```
5     1993          2                1  ...  0.273561  0.443718            2.736662
10    1993          3                1  ...  0.233417  0.315005            2.485468
15    1993          4                1  ...  0.122836  0.169198            1.306590
20    1993          5                1  ...  0.715590  1.018217            7.184956
...    ...        ...              ...  ...       ...       ...                 ...
1549  2018          7                1  ...  1.019434  0.771468            6.264285
1555  2018          8                1  ...  1.196863  1.118564            7.303023
1561  2018          9                1  ...  2.182986  1.320855           14.588205
1567  2018         10                1  ...  1.187922  1.193949            8.393952
1573  2018         11                1  ...  0.114368  0.078262            0.719859

[286 rows x 12 columns]
C:\Users\Utente\Desktop\DataScience\code\util.py:30: 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
  categorySlice['SMA_5'] = window.mean()
C:\Users\Utente\Desktop\DataScience\code\util.py:31: 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
  categorySlice['min'] = window.min()
C:\Users\Utente\Desktop\DataScience\code\util.py:32: 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
  categorySlice['max'] = window.max()
C:\Users\Utente\Desktop\DataScience\code\util.py:33: 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
  categorySlice['std'] = window.std()


************************************************************
MODEL Random forest regression with synthetic features
features= Index(['min', 'max', 'std', 'SMA_5', 'year', 'total_link_length_miles',
       'road_category_id'],
      dtype='object')
score model_forest: 0.9972
regression_error: 1.2381
<Figure size 432x288 with 0 Axes>

In [29]: runfile('C:/Users/Utente/Desktop/DataScience/code/main.py', wdir='C:/Users/
Utente/Desktop/DataScience/code')
Reloaded modules: init, model, util, visualization, preprocessing
[autoreload of model failed: Traceback (most recent call last):
  File "C:\Users\Utente\Anaconda3\lib\site-packages\IPython\extensions\autoreload.py",
line 245, in check
    superreload(m, reload, self.old_objects)
  File "C:\Users\Utente\Anaconda3\lib\site-packages\IPython\extensions\autoreload.py",
line 394, in superreload
    module = reload(module)
  File "C:\Users\Utente\Anaconda3\lib\imp.py", line 314, in reload
    return importlib.reload(module)
  File "C:\Users\Utente\Anaconda3\lib\importlib\__init__.py", line 169, in reload
    _bootstrap._exec(spec, module)
  File "<frozen importlib._bootstrap>", line 630, in _exec
  File "<frozen importlib._bootstrap_external>", line 724, in exec_module
  File "<frozen importlib._bootstrap_external>", line 860, in get_code
  File "<frozen importlib._bootstrap_external>", line 791, in source_to_code
  File "<frozen importlib._bootstrap>", line 219, in _call_with_frames_removed
  File "C:\Users\Utente\Desktop\DataScience\code\model.py", line 166
    def model_linear(frame):
      ^
IndentationError: expected an indented block
]
  File "C:\Users\Utente\Desktop\DataScience\code\model.py", line 166
    def model_linear(frame):
      ^
IndentationError: expected an indented block


In [30]: runfile('C:/Users/Utente/Desktop/DataScience/code/main.py', wdir='C:/Users/
Utente/Desktop/DataScience/code')
Reloaded modules: init


************************************************************
UNIQUE DISTRIBUTION OF FEATURES
Total number of regions:  11
Total number of road categories:  6
              year    region_id  ...      lorries  all_motor_vehicles
count  1547.000000  1547.000000  ...  1547.000000         1547.000000
mean   2005.475760     6.212023  ...     0.283026            5.028188
std       7.517803     3.187790  ...     0.292345            3.616128
min    1993.000000     1.000000  ...     0.000648            0.010438
25%    1999.000000     3.000000  ...     0.067333            2.238600
50%    2005.000000     6.000000  ...     0.168881            4.270186
75%    2012.000000     9.000000  ...     0.414246            7.396709
max    2018.000000    11.000000  ...     1.386260           15.845616

[8 rows x 12 columns]


************************************************************
GROUP DISTRIBUTION ON TRAFFIC BY YEAR
    year  region_id  road_category_id  ...       vans    lorries  all_motor_vehicles
0   1993        373               219  ...  11.087950  10.849047           16.140504
1   1994        373               219  ...  11.553254  11.082147           16.501645
2   1995        373               219  ...  11.867358  11.382132           16.822704
3   1996        373               219  ...  12.317925  11.736691           17.270079
4   1997        373               219  ...  12.949367  12.014887           17.631146
5   1998        373               219  ...  13.548895  12.411285           17.951606
6   1999        373               219  ...  13.760718  12.586221           18.283915
7   2000        373               219  ...  13.925848  12.627528           18.252833
8   2001        367               216  ...  14.257194  12.544484           18.507065
9   2002        367               216  ...  14.585115  12.653575           18.941853
10  2003        367               216  ...  15.315447  12.717592           19.057797
11  2004        367               216  ...  16.064099  13.102573           19.343582
12  2005        368               218  ...  16.498701  12.951695           19.339939
13  2006        368               218  ...  17.158052  12.996429           19.622870
14  2007        367               216  ...  17.981100  13.093525           19.794497
15  2008        367               216  ...  17.865668  12.799325           19.603296
16  2009        367               216  ...  17.480556  11.743582           19.417265
17  2010        367               216  ...  17.630435  11.783215           19.107573
18  2011        367               216  ...  17.775096  11.466826           19.144793
19  2012        367               216  ...  17.732849  11.185769           19.072997
```

```
20  2013        367               216  ...  18.292121  11.284625           19.140180
21  2014        367               216  ...  19.337688  11.576409           19.640368
22  2015        367               216  ...  20.154004  12.003396           19.962072
23  2016        373               219  ...  21.142740  12.115465           20.354558
24  2017        373               219  ...  21.704466  12.258545           20.617241
25  2018        373               219  ...  21.908306  12.300529           20.681070

[26 rows x 12 columns]


************************************************************
TRAFFIC BY REGION IN 2018
                        name   year  ...  all_motor_vehicles  other_vehicles
0                 North East  12108  ...           12.274201        0.221726
1                      Wales  10090  ...           18.260918        0.309984
2                     London  10090  ...           18.354733        0.744356
3              East Midlands  10090  ...           27.647983        0.365226
4   Yorkshire and The Humber  12108  ...           27.760255        0.374354
5                   Scotland  10090  ...           29.716436        0.507561
6              West Midlands  12108  ...           31.554014        0.372814
7                 South West  10090  ...           33.238699        0.513533
8                 North West  12108  ...           35.697093        0.399917
9            East of England  10090  ...           38.701718        0.512950
10                South East  12108  ...           54.908644        0.713793

[11 rows x 14 columns]


************************************************************
VEHICLE TYPE COMPARISON
    year  region_id  ...  all_motor_vehicles  other_vehicles
0   1993        373  ...          256.214013        5.211234
25  2018        373  ...          328.114695        5.036214

[2 rows x 13 columns]
+----------------------------+-------------------------+
| Vehicle type               |   Billion vehicle miles |
|----------------------------+-------------------------|
| all_motor_vehicles         |                 328.115   |
| cars_and_taxis             |                 255.013   |
| vans                       |                  50.9832  |
| lorries                    |                  17.0826  |
| pedal_cycles               |                   3.32911 |
| two_wheeled_motor_vehicles |                   2.73903 |
| buses_and_coaches          |                   2.29719 |
+----------------------------+-------------------------+


************************************************************
Average traffic for road category 1
      year  region_id  road_category_id  ...      vans   lorries  all_motor_vehicles
0     1993          1                 1  ...  0.309954  0.428961            3.465840
5     1993          2                 1  ...  0.273561  0.443718            2.736662
10    1993          3                 1  ...  0.233417  0.315005            2.485468
15    1993          4                 1  ...  0.122836  0.169198            1.306590
20    1993          5                 1  ...  0.715590  1.018217            7.184956
...    ...        ...               ...  ...       ...       ...                 ...
1549  2018          7                 1  ...  1.019434  0.771468            6.264285
1555  2018          8                 1  ...  1.196863  1.118564            7.303023
1561  2018          9                 1  ...  2.182986  1.320855           14.588205
1567  2018         10                 1  ...  1.187922  1.193949            8.393952
1573  2018         11                 1  ...  0.114368  0.078262            0.719859
```

```
[286 rows x 12 columns]


************************************************************
MODEL Random forest regression with synthetic features
features=C:\Users\Utente\Desktop\DataScience\code\util.py:30: 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
  categorySlice['SMA_5'] = window.mean()
C:\Users\Utente\Desktop\DataScience\code\util.py:31: 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
  categorySlice['min'] = window.min()
C:\Users\Utente\Desktop\DataScience\code\util.py:32: 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
  categorySlice['max'] = window.max()
C:\Users\Utente\Desktop\DataScience\code\util.py:33: 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
  categorySlice['std'] = window.std()
 Index(['min', 'max', 'std', 'SMA_5', 'year', 'total_link_length_miles',
       'road_category_id'],
      dtype='object')
score model_forest: 0.9981
regression_error: 0.9478
<Figure size 432x288 with 0 Axes>

In [31]: runfile('C:/Users/Utente/Desktop/DataScience/code/main.py', wdir='C:/Users/
Utente/Desktop/DataScience/code')
Reloaded modules: init, model, util, visualization, preprocessing


************************************************************
UNIQUE DISTRIBUTION OF FEATURES
Total number of regions:  11
Total number of road categories:  6
              year    region_id  ...      lorries  all_motor_vehicles
count  1547.000000  1547.000000  ...  1547.000000         1547.000000
mean   2005.475760     6.212023  ...     0.283026            5.028188
std       7.517803     3.187790  ...     0.292345            3.616128
min    1993.000000     1.000000  ...     0.000648            0.010438
25%    1999.000000     3.000000  ...     0.067333            2.238600
50%    2005.000000     6.000000  ...     0.168881            4.270186
75%    2012.000000     9.000000  ...     0.414246            7.396709
max    2018.000000    11.000000  ...     1.386260           15.845616

[8 rows x 12 columns]


************************************************************
```

```
GROUP DISTRIBUTION ON TRAFFIC BY YEAR
    year  region_id  road_category_id  ...       vans    lorries  all_motor_vehicles
0   1993        373               219  ...  11.087950  10.849047           16.140504
1   1994        373               219  ...  11.553254  11.082147           16.501645
2   1995        373               219  ...  11.867358  11.382132           16.822704
3   1996        373               219  ...  12.317925  11.736691           17.270079
4   1997        373               219  ...  12.949367  12.014887           17.631146
5   1998        373               219  ...  13.548895  12.411285           17.951606
6   1999        373               219  ...  13.760718  12.586221           18.283915
7   2000        373               219  ...  13.925848  12.627528           18.252833
8   2001        367               216  ...  14.257194  12.544484           18.507065
9   2002        367               216  ...  14.585115  12.653575           18.941853
10  2003        367               216  ...  15.315447  12.717592           19.057797
11  2004        367               216  ...  16.064099  13.102573           19.343582
12  2005        368               218  ...  16.498701  12.951695           19.339939
13  2006        368               218  ...  17.158052  12.996429           19.622870
14  2007        367               216  ...  17.981100  13.093525           19.794497
15  2008        367               216  ...  17.865668  12.799325           19.603296
16  2009        367               216  ...  17.480556  11.743582           19.417265
17  2010        367               216  ...  17.630435  11.783215           19.107573
18  2011        367               216  ...  17.775096  11.466826           19.144793
19  2012        367               216  ...  17.732849  11.185769           19.072997
20  2013        367               216  ...  18.292121  11.284625           19.140180
21  2014        367               216  ...  19.337688  11.576409           19.640368
22  2015        367               216  ...  20.154004  12.003396           19.962072
23  2016        373               219  ...  21.142740  12.115465           20.354558
24  2017        373               219  ...  21.704466  12.258545           20.617241
25  2018        373               219  ...  21.908306  12.300529           20.681070

[26 rows x 12 columns]


************************************************************
TRAFFIC BY REGION IN 2018
                        name   year  ...  all_motor_vehicles  other_vehicles
0                 North East  12108  ...           12.274201        0.221726
1                      Wales  10090  ...           18.260918        0.309984
2                     London  10090  ...           18.354733        0.744356
3              East Midlands  10090  ...           27.647983        0.365226
4   Yorkshire and The Humber  12108  ...           27.760255        0.374354
5                   Scotland  10090  ...           29.716436        0.507561
6              West Midlands  12108  ...           31.554014        0.372814
7                 South West  10090  ...           33.238699        0.513533
8                 North West  12108  ...           35.697093        0.399917
9            East of England  10090  ...           38.701718        0.512950
10                South East  12108  ...           54.908644        0.713793

[11 rows x 14 columns]


************************************************************
VEHICLE TYPE COMPARISON
    year  region_id  ...  all_motor_vehicles  other_vehicles
0   1993        373  ...          256.214013        5.211234
25  2018        373  ...          328.114695        5.036214

[2 rows x 13 columns]
+---------------------------+------------------------+
| Vehicle type              |   Billion vehicle miles |
|---------------------------+------------------------|
| all_motor_vehicles        |                328.115  |
| cars_and_taxis            |                255.013  |
| vans                      |                 50.9832 |
| lorries                     |                 17.0826  |
| pedal_cycles                |                  3.32911 |
| two_wheeled_motor_vehicles  |                  2.73903 |
| buses_and_coaches           |                  2.29719 |
+----------------------------+-------------------------+
C:\Users\Utente\Anaconda3\lib\site-packages\pandas\core\indexing.py:671:
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
  self._setitem_with_indexer(indexer, value)
Traceback (most recent call last):

  File "C:\Users\Utente\Anaconda3\lib\site-packages\pandas\core\indexes\base.py", line
2646, in get_loc
    return self._engine.get_loc(key)

  File "pandas\_libs\index.pyx", line 111, in pandas._libs.index.IndexEngine.get_loc

  File "pandas\_libs\index.pyx", line 138, in pandas._libs.index.IndexEngine.get_loc

  File "pandas\_libs\hashtable_class_helper.pxi", line 1619, in
pandas._libs.hashtable.PyObjectHashTable.get_item

  File "pandas\_libs\hashtable_class_helper.pxi", line 1627, in
pandas._libs.hashtable.PyObjectHashTable.get_item

KeyError: 'SMA_5'


During handling of the above exception, another exception occurred:

Traceback (most recent call last):

  File "C:\Users\Utente\Desktop\DataScience\code\main.py", line 62, in <module>
    frameSynthetic= add_synthetic_features(frame)

  File "C:\Users\Utente\Desktop\DataScience\code\util.py", line 38, in
add_synthetic_features
    newFrame['SMA_5'].fillna( method ='bfill', inplace = True)

  File "C:\Users\Utente\Anaconda3\lib\site-packages\pandas\core\frame.py", line 2800, in
__getitem__
    indexer = self.columns.get_loc(key)

  File "C:\Users\Utente\Anaconda3\lib\site-packages\pandas\core\indexes\base.py", line
2648, in get_loc
    return self._engine.get_loc(self._maybe_cast_indexer(key))

  File "pandas\_libs\index.pyx", line 111, in pandas._libs.index.IndexEngine.get_loc

  File "pandas\_libs\index.pyx", line 138, in pandas._libs.index.IndexEngine.get_loc

  File "pandas\_libs\hashtable_class_helper.pxi", line 1619, in
pandas._libs.hashtable.PyObjectHashTable.get_item

  File "pandas\_libs\hashtable_class_helper.pxi", line 1627, in
pandas._libs.hashtable.PyObjectHashTable.get_item

KeyError: 'SMA_5'

<Figure size 432x288 with 0 Axes>
```

```
In [32]: runfile('C:/Users/Utente/Desktop/DataScience/code/main.py', wdir='C:/Users/
Utente/Desktop/DataScience/code')
Reloaded modules: init, model, util, visualization, preprocessing


************************************************************
UNIQUE DISTRIBUTION OF FEATURES
Total number of regions:  11
Total number of road categories:  6
              year    region_id  ...      lorries  all_motor_vehicles
count  1547.000000  1547.000000  ...  1547.000000         1547.000000
mean   2005.475760     6.212023  ...     0.283026            5.028188
std       7.517803     3.187790  ...     0.292345            3.616128
min    1993.000000     1.000000  ...     0.000648            0.010438
25%    1999.000000     3.000000  ...     0.067333            2.238600
50%    2005.000000     6.000000  ...     0.168881            4.270186
75%    2012.000000     9.000000  ...     0.414246            7.396709
max    2018.000000    11.000000  ...     1.386260           15.845616

[8 rows x 12 columns]


************************************************************
GROUP DISTRIBUTION ON TRAFFIC BY YEAR
    year  region_id  road_category_id  ...       vans    lorries  all_motor_vehicles
0   1993        373               219  ...  11.087950  10.849047           16.140504
1   1994        373               219  ...  11.553254  11.082147           16.501645
2   1995        373               219  ...  11.867358  11.382132           16.822704
3   1996        373               219  ...  12.317925  11.736691           17.270079
4   1997        373               219  ...  12.949367  12.014887           17.631146
5   1998        373               219  ...  13.548895  12.411285           17.951606
6   1999        373               219  ...  13.760718  12.586221           18.283915
7   2000        373               219  ...  13.925848  12.627528           18.252833
8   2001        367               216  ...  14.257194  12.544484           18.507065
9   2002        367               216  ...  14.585115  12.653575           18.941853
10  2003        367               216  ...  15.315447  12.717592           19.057797
11  2004        367               216  ...  16.064099  13.102573           19.343582
12  2005        368               218  ...  16.498701  12.951695           19.339939
13  2006        368               218  ...  17.158052  12.996429           19.622870
14  2007        367               216  ...  17.981100  13.093525           19.794497
15  2008        367               216  ...  17.865668  12.799325           19.603296
16  2009        367               216  ...  17.480556  11.743582           19.417265
17  2010        367               216  ...  17.630435  11.783215           19.107573
18  2011        367               216  ...  17.775096  11.466826           19.144793
19  2012        367               216  ...  17.732849  11.185769           19.072997
20  2013        367               216  ...  18.292121  11.284625           19.140180
21  2014        367               216  ...  19.337688  11.576409           19.640368
22  2015        367               216  ...  20.154004  12.003396           19.962072
23  2016        373               219  ...  21.142740  12.115465           20.354558
24  2017        373               219  ...  21.704466  12.258545           20.617241
25  2018        373               219  ...  21.908306  12.300529           20.681070

[26 rows x 12 columns]


************************************************************
TRAFFIC BY REGION IN 2018
                      name   year  ...  all_motor_vehicles  other_vehicles
0               North East  12108  ...           12.274201        0.221726
1                    Wales  10090  ...           18.260918        0.309984
2                   London  10090  ...           18.354733        0.744356
3            East Midlands  10090  ...           27.647983        0.365226
```

```
4   Yorkshire and The Humber  12108  ...          27.760255        0.374354
5                   Scotland  10090  ...          29.716436        0.507561
6              West Midlands  12108  ...          31.554014        0.372814
7                 South West  10090  ...          33.238699        0.513533
8                 North West  12108  ...          35.697093        0.399917
9            East of England  10090  ...          38.701718        0.512950
10                South East  12108  ...          54.908644        0.713793

[11 rows x 14 columns]


************************************************************
VEHICLE TYPE COMPARISON
    year  region_id  ...  all_motor_vehicles  other_vehicles
0   1993        373  ...          256.214013        5.211234
25  2018        373  ...          328.114695        5.036214

[2 rows x 13 columns]
+----------------------------+-------------------------+
| Vehicle type               |   Billion vehicle miles |
|----------------------------+-------------------------|
| all_motor_vehicles         |               328.115   |
| cars_and_taxis             |               255.013   |
| vans                       |                50.9832  |
| lorries                    |                17.0826  |
| pedal_cycles               |                 3.32911 |
| two_wheeled_motor_vehicles |                 2.73903 |
| buses_and_coaches          |                 2.29719 |
+----------------------------+-------------------------+


************************************************************
Average traffic for road category 1
C:\Users\Utente\Desktop\DataScience\code\util.py:30: 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
  categorySlice.loc['SMA_5'] = window.mean()
C:\Users\Utente\Desktop\DataScience\code\util.py:31: 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
  categorySlice.loc['min'] = window.min()
C:\Users\Utente\Desktop\DataScience\code\util.py:32: 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
  categorySlice.loc['max'] = window.max()
C:\Users\Utente\Desktop\DataScience\code\util.py:33: 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
  categorySlice.loc['std'] = window.std()
      year  region_id  road_category_id  ...      vans   lorries  all_motor_vehicles
0     1993          1                 1  ...  0.309954  0.428961            3.465840
```

```
5     1993          2                 1  ...  0.273561  0.443718            2.736662
10    1993          3                 1  ...  0.233417  0.315005            2.485468
15    1993          4                 1  ...  0.122836  0.169198            1.306590
20    1993          5                 1  ...  0.715590  1.018217            7.184956
...    ...        ...               ...  ...       ...       ...                 ...
1549  2018          7                 1  ...  1.019434  0.771468            6.264285
1555  2018          8                 1  ...  1.196863  1.118564            7.303023
1561  2018          9                 1  ...  2.182986  1.320855           14.588205
1567  2018         10                 1  ...  1.187922  1.193949            8.393952
1573  2018         11                 1  ...  0.114368  0.078262            0.719859

[286 rows x 12 columns]


************************************************************
MODEL Random forest regression with synthetic features
features= Index(['min', 'max', 'std', 'SMA_5', 'year', 'total_link_length_miles',
       'road_category_id'],
      dtype='object')
score model_forest: 0.9972
regression_error: 1.2390
<Figure size 432x288 with 0 Axes>
```

In [**33**]: