1 pre-processing

[1]:
```
#import os
#os.chdir("Applied Data Science\project2\code")
#os.getcwd()
```

[2]:
```
import numpy as np
import pandas as pd
```

[3]:
```
df = pd.read_csv('data/yellow_tripdata_2019-02.csv')
df.dropna(inplace=True)
df
```

[3]:

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	\
0	1	2019-02-01 00:59:04	2019-02-01 01:07:27	1	
1	1	2019-02-01 00:33:09	2019-02-01 01:03:58	1	
2	1	2019-02-01 00:09:03	2019-02-01 00:09:16	1	
3	1	2019-02-01 00:45:38	2019-02-01 00:51:10	1	
4	1	2019-02-01 00:25:30	2019-02-01 00:28:14	1	
…	…	…	…	…	
7019370	2	2019-02-28 23:29:08	2019-02-28 23:29:11	1	
7019371	2	2019-02-28 22:48:47	2019-02-28 23:50:19	1	
7019372	2	2019-02-28 23:41:23	2019-02-28 23:42:23	1	
7019373	2	2019-02-28 23:12:52	2019-02-28 23:14:16	1	
7019374	2	2019-02-28 23:10:35	2019-02-28 23:10:37	1	

	trip_distance	RatecodeID	store_and_fwd_flag	PULocationID	\
0	2.1	1	N	48	
1	9.8	1	N	230	
2	0.0	1	N	145	
3	0.8	1	N	95	
4	0.8	1	N	140	
…	…	…	…	…	
7019370	0.0	1	N	193	
7019371	0.0	1	N	141	
7019372	0.0	1	N	264	
7019373	0.0	1	N	264	
7019374	0.0	1	N	264	

```
         DOLocationID  payment_type  fare_amount  extra  mta_tax  tip_amount  \
0                 234             1          9.0    0.5      0.5         2.0
1                  93             2         32.0    0.5      0.5         0.0
2                 145             2          2.5    0.5      0.5         0.0
3                  95             2          5.5    0.5      0.5         0.0
4                 263             2          5.0    0.5      0.5         0.0
…                 …               …          …      …        …           …
7019370           193             1          0.0    0.0      0.0         0.0
7019371           193             2          0.0    0.0      0.0         0.0
7019372           264             1          0.0    0.0      0.0         0.0
7019373           193             1          0.0    0.0      0.0         0.0
7019374           264             2          0.0    0.0      0.0         0.0

         tolls_amount  improvement_surcharge  total_amount  \
0                 0.0                    0.3          12.3
1                 0.0                    0.3          33.3
2                 0.0                    0.3           3.8
3                 0.0                    0.3           6.8
4                 0.0                    0.3           6.3
…                 …                      …             …
7019370           0.0                    0.0           0.0
7019371           0.0                    0.0           0.0
7019372           0.0                    0.0           0.0
7019373           0.0                    0.0           0.0
7019374           0.0                    0.0           0.0

         congestion_surcharge
0                         0.0
1                         0.0
2                         0.0
3                         0.0
4                         0.0
…                         …
7019370                   0.0
7019371                   2.5
7019372                   0.0
7019373                   0.0
7019374                   0.0

[7019375 rows x 18 columns]
```

1 add new attributes

[4]:
```
df['tpep_pickup_datetime'] = pd.to_datetime(df['tpep_pickup_datetime'],
                                format='%Y/%m/%d %H:%M', errors='coerce')
df['tpep_dropoff_datetime'] = pd.to_datetime(df['tpep_dropoff_datetime'],
                                format='%Y/%m/%d %H:%M', errors='coerce')
df['duration'] = (df['tpep_dropoff_datetime'] -
                  df['tpep_pickup_datetime']).dt.seconds.astype(int) /60
df["revenue"] = (df['tip_amount'] + df['fare_amount'])/df['duration'] *60
df['start_hour'] =  df['tpep_pickup_datetime'].dt.hour
df['day_of_week'] = df['tpep_pickup_datetime'].dt.day_name()
df['date_hour'] = df['tpep_pickup_datetime'].dt.strftime('%d %H')
```

[5]:
```
weather = pd.read_csv('data/2019_2_weather.csv')
weather['date_time'] = weather['date'].map(str) + " " + weather['time']
weather['date_hour'] = pd.to_datetime(weather['date_time'], format='%d %I:%M␣
 ↪%p',
                                      errors='coerce').dt.strftime('%d %H')

weather.set_index('date_hour', inplace=True)
weather
```

[5]:

date_hour	date	time	condition	date_time
01 23	1	11:51 PM	Fair	1 11:51 PM
01 00	1	12:51 AM	Fair	1 12:51 AM
01 01	1	1:51 AM	Fair	1 1:51 AM
01 02	1	2:51 AM	Fair	1 2:51 AM
01 03	1	3:51 AM	Fair	1 3:51 AM
…	…	…	…	…
28 18	28	6:51 PM	Mostly Cloudy	28 6:51 PM
28 19	28	7:51 PM	Mostly Cloudy	28 7:51 PM
28 20	28	8:51 PM	Mostly Cloudy	28 8:51 PM
28 21	28	9:51 PM	Mostly Cloudy	28 9:51 PM
28 22	28	10:51 PM	Mostly Cloudy	28 10:51 PM

[672 rows x 4 columns]

[6]:
```
preci = ['Heavy Rain','Light Drizzle','Light Freezing Drizzle','Light␣
 ↪Rain','Light Snow','Light Snow / Windy',
         'Rain','Snow','Snow and Sleet','Wintry Mix']

remain = ['Cloudy','Fair','Fair / Windy', 'Fog','Mostly Cloudy','Mostly Cloudy /
 ↪ Windy','Partly Cloudy',
          'Partly Cloudy / Windy', 'Sleet']

def precipitation_or_not(condition):
```

```
    if condition in preci:
        return 'preci'
    elif condition in remain:
        return 'remain'

weather['condition'] = weather.condition.map(precipitation_or_not)
weather
```

[6]:
```
            date      time condition    date_time
date_hour
01 23          1  11:51 PM    remain   1 11:51 PM
01 00          1  12:51 AM    remain   1 12:51 AM
01 01          1   1:51 AM    remain    1 1:51 AM
01 02          1   2:51 AM    remain    1 2:51 AM
01 03          1   3:51 AM    remain    1 3:51 AM
…          …        …        …           …
28 18         28   6:51 PM    remain   28 6:51 PM
28 19         28   7:51 PM    remain   28 7:51 PM
28 20         28   8:51 PM    remain   28 8:51 PM
28 21         28   9:51 PM    remain   28 9:51 PM
28 22         28  10:51 PM    remain  28 10:51 PM

[672 rows x 4 columns]
```

[7]:
```
def fill_weather(time):
    try:
        return weather['condition'].loc[time]
    except:
        return np.NaN
```

[8]:
```
df['weather'] = df['date_hour'].map(fill_weather)
```

[9]:
```
mta_tax = df['mta_tax'].value_counts()
mta_tax
```

[9]:
```
 0.50     6971733
 0.00       38469
-0.50        9165
 3.00           4
 0.89           1
 2.80           1
 3.30           1
 24.39          1
Name: mta_tax, dtype: int64
```

[10]:
```
improvement_surcharge = df['improvement_surcharge'].value_counts()
improvement_surcharge
```

[10]:
```
 0.3    7007247
-0.3       9468
 0.0       2648
 1.0         12
Name: improvement_surcharge, dtype: int64
```

[11]:
```
df
```

[11]:

	VendorID	tpep_pickup_datetime	tpep_dropoff_datetime	passenger_count	\
0	1	2019-02-01 00:59:04	2019-02-01 01:07:27	1	
1	1	2019-02-01 00:33:09	2019-02-01 01:03:58	1	
2	1	2019-02-01 00:09:03	2019-02-01 00:09:16	1	
3	1	2019-02-01 00:45:38	2019-02-01 00:51:10	1	
4	1	2019-02-01 00:25:30	2019-02-01 00:28:14	1	
…	…	…	…	…	
7019370	2	2019-02-28 23:29:08	2019-02-28 23:29:11	1	
7019371	2	2019-02-28 22:48:47	2019-02-28 23:50:19	1	
7019372	2	2019-02-28 23:41:23	2019-02-28 23:42:23	1	
7019373	2	2019-02-28 23:12:52	2019-02-28 23:14:16	1	
7019374	2	2019-02-28 23:10:35	2019-02-28 23:10:37	1	

	trip_distance	RatecodeID	store_and_fwd_flag	PULocationID	\
0	2.1	1	N	48	
1	9.8	1	N	230	
2	0.0	1	N	145	
3	0.8	1	N	95	
4	0.8	1	N	140	
…	…	…	…	…	
7019370	0.0	1	N	193	
7019371	0.0	1	N	141	
7019372	0.0	1	N	264	
7019373	0.0	1	N	264	
7019374	0.0	1	N	264	

	DOLocationID	payment_type	…	tolls_amount	improvement_surcharge	\
0	234	1	…	0.0	0.3	
1	93	2	…	0.0	0.3	
2	145	2	…	0.0	0.3	
3	95	2	…	0.0	0.3	
4	263	2	…	0.0	0.3	
…	…	…	…	…	…	
7019370	193	1	…	0.0	0.0	
7019371	193	2	…	0.0	0.0	
7019372	264	1	…	0.0	0.0	
7019373	193	1	…	0.0	0.0	
7019374	264	2	…	0.0	0.0	

```
         total_amount  congestion_surcharge   duration     revenue  \
0                12.3                   0.0   8.383333   78.727634
1                33.3                   0.0  30.816667   62.303948
2                 3.8                   0.0   0.216667  692.307692
3                 6.8                   0.0   5.533333   59.638554
4                 6.3                   0.0   2.733333  109.756098
…                 …                     …        …          …
7019370           0.0                   0.0   0.050000    0.000000
7019371           0.0                   2.5  61.533333    0.000000
7019372           0.0                   0.0   1.000000    0.000000
7019373           0.0                   0.0   1.400000    0.000000
7019374           0.0                   0.0   0.033333    0.000000

         start_hour  day_of_week  date_hour  weather
0                 0       Friday      01 00   remain
1                 0       Friday      01 00   remain
2                 0       Friday      01 00   remain
3                 0       Friday      01 00   remain
4                 0       Friday      01 00   remain
…                 …            …        …       …
7019370          23     Thursday      28 23   remain
7019371          22     Thursday      28 22   remain
7019372          23     Thursday      28 23   remain
7019373          23     Thursday      28 23   remain
7019374          23     Thursday      28 23   remain

[7019375 rows x 24 columns]
```

2 Data cleaning

[12]:
```
df = df.loc[(df['passenger_count'] > 0) & (df['passenger_count'] <= 4) &
            (df["fare_amount"] > 0) & (df['fare_amount'] <=80) &
            (df["duration"] >= 0.25) & (df["duration"] <= 100) &
            (df["tip_amount"] >= 0) & (df["tip_amount"] <=20) &
            (df["total_amount"]>0) & (df["total_amount"]<=100) &
            (df["trip_distance"] >= 0.01) &(df['trip_distance'] <= 30) &
            (df["revenue"] >0) &  (df["revenue"] <= 200) &
            (df["extra"] >= 0) & (df["mta_tax"] == 0.5) &␣
↪(df["congestion_surcharge"] >=0)&
            (df["tolls_amount"] >=0) &  (df["improvement_surcharge"] == 0.3) &
            (df['payment_type'] == 1) & (df['RatecodeID'] == 1)]
```

[13]:
```
drop_col = ['VendorID','tpep_pickup_datetime',␣
↪'tpep_dropoff_datetime','store_and_fwd_flag',
            'date_hour', 'congestion_surcharge',␣
↪'extra','mta_tax','improvement_surcharge',
```

```
            'payment_type','tolls_amount','total_amount','RatecodeID',␣
 ↪'PULocationID',
            'DOLocationID']
df.drop(drop_col, axis=1, inplace = True)
df.dropna(inplace=True)
df.reset_index(inplace=True, drop=True)
df
```

```
C:\Users\enzon\anaconda3\lib\site-packages\pandas\core\frame.py:3990:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  return super().drop(
<ipython-input-13-9140f16c5c92>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  df.dropna(inplace=True)
```

[13]:

	passenger_count	trip_distance	fare_amount	tip_amount	duration	\
0	1	2.10	9.0	2.00	8.383333	
1	2	1.80	9.0	2.05	9.800000	
2	1	0.80	5.5	2.00	4.950000	
3	1	9.00	26.0	7.50	16.816667	
4	1	1.80	8.5	1.95	8.650000	
…	…	…	…	…	…	
4535200	1	7.90	24.0	10.05	20.000000	
4535201	1	2.86	13.0	4.20	15.950000	
4535202	3	3.20	14.5	3.66	18.333333	
4535203	1	1.22	6.5	1.00	6.183333	
4535204	2	1.75	8.5	1.84	10.616667	

	revenue	start_hour	day_of_week	weather
0	78.727634	0	Friday	remain
1	67.653061	0	Friday	remain
2	90.909091	0	Friday	remain
3	119.524281	0	Friday	remain
4	72.485549	0	Friday	remain
…	…	…	…	…
4535200	102.150000	23	Thursday	remain
4535201	64.702194	23	Thursday	remain
4535202	59.432727	23	Thursday	remain
4535203	72.776280	23	Thursday	remain
4535204	58.436421	23	Thursday	remain

```
[4535205 rows x 9 columns]
```

[14]:
```
df.to_feather('data/clean.feather')
```

[ ]:
```

```

2 Box-Cox transformation

[1]:
```
#import os
#os.chdir("Applied Data Science\project2\code")
#os.getcwd()
```

[2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
#from scipy.stats import chi2_contingency
```

[3]:
```
df = pd.read_feather('data/clean.feather')
df
```

[3]:

	passenger_count	trip_distance	fare_amount	tip_amount	duration	\
0	1	2.10	9.0	2.00	8.383333	
1	2	1.80	9.0	2.05	9.800000	
2	1	0.80	5.5	2.00	4.950000	
3	1	9.00	26.0	7.50	16.816667	
4	1	1.80	8.5	1.95	8.650000	
…	…	…	…	…	…	
4535200	1	7.90	24.0	10.05	20.000000	
4535201	1	2.86	13.0	4.20	15.950000	
4535202	3	3.20	14.5	3.66	18.333333	
4535203	1	1.22	6.5	1.00	6.183333	
4535204	2	1.75	8.5	1.84	10.616667	

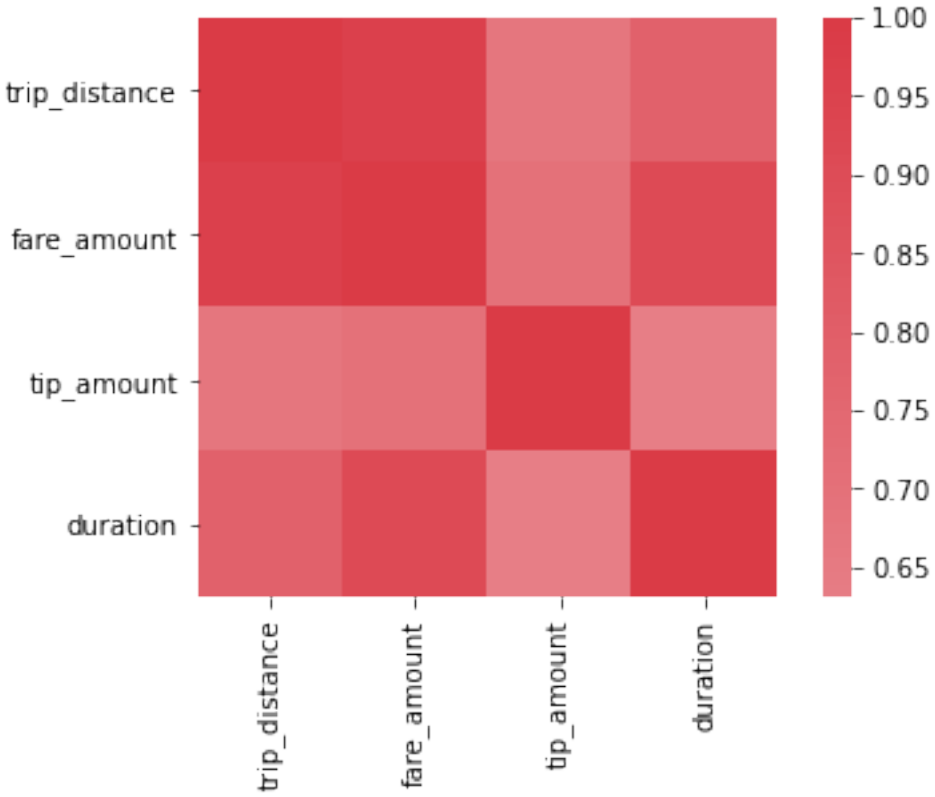
	revenue	start_hour	day_of_week	weather
0	78.727634	0	Friday	remain
1	67.653061	0	Friday	remain
2	90.909091	0	Friday	remain
3	119.524281	0	Friday	remain
4	72.485549	0	Friday	remain
…	…	…	…	…
4535200	102.150000	23	Thursday	remain
4535201	64.702194	23	Thursday	remain
4535202	59.432727	23	Thursday	remain
4535203	72.776280	23	Thursday	remain

```
4535204   58.436421          23    Thursday  remain

[4535205 rows x 9 columns]
```

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

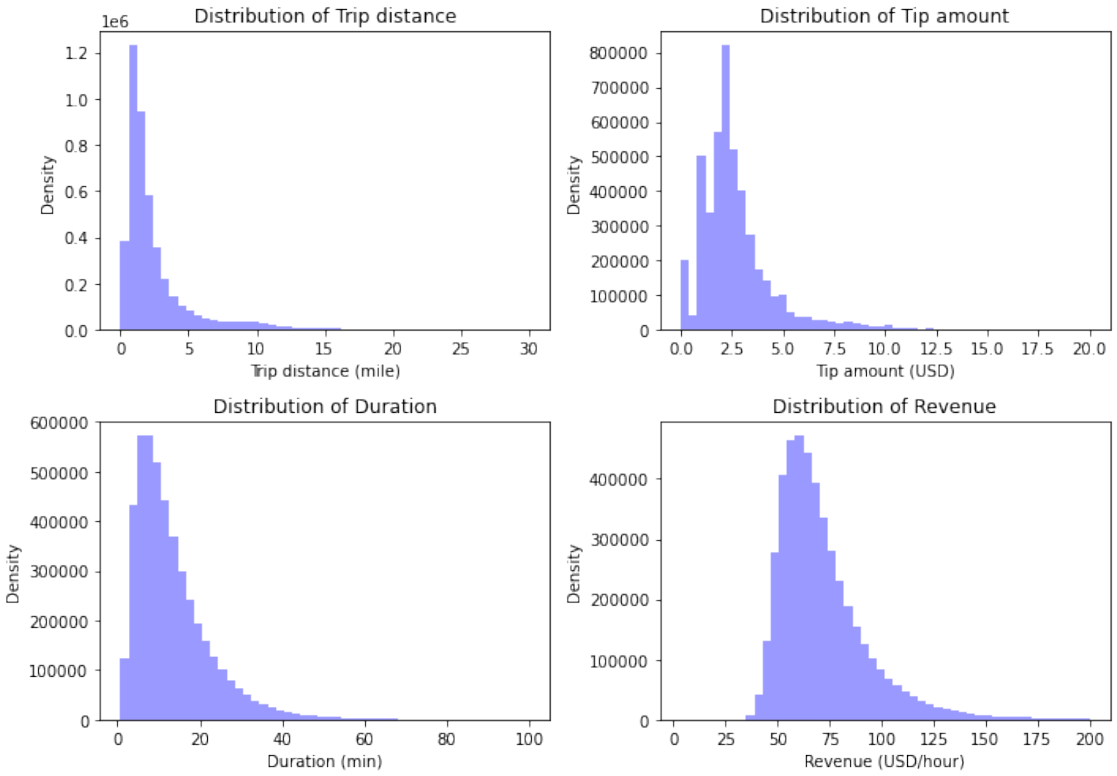
[4]:
```
(4535205, 9)
```

[5]:
```
corr_attr =['trip_distance', 'fare_amount', 'tip_amount', 'duration']
corr = df[corr_attr].corr()
sns.heatmap(corr,cmap = sns.diverging_palette(220, 10, as_cmap=True),␣
 ↪square=True, center=0)
#plt.title('Pearson correlation matrix')
plt.savefig('plots/correlation.png')
plt.show()
```

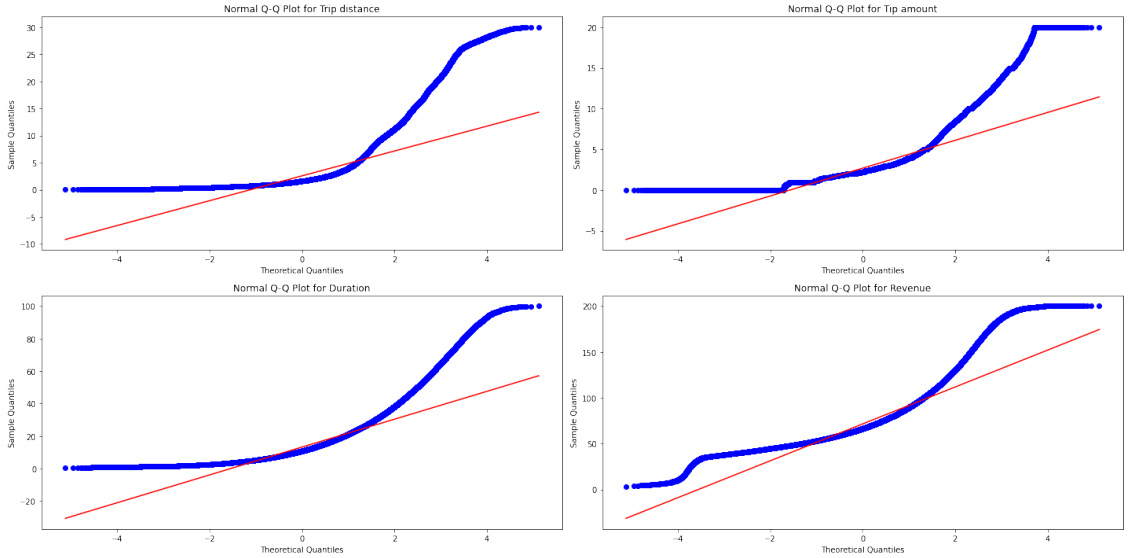


[6]:
```
fig = plt.figure(figsize=(10, 7))
plt.subplot(221)
sns.distplot(df['trip_distance'], kde = False, label = "trip_distance", color␣
 ↪="blue")
plt.title("Distribution of Trip distance")
```

```
plt.xlabel('Trip distance (mile)')
plt.ylabel("Density")

plt.subplot(222)
sns.distplot(df['tip_amount'], kde = False, label = "tip_amount", color ="blue")
plt.title("Distribution of Tip amount")
plt.xlabel('Tip amount (USD)')
plt.ylabel("Density")

plt.subplot(223)
sns.distplot(df['duration'], kde = False, label = "duration", color ="blue")
plt.title("Distribution of Duration")
plt.xlabel('Duration (min)')
plt.ylabel("Density")

plt.subplot(224)
sns.distplot(df['revenue'], kde = False, label = "revenue", color ="blue")
plt.title("Distribution of Revenue")
plt.xlabel('Revenue (USD/hour)')
plt.ylabel("Density")

plt.tight_layout()
plt.show()
```



[7]:
```
fig = plt.figure(figsize=(20, 10))
plt.subplot(221)
stats.probplot(df['trip_distance'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for Trip distance")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(222)
stats.probplot(df['tip_amount'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for Tip amount")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(223)
stats.probplot(df['duration'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for Duration")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(224)
stats.probplot(df['revenue'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for Revenue")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.tight_layout()
plt.show()
```



1 Transform

[8]:
```
def check_if_0(x):
    if x==0:
        return 0.00001
    else:
        return x

df['tip_amount'] = df['tip_amount'].map(check_if_0)
```

[ ]:
```
```

[9]:
```
from sklearn.model_selection import train_test_split

train, test = train_test_split(df, test_size=0.3, random_state=60, shuffle=True)
train.reset_index(inplace = True, drop=True)
test.reset_index(inplace = True, drop=True)
```

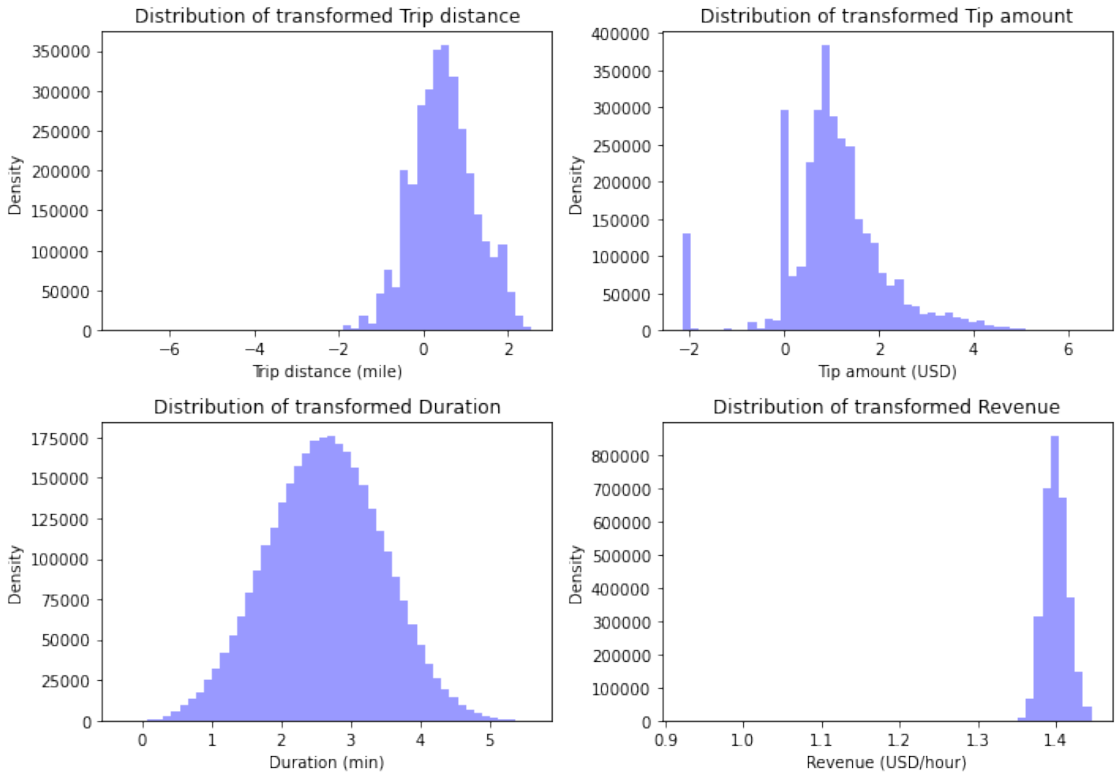
[10]:
```
train.to_feather('data/train.feather')
test.to_feather('data/test.feather')
```

[11]:
```
lambda_list = []
continuous_col = ['trip_distance', 'fare_amount', 'tip_amount','duration',␣
 ↪'revenue']
for i in continuous_col:
    out = stats.boxcox(train[i], lmbda=None)
    train[i] = out[0]
    lambda_list.append(out[1])
```

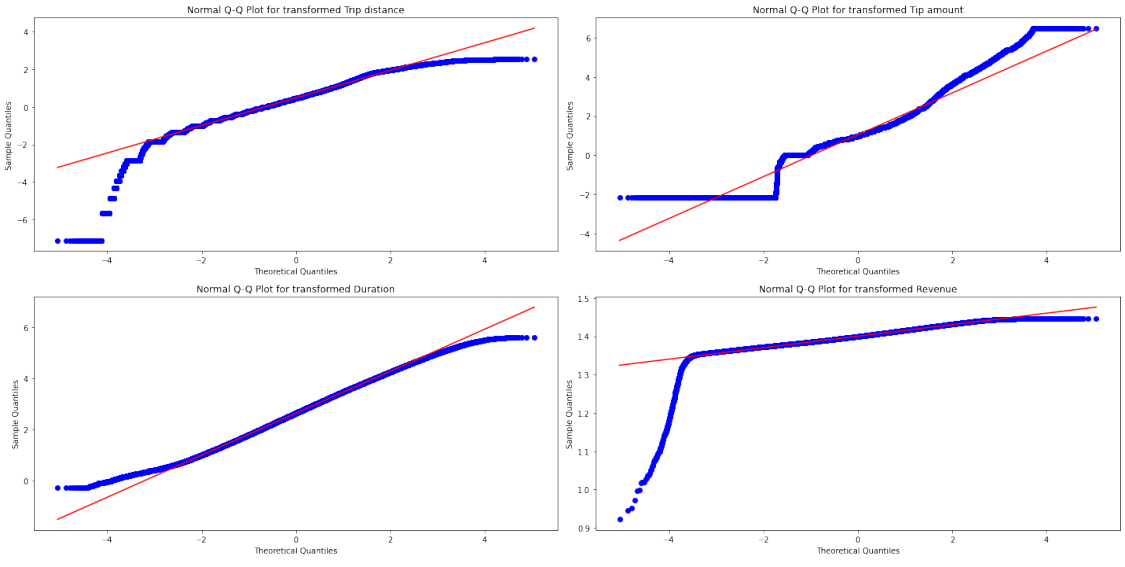
```
<ipython-input-11-ce667e7d2bff>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  train[i] = out[0]
```

[12]:
```
fig = plt.figure(figsize=(10, 7))
plt.subplot(221)
sns.distplot(train['trip_distance'], kde = False, label = "trip_distance",␣
 ↪color ="blue")
plt.title("Distribution of transformed Trip distance")
plt.xlabel('Trip distance (mile)')
plt.ylabel("Density")

plt.subplot(222)
```

```
sns.distplot(train['tip_amount'], kde = False, label = "tip_amount", color␣
↪="blue")
plt.title("Distribution of transformed Tip amount")
plt.xlabel('Tip amount (USD)')
plt.ylabel("Density")

plt.subplot(223)
sns.distplot(train['duration'], kde = False, label = "duration", color ="blue")
plt.title("Distribution of transformed Duration")
plt.xlabel('Duration (min)')
plt.ylabel("Density")

plt.subplot(224)
sns.distplot(train['revenue'], kde = False, label = "income", color ="blue")
plt.title("Distribution of transformed Revenue")
plt.xlabel('Revenue (USD/hour)')
plt.ylabel("Density")


plt.tight_layout()
plt.show()
```



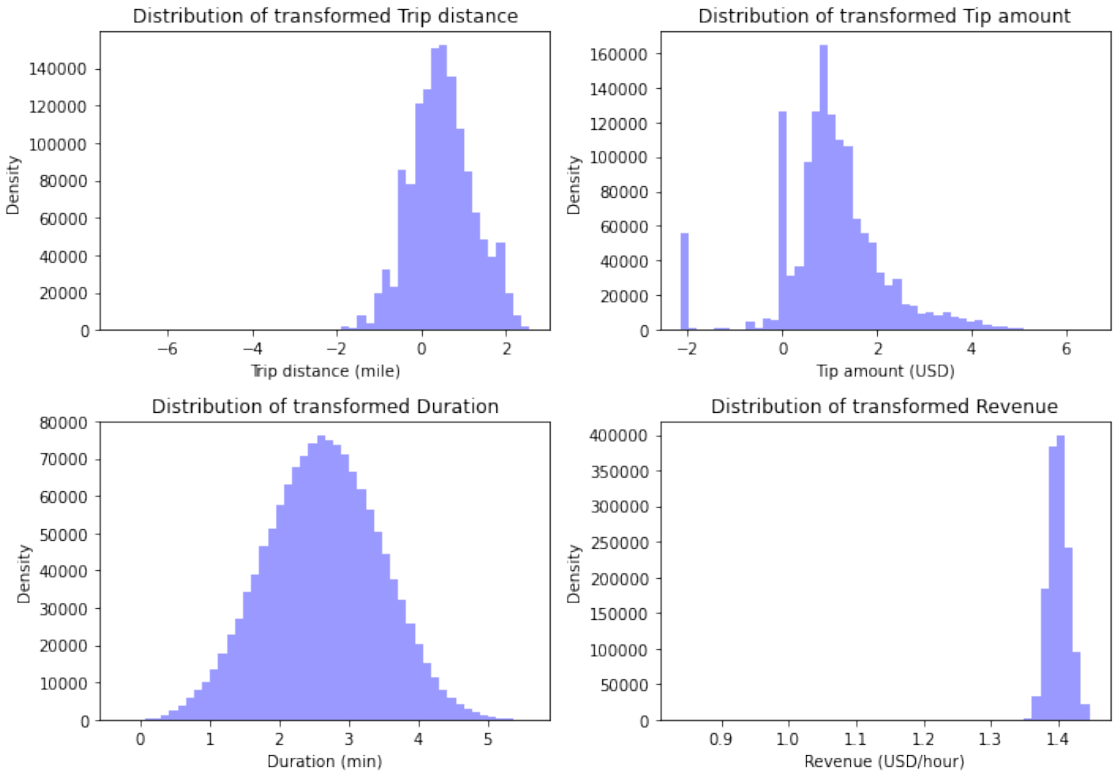
[13]:
```
fig = plt.figure(figsize=(20, 10))
plt.subplot(221)
stats.probplot(train['trip_distance'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Trip distance")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(222)
stats.probplot(train['tip_amount'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Tip amount")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(223)
stats.probplot(train['duration'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Duration")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(224)
stats.probplot(train['revenue'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Revenue")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.tight_layout()
plt.show()
```



[14]:
```
for i,j  in enumerate(continuous_col):
    out = stats.boxcox(test[j], lmbda=lambda_list[i])
    test[j] = out
```

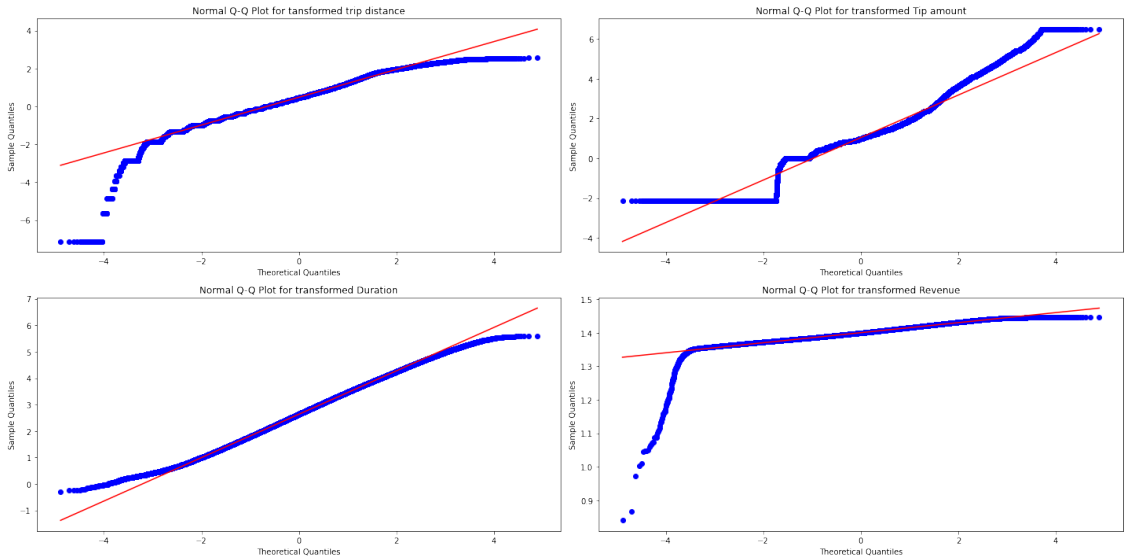
```
<ipython-input-14-be7d8a283fd5>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  test[j] = out
```

[15]:
```
fig = plt.figure(figsize=(10, 7))
plt.subplot(221)
sns.distplot(test['trip_distance'], kde = False, label = "trip_distance", color␣
 ↪="blue")
plt.title("Distribution of transformed Trip distance")
plt.xlabel('Trip distance (mile)')
plt.ylabel("Density")

plt.subplot(222)
sns.distplot(test['tip_amount'], kde = False, label = "tip_amount", color␣
 ↪="blue")
plt.title("Distribution of transformed Tip amount")
plt.xlabel('Tip amount (USD)')
plt.ylabel("Density")

plt.subplot(223)
sns.distplot(test['duration'], kde = False, label = "duration", color ="blue")
plt.title("Distribution of transformed Duration")
plt.xlabel('Duration (min)')
plt.ylabel("Density")

plt.subplot(224)
sns.distplot(test['revenue'], kde = False, label = "revenue", color ="blue")
plt.title("Distribution of transformed Revenue")
plt.xlabel('Revenue (USD/hour)')
plt.ylabel("Density")


plt.tight_layout()
plt.show()
```



[16]:
```
fig = plt.figure(figsize=(20, 10))
plt.subplot(221)
stats.probplot(test['trip_distance'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for tansformed trip distance")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(222)
stats.probplot(test['tip_amount'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Tip amount")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(223)
stats.probplot(test['duration'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Duration")
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.subplot(224)
stats.probplot(test['revenue'], dist="norm", plot=plt)
plt.title("Normal Q-Q Plot for transformed Revenue")
```

```
plt.xlabel('Theoretical Quantiles')
plt.ylabel("Sample Quantiles")

plt.tight_layout()
plt.show()
```



[17]:
```
train.to_csv('data/train_scaled.csv', index=False)
train.to_feather('data/train_scaled.feather')
test.to_csv('data/test_scaled.csv', index=False)
test.to_feather('data/test_scaled.feather')
```

[18]:
```
lambda_list =  pd.DataFrame ([lambda_list],columns=continuous_col)
lambda_list
```

[18]:

	trip_distance	fare_amount	tip_amount	duration	revenue
0	-0.177254	-0.440714	0.463613	0.082524	-0.671899

[19]:
```
lambda_list.to_csv('data/lambda_list.csv')
```

3 train, refine model

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
scaled.df = read.csv("data/train_scaled.csv")
str(scaled.df)
```

```
## 'data.frame':    200000 obs. of  9 variables:
##  $ passenger_count: int  1 1 1 2 4 1 1 1 2 1 ...
##  $ trip_distance  : num  1.4753 0.9427 0.2033 0.0673 1.8308 ...
##  $ fare_amount    : num  1.72 1.49 1.33 1.38 1.78 ...
##  $ tip_amount     : num  2.09 1.4 1.26 1.05 4.44 ...
##  $ duration       : num  4.06 2.94 2.4 2.67 4.38 ...
##  $ revenue        : num  1.37 1.38 1.38 1.37 1.38 ...
##  $ start_hour     : int  10 12 8 14 13 12 10 15 11 16 ...
##  $ day_of_week    : chr  "Friday" "Thursday" "Tuesday" "Thursday" ...
##  $ weather        : chr  "remain" "remain" "remain" "remain" ...
```

```
scaled.df$passenger_count = factor(scaled.df$passenger_count,levels = c(1,2,3,4), ordered = TRUE)
scaled.df$start_hour = factor(scaled.df$start_hour)
scaled.df$day_of_week = factor(scaled.df$day_of_week)
scaled.df$weather = factor(scaled.df$weather)
str(scaled.df)
```

```
## 'data.frame':    200000 obs. of  9 variables:
##  $ passenger_count: Ord.factor w/ 4 levels "1"<"2"<"3"<"4": 1 1 1 2 4 1 1 1 2 1 ...
##  $ trip_distance  : num  1.4753 0.9427 0.2033 0.0673 1.8308 ...
##  $ fare_amount    : num  1.72 1.49 1.33 1.38 1.78 ...
##  $ tip_amount     : num  2.09 1.4 1.26 1.05 4.44 ...
##  $ duration       : num  4.06 2.94 2.4 2.67 4.38 ...
##  $ revenue        : num  1.37 1.38 1.38 1.37 1.38 ...
##  $ start_hour     : Factor w/ 24 levels "0","1","2","3",..: 11 13 9 15 14 13 11 16 12 17 ...
##  $ day_of_week    : Factor w/ 7 levels "Friday","Monday",..: 1 5 6 5 7 2 6 7 5 5 ...
##  $ weather        : Factor w/ 2 levels "preci","remain": 2 2 2 2 1 2 2 2 2 2 ...
```

```
##
## Call:
## lm(formula = revenue ~ passenger_count + trip_distance + duration +
##     start_hour + weather + day_of_week + duration * start_hour +
##     duration * day_of_week + duration * weather, data = scaled.df)
##
```

```
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.300309 -0.004241 -0.000859  0.003657  0.141984
##
## Coefficients:
##                                Estimate Std. Error  t value Pr(>|t|)
## (Intercept)                   1.441e+00  4.259e-04 3382.979  < 2e-16 ***
## passenger_count.L            -8.727e-05  8.917e-05   -0.979 0.327696
## passenger_count.Q             1.243e-04  7.980e-05    1.557 0.119361
## passenger_count.C             7.012e-05  6.922e-05    1.013 0.311019
## trip_distance                 1.918e-02  4.857e-05  394.933  < 2e-16 ***
## duration                     -2.534e-02  1.590e-04 -159.310  < 2e-16 ***
## start_hour1                   3.097e-03  5.558e-04    5.572 2.52e-08 ***
## start_hour2                   2.516e-03  6.245e-04    4.029 5.60e-05 ***
## start_hour3                   1.948e-03  6.995e-04    2.786 0.005343 **
## start_hour4                  -1.841e-03  8.219e-04   -2.240 0.025082 *
## start_hour5                  -3.342e-03  7.227e-04   -4.625 3.75e-06 ***
## start_hour6                   5.445e-04  5.179e-04    1.051 0.293140
## start_hour7                   2.067e-03  4.663e-04    4.432 9.34e-06 ***
## start_hour8                   5.052e-03  4.499e-04   11.227  < 2e-16 ***
## start_hour9                   4.551e-03  4.479e-04   10.160  < 2e-16 ***
## start_hour10                  4.983e-03  4.550e-04   10.951  < 2e-16 ***
## start_hour11                  4.829e-03  4.604e-04   10.489  < 2e-16 ***
## start_hour12                  4.992e-03  4.545e-04   10.984  < 2e-16 ***
## start_hour13                  3.998e-03  4.598e-04    8.696  < 2e-16 ***
## start_hour14                  3.533e-03  4.471e-04    7.903 2.74e-15 ***
## start_hour15                  3.321e-03  4.466e-04    7.436 1.04e-13 ***
## start_hour16                  3.658e-03  4.446e-04    8.228  < 2e-16 ***
## start_hour17                  4.833e-03  4.380e-04   11.034  < 2e-16 ***
## start_hour18                  5.799e-03  4.311e-04   13.451  < 2e-16 ***
## start_hour19                  5.060e-03  4.359e-04   11.607  < 2e-16 ***
## start_hour20                  3.309e-03  4.425e-04    7.479 7.52e-14 ***
## start_hour21                  1.556e-03  4.453e-04    3.495 0.000475 ***
## start_hour22                  1.466e-03  4.486e-04    3.269 0.001081 **
## start_hour23                  4.765e-04  4.684e-04    1.017 0.308981
## weatherremain                -1.983e-03  1.875e-04  -10.579  < 2e-16 ***
## day_of_weekMonday            -6.461e-04  2.228e-04   -2.899 0.003741 **
## day_of_weekSaturday           2.292e-03  2.185e-04   10.489  < 2e-16 ***
## day_of_weekSunday            -1.525e-04  2.291e-04   -0.666 0.505583
## day_of_weekThursday           6.409e-05  2.129e-04    0.301 0.763383
## day_of_weekTuesday            7.759e-04  2.195e-04    3.536 0.000407 ***
## day_of_weekWednesday          1.136e-03  2.166e-04    5.243 1.58e-07 ***
## duration:start_hour1         -1.301e-03  2.103e-04   -6.183 6.29e-10 ***
## duration:start_hour2         -1.099e-03  2.403e-04   -4.572 4.84e-06 ***
## duration:start_hour3         -5.937e-04  2.710e-04   -2.191 0.028442 *
## duration:start_hour4          1.499e-03  3.155e-04    4.752 2.01e-06 ***
## duration:start_hour5          2.164e-03  2.887e-04    7.494 6.71e-14 ***
## duration:start_hour6         -3.089e-04  2.013e-04   -1.534 0.124979
## duration:start_hour7         -1.346e-03  1.746e-04   -7.706 1.31e-14 ***
## duration:start_hour8         -2.309e-03  1.653e-04  -13.969  < 2e-16 ***
## duration:start_hour9         -1.818e-03  1.640e-04  -11.085  < 2e-16 ***
## duration:start_hour10        -2.004e-03  1.670e-04  -12.000  < 2e-16 ***
## duration:start_hour11        -1.939e-03  1.686e-04  -11.496  < 2e-16 ***
## duration:start_hour12        -1.932e-03  1.667e-04  -11.591  < 2e-16 ***
```

```
## duration:start_hour13         -1.751e-03  1.682e-04  -10.408  < 2e-16 ***
## duration:start_hour14         -1.498e-03  1.634e-04   -9.169  < 2e-16 ***
## duration:start_hour15         -1.609e-03  1.630e-04   -9.868  < 2e-16 ***
## duration:start_hour16         -1.579e-03  1.626e-04   -9.706  < 2e-16 ***
## duration:start_hour17         -2.126e-03  1.602e-04  -13.267  < 2e-16 ***
## duration:start_hour18         -2.520e-03  1.583e-04  -15.922  < 2e-16 ***
## duration:start_hour19         -2.262e-03  1.609e-04  -14.059  < 2e-16 ***
## duration:start_hour20         -1.600e-03  1.638e-04   -9.768  < 2e-16 ***
## duration:start_hour21         -8.242e-04  1.645e-04   -5.009 5.47e-07 ***
## duration:start_hour22         -6.943e-04  1.654e-04   -4.197 2.71e-05 ***
## duration:start_hour23         -1.907e-04  1.722e-04   -1.107 0.268243
## duration:day_of_weekMonday     5.146e-04  8.135e-05    6.326 2.52e-10 ***
## duration:day_of_weekSaturday  -1.034e-03  7.975e-05  -12.967  < 2e-16 ***
## duration:day_of_weekSunday     5.231e-05  8.534e-05    0.613 0.539896
## duration:day_of_weekThursday   2.135e-04  7.544e-05    2.831 0.004646 **
## duration:day_of_weekTuesday   -7.931e-05  7.905e-05   -1.003 0.315714
## duration:day_of_weekWednesday -1.717e-04  7.741e-05   -2.218 0.026567 *
## duration:weatherremain         7.840e-04  6.894e-05   11.373  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007824 on 199934 degrees of freedom
## Multiple R-squared:  0.7068, Adjusted R-squared:  0.7067
## F-statistic:  7413 on 65 and 199934 DF,  p-value: < 2.2e-16
```

```
model = step(model,scope = revenue ~passenger_count+ trip_distance+ duration    + start_hour + weather+
             duration*start_hour + duration*day_of_week + duration * weather,k = log(dim(scaled.df)[1]))
```

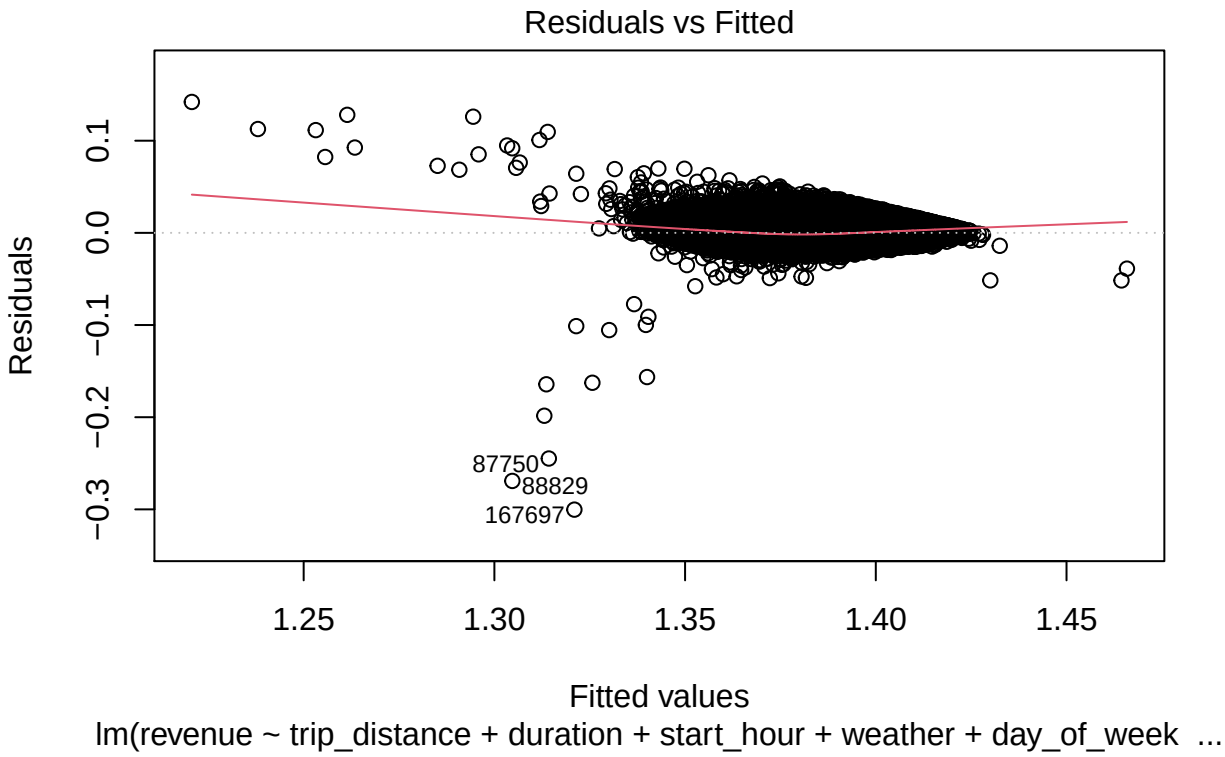
```
## Start:  AIC=-1939487
## revenue ~ passenger_count + trip_distance + duration + start_hour +
##     weather + day_of_week + duration * start_hour + duration *
##     day_of_week + duration * weather
##
##                         Df Sum of Sq    RSS      AIC
## - passenger_count        3    0.0007 12.239 -1939513
## <none>                               12.239 -1939487
## - duration:weather       1    0.0079 12.247 -1939370
## - duration:day_of_week   6    0.0244 12.263 -1939163
## - duration:start_hour   23    0.0744 12.313 -1938555
## - trip_distance          1    9.5477 21.787 -1824163
##
## Step:  AIC=-1939513
## revenue ~ trip_distance + duration + start_hour + weather + day_of_week +
##     duration:start_hour + duration:day_of_week + duration:weather
##
##                         Df Sum of Sq    RSS      AIC
## <none>                               12.239 -1939513
## + passenger_count        3    0.0007 12.239 -1939487
## - duration:weather       1    0.0079 12.247 -1939396
## - duration:day_of_week   6    0.0244 12.264 -1939188
## - duration:start_hour   23    0.0744 12.314 -1938582
## - trip_distance          1    9.5506 21.790 -1824167
```

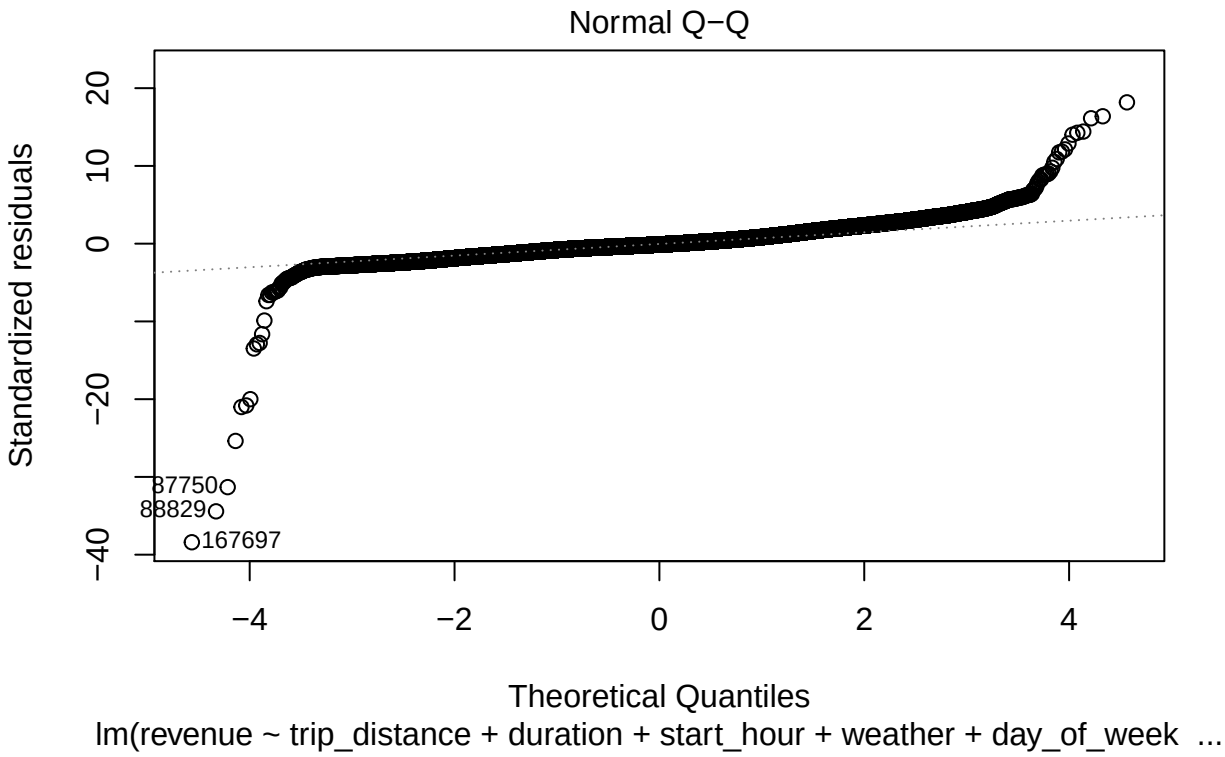
```r
summary(model)
```

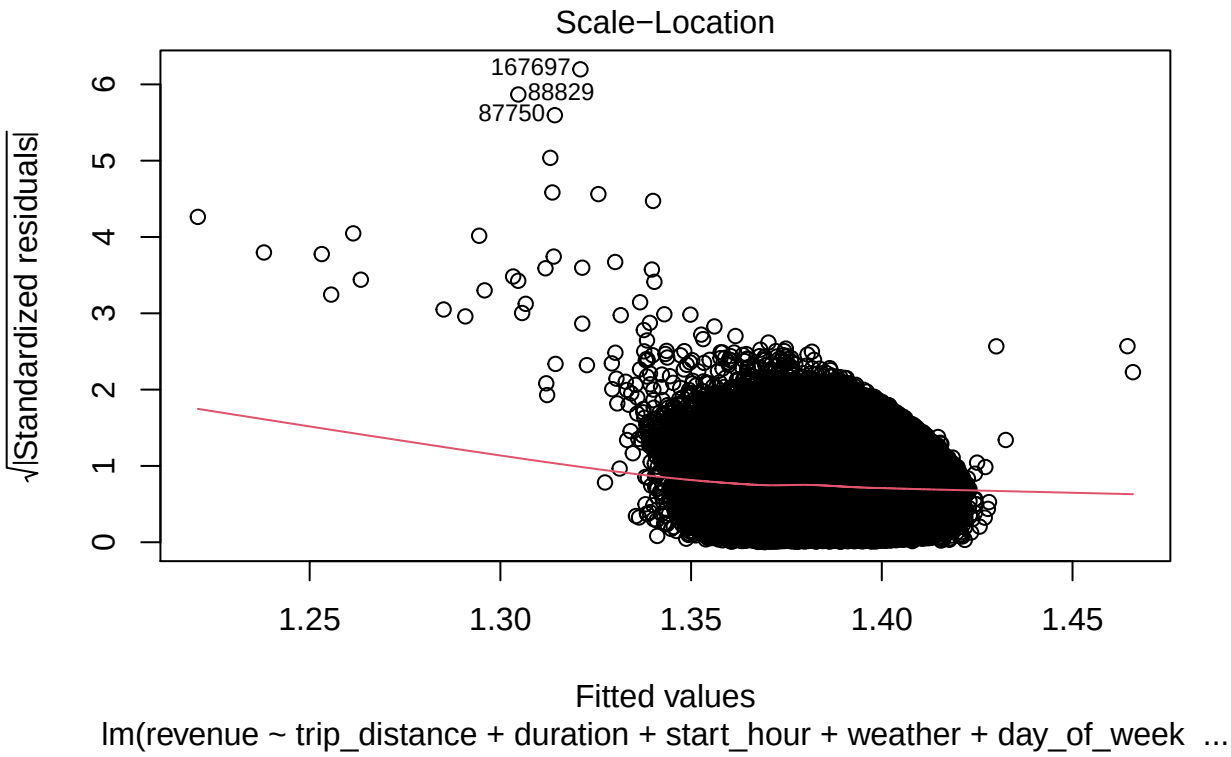
```
##
## Call:
## lm(formula = revenue ~ trip_distance + duration + start_hour +
##     weather + day_of_week + duration:start_hour + duration:day_of_week +
##     duration:weather, data = scaled.df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.300279 -0.004243 -0.000859  0.003658  0.142020
##
## Coefficients:
##                                Estimate Std. Error  t value Pr(>|t|)
## (Intercept)                   1.441e+00  4.243e-04 3395.290  < 2e-16 ***
## trip_distance                 1.918e-02  4.857e-05  394.985  < 2e-16 ***
## duration                     -2.534e-02  1.590e-04 -159.313  < 2e-16 ***
## start_hour1                   3.102e-03  5.558e-04    5.582 2.39e-08 ***
## start_hour2                   2.515e-03  6.245e-04    4.026 5.66e-05 ***
## start_hour3                   1.949e-03  6.995e-04    2.787 0.005323 **
## start_hour4                  -1.833e-03  8.219e-04   -2.230 0.025763 *
## start_hour5                  -3.323e-03  7.226e-04   -4.599 4.25e-06 ***
## start_hour6                   5.622e-04  5.179e-04    1.085 0.277732
## start_hour7                   2.079e-03  4.663e-04    4.458 8.27e-06 ***
## start_hour8                   5.058e-03  4.499e-04   11.240  < 2e-16 ***
## start_hour9                   4.563e-03  4.479e-04   10.187  < 2e-16 ***
## start_hour10                  4.996e-03  4.550e-04   10.980  < 2e-16 ***
## start_hour11                  4.840e-03  4.604e-04   10.514  < 2e-16 ***
## start_hour12                  5.007e-03  4.545e-04   11.015  < 2e-16 ***
## start_hour13                  4.009e-03  4.598e-04    8.718  < 2e-16 ***
## start_hour14                  3.546e-03  4.471e-04    7.930 2.20e-15 ***
## start_hour15                  3.328e-03  4.466e-04    7.451 9.29e-14 ***
## start_hour16                  3.662e-03  4.446e-04    8.238  < 2e-16 ***
## start_hour17                  4.836e-03  4.380e-04   11.041  < 2e-16 ***
## start_hour18                  5.803e-03  4.311e-04   13.461  < 2e-16 ***
## start_hour19                  5.062e-03  4.359e-04   11.612  < 2e-16 ***
## start_hour20                  3.307e-03  4.425e-04    7.475 7.77e-14 ***
## start_hour21                  1.558e-03  4.453e-04    3.498 0.000469 ***
## start_hour22                  1.468e-03  4.486e-04    3.272 0.001069 **
## start_hour23                  4.758e-04  4.684e-04    1.016 0.309664
## weatherremain                -1.984e-03  1.875e-04  -10.584  < 2e-16 ***
## day_of_weekMonday            -6.407e-04  2.228e-04   -2.875 0.004037 **
## day_of_weekSaturday           2.291e-03  2.185e-04   10.481  < 2e-16 ***
## day_of_weekSunday            -1.544e-04  2.291e-04   -0.674 0.500375
## day_of_weekThursday           6.904e-05  2.129e-04    0.324 0.745697
## day_of_weekTuesday            7.777e-04  2.195e-04    3.543 0.000395 ***
## day_of_weekWednesday          1.139e-03  2.166e-04    5.260 1.45e-07 ***
## duration:start_hour1         -1.302e-03  2.103e-04   -6.189 6.06e-10 ***
## duration:start_hour2         -1.097e-03  2.403e-04   -4.566 4.98e-06 ***
## duration:start_hour3         -5.937e-04  2.710e-04   -2.191 0.028448 *
## duration:start_hour4          1.498e-03  3.155e-04    4.750 2.04e-06 ***
## duration:start_hour5          2.160e-03  2.887e-04    7.482 7.36e-14 ***
## duration:start_hour6         -3.115e-04  2.013e-04   -1.547 0.121825
## duration:start_hour7         -1.347e-03  1.746e-04   -7.713 1.23e-14 ***
```

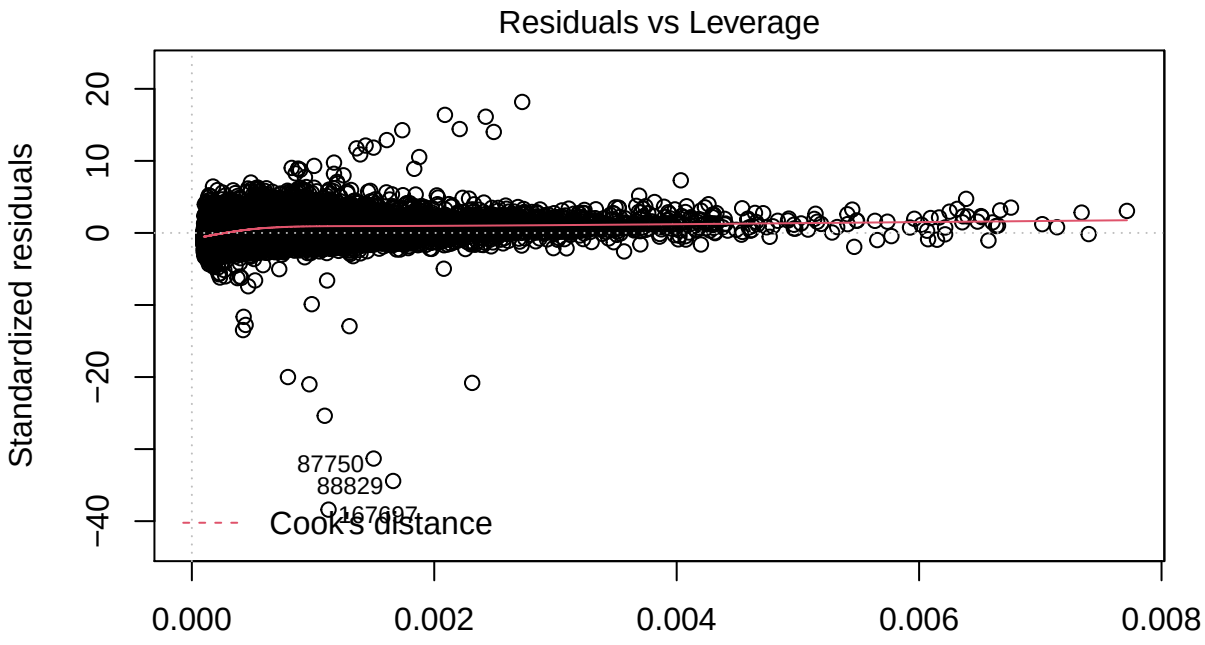
```
## duration:start_hour8          -2.308e-03  1.653e-04  -13.964  < 2e-16 ***
## duration:start_hour9          -1.819e-03  1.640e-04  -11.092  < 2e-16 ***
## duration:start_hour10         -2.006e-03  1.670e-04  -12.011  < 2e-16 ***
## duration:start_hour11         -1.940e-03  1.686e-04  -11.506  < 2e-16 ***
## duration:start_hour12         -1.935e-03  1.667e-04  -11.608  < 2e-16 ***
## duration:start_hour13         -1.752e-03  1.682e-04  -10.419  < 2e-16 ***
## duration:start_hour14         -1.501e-03  1.634e-04   -9.182  < 2e-16 ***
## duration:start_hour15         -1.610e-03  1.630e-04   -9.876  < 2e-16 ***
## duration:start_hour16         -1.579e-03  1.626e-04   -9.708  < 2e-16 ***
## duration:start_hour17         -2.126e-03  1.602e-04  -13.265  < 2e-16 ***
## duration:start_hour18         -2.520e-03  1.583e-04  -15.925  < 2e-16 ***
## duration:start_hour19         -2.261e-03  1.609e-04  -14.056  < 2e-16 ***
## duration:start_hour20         -1.599e-03  1.638e-04   -9.760  < 2e-16 ***
## duration:start_hour21         -8.247e-04  1.645e-04   -5.012 5.39e-07 ***
## duration:start_hour22         -6.952e-04  1.654e-04   -4.202 2.64e-05 ***
## duration:start_hour23         -1.910e-04  1.722e-04   -1.109 0.267494
## duration:day_of_weekMonday     5.130e-04  8.135e-05    6.307 2.86e-10 ***
## duration:day_of_weekSaturday  -1.036e-03  7.975e-05  -12.986  < 2e-16 ***
## duration:day_of_weekSunday     5.129e-05  8.534e-05    0.601 0.547886
## duration:day_of_weekThursday   2.125e-04  7.544e-05    2.816 0.004857 **
## duration:day_of_weekTuesday   -7.936e-05  7.905e-05   -1.004 0.315392
## duration:day_of_weekWednesday -1.721e-04  7.741e-05   -2.223 0.026190 *
## duration:weatherremain         7.844e-04  6.894e-05   11.378  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.007824 on 199937 degrees of freedom
## Multiple R-squared:  0.7067, Adjusted R-squared:  0.7066
## F-statistic:  7772 on 62 and 199937 DF,  p-value: < 2.2e-16
```

```
plot(model)
```


Residuals vs Fitted
Residuals
0.1
0.0
−0.1
−0.2
−0.3
87750
88829
167697
1.25
1.30
1.35
1.40
1.45
Fitted values
lm(revenue ~ trip_distance + duration + start_hour + weather + day_of_week ...


Normal Q−Q
Standardized residuals
20
10
0
−20
−40
37750
88829
167697
−4
−2
0
2
4
Theoretical Quantiles
lm(revenue ~ trip_distance + duration + start_hour + weather + day_of_week ...


Scale−Location
167697
88829
87750
√|Standardized residuals|
0
1
2
3
4
5
6
1.25
1.30
1.35
1.40
1.45
Fitted values
lm(revenue ~ trip_distance + duration + start_hour + weather + day_of_week ...




```
test.df = read.csv("data/test_scaled.csv")
str(test.df)
```

```
## 'data.frame':    4335205 obs. of  9 variables:
##  $ passenger_count: int  1 1 1 1 4 1 1 1 2 1 ...
##  $ trip_distance  : num  0.1465 0.474 -0.5714 0.764 -0.0202 ...
##  $ fare_amount    : num  1.27 1.4 1.24 1.67 1.24 ...
##  $ tip_amount     : num  0.414 0 1.313 2.82 0 ...
##  $ duration       : num  1.95 2.59 2.22 4.15 2.04 ...
##  $ revenue        : num  1.39 1.37 1.38 1.36 1.38 ...
##  $ start_hour     : int  14 23 9 14 18 10 17 23 9 20 ...
##  $ day_of_week    : chr  "Monday" "Wednesday" "Tuesday" "Wednesday" ...
##  $ weather        : chr  "remain" "remain" "preci" "remain" ...
```

```
test.df$passenger_count = factor(test.df$passenger_count,levels = c(1,2,3,4), ordered = TRUE)
test.df$start_hour = factor(test.df$start_hour)
test.df$weather = factor(test.df$weather)
str(test.df)
```

```
## 'data.frame':    4335205 obs. of  9 variables:
##  $ passenger_count: Ord.factor w/ 4 levels "1"<"2"<"3"<"4": 1 1 1 1 4 1 1 1 2 1 ...
##  $ trip_distance  : num  0.1465 0.474 -0.5714 0.764 -0.0202 ...
##  $ fare_amount    : num  1.27 1.4 1.24 1.67 1.24 ...
##  $ tip_amount     : num  0.414 0 1.313 2.82 0 ...
##  $ duration       : num  1.95 2.59 2.22 4.15 2.04 ...
##  $ revenue        : num  1.39 1.37 1.38 1.36 1.38 ...
##  $ start_hour     : Factor w/ 24 levels "0","1","2","3",..: 15 24 10 15 19 11 18 24 10 21 ...
```

```
##  $ day_of_week    : chr  "Monday" "Wednesday" "Tuesday" "Wednesday" ...
##  $ weather        : Factor w/ 2 levels "preci","remain": 2 2 1 2 2 2 2 2 2 2 ...
```

```r
out = predict.lm(model, test.df, interval = "none")
out = data.frame("predict" = out)
write.csv(out, "data/test_output.csv", row.names = FALSE)
```

4 evaluate model

[1]:
```
#import os
#os.chdir("Applied Data Science\project2\code")
#os.getcwd()
```

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.metrics import r2_score, mean_absolute_error,mean_squared_error
from scipy.special import boxcox, inv_boxcox
from xgboost import XGBRegressor
from sklearn.preprocessing import OneHotEncoder
```

[3]:
```
true = pd.read_feather('data/test.feather')['revenue']
true
```

[3]:
```
0           77.658537
1           56.962025
2           68.051613
3           44.904652
4           63.636364
              …
4335200     74.925000
4335201    103.125000
4335202     55.851429
4335203     52.144330
4335204     58.417850
Name: revenue, Length: 4335205, dtype: float64
```

[4]:
```
pred_linear = pd.read_csv('data/test_output.csv')['predict']
```

[5]:
```
_lambda = pd.read_csv("data/lambda_list.csv")
_lambda
```

[5]:

	Unnamed: 0	trip_distance	fare_amount	tip_amount	duration	revenue
0	0	-0.173857	-0.443243	0.462488	0.079021	-0.681852

[6]:
```
pred_linear = inv_boxcox(pred_linear, -0.681852)
pred_linear.replace([np.inf, -np.inf], np.nan, inplace=True)
pred_linear.fillna(0, inplace=True)
pred_linear
```

[6]:
```
0           83.191800
1           69.235895
2           58.060569
3           40.650461
4           75.237809
              …
4335200     66.445531
4335201    106.667851
4335202     56.513942
4335203     52.225508
4335204     79.904453
Name: predict, Length: 4335205, dtype: float64
```

[7]:
```
scores = []
def store_score(scores, pred,true=true):
    curr = [r2_score(true, pred), mean_absolute_error(true, pred),
            mean_squared_error(true, pred)]
    scores.append(curr)
```

[8]:
```
store_score(scores, pred_linear)
```

[9]:
```
train = pd.read_feather('data/train_scaled.feather')
test = pd.read_feather('data/test_scaled.feather')
```

[10]:
```
def preprocessing(df):
    df['passenger_count'] = df['passenger_count'].astype('category')
    df['start_hour'] = df['start_hour'].astype('category')
    df['day_of_week'] = df['day_of_week'].astype('category')
    df['weather'] = df['weather'].astype('category')
    x = df.drop(['fare_amount','revenue'], axis=1)
    x = pd.get_dummies(x)
    y = df['revenue']

    return x, y
```

[11]:
```
train_x, train_y = preprocessing(train)
test_x, test_y = preprocessing(test)
```

[12]:
```
model = XGBRegressor()
model.fit(train_x, train_y)
pred_xgboost = model.predict(test_x)
```

```
pred_xgboost = inv_boxcox(pred_xgboost, -0.681852)
```

[13]:
```
pred_xgboost[np.isnan(pred_xgboost)]=0
store_score(scores, pred_xgboost)
```

[14]:
```
mean = [true.mean()] * len(true)
store_score(scores, mean)
```

[15]:
```
median = [true.median()] * len(true)
store_score(scores, median)
```

[16]:
```
scores
```

[16]:
```
[[0.6312128067940872, 7.646990861198339, 170.69201589626869],
 [0.9892818371805583, 1.577627790537623, 4.960868631176782],
 [0.0, 15.833505114833216, 462.8469183336379],
 [-0.05454166382931325, 15.27506086086051, 488.0913593578248]]
```

[ ]:
```

```

1 how the duration and trip distance change the revenue

[17]:
```
x=np.arange(1.35,1.45,0.01)
y= inv_boxcox(x, -0.681852)
y
```

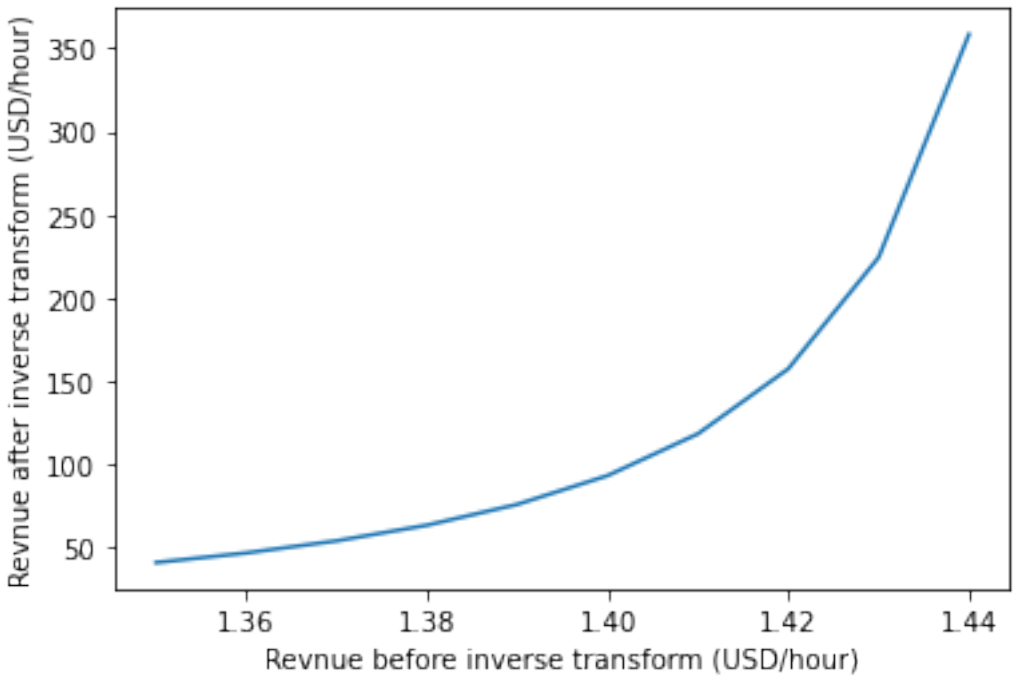
[17]:
```
array([ 40.99364064,  46.75528888,  54.02260299,  63.41376774,
        75.91767435,  93.20789982, 118.32848055, 157.37264446,
       224.28732237, 358.19011555])
```

[18]:
```
distance = inv_boxcox(x+0.07672, -0.681852) - inv_boxcox(x, -0.681852)
distance
```

[18]:
```
array([ 156.76256084,  255.26502818,  495.05783271, 1468.30202033,
                  nan,           nan,           nan,           nan,
                  nan,           nan])
```

[19]:
```
import matplotlib.pyplot as plt
plt.xlabel('Revnue before inverse transform (USD/hour)')
plt.ylabel("Revnue after inverse transform (USD/hour)")
plt.plot(x,y)
```

[19]:
```
[<matplotlib.lines.Line2D at 0x166834f7fa0>]
```


Revnue after inverse transform (USD/hour)
350
300
250
200
150
100
50
1.36
1.38
1.40
1.42
1.44
Revnue before inverse transform (USD/hour)

[ ]: