```
***********************
VEHICLE TYPE COMPARISON
   year region_id ... all_motor_vehicles other_vehicles
             373 ...
   1993
                             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
                                        3.32911
pedal cycles
                                        2.73903
| two wheeled motor vehicles |
                                        2.29719 İ
buses and coaches
+----+
***********************
MODEL Linear regression 2
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
score linear2: 0.0436
regression error: 2.7663
******************
MODEL SV Regression
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
score: 0.6936
regression_error: 1.2666
Execution time: 2.516101360321045
*******************
MODEL Random forest regression hyp
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
{'max_features': 'auto', 'n_estimators': 100, 'n_jobs': -1}
score model_forest_hyp: 0.9968
regression_error: 0.1091
Execution time: 9.04289722442627
C:\Users\Utente\Desktop\DataScience\code\util.py:28: 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:29: 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: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['max'] = window.max()

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['std'] = window.std()
<Figure size 432x288 with 0 Axes>

In [17]: 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 OF 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

```
******************
TRAFFIC BY REGION IN 2018
                name year ... all_motor_vehicles other_vehicles North East 12108 ... 12.274201 0.221726
             Wales 10090 ...
London 10090 ...
East Midlands 10090 ...
1
                                               18.260918
                                                               0.309984
2
                                               18.354733
                                                               0.744356
3
                                               27.647983
                                                               0.365226
   Yorkshire and The Humber 12108 ...
Scotland 10090 ...
West Midlands 12108 ...
South West 10090 ...
North West 12108 ...
East of England 10090 ...
4
                                               27.760255
                                                               0.374354
5
                                               29.716436
                                                               0.507561
6
                                               31.554014
                                                               0.372814
                                             33.238699
7
                                                               0.513533
8
                                             35.697093
                                                              0.399917
9
                                              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
   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
                                         255.013
cars_and_taxis
                                          50.9832
| vans
| lorries
                                          17.0826
                                          3.32911
| pedal_cycles
| two_wheeled_motor_vehicles |
                                           2.73903
buses_and_coaches
                                           2.29719
******************
MODEL Linear regression 2
features= Index(['year', 'region_id', 'road_category_id', 'total_link_length_km',
      'total_link_length_miles'],
     dtype='object')
score linear2: 0.1797
regression error: 2.4088
******************
MODEL SV Regression
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
score: 0.7081
regression error: 1.1189
Execution time: 2.6000030040740967
************************
MODEL Random forest regression hyp
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
{'max_features': 'auto', 'n_estimators': 100, 'n_jobs': -1}
```

```
regression_error: 0.0977
Execution time: 9.050088167190552
C:\Users\Utente\Desktop\DataScience\code\util.py:28: 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:29: 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: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['max'] = window.max()
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['std'] = window.std()
<Figure size 432x288 with 0 Axes>
In [18]: 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
                                       lorries all_motor_vehicles
            year
                    region_id ...
count 1547.000000 1547.000000 ... 1547.000000 1547.000000
                    6.212023 ... 0.283026
      2005.475760
                                                         5.028188
mean
                     3.187790 ...
                                    0.292345
         7.517803
                                                         3.616128
std
                    1.000000 ... 0.000648
                                                         0.010438
      1993.000000
min
      1999.000000
                    3.000000 ... 0.067333
25%
                                                         2.238600
50%
      2005.000000
                    6.000000 ... 0.168881
                                                        4.270186
75%
      2012.000000
                    9.000000 ... 0.414246
                                                         7.396709
      2018.000000 11.000000 ...
                                     1.386260
                                                        15.845616
max
[8 rows x 12 columns]
**********************
GROUP DISTRIBUTION OF TRAFFIC BY YEAR
   year region_id road_category_id ...
                                                     lorries all motor vehicles
                                              vans
                                219 ... 11.087950 10.849047
   1993
              373
                                                                      16.140504
   1994
1
              373
                                219 ... 11.553254 11.082147
                                                                      16.501645
                                219 ... 11.867358 11.382132
2
   1995
              373
                                                                      16.822704
                                219 ... 12.317925 11.736691
3
   1996
              373
                                                                      17.270079
   1997
                                219 ... 12.949367 12.014887
               373
                                                                      17.631146
```

score model forest hyp: 0.9978

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]

1	
Vehicle type	Billion vehicle miles
all_motor_vehicles   cars_and_taxis   vans   lorries   pedal_cycles   two_wheeled_motor_vehicles   buses_and_coaches	328.115 255.013 50.9832 17.0826 3.32911 2.73903 2.29719
<b>1</b>	L

```
*************************
MODEL Linear regression 2
features= Index(['year', 'region_id', 'road_category_id', 'total_link_length_km',
       'total_link_length_miles'],
     dtype='object')
score linear2: 0.1961
regression error: 2.5539
**********************
MODEL SV Regression
features= Index(['year', 'region_id', 'road_category_id', 'total_link_length_km',
       'total_link_length_miles'],
     dtype='object')
score: 0.6992
regression error: 1.1685
Execution time: 3.1753594875335693
******************
MODEL Random forest regression hyp
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
{'max_features': 'auto', 'n_estimators': 100, 'n_jobs': -1}
score model_forest_hyp: 0.9951
regression_error: 0.1165
Execution time: 9.633936405181885
C:\Users\Utente\Desktop\DataScience\code\util.py:28: 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:29: 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: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['max'] = window.max()
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['std'] = window.std()
<Figure size 432x288 with 0 Axes>
In [19]: 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
```

6

```
Total number of regions: 11
Total number of road categories: 6
              year
                      region_id ...
                                           lorries all_motor_vehicles
      1547.000000
                    1547.000000
                                       1547.000000
                                                           1547.000000
count
                                 . . .
       2005.475760
                       6.212023
                                          0.283026
                                                               5.028188
mean
          7.517803
                       3.187790
                                          0.292345
                                                               3.616128
std
       1993.000000
                       1.000000
                                          0.000648
                                                               0.010438
min
                                 . . .
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 OF 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 all\_motor\_vehicles other\_vehicles name year ... North East 12108 ... 12.274201 0.221726 Wales 10090 ... 1 18.260918 0.309984 London 10090 ... 2 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 South East 12108 ... 54.908644 0.713793 10

```
***********************
VEHICLE TYPE COMPARISON
   year region_id ... all_motor_vehicles other_vehicles
   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
                                        3.32911
pedal cycles
                                        2.73903
| two wheeled motor vehicles |
                                        2.29719 İ
buses and coaches
+----+
***********************
MODEL Linear regression 2
features= Index(['year', 'region_id', 'road_category_id', 'total_link_length_km',
      'total_link_length_miles'],
     dtype='object')
score linear2: 0.1807
regression_error: 2.6557
**********************
MODEL SV Regression
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
score: 0.6721
regression_error: 1.3146
Execution time: 2.567014455795288
**********************
MODEL Random forest regression hyp
features= Index(['year', 'region_id', 'road_category_id'], dtype='object')
{'max_features': 'auto', 'n_estimators': 100, 'n_jobs': -1}
score model_forest_hyp: 0.9977
regression_error: 0.0969
Execution time: 8.689334154129028
C:\Users\Utente\Desktop\DataScience\code\util.py:28: 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:29: 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: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['max'] = window.max()

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['std'] = window.std()

<Figure size 432x288 with 0 Axes>

In [20]:
```