1. Цель лабораторной работы

Изучить способы предварительной обработки данных для дальнейшего формирования моделей.

2. Задание:

Выбрать набор данных (датасет), содержащий категориальные и числовые признаки и пропуски в данных. Для выполнения следующих пунктов можно использовать несколько различных наборов данных (один для обработки пропусков, другой для категориальных признаков и т.д.) Просьба не использовать датасет, на котором данная задача решалась в лекции. Для выбранного датасета (датасетов) на основе материалов лекций решить следующие задачи: устранение пропусков в данных; кодирование категориальных признаков; нормализацию числовых признаков.

3.процесс выполнения работы

#Импортирование необходимых библиотек

In [16]:

```
import numpy as np
import pandas as pd
import seaborn as sns
import sklearn.impute
import sklearn.preprocessing
import matplotlib.pyplot as plt
%matplotlib inline
sns.set(style="ticks")
```

In [17]:

```
pd.set_option("display.width",70)
```

In [18]:

```
data = pd.read_csv(r"C:\Users\asus\Desktop\iu5\MMO\lab2\LA_Metro_BikeSharing_CLEANED_2016quater3-202
```

```
C:\ProgramData\Anaconda3\lib\site-packages\IPython\core\interactiveshell.py:3444: Dt
ypeWarning: Columns (4) have mixed types.Specify dtype option on import or set low_m
emory=False.
  exec(code_obj, self.user_global_ns, self.user_ns)
```

In [19]:

```
data.shape
```

Out[19]:

```
(1250835, 16)
```

In [20]:

```
data.head()
```

Out[20]:

	trip_id	duration	start_time	end_time	bike_id	trip_route_category	plan_duration	passh
0	1912818	3	07/07/2016 4:17	07/07/2016 4:20	6281	Round Trip	30	Mc
1	1919661	33	07/07/2016 6:00	07/07/2016 6:33	6281	Round Trip	30	Mc
2	1933383	5	07/07/2016 10:32	07/07/2016 10:37	5861	Round Trip	365	A
3	1940317	7	07/07/2016 12:51	07/07/2016 12:58	6674	Round Trip	0	
4	1943980	9	07/07/2016 13:50	07/07/2016 13:59	6108	One Way	30	Mc

In [21]:

```
data.dtypes
```

Out[21]:

```
trip_id                int64
duration               int64
start_time            object
end_time              object
bike_id               object
trip_route_category   object
plan_duration          int64
passholder_type       object
bike_type             object
start_station          int64
end_station            int64
start_lat            float64
start_lon            float64
end_lat              float64
end_lon              float64
taxicab_distance     float64
dtype: object
```

3.1. Обработка пропусков в данных

In [22]:

```
data_features = list(zip(
# признаки
[i for i in data.columns],
zip(
    # типы колонок
    [str(i) for i in data.dtypes],
    # проверим есть ли пропущенные значения
    [i for i in data.isnull().sum()]
)))
# Признаки с типом данных и количеством пропусков
data_features
```

Out[22]:

```
[('trip_id', ('int64', 0)),
 ('duration', ('int64', 0)),
 ('start_time', ('object', 0)),
 ('end_time', ('object', 0)),
 ('bike_id', ('object', 0)),
 ('trip_route_category', ('object', 0)),
 ('plan_duration', ('int64', 0)),
 ('passholder_type', ('object', 4585)),
 ('bike_type', ('object', 0)),
 ('start_station', ('int64', 0)),
 ('end_station', ('int64', 0)),
 ('start_lat', ('float64', 0)),
 ('start_lon', ('float64', 0)),
 ('end_lat', ('float64', 0)),
 ('end_lon', ('float64', 0)),
 ('taxicab_distance', ('float64', 0))]
```

Устранение пропусков

In [23]:

```
# Доля (процент) пропусков
[(c, data[c].isnull().mean()) for c in data.columns]
```

Out[23]:

```
[('trip_id', 0.0),
 ('duration', 0.0),
 ('start_time', 0.0),
 ('end_time', 0.0),
 ('bike_id', 0.0),
 ('trip_route_category', 0.0),
 ('plan_duration', 0.0),
 ('passholder_type', 0.003665551411657013),
 ('bike_type', 0.0),
 ('start_station', 0.0),
 ('end_station', 0.0),
 ('start_lat', 0.0),
 ('start_lon', 0.0),
 ('end_lat', 0.0),
 ('end_lon', 0.0),
 ('taxicab_distance', 0.0)]
```

In [24]:

```
# Удаление колонок, содержащих пустые значения
data.dropna(axis=1, how='any')
```

Out[24]:

	trip_id	duration	start_time	end_time	bike_id	trip_route_category	plan_duratio
0	1912818	3	07/07/2016 4:17	07/07/2016 4:20	6281	Round Trip	3
1	1919661	33	07/07/2016 6:00	07/07/2016 6:33	6281	Round Trip	3
2	1933383	5	07/07/2016 10:32	07/07/2016 10:37	5861	Round Trip	36
3	1940317	7	07/07/2016 12:51	07/07/2016 12:58	6674	Round Trip	
4	1943980	9	07/07/2016 13:50	07/07/2016 13:59	6108	One Way	3
...	...	...	...	...	...	...	.
1250830	172219113	7	9/30/2021 23:41	9/30/2021 23:48	19544	One Way	
1250831	172219114	13	9/30/2021 23:34	9/30/2021 23:47	19819	One Way	3
1250832	172219218	25	9/30/2021 23:26	9/30/2021 23:51	17317	One Way	3
1250833	172219313	7	9/30/2021 23:50	9/30/2021 23:57	19998	One Way	
1250834	172222121	57	9/30/2021 23:24	10/01/2021 0:21	5940	One Way	3

1250835 rows × 15 columns

In [25]:

```
# Удаление колонок с высоким процентом пропусков (боле
data.dropna(axis=1, thresh=730)
```

Out[25]:

	trip_id	duration	start_time	end_time	bike_id	trip_route_category	plan_duratio
0	1912818	3	07/07/2016 4:17	07/07/2016 4:20	6281	Round Trip	3
1	1919661	33	07/07/2016 6:00	07/07/2016 6:33	6281	Round Trip	3
2	1933383	5	07/07/2016 10:32	07/07/2016 10:37	5861	Round Trip	36
3	1940317	7	07/07/2016 12:51	07/07/2016 12:58	6674	Round Trip	
4	1943980	9	07/07/2016 13:50	07/07/2016 13:59	6108	One Way	3
...	...	...	...	...	...	...	.
1250830	172219113	7	9/30/2021 23:41	9/30/2021 23:48	19544	One Way	
1250831	172219114	13	9/30/2021 23:34	9/30/2021 23:47	19819	One Way	3
1250832	172219218	25	9/30/2021 23:26	9/30/2021 23:51	17317	One Way	3
1250833	172219313	7	9/30/2021 23:50	9/30/2021 23:57	19998	One Way	
1250834	172222121	57	9/30/2021 23:24	10/01/2021 0:21	5940	One Way	3

1250835 rows × 16 columns

In [26]:

```
data.isnull().sum()
```

Out[26]:

```
trip_id                  0
duration                 0
start_time               0
end_time                 0
bike_id                  0
trip_route_category      0
plan_duration            0
passholder_type       4585
bike_type                0
start_station            0
end_station              0
start_lat                0
start_lon                0
end_lat                  0
end_lon                  0
taxicab_distance         0
dtype: int64
```

In [27]:

```
cols_with_na_temp = ['passholder_type']
```

In [28]:

```
data_drop = data[cols_with_na_temp].dropna()
data_drop.shape
```

Out[28]:

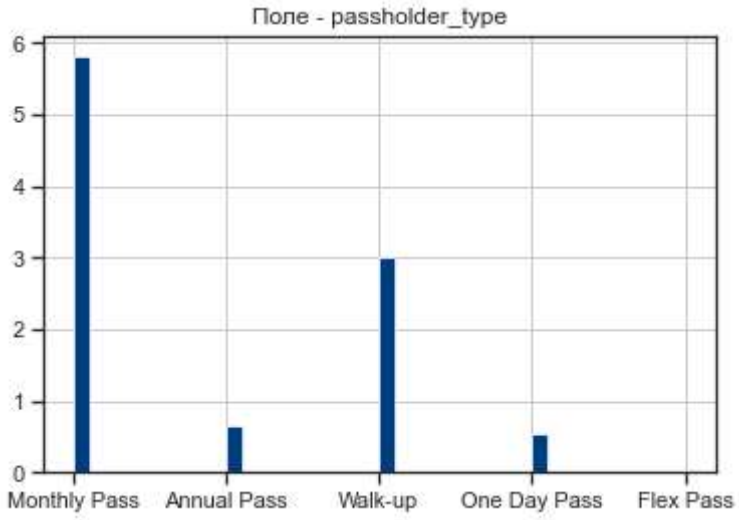
```
(1246250, 1)
```

In [29]:

```
def plot_hist_diff(old_ds, new_ds, cols):
    for c in cols:
        fig = plt.figure()
        ax = fig.add_subplot(111)
        ax.title.set_text('П о л е - ' + str(c))
        old_ds[c].hist(bins=40, ax=ax, density=True, color='green')
        new_ds[c].hist(bins=40, ax=ax, density=True, color='blue', alpha=0.5)
        plt.show()
```

In [30]:

```
plot_hist_diff(data, data_drop, cols_with_na_temp)
```



3.2. Кодирование категориальных признаков

In [31]:

```
trip=data["trip_route_category"].dropna().astype(str)
trip.value_counts()
```

Out[31]:

```
One Way        999145
Round Trip     194808
Monthly Pa      26402
Walk-up         18762
One Day Pa       7192
Annual Pas       4526
Name: trip_route_category, dtype: int64
```

In [32]:

```
le=sklearn.preprocessing.LabelEncoder()
type_le = le.fit_transform(trip)
print(np.unique(type_le))
le.inverse_transform(np.unique(type_le))
```

```
[0 1 2 3 4 5]
```

Out[32]:

```
array(['Annual Pas', 'Monthly Pa', 'One Day Pa', 'One Way', 'Round Trip',
       'Walk-up'], dtype=object)
```

In [33]:

```
trip_one_hot=pd.get_dummies(trip)
trip_one_hot.head()
```

Out[33]:

	Annual Pas	Monthly Pa	One Day Pa	One Way	Round Trip	Walk-up
0	0	0	0	0	1	0
1	0	0	0	0	1	0
2	0	0	0	0	1	0
3	0	0	0	0	1	0
4	0	0	0	1	0	0

In [34]:

```
trip_one_hot[trip_one_hot["One Way"]==1].head()
```

Out[34]:

	Annual Pas	Monthly Pa	One Day Pa	One Way	Round Trip	Walk-up
4	0	0	0	1	0	0
6	0	0	0	1	0	0
8	0	0	0	1	0	0
10	0	0	0	1	0	0
11	0	0	0	1	0	0

3.3. Нормализацию числовых признаков.

In [35]:

```
import scipy.stats as stats
```

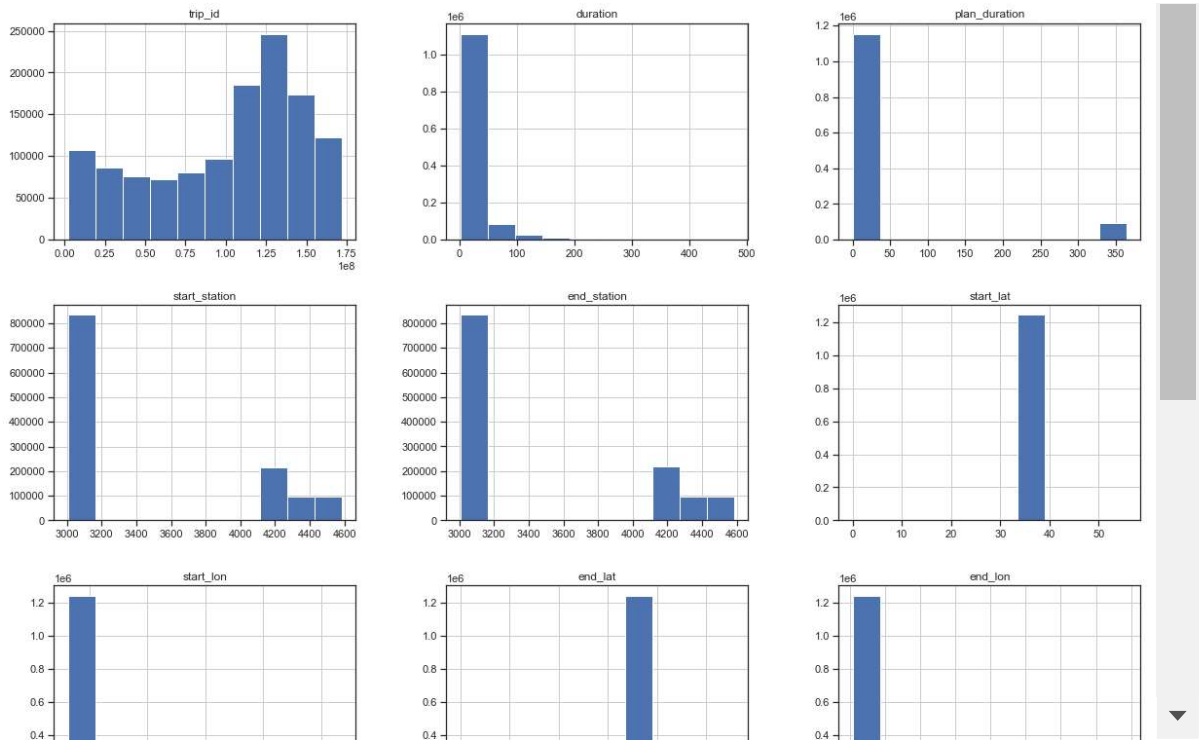
In [36]:

```
def diagnostic_plots(df, variable):
    plt.figure(figsize=(15,6))
    plt.subplot(1, 2, 1)
    df[variable].hist(bins=30)
    plt.subplot(1, 2, 2)
    stats.probplot(df[variable], dist="norm", plot = plt)
    plt.show()
```

In [37]:

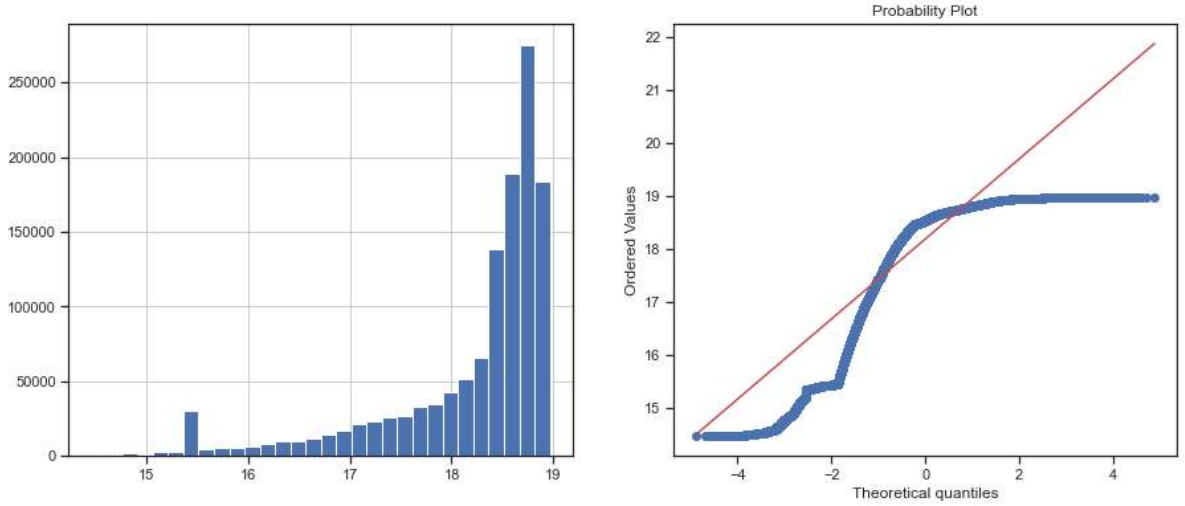
```
data_clean = data.dropna()
data_clean.info()
data_clean.hist(figsize=(20,20))
plt.show()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1246250 entries, 0 to 1250834
Data columns (total 16 columns):
 #   Column               Non-Null Count    Dtype  
---  ------               --------------    -----  
 0   trip_id              1246250 non-null  int64  
 1   duration             1246250 non-null  int64  
 2   start_time           1246250 non-null  object 
 3   end_time             1246250 non-null  object 
 4   bike_id              1246250 non-null  object 
 5   trip_route_category  1246250 non-null  object 
 6   plan_duration        1246250 non-null  int64  
 7   passholder_type      1246250 non-null  object 
 8   bike_type            1246250 non-null  object 
 9   start_station        1246250 non-null  int64  
 10  end_station          1246250 non-null  int64  
 11  start_lat            1246250 non-null  float64
 12  start_lon            1246250 non-null  float64
 13  end_lat              1246250 non-null  float64
 14  end_lon              1246250 non-null  float64
 15  taxicab_distance     1246250 non-null  float64
dtypes: float64(5), int64(5), object(6)
memory usage: 161.6+ MB
```



In [46]:

```
diagnostic_plots(data_clean, 'trip_id')
```

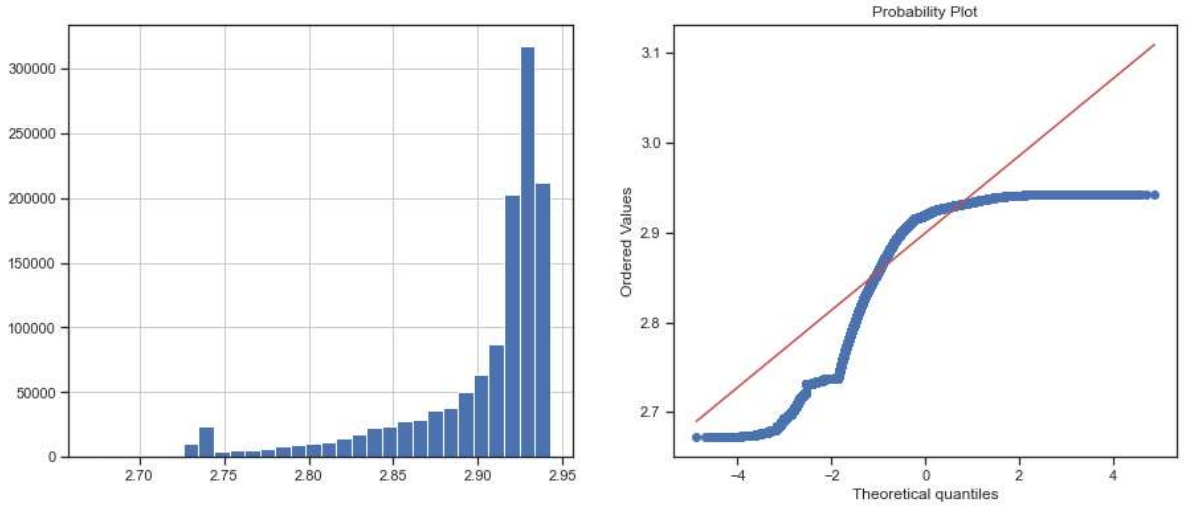


In [40]:

```
data_clean['trip_id_log'] = np.log(data_clean['trip_id'])
diagnostic_plots(data_clean, 'trip_id_log')
```

```
C:\Users\asus\AppData\Local\Temp/ipykernel_2344/1555149951.py:1: SettingWithCopyWarn
ing:
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/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data_clean['trip_id_log'] = np.log(data_clean['trip_id'])
```

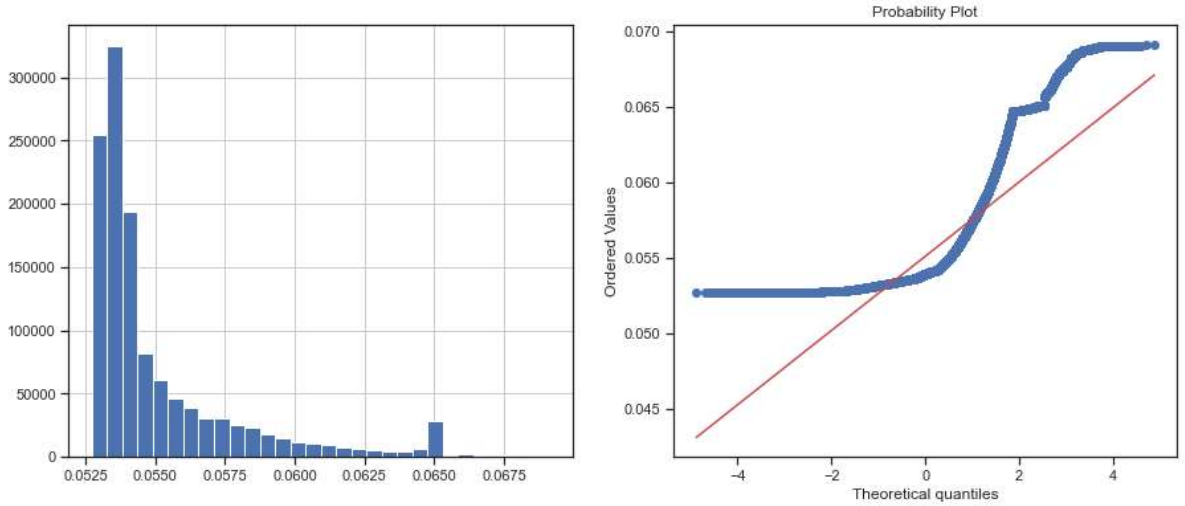


In [42]:

```
data_clean['trip_id_reciprocal'] = 1 / (data_clean['trip_id'])
diagnostic_plots(data_clean, 'trip_id_reciprocal')
```

```
C:\Users\asus\AppData\Local\Temp/ipykernel_2344/3592647394.py:1: SettingWithCopyWarn
ing:
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/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data_clean['trip_id_reciprocal'] = 1 / (data_clean['trip_id'])
```

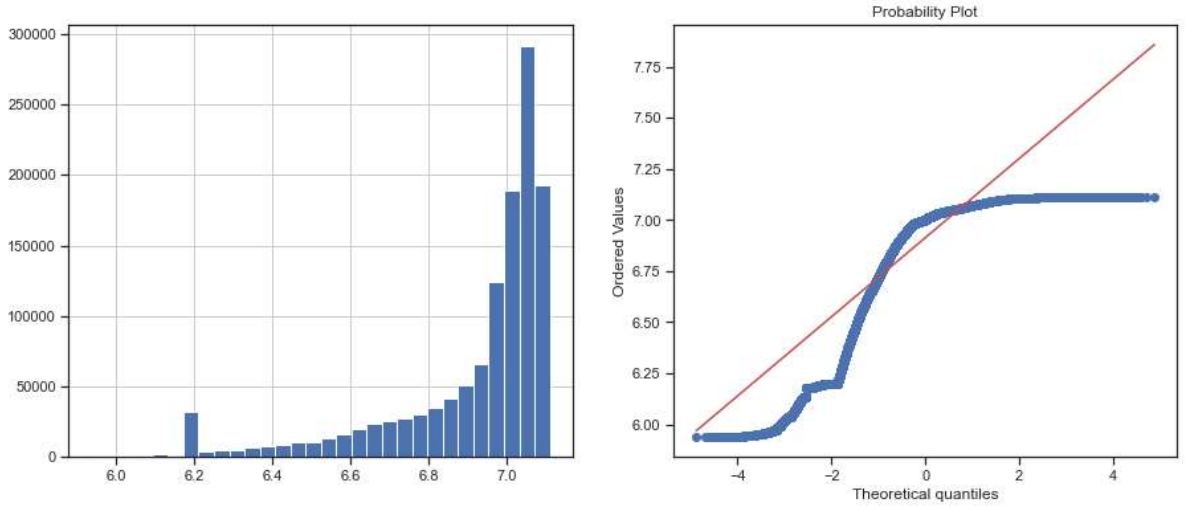


In [43]:

```
data_clean['trip_id_exp1']=data_clean['trip_id']**(1/1.5)
diagnostic_plots(data_clean, 'trip_id_exp1')
```

```
C:\Users\asus\AppData\Local\Temp/ipykernel_2344/3759703456.py:1: SettingWithCopyWarn
ing:
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/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data_clean['trip_id_exp1']=data_clean['trip_id']**(1/1.5)
```

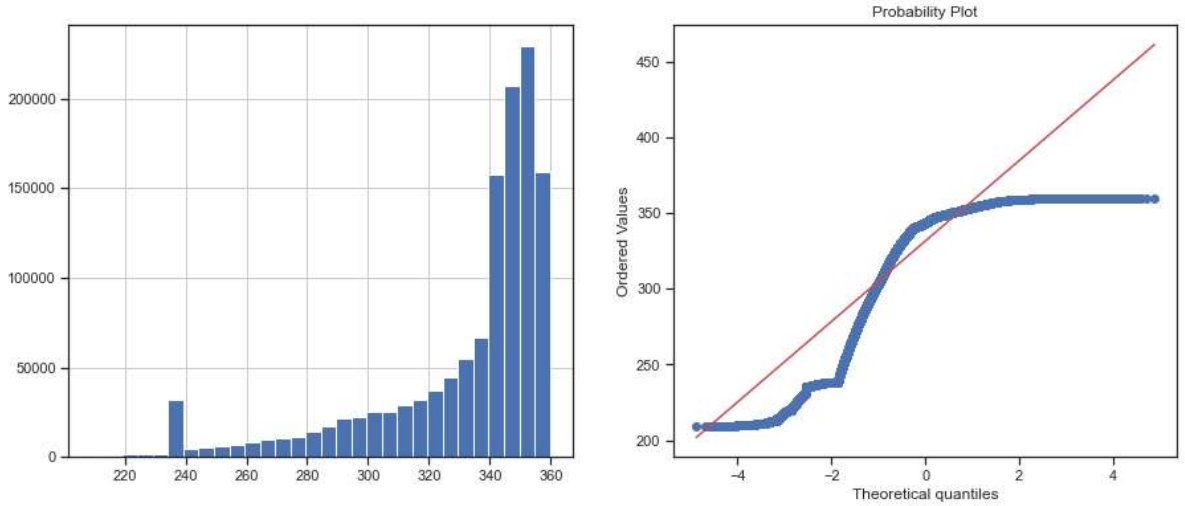


In [44]:

```
data_clean['trip_id_exp2'] = data_clean['trip_id']**(2)
diagnostic_plots(data_clean, 'trip_id_exp2')
```

```
C:\Users\asus\AppData\Local\Temp/ipykernel_2344/530966480.py:1: SettingWithCopyWarni
ng:
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/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data_clean['trip_id_exp2'] = data_clean['trip_id']**(2)
```

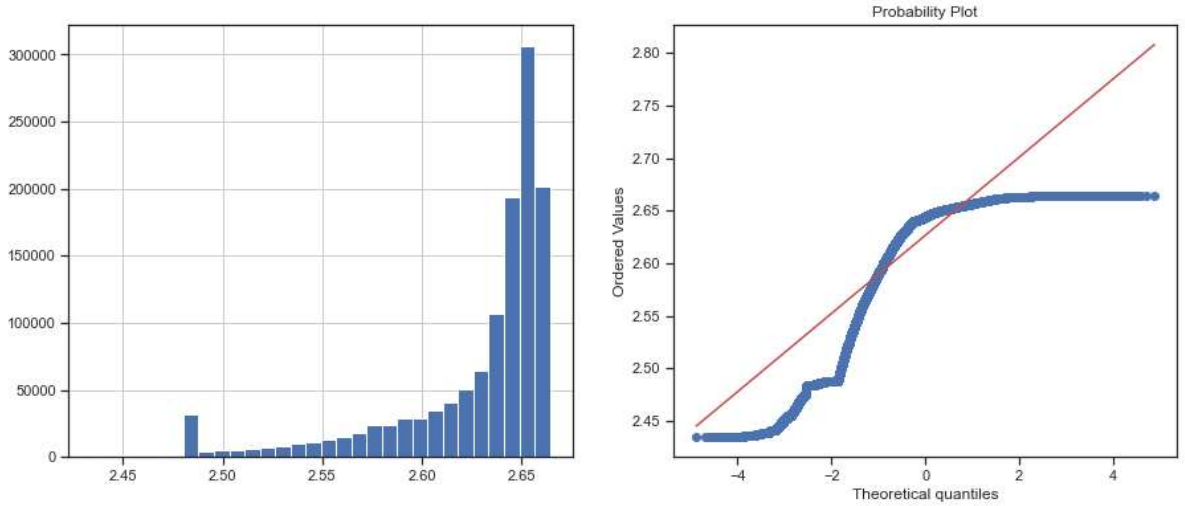


In [47]:

```
data_clean['trip_id_exp3'] = data_clean['trip_id']**(0.333)
diagnostic_plots(data_clean, 'trip_id_exp3')
```

```
C:\Users\asus\AppData\Local\Temp/ipykernel_2344/3811952672.py:1: SettingWithCopyWarn
ing:
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/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data_clean['trip_id_exp3'] = data_clean['trip_id']**(0.333)
```



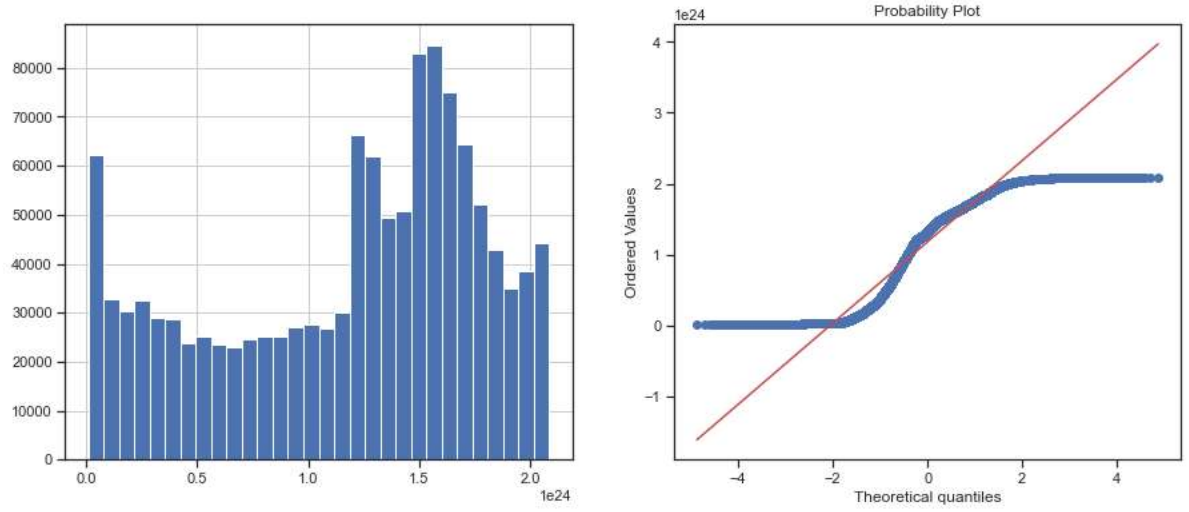
In [48]:

```
data_clean['trip_id_boxcox'], param = stats.boxcox(data_clean['trip_id'])
print('Оптимальное значение λ = {}'.format(param))
diagnostic_plots(data_clean, 'trip_id_boxcox')
```

```
C:\Users\asus\AppData\Local\Temp/ipykernel_2344/4007994848.py:1: SettingWithCopyWarn
ing:
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/u
ser_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pa
ndas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  data_clean['trip_id_boxcox'], param = stats.boxcox(data_clean['trip_id'])
```

```
Оптимальное значение λ = 20.049273258656413
```



In [ ]: