | VendorID | passenger_count | trip_distance | RatecodeID | store_and_fwd_flag | payment_type | far |
|---|---|---|---|---|---|---|---|
| **0** | 1 | 2.0 | 13.60 | 1.0 | N | 2 | |
| **1** | 1 | 0.0 | 3.50 | 1.0 | N | 1 | |
| **2** | 2 | 4.0 | 18.61 | 2.0 | N | 1 | |
| **3** | 2 | 1.0 | 0.39 | 1.0 | N | 1 | |
| **4** | 2 | 1.0 | 1.20 | 1.0 | N | 1 | |

In [152]:
```
1 df1.dtypes=="object"
```

Out[152]:
```
VendorID                False
passenger_count         False
trip_distance           False
RatecodeID              False
store_and_fwd_flag       True
payment_type            False
fare_amount             False
extra                   False
mta_tax                 False
tip_amount              False
tolls_amount            False
improvement_surcharge   False
total_amount            False
congestion_surcharge    False
Airport_fee             False
dtype: bool
```

In [153]:
```
1 cat_data=df1[df1.dtypes[df.dtypes=="object"].index]   # seperating object
2 cat_data
```

Out[153]:

| | store_and_fwd_flag |
|---|---|
| **0** | N |
| **1** | N |
| **2** | N |
| **3** | N |
| **4** | N |
| **...** | ... |
| **3207035** | N |
| **3207036** | N |
| **3207037** | N |
| **3207038** | N |
| **3207039** | N |

3055963 rows × 1 columns

In [154]:
```
num_data=df1[df1.dtypes[df.dtypes!="object"].index]
num_data.head()
```

Out[154]:

| | VendorID | passenger_count | trip_distance | RatecodeID | payment_type | fare_amount | extra | mt |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 2.0 | 13.60 | 1.0 | 2 | 61.8 | 2.75 | |
| **1** | 1 | 0.0 | 3.50 | 1.0 | 1 | 20.5 | 3.50 | |
| **2** | 2 | 4.0 | 18.61 | 2.0 | 1 | 70.0 | 0.00 | |
| **3** | 2 | 1.0 | 0.39 | 1.0 | 1 | 4.4 | 1.00 | |
| **4** | 2 | 1.0 | 1.20 | 1.0 | 1 | 10.0 | 1.00 | |

In [155]:
```
cat_data.isnull().sum()
```

Out[155]:
```
store_and_fwd_flag    0
dtype: int64
```

In [156]:
```
num_data.isnull().sum()
```

Out[156]:
```
VendorID                 0
passenger_count          0
trip_distance            0
RatecodeID               0
payment_type             0
fare_amount              0
extra                    0
mta_tax                  0
tip_amount               0
tolls_amount             0
improvement_surcharge    0
total_amount             0
congestion_surcharge     0
Airport_fee              0
dtype: int64
```

In [157]:
```
def outliercap(x):
    x=x.clip(upper=x.quantile(.99))
    x=x.clip(lower=x.quantile(.01))
    return x
```

In [158]:
```
num_data=num_data.apply(outliercap)
```

In [159]:
```
num_data.describe(percentiles=[.01,.02,.03,.04,.05,.1,.25,.5,.75,.9,.95,.9
```

Out[159]:

| | count | mean | std | min | 1% | 2% | 3% | 4% | 5% | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| **VendorID** | 3055963.0 | 1.767355 | 0.422518 | 1.0 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 | 1 |
| **passenger_count** | 3055963.0 | 1.349575 | 0.830757 | 0.0 | 0.0 | 1.00 | 1.00 | 1.00 | 1.00 | 1 |
| **trip_distance** | 3055963.0 | 2.941548 | 3.602134 | 0.0 | 0.0 | 0.28 | 0.38 | 0.42 | 0.49 | 0 |
| **RatecodeID** | 3055963.0 | 1.032961 | 0.178536 | 1.0 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 | 1 |
| **payment_type** | 3055963.0 | 1.187127 | 0.424228 | 1.0 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 | 1 |
| **fare_amount** | 3055963.0 | 18.247556 | 14.341248 | 4.4 | 4.4 | 5.10 | 5.10 | 5.80 | 5.80 | 6 |
| **extra** | 3055963.0 | 1.519776 | 1.669640 | 0.0 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0 |
| **mta_tax** | 3055963.0 | 0.500000 | 0.000000 | 0.5 | 0.5 | 0.50 | 0.50 | 0.50 | 0.50 | 0 |
| **tip_amount** | 3055963.0 | 3.424330 | 3.274543 | 0.0 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0 |
| **tolls_amount** | 3055963.0 | 0.431328 | 1.672591 | 0.0 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0 |
| **improvement_surcharge** | 3055963.0 | 1.000000 | 0.000000 | 1.0 | 1.0 | 1.00 | 1.00 | 1.00 | 1.00 | 1 |
| **total_amount** | 3055963.0 | 27.044336 | 18.318759 | 8.7 | 8.7 | 9.80 | 10.50 | 11.10 | 11.50 | 12 |
| **congestion_surcharge** | 3055963.0 | 2.362706 | 0.569547 | 0.0 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 | 2 |
| **Airport_fee** | 3055963.0 | 0.124446 | 0.449771 | 0.0 | 0.0 | 0.00 | 0.00 | 0.00 | 0.00 | 0 |

### Dummy Creation

In [160]:
```
cat_data1=pd.get_dummies(cat_data,drop_first=True)
cat_data1.nunique()
```

Out[160]:
```
store_and_fwd_flag_Y    2
dtype: int64
```

In [161]:
```
# cat_data.unique()
```

In [162]:
```
final_data0=pd.concat([num_data,cat_data1],axis=1)
final_data0.head(2)
```

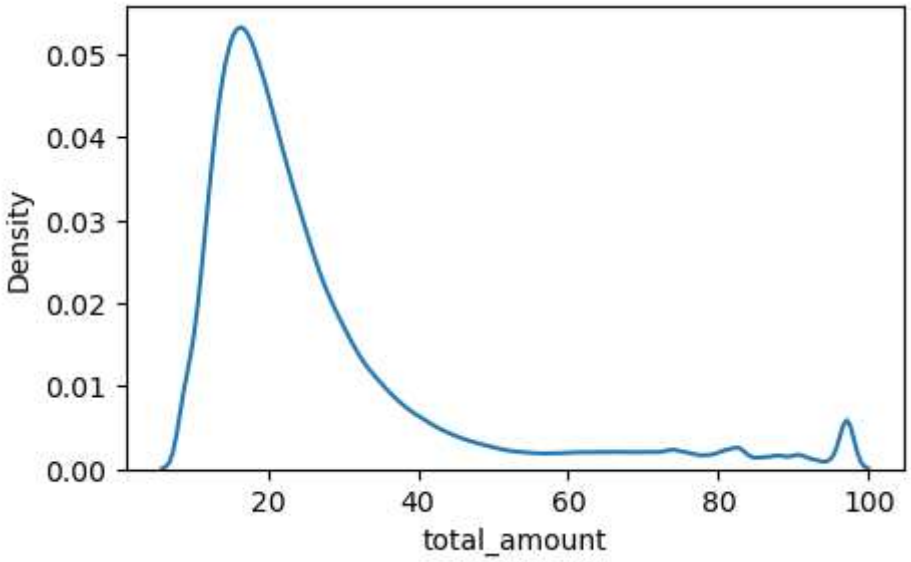
Out[162]:

| | VendorID | passenger_count | trip_distance | RatecodeID | payment_type | fare_amount | extra | mt |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 2.0 | 13.6 | 1.0 | 2 | 61.8 | 2.75 | |
| **1** | 1 | 0.0 | 3.5 | 1.0 | 1 | 20.5 | 3.50 | |

In [163]:
```
# Assumptions
# 1. Y must be continuous
# 2. Y should have Relationsip with X variables
# 3. Y should be normally distributed
# 4. Multicollinearity
# 5. No autocorrelation
# 6. No Hetroscedasticity
```

In [164]:
```
plt.figure(figsize=(5,3))
sns.kdeplot(data=final_data0, x=final_data0["total_amount"])
```

Out[164]: <Axes: xlabel='total_amount', ylabel='Density'>




In [165]:
```
# Shapiro Hypothesis testing
# null hypothesis =  Data is normal
# alternative hypothesis = data is not normal
# 95% - alpha = 0.05
from scipy import stats
stats.shapiro(final_data0["total_amount"])
```

```
D:\Anaconda\lib\site-packages\scipy\stats\_morestats.py:1816: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
```

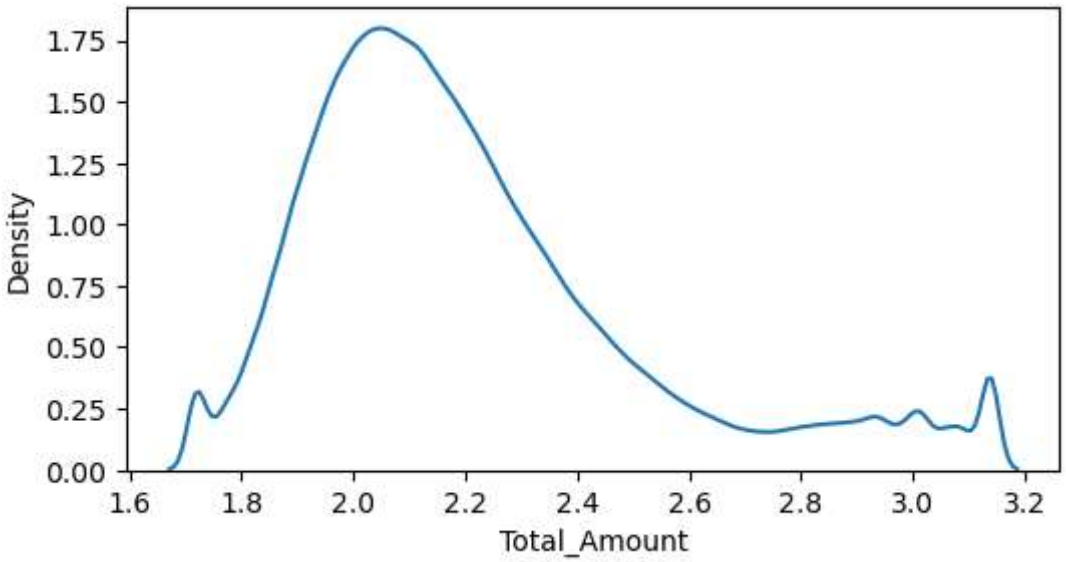
Out[165]: ShapiroResult(statistic=0.7387919425964355, pvalue=0.0)

In [166]:
```
# How to make it normal
# try - square , square root, cube, cube root, etc
# try log and its varient
```

In [167]:
```
final_data0["Total_Amount"]=(final_data0["total_amount"])**(1/4)

print(stats.shapiro(final_data0["Total_Amount"]))

plt.figure(figsize=(6,3))
sns.kdeplot(data=final_data0, x=final_data0["Total_Amount"])
```

```
D:\Anaconda\lib\site-packages\scipy\stats\_morestats.py:1816: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")

ShapiroResult(statistic=0.9076067805290222, pvalue=0.0)
```

Out[167]:
```
<Axes: xlabel='Total_Amount', ylabel='Density'>
```



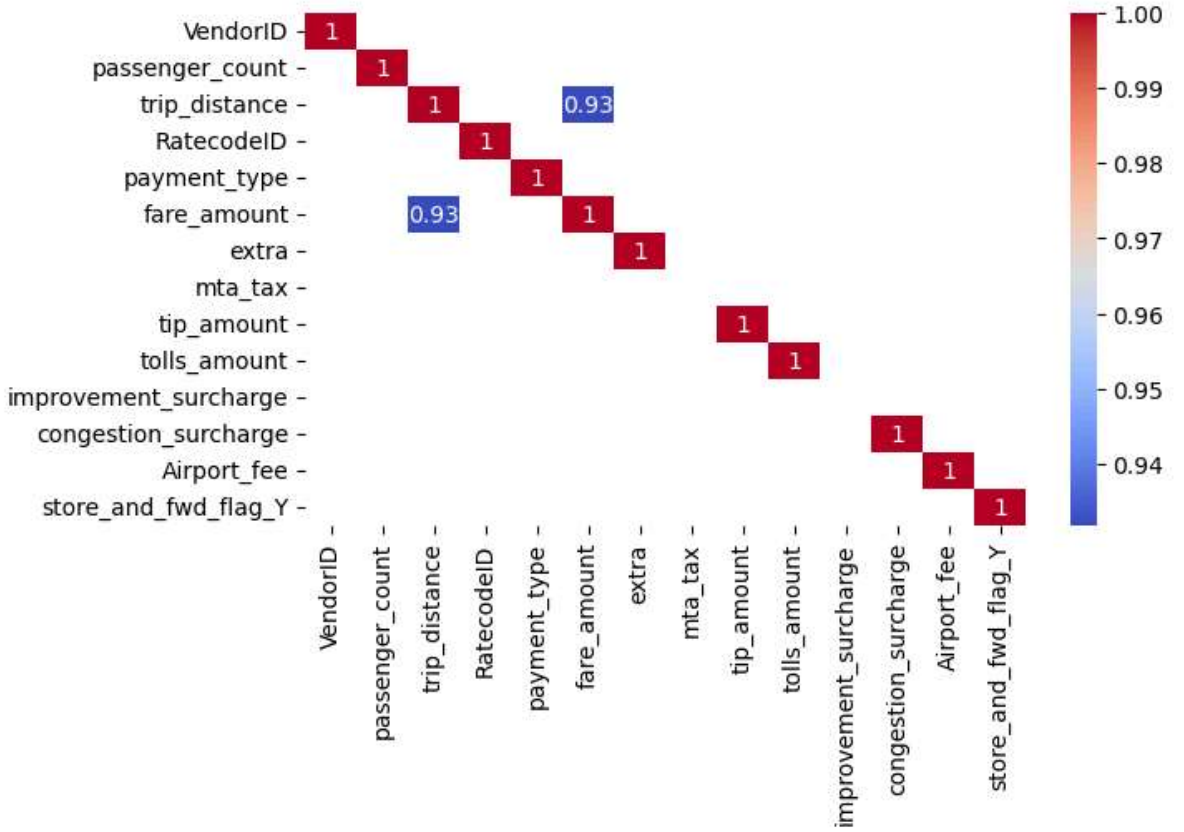

In [168]:
```
final_data0.columns
```

Out[168]:
```
Index(['VendorID', 'passenger_count', 'trip_distance', 'RatecodeID',
       'payment_type', 'fare_amount', 'extra', 'mta_tax', 'tip_amount',
       'tolls_amount', 'improvement_surcharge', 'total_amount',
       'congestion_surcharge', 'Airport_fee', 'store_and_fwd_flag_Y',
       'Total_Amount'],
      dtype='object')
```

In [169]:
```
cr=final_data0[['VendorID', 'passenger_count', 'trip_distance', 'RatecodeI
       'payment_type', 'fare_amount', 'extra', 'mta_tax', 'tip_amount',
       'tolls_amount', 'improvement_surcharge',
       'congestion_surcharge', 'Airport_fee', 'store_and_fwd_flag_Y']].cor
```

In [170]:
```
cr1=cr[abs(cr)>.7]
plt.figure(figsize=(7,4))
sns.heatmap(cr1, cmap = "coolwarm", annot=True)
```

Out[170]: <Axes: >




In [171]:
```
y=final_data0["Total_Amount"]
x= final_data0[['VendorID', 'passenger_count', 'trip_distance', 'RatecodeI
       'payment_type', 'fare_amount', 'extra', 'mta_tax', 'tip_amount',
       'tolls_amount', 'improvement_surcharge',
       'congestion_surcharge', 'Airport_fee', 'store_and_fwd_flag_Y']]
```

In [172]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=.3, ran
```

In [173]:
```
x_train.head(2)
```

Out[173]:

| | VendorID | passenger_count | trip_distance | RatecodeID | payment_type | fare_amount | ext |
|---|---|---|---|---|---|---|---|
| **1504750** | 2 | 1.0 | 0.67 | 1.0 | 1 | 8.6 | 0 |
| **2898254** | 2 | 1.0 | 1.56 | 1.0 | 2 | 12.8 | 0 |

In [174]:

```
import statsmodels.api as sm

x_train1=x_train[['VendorID', 'passenger_count', 'trip_distance', 'Rateco
       'payment_type', 'fare_amount', 'extra', 'tip_amount',
       'tolls_amount', 'improvement_surcharge',
       'congestion_surcharge', 'Airport_fee']]
xtrain1 = sm.add_constant(x_train1)
model = sm.OLS(y_train, x_train1)
results= model.fit()
print(results.summary())
```

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Total_Amount | R-squared: | 0.968 |
| Model: | OLS | Adj. R-squared: | 0.968 |
| Method: | Least Squares | F-statistic: | 5.962e+06 |
| Date: | Tue, 27 Feb 2024 | Prob (F-statistic): | 0.00 |
| Time: | 09:42:19 | Log-Likelihood: | 3.1773e+06 |
| No. Observations: | 2139174 | AIC: | -6.355e+06 |
| Df Residuals: | 2139162 | BIC: | -6.354e+06 |
| Df Model: | 11 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| VendorID | 0.0529 | 0.000 | 445.321 | 0.000 | 0.053 | 0.053 |
| passenger_count | 0.0008 | 4.53e-05 | 16.872 | 0.000 | 0.001 | 0.001 |
| trip_distance | -0.0102 | 3.18e-05 | -319.705 | 0.000 | -0.010 | -0.010 |
| RatecodeID | -0.2229 | 0.000 | -777.162 | 0.000 | -0.223 | -0.222 |
| payment_type | -0.0377 | 0.000 | -342.868 | 0.000 | -0.038 | -0.038 |
| fare_amount | 0.0226 | 7.69e-06 | 2937.918 | 0.000 | 0.023 | 0.023 |
| extra | 0.0216 | 3.29e-05 | 655.901 | 0.000 | 0.022 | 0.022 |
| tip_amount | 0.0131 | 1.86e-05 | 704.557 | 0.000 | 0.013 | 0.013 |
| tolls_amount | 0.0008 | 3.4e-05 | 23.355 | 0.000 | 0.001 | 0.001 |
| improvement_surcharge | 1.8834 | 0.000 | 4092.478 | 0.000 | 1.882 | 1.884 |
| congestion_surcharge | 0.0212 | 7.81e-05 | 271.824 | 0.000 | 0.021 | 0.021 |
| Airport_fee | -0.0151 | 0.000 | -111.111 | 0.000 | -0.015 | -0.015 |

| | | | |
|---|---|---|---|
| Omnibus: | 395865.887 | Durbin-Watson: | 2.000 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 1741359.753 |
| Skew: | -0.851 | Prob(JB): | 0.00 |

```
Kurtosis:                       7.079   Cond. No.                         33
1.
==============================================================================
=

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
```

In [175]:

```
# Linear Regr:
# Null Hyp : all m values are zeros
# Alter Hyp: atleast one m value is not zero

# Alpha - o.o5
```

In [176]:

```
import statsmodels.api as sm
x_train1=x_train[['VendorID', 'passenger_count', 'trip_distance', 'Rateco
       'payment_type', 'fare_amount', 'extra', 'tip_amount',
       'tolls_amount', 'improvement_surcharge',
       'congestion_surcharge', 'Airport_fee']]

x_train1 = sm.add_constant(x_train1)
model = sm.OLS(y_train,x_train1)
results=model.fit()
print(results.summary())
```

OLS Regression Results

| | | | |
|---|---|---|---|
| Dep. Variable: | Total_Amount | R-squared: | 0.968 |
| Model: | OLS | Adj. R-squared: | 0.968 |
| Method: | Least Squares | F-statistic: | 5.962e+06 |
| Date: | Tue, 27 Feb 2024 | Prob (F-statistic): | 0.00 |
| Time: | 09:42:22 | Log-Likelihood: | 3.1773e+06 |
| No. Observations: | 2139174 | AIC: | -6.355e+06 |
| Df Residuals: | 2139162 | BIC: | -6.354e+06 |
| Df Model: | 11 | | |
| Covariance Type: | nonrobust | | |

| | coef | std err | t | P>\|t\| | [0.025 | 0.975] |
|---|---|---|---|---|---|---|
| VendorID | 0.0529 | 0.000 | 445.321 | 0.000 | 0.053 | 0.053 |
| passenger_count | 0.0008 | 4.53e-05 | 16.872 | 0.000 | 0.001 | 0.001 |
| trip_distance | -0.0102 | 3.18e-05 | -319.705 | 0.000 | -0.010 | -0.010 |
| RatecodeID | -0.2229 | 0.000 | -777.162 | 0.000 | -0.223 | -0.222 |
| payment_type | -0.0377 | 0.000 | -342.868 | 0.000 | -0.038 | -0.038 |
| fare_amount | 0.0226 | 7.69e-06 | 2937.918 | 0.000 | 0.023 | 0.023 |
| extra | 0.0216 | 3.29e-05 | 655.901 | 0.000 | 0.022 | 0.022 |
| tip_amount | 0.0131 | 1.86e-05 | 704.557 | 0.000 | 0.013 | 0.013 |
| tolls_amount | 0.0008 | 3.4e-05 | 23.355 | 0.000 | 0.001 | 0.001 |
| improvement_surcharge | 1.8834 | 0.000 | 4092.478 | 0.000 | 1.882 | 1.884 |
| congestion_surcharge | 0.0212 | 7.81e-05 | 271.824 | 0.000 | 0.021 | 0.021 |
| Airport_fee | -0.0151 | 0.000 | -111.111 | 0.000 | -0.015 | -0.015 |

| | | | |
|---|---|---|---|
| Omnibus: | 395865.887 | Durbin-Watson: | 2.000 |
| Prob(Omnibus): | 0.000 | Jarque-Bera (JB): | 1741359.753 |
| Skew: | -0.851 | Prob(JB): | 0.00 |

```
Kurtosis:                       7.079   Cond. No.                         33
1.
==============================================================================
=

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correc
tly specified.
```

**With SkLearn**

In [177]:
```
from sklearn.linear_model import LinearRegression
```

In [178]:
```
y=final_data0["Total_Amount"]
x=final_data0[['VendorID', 'passenger_count', 'trip_distance', 'RatecodeID
       'payment_type', 'fare_amount', 'extra', 'tip_amount',
       'tolls_amount', 'improvement_surcharge',
       'congestion_surcharge', 'Airport_fee']]
```

In [179]:
```
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.3, ra
```

In [180]:
```
lr=LinearRegression()
lr.fit(x_train, y_train) # model traiing
```

Out[180]:
```
LinearRegression
LinearRegression()
```

In [181]:
```
lr.coef_
```

Out[181]:
```
array([ 5.28802778e-02,  7.65066113e-04, -1.01642074e-02, -2.22862531e-01,
       -3.77377506e-02,  2.25922678e-02,  2.15656334e-02,  1.30729698e-02,
        7.94163541e-04,  1.66533454e-16,  2.12276052e-02, -1.50865359e-02])
```

In [182]:
```
lr.intercept_   # constant
```

Out[182]:
```
1.8833508536858237
```

In [183]:
```
#Model evalution
# MSE, RMSE, MAE, MAPE
# MSE=(Pred-Act)^2/ N
# RMSE= (MSE)**(1/2) or RMSE= ((( Pred-Act)^(2))/N)^(1/2)
# MAE - Mean absolute Error - = mean(abs(Pred-Act))
# MAPE - Mean Absolute percentage Error - = mean(abs((Pred-Act)/Act))
```

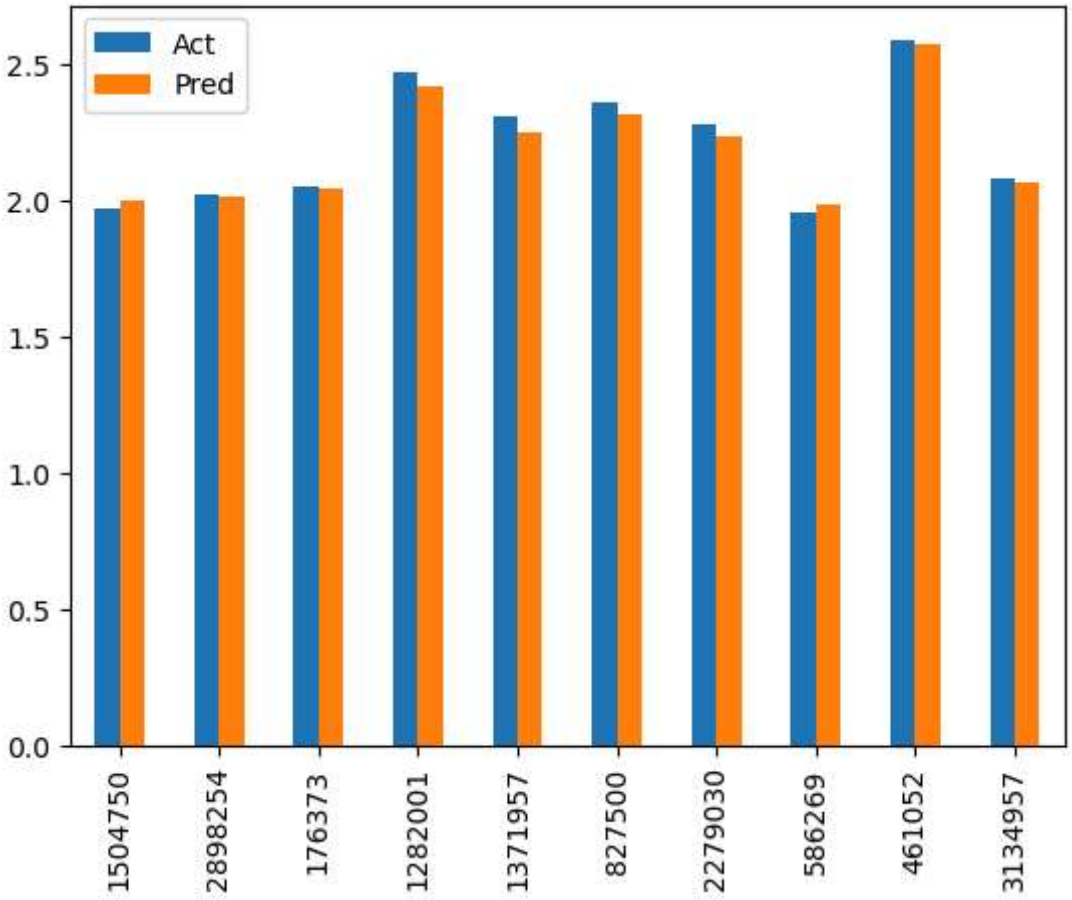
In [184]:
```
pred_train = lr.predict(x_train)
pred_test = lr.predict(x_test)
```

In [185]:
```
1 pred_train
```

Out[185]:
```
array([2.00277257, 2.01793232, 2.04340829, ..., 1.90322349, 2.95554503,
       2.2616247 ])
```

In [186]:
```
1 pd.DataFrame({"Act":y_train, "Pred":pred_train}).head(10).plot(kind="bar")
```

Out[186]: <Axes: >




In [187]:
```
1 pd.DataFrame({"Act":y_train,"Pred":pred_train}).head()
```

Out[187]:

| | Act | Pred |
|---|---|---|
| **1504750** | 1.971914 | 2.002773 |
| **2898254** | 2.024544 | 2.017932 |
| **176373** | 2.049390 | 2.043408 |
| **1282001** | 2.472965 | 2.423467 |
| **1371957** | 2.307889 | 2.251287 |

In [188]:
```
1 print("Train MSE:", np.mean(pred_train-y_train)**2)
2 print("Test MSE:", np.mean(pred_test-y_test)**2)
```

```
Train MSE: 8.517346153726308e-31
Test MSE: 3.1564714348892645e-10
```

The MSE of the Train and Test Data is very close to zero, it means that our model is good.

In [189]:
```
print("Train RMSE:",np.sqrt(np.mean(pred_train-y_train)))
print("Test RMSE:",np.sqrt(np.mean(pred_test-y_test)))
```

```
Train RMSE: 3.0379181910389204e-08
Test RMSE: 0.004215028018612361
```

The RMSE of the train and test data is also close to zero it means that our model is good and there are minimul deviations.

In [190]:
```
print("Train MAE:", np.mean(abs(pred_train-y_train)))
print("Test MAE:", np.mean(abs(pred_test-y_test)))
```

```
Train MAE: 0.040669077720758076
Test MAE: 0.04071163159576854
```

Since MAE of the train and Test data is closer to zero it means that our model is accurate

In [191]:
```
print("Train MAPE:", np.mean(abs((pred_train-y_train)/y_train)))
print("Test MAPE:", np.mean(abs((pred_test-y_test)/y_test)))
```

```
Train MAPE: 0.018442932724259276
Test MAPE: 0.018463548355052442
```

Since our MAPE is less than 10% therefore Accurate forcasting

In [196]:
```
x_train.head(10)
```

Out[196]:

| | VendorID | passenger_count | trip_distance | RatecodeID | payment_type | fare_amount | ext |
|---|---|---|---|---|---|---|---|
| **1504750** | 2 | 1.0 | 0.67 | 1.0 | 1 | 8.6 | 0 |
| **2898254** | 2 | 1.0 | 1.56 | 1.0 | 2 | 12.8 | 0 |
| **176373** | 2 | 1.0 | 1.88 | 1.0 | 1 | 10.7 | 0 |
| **1282001** | 2 | 5.0 | 7.65 | 1.0 | 2 | 32.4 | 1 |
| **1371957** | 2 | 1.0 | 2.62 | 1.0 | 1 | 17.7 | 1 |
| **827500** | 2 | 1.0 | 2.48 | 1.0 | 1 | 21.9 | 0 |
| **2279030** | 2 | 1.0 | 2.16 | 1.0 | 1 | 18.4 | 0 |
| **586269** | 2 | 1.0 | 1.76 | 1.0 | 1 | 8.6 | 0 |
| **461052** | 2 | 1.0 | 5.69 | 1.0 | 1 | 32.4 | 1 |
| **3134957** | 2 | 2.0 | 1.23 | 1.0 | 2 | 14.9 | 0 |

In [193]:
```
x_train.columns
```

Out[193]:
```
Index(['VendorID', 'passenger_count', 'trip_distance', 'RatecodeID',
       'payment_type', 'fare_amount', 'extra', 'tip_amount', 'tolls_amount',
       'improvement_surcharge', 'congestion_surcharge', 'Airport_fee'],
      dtype='object')
```

In [199]:
```
new_data=pd.DataFrame({"VendorID":[2,1,2],
                       "passenger_count":[0,1,1],
                       "trip_distance":[7.65,1.88,0.67],
                       "RatecodeID":[4,2,1],
                       "payment_type":[2,1,1],
                       "fare_amount":[8.6,32.4,8.6],
                       "extra":[0,1,0],
                       "tip_amount":[2,0,2.52],
                       "tolls_amount":[0,6.94,0],
                       "improvement_surcharge":[1,1,1],
                       "congestion_surcharge":[2.5,0,2.5],
                       "Airport_fee":[0,1.75,0]})

np.expm1(lr.predict(new_data))
```

Out[199]:
```
array([2.38021072, 7.7328331 , 6.40957123])
```

In [ ]:
```

```

In [ ]:
```

```